TECHNISCHE UNIVERSITÄT
CHEMNITZ

Fakultät für Informatik

CSR-25-07

Integration of Learning Analytics into the ARC-Tutoring Workbench

Chaitanya Sravanthi Akula · Ummay Ubaida Shegupta ·
Wolfram Hardt

August 2025

Chemnitzer Informatik-Berichte

TECHNISCHE UNIVERSITÄT
CHEMNITZ

Integration of Learning Analytics into the ARC-Tutoring Workbench

Master Thesis

Submitted in Fulfilment of the
Requirements for the Academic Degree
M.Sc.

Dept. of Computer Science
Chair of Computer Engineering

Submitted by: Chaitanya Sravanthi Akula
Student ID: 558712
Date: 28.11.2024

Supervising tutor: Prof. Dr. W. Hardt
Ummay Ubaida Shegupta, M.Ed.

Acknowledgement

I would like to express my deepest gratitude to everyone who contributed to the successful completion of this research work on the ARC Tutoring Workbench.

First and foremost, I would like to thank my supervisor, Ummay Ubaida Shegupta, for her invaluable guidance, support, and encouragement throughout the course of this research. Her insightful feedback and expertise were instrumental in shaping this project and ensuring its success.

I am also grateful to the Prof. Dr. Dr. h. c. Wolfram Hardt for his constructive suggestions during the development of the ARC Tutoring Workbench. Also, grateful to the faculty members and tutors at Technical University of Chemnitz, their inputs helped refine the functionalities of the dashboard, making it a more effective tool for both students and instructors.

A special thanks to my team members, whose collaboration and dedication made this project possible. I appreciate their hard work in developing key components such as the Avatar-Based Chatbot, Learning Analytics Modules, and the Time Management Tool. Working together as a team has been an enriching experience, and I am grateful for their contributions.

I would also like to acknowledge the students who participated in the surveys and provided valuable feedback on the user experience of the dashboard. Their insights were crucial in evaluating the effectiveness of the platform.

Lastly, I would like to thank my family and friends for their unwavering support and encouragement throughout this journey. Their belief in me kept me motivated during challenging times.

This research would not have been possible without the collective efforts of all these individuals, and I am deeply thankful for their contributions.

Abstract

The evolving educational landscape increasingly demands tools that provide personalized and data-driven support for students and tutors. This research presents the design and development of the ARC Tutoring Workbench, a comprehensive dashboard that integrates multiple modules—Login, Descriptive Learning Analytics (LA), Diagnostic LA, Predictive LA, an avatar-based chatbot, and a Time Management tool. The primary goal of this workbench is to enhance academic performance and streamline task management for students while enabling tutors to provide targeted assistance.

For students, the workbench leverages Learning Analytics to present current performance scores, diagnose learning gaps, and predict future outcomes. This predictive capability empowers students to focus on areas requiring improvement, ultimately supporting better academic achievement. Additionally, the Time Management tool facilitates effective planning by allowing students to organize their tasks and milestones, while the chatbot provides valuable support in crafting reports and designing presentations.

Tutors benefit from comprehensive data visualization and analysis tools that enable them to monitor the performance of all their students. The dashboard highlights students in need of additional help, allowing tutors to provide tailored guidance. Advanced filtering options allow tutors to dig deeper into student data, fostering a more personalized and efficient approach to education.

Access to the dashboard is secured through a robust login system, ensuring that students and tutors can only view their respective dashboards. By integrating these modules into a unified system, the ARC Tutoring Workbench fosters an environment of informed decision-making and proactive intervention, ultimately enhancing the teaching and learning experience. This research contributes to the field of educational technology by demonstrating how integrated dashboards can support personalized learning and targeted instruction.

Keywords: Learning Analytics, Learning Analytics Dashboards, Dashboards, Workbench

Contents

List of Figures

List of Tables

List of Abbreviations

LA Learning Analytics

LAD Learning Analytics Dashboards

SQL Structured Query Language

ARC Assessment, Recommendations and Conversational

ARS Audience Response System

LMS Learning Management System

DB Database

HCI Human Computer Interaction

UX User Experience

UI User Interface

LX Learner Experience

LXD Learner Experience Design

LIDT Learning and Instructional Design Technology

ID Instructional Design

FIT Feedback Intervention Theory

SRL Self Regulated Learning

CLT Cognitive Load Theory

SCT Social Comparison Theory

GDPR General Data Protection Regulation

IDE Integrated Development Environment

NPM Node Package Manager

JS Java Script

MLR Multiple Linear Regression

ETL Extract Transform and Load

ML Machine Learning

AI Artificial Intelligence

OPAL Online Platform for Academic Learning

URL Uniform Resource Locator

LAV Learning Analytics Visualizations

1 Introduction

In recent years, the adoption of Learning Analytics (LA) has grown significantly within educational settings, particularly in online and blended learning environments. The target of LA is to understand and to make best use of both learning processes and the environments in which they occur. The increasing demand for personalized, data-driven learning experiences has improved the development of advanced systems that utilize LA to improve both teaching methods and student outcomes [5]. This chapter explores the basic concepts of Learning Analytics and its role in enhancing educational practices [6].

1.1 Learning Analytics

Learning Analytics refers to the process of gathering, analyzing, and reporting data about students and their learning environments. The objective is to understand how learning happens and to improve both the learning process and the settings in which it occurs [7, 8]. The main goal of Learning Analytics in education is to utilize student data to enhance their learning outcomes. By providing valuable insights into the learning process, Learning Analytics helps students improve their performance and in enabling tutors to adjust and enhance their teaching methods accordingly.

1.1.1 Importance in Online Education

The significance of Learning Analytics in online education has grown exponentially, particularly in recent years. The COVID-19 pandemic acted as a catalyst to give a rapid shift towards digital learning platforms, requiring innovative approaches to support both students and tutors in this new environment [9]. Learning Analytics has emerged as a great tool in addressing the unique challenges faced in remote teaching and learning scenarios [10]. It provides Real-time insights into student engagement and progress, Early identification of at-risk students, Personalized learning pathways and Data-driven decision-making for curriculum development [10].

1.1.2 Types of Learning Analytics

Learning Analytics are mainly categorized into four types. Each addressing different aspects of the educational process [1]. This classification is mainly based on the task each LA is concerned about. The same can be seen from the figure 1.1.

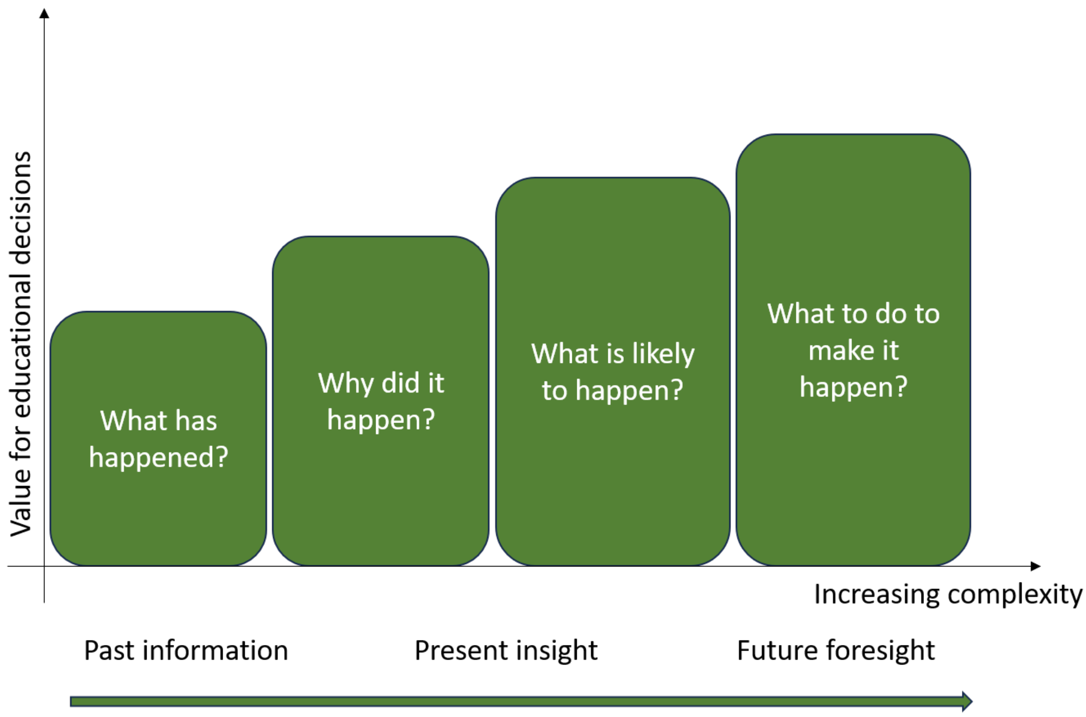

Figure 1.1: Types of Learning Analytics [1]

Descriptive Analytics

Answers questions related to 'What has happened in the past?' It uses aggregation and data mining techniques to extract underlying patterns in the data [11]. Descriptive analytics provide a foundation for understanding historical trends and behaviors in educational data.

Diagnostic Analytics

Utilizes techniques such as drill-down analysis, data mining, and correlation studies to answer the questions related to 'Why did something happen?' [11]. In the domain of learning analytics, this helps tutors understand the root causes of specific learning outcomes or behaviors.

Predictive Analytics

Employs statistical models and forecasting techniques to answer questions related to 'What can happen in the future?' [11]. Predictive analytics enable proactive interventions by identifying potential future trends or challenges in student learning.

Prescriptive Analytics

Leverages optimization and simulation algorithms to address questions such as 'What to do if I need this outcome?' or 'How should I make this happen?' [11]. This advanced level of analytics provides actionable recommendations to optimize learning outcomes.

These four levels of Learning Analytics together create a comprehensive framework for understanding, predicting, and enhancing educational processes. By using the power of data, Learning Analytics has the potential to transform education, making it more responsive, efficient, and tailored to individual learner needs.

1.1.3 Learning Analytics Visualization

Learning Analytics Visualizations (LAVs) are essential elements designed to assist stakeholders in interpreting complex educational data. These visualizations convert raw data into actionable insights by presenting it in user-friendly formats such as graphs, charts, heatmaps, or interactive dashboards. LAVs play a pivotal role in supporting decision-making processes within education by enabling stakeholders to track performance, identify at-risk students, and conduct timely interventions. By simplifying the interpretation of large datasets, LAVs enhance the ability of tutors and learners to make informed decisions that improve learning outcomes [12]. Additionally, LAVs have been shown to contribute to study success by providing personalized feedback and actionable insights that align with individual learning goals [13]. LAVs are combined in one place called Learning Analytics Dashboards to offer a variety of outcomes. Let us see more about LADs in the next section of this paper. This section is mainly concerned in providing a basic picture of Learning Analytics Visualizations and their contributions in enhancing educational practices.

Types of Visualizations LAVs come in various forms, each suited to specific types of data and educational contexts:

- **Bar Charts and Line Graphs**: These are commonly used to show trends over time, such as student performance across different assessments or engagement levels in an online course [14].
- **Heatmaps**: Heatmaps visualize areas of high or low activity within a dataset. For example, they can be used to show which sections of a course are most frequently accessed by students [15].
- **Pie Charts**: Pie charts break down categorical data into proportions. They can be used to represent the distribution of grades or completion rates for different assignments [16].
- **Scatter Plots**: Scatter plots help visualize correlations between two variables, such as the relationship between time spent on an activity and performance outcomes [17].

- **Dashboards**: Dashboards aggregate multiple visualizations into a single interface, providing a comprehensive view of student activities and performance metrics. Dashboards integrate machine learning models to offer predictive insights into student behaviors and performance trends [18].

1.2 Learning Analytics Dashboards

Learning Analytics Dashboards (LADs) are defined as "displays that contain different indicators about learners, learning processes and/or learning contexts and outcomes using one or multiple visualizations" [19, 20]. LADs are the most common way of representing Learning Analytics. They are mainly helpful in identifying at-risk students, facilitating group work, and other educational processes [21].

The present-day LAD research is also exploring advanced applications, such as Monitoring students' emotions based on their facial expressions in online learning sessions, providing personalized insights to enhance self-regulated learning, supporting tutors in data-driven decision-making for instructional improvements etc.

Students and tutors are the primary stakeholders of these LADs [22]. For students, visualizing learning data can help in obtaining self-awareness of their learning behaviors [23]. For tutors, LADs are instrumental in identifying students who need extra help [19].

Recent advancements in LAD design focus on:

- **Adaptive Visualizations:** Dashboards that adjust their display based on user preferences and data relevance.
- **Predictive Analytics:** Incorporating machine learning models to forecast student outcomes.
- **Multimodal Data Integration:** Combining traditional academic data with new data sources for a more comprehensive view.

As part of this research, a LAD named 'ARC Tutoring Workbench' was developed. This dashboard was tested with both students and tutors to gain critical insights and identify areas for improvement. Our testing protocol focused on usability and user experience, effectiveness of data interpretation and impact on learning outcomes. This iterative development and testing process aligns with best practices in LAD design, emphasizing user-centered approaches and continuous refinement based on the stakeholder feedback.

1.3 Problem Background

This research is mainly about integrating Learning Analytics into the Tutoring Workbench. The Learning Analytics part of this research are Descriptive, Diagnostic and Predictive LA. And tutoring is offered in terms of performance in online assessments, recommendation of topics based on students interests, reminder based time management tool and a conversational chatbot for quick help. The same is diagrammatically represented and shown in the figure 1.2. The yellow boxes are the components to be integrated and the purple box in the figure 1.2, highlights the primary task of this research.

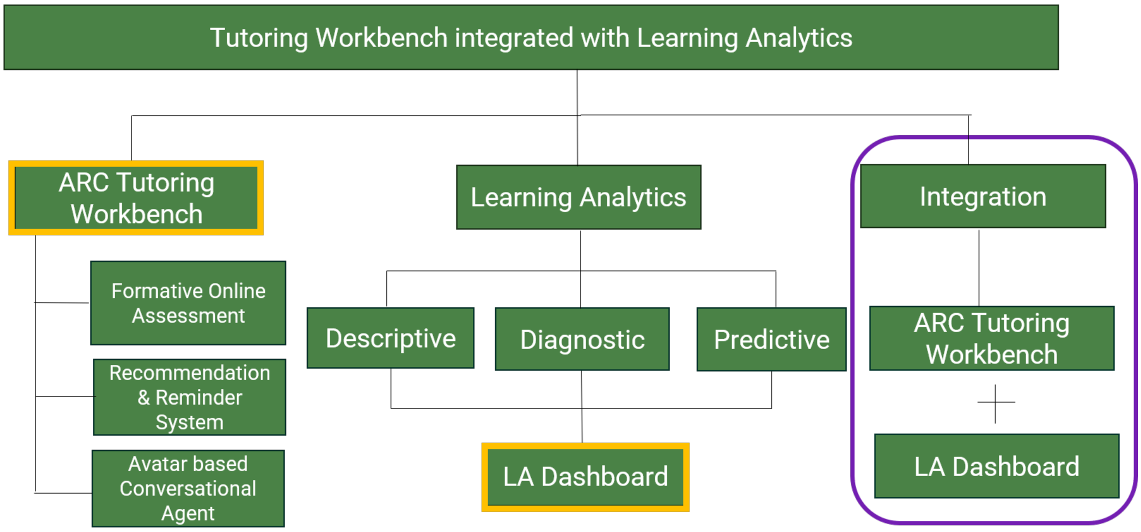

Figure 1.2: Components of ARC Tutoring Workbench Dashboard

Nowadays, we have many recent advancements in offering support to students via e-learning systems, Learning Management Systems(LMS), Artificial Intelligence(AI), Machine Learning(ML) models etc.

- In online learning environments, students frequently experience a lack of individualized support and timely feedback. Conventional e-learning platforms are often rigid and fail to adjust to the unique needs of each learner. This absence of personalization can lead to disengagement, lower academic performance, and increased dropout rates [24].

- Learning Management Systems (LMS) such as Moodle and Blackboard typically do not offer sufficient insights into the reasons behind student difficulties or provide predictive capabilities for future performance [25, 26, 27]. These platforms are often limited in their ability to deliver diagnostic or predictive analytics that could help tutors intervene proactively [28].

- Artificial Intelligence(AI) is underutilized due to a lack of integration with existing systems or insufficient awareness of its potential benefits [29].

- Machine Learning(ML) models have potential but are rarely applied effectively within traditional educational platforms [30].

In order to enhance personalized learning support through data-driven insights and interventions. This workbench delivers personalized tutoring support. It integrates various tools to assess student performance, provide tailored recommendations, and engage students through conversational agents.

Each component as shown in the figure 1.2, was developed by different students. My main task is to integrate them all into a single dashboard. This is mainly consumed by tutors and students as the primary stakeholders. Tutors can use this dashboard to monitor their students progress, identify at-risk students, and make informed decisions about interventions. Where as students can view their own performance metrics. This helps them to track their own progress over time.

1.4 Motivation

The integration of Learning Analytics (LA) into educational platforms has become increasingly important as educational institutions seek to improve student engagement, personalize learning experiences, and make data-driven decisions. The ARC Tutoring Workbench, leverages LA to provide personalized support through formative assessments, recommendations, and conversational agents. This section outlines the motivation behind integrating LA into the tutoring workbench. The points mentioned in the figure 1.3 are explained in detail as follows:

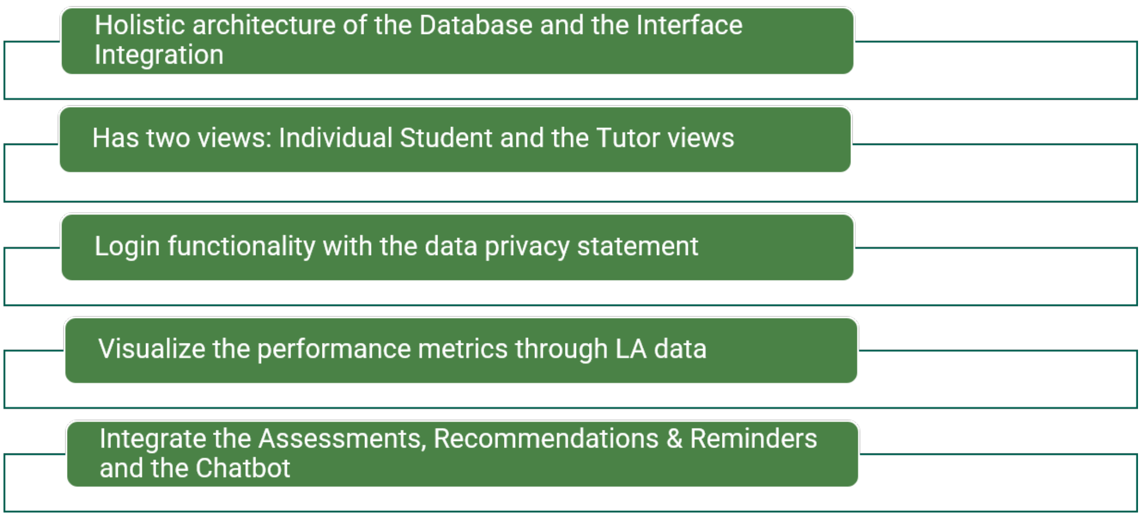

Figure 1.3: Motivation behind ARC Tutoring Workbench

1. Holistic Architecture of the Database and Interface Integration
One of the primary motivations for integrating LA into the ARC Tutoring Workbench is to create a comprehensive system that seamlessly integrates data collection, analysis, and presentation processes. A well-designed holistic architecture ensures

that data from various sources, such as student interactions, assessments, and learning activities, can be efficiently processed and stored in a centralized database. This architecture allows for real-time data retrieval and analysis, enabling educators to make informed decisions based on up-to-date information. The integration of LA into this architecture ensures that data is not only collected but also analyzed to provide meaningful insights that can be used to enhance the learning experience.

2. Two Views: Individual Student and Tutor Views
The ARC Tutoring Workbench is designed to cater both students and tutors by providing two distinct views i.e., the individual Student View and the Tutor View. This dual-view system allows students to monitor their own progress while enabling tutors to track student performance across various metrics. By integrating LA into these views, students can receive personalized feedback on their performance, while tutors can access detailed analytics on student engagement, assessment scores, and learning behaviors. This dual-view approach ensures that both students and tutors have access to relevant data that can inform their learning or teaching strategies [31].

3. Login Functionality with Data Privacy Statement
Data privacy is a critical concern in any educational platform that collects and analyzes student data. The ARC Tutoring Workbench includes a robust login functionality that ensures only authorized users can access sensitive data. Additionally, the system incorporates a data privacy statement that informs users about how their data will be collected, stored, and used in compliance with regulations such as GDPR (General Data Protection Regulation). By integrating LA into a secure platform with clear data privacy protocols, the system ensures that student data is protected while still allowing for meaningful analysis [32]. This approach addresses concerns about data security while providing transparency about how learning analytics are used.

4. Visualizing Performance Metrics through LA Data
One of the key benefits of integrating LA into the ARC Tutoring Workbench is the ability to visualize performance metrics in a way that is easy to understand for both students and tutors. The system uses LA to analyze data from assessments, quizzes, and other learning activities and presents this information through intuitive visualizations such as charts, graphs, and dashboards. These visualizations help students track their progress over time and identify areas where they need improvement. For tutors, visualizing performance metrics allows them to quickly identify at-risk students or trends in class performance that may require intervention [33]. By making complex data more accessible through visualization tools, LA enhances both learning and teaching outcomes.

5. Integration of Assessments, Recommendations & Reminders with Chatbot
The integration of formative assessments with personalized recommendations and

reminders is another key motivation for incorporating LA into the ARC Tutoring Workbench. The system uses LA to analyze student performance on assessments and generate personalized recommendations for further study or improvement. Additionally, the system sends reminders to students about upcoming deadlines or tasks they need to complete based on their progress. These recommendations and reminders are delivered through an integrated chat-bot, which interacts with students in real-time to provide guidance and support [34]. This integration ensures that students receive timely feedback and support tailored to their individual needs.

The integration of Learning Analytics (LA) into the ARC Tutoring Workbench is not merely an enhancement but a necessity to address the growing complexities and demands of modern education. LA provides a data-driven approach to understanding and improving learning processes, offering significant benefits to both students and tutors. Below are detailed reasons, supported by research, that highlight the importance of integrating LA into the ARC Tutoring Workbench.

1. Enhancing Personalized Learning
One of the primary goals of Learning Analytics is to enable personalized learning experiences tailored to individual student needs. By analyzing data such as test scores, participation rates, and behavioral patterns, LA can provide insights into each student's strengths and weaknesses. This allows the ARC Tutoring Workbench to offer personalized recommendations, such as research topics which align with a student's academic goals and progress. Research has shown that personalized learning significantly improves student engagement and outcomes [35].

2. Supporting Data-Driven Decision-Making
Learning Analytics equips both students and tutors with actionable insights derived from the data collected. The ability to make informed decisions based on data is crucial in modern education. Data-driven decision-making enables tutors to refine their teaching strategies, allocate resources effectively [36]. The ARC Tutoring Workbench leverages these capabilities to empower tutors with tools for monitoring class-wide performance while providing individualized support.

3. Proactive Interventions Through Predictive Analytics Predictive Learning Analytics is a powerful feature that forecasts future outcomes based on current data. By integrating predictive models into the ARC Tutoring Workbench, tutors can identify students who are at risk of failing or dropping out and take required measures to prevent them from failing. For example, predictive analytics can forecast final grades enabling tutors to focus their efforts on students who require additional support [37].

4. Improving Engagement Through Real-Time Feedback
Learning Analytics improves student engagement by providing feedback on their progress. Features like self-tests in the ARC Tutoring Workbench allow students to assess their own knowledge on specific topics and receive immediate feedback

on areas for improvement. Real-time feedback has been shown to increase student motivation and accountability [38].

5. Streamlining Time Management
Time management is an important skill for academic success, particularly for students juggling multiple responsibilities such as coursework, internships, and thesis work. The Time Management Tool in the ARC Tutoring Workbench integrates Learning Analytics to help users organize tasks, set milestones, and track deadlines effectively. By analyzing task completion patterns and providing reminders or suggestions tit helps users stay on track with their academic responsibilities.

6. Enhancing Tutor Effectiveness
For tutors managing large groups of students, monitoring individual progress can be overwhelming without the right tools. Learning Analytics simplifies this process by consolidating data into dashboards that provide a holistic view of class performance.

The integration of Learning Analytics into the ARC Tutoring Workbench addresses several key challenges in modern education. Such as providing personalized support through formative assessments, predictive analytics, and real-time interaction via chat-bots. By leveraging real-time data on student behavior and performance, this system enables tutors to make informed decisions. Tutors can offer interventions while ensuring that students receive timely feedback tailored to their individual needs.

As research continues to demonstrate the benefits of Learning Analytics in improving student engagement and academic outcomes [39], it becomes increasingly clear that integrating these tools into educational platforms will play a crucial role in shaping the future of personalized learning.

Chapter one gives introduction to the concepts that are the foundation for conducting the rest of the research. The Research Background chapter gives an overview of the modules that are part of ARC Tutoring Workbench and how they were implemented. The State of the Art chapter talks about the concepts of User Experience and Learning Experience Design and how they were applied to design an appealing LAD and also highlights the research papers considered for this study. Next comes the Concept chapter, which explains the concepts that are involved in building the LAD. The actual implementation process and the tools and technologies used are explained in detail in the chapter named Implementation. Then the results obtained are showcased and the developed dashboard is evaluated with the corresponding users. All this is mentioned in the Results and Evaluation chapter. Additional information is provided in the Appendix chapter.

2 Research Background

The Research Background chapter provides an in-depth overview of the key components and functionalities developed as part of the ARC Tutoring Workbench. This chapter outlines the foundational modules that support the overall functionality of the dashboard, including Login functionality, Descriptive and Diagnostic Learning Analytics (LA), Predictive LA, the Avatar-Based Chat-bot and the Time Management tool. Each of these modules plays a critical role in enhancing the user experience, improving student performance tracking, and providing real-time support for both students and tutors.

All these modules have been already designed and developed by my fellow students. My task is to integrate all of them into a dashboard. So, the integration task of mine will be explained in-detail in the implementation chapter of this report. As part of this chapter named Research Background, I will introduce the existing components and the tools and technologies used in their development. A highlevel picture of the components involved are shown in the figure 2.1. The Login module ensures secure access to the dashboard by managing user authentication and session handling. The Descriptive and Diagnostic LA modules provide insights into student performance through real-time data visualization and pattern recognition. The Predictive LA module uses advanced algorithms to forecast future student performance based on current data. The Avatar-Based Chat-bot serves as an interactive assistant, offering personalized support for tasks such as report writing and presentation preparation. Finally, the Time Management tool helps students and tutors in listing out their tasks and plan them accordingly. This chapter will explore each of these components in detail, explaining their development, functionality, and significance within the ARC Tutoring Workbench. The overview of the modules involved in the ARC Tutoring Workbench are summarized in the table 2.1.

2.1 Login

In addition to the various LA modules, chat-bot functionality and the time planner tool, the ARC Tutoring Workbench also includes a robust login and authentication system. This functionality ensures secure access to the dashboard for both students and tutors, allowing them to view personalized data and perform tasks based on their roles [40].

A. Backend Development: Python and Flask
The login and authentication system was developed using Python and Flask for

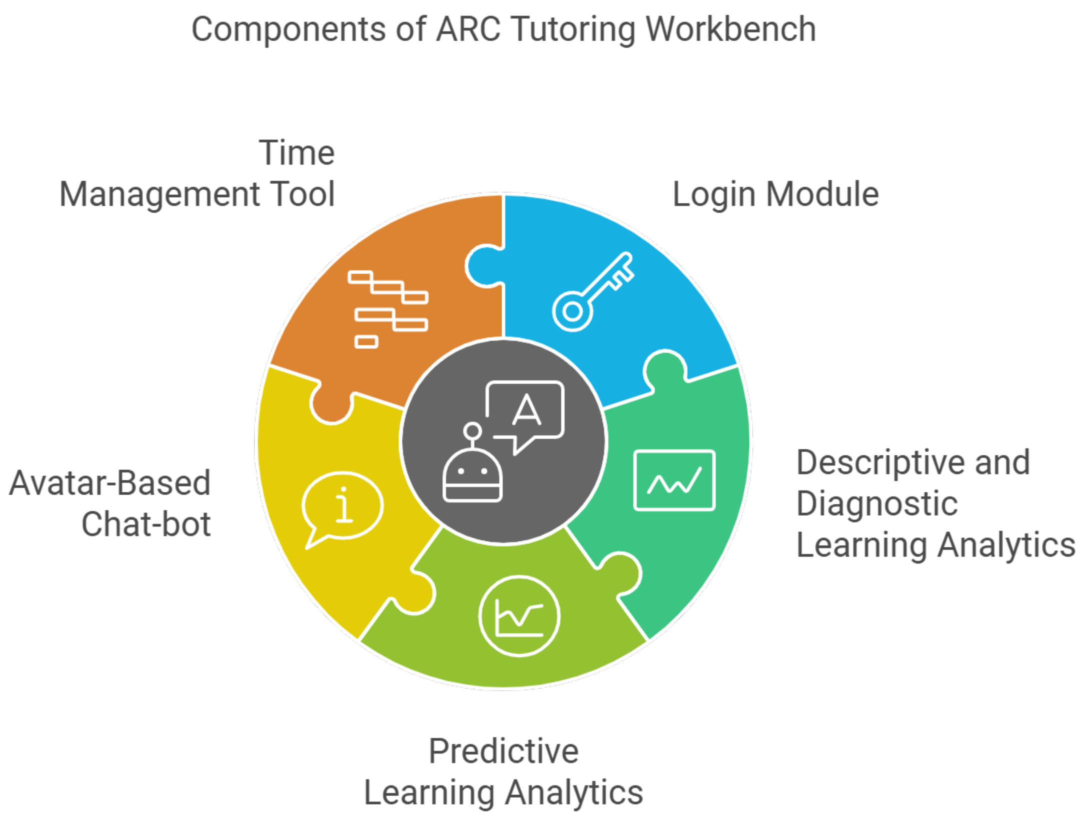

Figure 2.1: Components involved in the ARC Tutoring Workbench

the backend. Flask, being a lightweight web framework, provides an efficient way to handle HTTP requests, manage user sessions, and interact with databases. The authentication system is designed to ensure that only authorized users can access the dashboard. It offers different levels of access based on whether the user is a student or a tutor [40]. The user credentials (such as usernames, passwords, and roles) are securely stored in a MySQL database. Passwords are encrypted using hashing algorithms to ensure that sensitive information is protected. Once a user logs in, a session is created, which keeps track of the user's activity within the dashboard. The session data is also stored in the MySQL database under a dedicated table for sessions. This allows users to maintain their login status across different pages of the dashboard [40].

B. Frontend Development: ReactJS

The frontend of the login functionality was developed using ReactJS, which ensures a smooth and responsive user experience. ReactJS allows for dynamic updates to the user interface (UI) without requiring full-page reloads, making interactions with the dashboard seamless [40, 41]. The login form is built using React components that

Table 2.1: Modules of ARC Tutoring Workbench and their intended functionalities and use cases.

Module Name	Functionality	Use Case
Login	Ensures secure access to the dashboard for both students and tutors	Access role-specific dashboards
Descriptive & Diagnostic LA	Descriptive LA focuses on providing insights into student performance across various test Diagnostic LA, provides a more in-depth analysis through a correlation matrix that compares performance across different types of tests.	Provide actionable insights about student learning behaviors and outcomes. Make data-driven decisions to personalize teaching.
Predictive LA	To predict the future scores of students based on the current performance trends.	Topic recommendations and alerting students. For tutors in offering extra support to low performing students.
Chatbot	Serves as an interactive assistant for students, answering their queries related to various academic tasks. It enhances the user experience by providing real-time support, helping students navigate through their academic responsibilities	Structuring reports and presentations.
Time Management Tool	Allows users to plan, track, and update their academic tasks. Helps in better organization of their academic schedules.	Students for planning tasks and milestones in Hauptseminar, Research Internship and Master Thesis

capture user input (username and password) and send it to the Flask backend for validation via an API call [40]. Based on the user's role (student or tutor), different views are rendered on the frontend after successful login to the dashboard[40]. For example, students can view their own performance data, including test scores and predictive analytics.

C. Cookie-Based User Identification
To enhance user experience and streamline session management, cookies are used to store session-related data on the client side [40]. When a user logs in, a cookie is created that contains essential information such as Matriculation Number, Email ID, password in encrypted format, user role whether a student or a tutor. Storing such data in the cookie, allows the system to fetch personalized data from the database based on this matriculation number. For example, fetching specific test scores or predictive analytics results—based on who is currently logged in [40].

D. Security Considerations
Several security measures have been implemented to ensure that user data remains safe. Passwords are hashed before being stored in the database, ensuring that even if unauthorized access occurs, sensitive information remains protected. Also, to prevent unauthorized access through stale sessions, session tokens stored in cookies have an expiry time. Once expired, users are required to log in again. The system ensures that students can only view their own data while tutors have broader access based on their role [40].

The integration of login functionality into ARC Tutoring Workbench ensures secure access for both students and tutors. By leveraging Python/Flask for backend development and ReactJS for frontend design, this system provides smooth authentication processes while ensuring that sensitive data remains protected through encryption and secure session management using cookies. Overall, this login system acts as a protector in governing data privacy and security within ARC Tutoring Workbench while enabling users to interact with personalized content efficiently.

2.2 Descriptive and Diagnostic LA

The Descriptive and Diagnostic Learning Analytics (LA) sections have been developed by one of my team members, with the front-end built using JavaScript (.js) and the back-end developed using Python and Flask [42]. All data processing and storage are handled through Pandas DataFrames, which provide an efficient way to manage and manipulate the data for analysis [42].

A. In Student View
In the student view, descriptive learning analytics focuses on providing insights into student performance across various tests. The key metrics include, the number of students who participated in each test, total marks obtained by each student across all tests, the marks obtained in each test, with the ability to drill down further to view detailed performance.

For example, students can see their scores for each question compared to the maximum points possible for that question. This granular level of detail helps students understand their strengths and weaknesses in specific areas of each test [42].

B. In Tutor View
Descriptive learning analytics provides an overview of student participation, showing the number of students who participated in each test. This gives tutors a high-level understanding of student engagement across different assessments [42].

For diagnostic learning analytics, a more in-depth analysis is provided through a correlation matrix that compares performance across different types of tests. These tests include 'Self_Test_Search', 'Self_Test_Discussion', 'Self_Test_Presentation', 'Self_Test_Report', 'ARS_Search', 'ARS_Discussion', 'ARS_Presentation', 'ARS_Report'.

The correlation matrix allows tutors to identify relationships between different types of assessments, helping them understand how performance in one type of test (e.g., presentations) might correlate with performance in another (e.g., reports). This diagnostic insight can be used to identify patterns in student learning behaviors and inform instructional adjustments. Additionally, tutors have access to all students' scores, enabling them to monitor individual and group performance over time [42].

This system is initially designed to take input data from the TUC server and then perform data cleansing and storing the pre-processed data in pandas dataframes. Later to remove the dependency on server issues, the input data is stored in the form of excel sheets locally. This excel data has anamolies and redundancies. All of them were normalized and then stored in the MySQL database. Now this system is designed to work in both the ways i.e., it can fetch the data directly from server or fetch the cleansed data from the MySQL Database.

By integrating both descriptive and diagnostic learning analytics, this system provides a comprehensive view of student performance from both the student's and tutor's perspectives. The use of Pandas DataFrames ensures that data is processed efficiently, allowing for real-time updates and detailed insights into student learning outcomes [42].

2.3 Predictive LA

The Predictive Learning Analytics (LA) component was initially developed by another team member, using a combination of Python for data pre-processing and C# for implementing the predictive functionality. The goal of this component is to predict students' future performance based on their current and past performance data. The key predictive model used is Multiple Linear Regression (MLR), which helps forecast future outcomes by analyzing the relationships between multiple independent variables and a dependent variable [2].

A. Tools and Technologies Used:
Python was employed for pre-processing the data, where operations such as cleaning, merging, and transforming datasets were performed using Pandas DataFrames. This

ensured that the data was in a suitable format for further analysis. The predictive model itself was developed using C#, where Multiple Linear Regression (MLR) was applied to predict students' future scores. The three primary variables used in the MLR model are Self_Test scores, ARS_Test scores, Topic_Recommender scores [2].

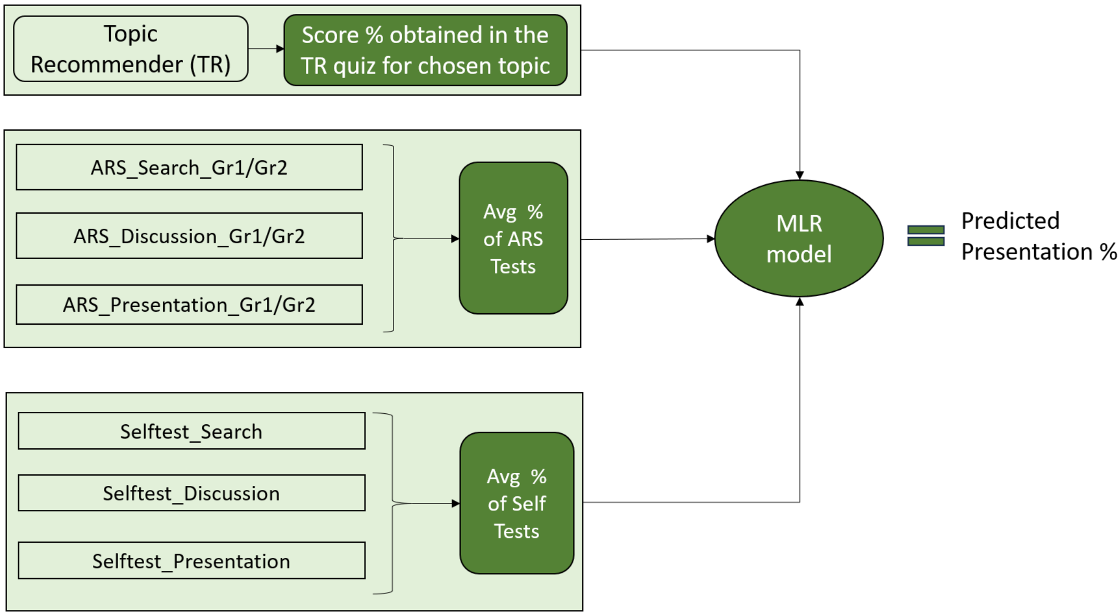

Figure 2.2: Calculation of presentation prediction % [2]

B. Data Sources and Merging:
The Self_Test and ARS_Test scores were extracted from the database table Student_Test_Data. These tables were merged with the Student and Semester tables based on common columns like Student_ID and Semester_ID respectively. This allowed for a comprehensive dataset that linked student performance across different tests and semesters [2, 43].

The Topic_Recommender score is derived from a quiz where students are recommended the top five topics based on their quiz performance. The score from the topic selected by the student is considered as an input variable for MLR [2, 43].

The calculations involved in deriving the score predictions using the independent variables were shown in the figures 2.2, 2.3, 2.4. The final MLR model uses these three variables to predict future student performance, providing valuable insights into how students might perform in upcoming assessments based on their current progress [2].

C. Front-end Development:
The front-end of this predictive LA module was developed using ASP .Net MVC, which provides an interactive interface where tutors can view predictions for each

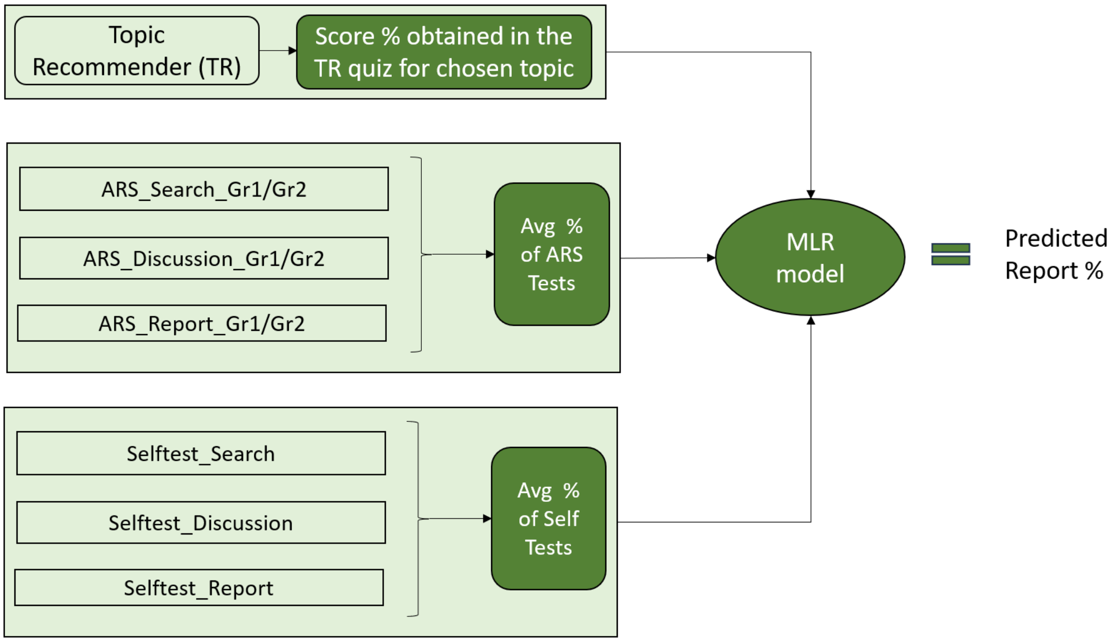

Figure 2.3: Calculation of report prediction % [2]

student. This interface allows tutors to analyze predicted performance trends and make informed decisions about interventions or additional support [2].

D. Data Storage:
All relevant data, including student test scores and quiz results, is stored in an SQL database. SQL is used to manage and query this data efficiently, ensuring that the backend can quickly retrieve necessary information for both pre-processing in Python and predictive modeling in C# [2].

E. Re-Implementation of Predictive LA
The above mentioned Predictive LA functionality was later re-implemented using Python and Flask for the back-end, along with ReactJS and NPM for the front-end. The calculations used can be seen from the figures 2.2, 2.3, 2.4. The same Multiple Linear Regression (MLR) algorithm was used to predict student scores based on the same variables: 'Self_Test', 'ARS_Test', and 'Topic_Recommender' scores.

This re-implementation was necessary because all other modules developed as part of the ARC Tutoring Workbench Dashboard were built using Python/Flask for the back-end and React/NPM for the front-end. To maintain consistency across all modules within the dashboard, it was important to re-implement this functionality using the same technologies. By doing so, seamless integration of all components is ensured through Flask's blueprint architecture, which allows different modules to be combined into a unified dashboard.

In summary, this re-implementation aligns with the overall architecture of the

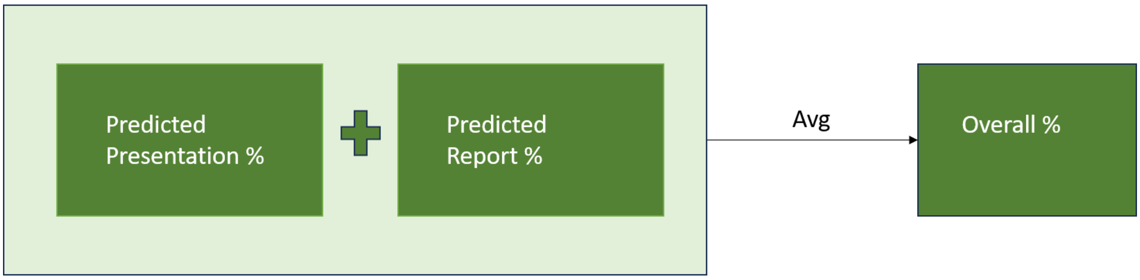

Figure 2.4: Calculation of overall prediction % [2]

ARC Tutoring Workbench Dashboard, ensuring that all modules—whether descriptive, diagnostic or predictive are integrated efficiently into a single platform.

2.4 Avatar based Chatbot

As part of the ARC Tutoring Workbench, an Avatar-based chatbot was developed and integrated into the dashboard by one of the team members. This chatbot serves as an interactive assistant for students, answering their queries related to various academic tasks such as report writing, presentation preparation, and other course-related activities. The chatbot enhances the user experience by providing real-time support, helping students navigate through their academic responsibilities more efficiently [44].

A. Key Features of the Avatar-Based Chatbot
1. Avatar Selection: The chatbot allows users to switch between two avatars i.e., a female avatar named "VITAF" and a male avatar named "VITAM". These avatars provide a personalized and engaging interaction experience, making the chatbot more relatable and user-friendly. Students can select their preferred avatar based on personal preference, which adds a layer of customization to the interaction [44].

2. Emotions:
The chatbot is equipped with five different emotions, which are displayed depending on the context of the conversation or the type of the query being asked. These emotions help humanize the interaction, making it feel more natural and less robotic. The emotional responses are designed to adapt to different scenarios, such as offering encouragement during stressful times (e.g., near submission deadlines) or providing neutral responses when answering factual queries [44].

3. Functionality: The primary purpose of this chatbot is to assist students with academic queries related to Report Writing and Presentation Preparation. The chatbot provides guidelines, tips, and templates for structuring reports effectively. Also, offers advice on how to prepare slides, structure presentations, and deliver them confidently. This functionality ensures that students receive timely assistance without needing to consult external resources or wait for instructor feedback. The chatbot serves as a first point of contact for common questions, allowing students to resolve issues quickly and efficiently [44].

4. JWT Token Integration:
To ensure secure communication between the chatbot and the other components of the dashboard, JWT (JSON Web Token) tokens are used for authentication. JWT tokens help maintain secure sessions by verifying user identities and ensuring that only authorized users can access certain functionalities within the chatbot. This security measure is crucial in maintaining data privacy and protecting sensitive student information [44].

5. Frontend Integration using iFrame:
On the frontend, the chatbot is embedded into the ARC Tutoring Workbench using an iFrame object. This allows seamless integration without disrupting other elements of the dashboard's user interface (UI). The iFrame ensures that the chatbot remains accessible at all times while maintaining a clean and organized layout within the dashboard [44].

B. Use Cases
a. Report Writing Assistance:
Students often struggle with structuring their reports according to academic standards. The Avatar-based chatbot provides step-by-step guidance on how to format reports, including sections like introduction, methodology, results, discussion, and conclusion. It also offers tips on citation styles (e.g., APA or MLA) and helps students avoid common pitfalls such as plagiarism or improper formatting [44].
b. Presentation Preparation Guidance:
Preparing for presentations can be daunting for many students, especially when delivering in front of peers or instructors. The chatbot assists by offering advice on slide design (e.g., keeping slides concise), presentation flow (e.g., organizing content logically), and delivery techniques (e.g., maintaining eye contact or managing time effectively). By providing this support in real-time, students can refine their presentations before seeking further feedback from instructors [44].

The Avatar-based chatbot integrated into the ARC Tutoring Workbench plays a crucial role in enhancing student engagement and providing real-time academic support. By offering customizable avatars (VITAF and VITAM), emotional responses, and specific functionalities related to report writing and presentation preparation, this chatbot serves as an effective tool for improving student productivity while reducing dependency on instructor intervention for routine queries. The use of JWT tokens ensures secure interactions within the dashboard environment, while iFrame integration maintains a seamless user experience without cluttering the interface [44]. Overall, this Avatar-based chatbot is a valuable addition to the ARC Tutoring Workbench's suite of tools designed to support student learning outcomes in a personalized and efficient manner.

2.5 Time Management Tool

The Time Management tool is an essential feature integrated into the ARC Tutoring Workbench, designed to help both students and tutors effectively manage their time and tasks. This tool allows users to plan, track, and update their academic tasks. It helps in better organization and productivity throughout their academic journey. The Time Management tool is particularly useful for managing tasks related to Hauptseminar, Research Internship, and Master Thesis [45].

A. Key Functionalities
1. Default Tasks:
Upon accessing the Time Management tool users are presented with a set of default tasks that are pre-configured based on common academic milestones [45]. These default tasks include activities such as "Literature Research", "Conceptualization", "Programming", "Testing", "Writing" (e.g., reports or thesis). These tasks can be edited, updated, or deleted based on the user's specific needs. For example, if a student has completed a task earlier than expected, they can mark it as complete or update its status [45].

2. Task and Milestone Management:
Users have the flexibility to create new tasks and milestones in addition to the default ones. Each task or milestone can be assigned a start date, end date, status (e.g., "Not Started", "In Progress", "Completed"), and a percentage of completion. This allows students and tutors to break down complex projects into manageable chunks and track progress over it [45]. Students or tutors can add new tasks by specifying the task name, start date, end date, status, and completion percentage. They can also add milestones to represent significant achievements within a project (e.g., completing a chapter of a thesis or submitting a report). These milestones help users focus on key deliverables [45].

3. Task Editing and Deletion:
Each task or milestone can be edited or deleted as needed. For example, if a task deadline changes or if additional work is required on a certain task, users can easily update the task details through the interface. Similarly, if a task is no longer relevant, it can be deleted from the plan [45].

4. Visualization of Tasks:
The Time Management tool provides a visual timeline that displays all tasks and milestones in chronological order. Users can switch between Week View and Day View, depending on their preference for viewing upcoming deadlines or milestones. This visual representation helps users understand how their tasks are distributed

over time and ensures that they stay on track with their academic responsibilities [45].

B. UseCases
The Time Management tool is designed to cater to three primary use cases:

- Hauptseminar: Students enrolled in Hauptseminar courses can use the Time Management tool to organize their seminar-related tasks such as topic selection, literature review, presentation preparation, and final report submission [45].
- Research Internship: For students engaged in research internships, the Time Management tool helps them manage research activities, meetings with supervisors, data collection phases, and report writing [45].
- Master Thesis: The tool is particularly useful for students working on their Master Thesis projects. It allows them to break down the thesis into smaller tasks such as literature review, conceptualization of ideas, programming, testing, writing chapters and preparing for the final defense [45].

C. Backend and Frontend Technologies
The Time Management tool has been developed using modern web technologies. The backend functionality is implemented using Python and Flask, which handle user authentication, task management logic, and database interactions. The frontend interface is built using ReactJS, ensuring a smooth and interactive user experience. Task-related information is stored in an SQL Lite local database, which allows for efficient storage and retrieval of tasks based on user input [45].

D. User Authentication and Data Retrieval
To ensure personalized task management for each user, the system uses cookies to store user-specific data such as the student's matriculation number. Based on the matriculation number stored in the cookie, the corresponding list of tasks is fetched from the database and displayed to the user. This ensures that each student or tutor only sees their own set of tasks and milestones when they log in to the dashboard [45].

E. Benefits of the Time Management tool
1. Better Time Management: By providing a clear overview of all upcoming tasks and deadlines, the Time Management tool helps students manage their time more effectively. It encourages them to plan ahead and stay organized throughout their academic journey [45].
2. Task Tracking: The ability to mark tasks as "In Progress" or "Completed" gives users a sense of accomplishment as they work through their projects [45].
3. Customization: The flexibility to add new tasks or milestones ensures that students can tailor the planner according to their specific academic needs [45].

4. Visual Representation: The visual timeline provides an easy-to-understand overview of how tasks are distributed over time, helping users avoid last-minute rushes by staying on top of deadlines [45].

The integration of the Time Management tool into the ARC Tutoring Workbench adds significant value by helping students and tutors manage their academic workload more efficiently. With features like default tasks, custom task creation, milestone tracking, and real-time updates based on user input, this tool ensures that users stay organized throughout critical academic phases such as Hauptseminar courses, research internships, and Master Thesis projects.

In conclusion, the Research Background chapter has highlighted the core functionalities integrated into the ARC Tutoring Workbench. The Login module ensures secure access and personalized data retrieval for students and tutors. The Descriptive and Diagnostic LA modules provide valuable insights by highlighting trends in student performance. This helps both students and tutors make informed decisions. The Predictive LA module enables proactive interventions by forecasting future performance based on current data. The Avatar-Based Chatbot enhances user engagement by providing real-time assistance with academic tasks.

Together, these modules form a cohesive system that supports both students' learning journeys and tutors' instructional strategies. By combining data-driven insights with interactive tools, the ARC Tutoring Workbench offers a comprehensive solution for improving academic outcomes in a user-friendly environment.

3 State of the Art

The State of the Art chapter provides an in-depth analysis of the current advancements and research in the field of Learning Analytics (LA), and Learning Analytics Dashboards (LADs). To design a good interface, the User Experience Design (UXD) is considered in the creation of Learning Analytics Dashboards. This chapter also conveys information on UX design. This chapter aims to establish a comprehensive understanding of the existing technologies, frameworks, and methodologies that are integrated into modern educational systems, particularly focusing on how they enhance personalized learning experiences.

3.1 Learning Analytics Infrastructure

The Learning Analytics Infrastructure is the foundational system that enables the implementation of LA in educational environments. It typically consists of several key components, including data collection mechanisms, storage systems, analytics engines, and user interfaces that present insights to tutors and learners [46]. At its core, the infrastructure integrates data from various sources such as Learning Management Systems (LMS), formative assessments, student interactions, and other learning tools. This data is then processed through different types of analytics such as Descriptive, Diagnostic, and Predictive analytics.

The ARC Tutoring Workbench, is an example of a system that leverages Learning Analytics infrastructure. It integrates formative online assessments, a recommendation & reminder system, and an avatar-based conversational agent to provide personalized support to students. The workbench interacts with the LA Dashboard, which visualizes insights derived from descriptive, diagnostic, and predictive analytics. This dashboard allows tutors to monitor student progress in real-time and make informed decisions for timely interventions. The integration layer ensures communication between the ARC Tutoring Workbench and the Learning Analytics components by facilitating data exchange between these systems.

3.2 User Experience Design

User Experience (UX) Design is a critical aspect of product development that focuses on enhancing user satisfaction by improving the usability, accessibility and pleasure during the interaction between the user and the product. It encompasses the entire spectrum of the user's interaction with the product. Helps in answering the fundamental question, 'How did the user feel when using a product or an interface?' [47].

UX Design is pivotal in increasing the usability and adoption of a product, serving as a key differentiator in today's competitive market. It goes beyond mere functionality to create meaningful and relevant experiences for users. The discipline involves several key components:

- **User Research:** Understanding the needs, behaviors, and motivations of users through various research methodologies.
- **Information Architecture:** Organizing and structuring content in a way that is intuitive and easily navigable.
- **Interaction Design:** Designing the interactive elements of a product to create a seamless and engaging user journey.
- **Visual Design:** Creating aesthetically pleasing interfaces that align with the product's brand and enhance usability.
- **Usability Testing:** Evaluating the product with real users to identify areas for improvement.

UX metrics play a crucial role in quantifying and measuring users' behavior and attributes when interacting with the product. These metrics may include Task success rate, Time-on-task, Error rate, User satisfaction scores, Net Promoter Score (NPS). The UX design process is iterative and detail-oriented, considering every aspect of the user's journey to offer an optimal experience. This meticulous approach ensures that even the smallest elements of the design are scrutinized and refined to enhance the overall user experience [48].

In the context of educational technology and learning analytics dashboards (LADs), user experience (UX) design plays a pivotal role in ensuring the effectiveness of these tools. A good UX design directly impacts four critical areas: user engagement, learning outcomes, data interpretation, and accessibility.

First, a thoughtfully designed interface can significantly enhance user engagement, encouraging students and tutors to interact with the platform more frequently and for longer durations. Research highlights that dashboards with intuitive designs promote sustained participation, which is particularly important in online and blended learning environments where physical interaction is limited [49].

Second, UX design influences learning outcomes by reducing cognitive load. When users can navigate the system effortlessly, they can focus more on the educational content rather than struggling with the interface. This aligns with findings that emphasize the need for user-centered design principles to improve usability and ensure high-quality data interaction [50].

Third, effective UX design improves data interpretation by presenting complex learning analytics in clear, actionable formats such as graphs or progress indicators.

Studies show that well-visualized data enhances users' ability to make informed decisions about their learning strategies or teaching approaches [51].

Finally, accessibility is another critical dimension of UX design, ensuring that LADs cater to diverse user needs and abilities, including those with disabilities. By addressing these aspects, UX design becomes a cornerstone for creating impactful learning analytics dashboards that foster better engagement, understanding, and inclusivity in education [49, 52].

As technology continues to evolve, UX design remains at the forefront of creating products that are not only functional but also enjoyable and meaningful to use. In the realm of educational technology, it serves as a bridge between complex data analytics and the end-users, making sophisticated tools accessible and beneficial to both tutors and learners.

Comparison of UXD and LXD
The below table 3.1 gives a good comparison of UXD with LXD.

Table 3.1: UXD vs LXD

Aspect	UXD	LXD
Focus	Seamless, enjoyable user experience [53]	Meaningful and engaging learning experience [54]
Target Audience	Users (general consumers)	Learners (students, professionals) [54, 55]
Outcomes	Task completion, satisfaction [56]	Learning goals, knowledge retention [54]
Tools	Figma, Adobe XD, usability tests [53]	LMS platforms, learning analytics, gamification [54, 55]

LXD is essentially UXD tailored to learning contexts. It builds upon the foundation of UXD principles. It can address the unique challenges faced at the time of creating impactful, goal-driven learning experiences. A successful LXD practitioner must be skilled in UXD practices but also able to craft meaningful learning environments.

3.2.1 Learning Experience Design

As the field of Learning and Instructional Design Theory evolves towards more human-centered design practices, the concept of Learning Experience Design (LXD) has emerged as a pivotal approach [57]. This shift represents a significant paradigm change in how we conceptualize and create educational environments.

LXD emphasizes the complex dynamics of how individuals learn and engage with educational materials [3]. This comprehensive approach goes beyond content delivery, addressing the entire learning experience to influence a variety of learning

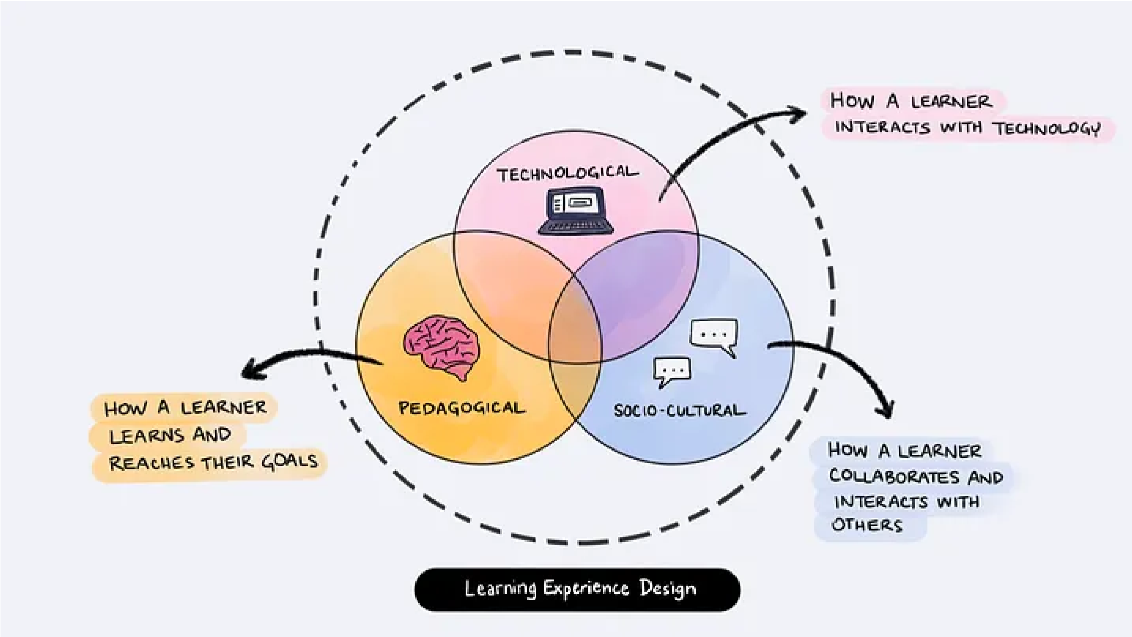

Figure 3.1: Multi-dimensional nature of LXD [3]

outcomes. The significance of LXD is especially clear when examining the challenges posed by poorly designed learning technologies, which can result in learner frustration and disengagement [55].

The integration of User Experience (UX) design principles into learning design practices has paved the way for more productive and engaging digital learning environments. This synthesis acknowledges that effective learning is not just about information transfer but about creating meaningful, enjoyable, and impactful experiences.

Figure 3.1 shows, LXD posits that an individual learner's experience is multidimensional, encompassing three key aspects:

1. **The Technological Dimension:** This aspect focuses on how learners interact with technology. It considers factors such as interface design and usability, accessibility features, device compatibility and responsiveness, integration of multimedia elements [57].

2. **The Pedagogical Dimension:** This encompasses learner interaction with designed materials, instructions, activities, assessments, and how these elements contribute to achieving learning goals. Key considerations include alignment of content with learning objectives, scaffolding of complex concepts, variety in assessment methods and incorporation of active learning strategies [57].

3. **The Sociocultural Dimension:** This dimension explores how learners inter-

act with others in the learning environment. It addresses collaborative learning opportunities, cultural sensitivity and inclusivity, community building within learning platforms, integration of social learning theories [57].

The interplay of these dimensions creates a comprehensive framework for designing learning experiences that are not only effective but also engaging and meaningful [58, 59]. By considering these aspects, LXD practitioners can create learning environments that enhance learner motivation and engagement, improve knowledge retention and application, foster critical thinking and problem-solving skills, accommodate diverse learning styles and preferences and promote lifelong learning attitudes [58, 59].

In the context of digital learning platforms and Learning Analytics Dashboards, LXD principles can be applied to create interfaces that are not only informative but also intuitive and motivating. This approach ensures that learners can easily navigate through content, understand their progress, and feel empowered in their learning journey [60, 61].

As educational technologies continue to evolve, the principles of LXD will play an increasingly crucial role in shaping the future of learning. By placing the learner's experience at the center of design considerations, LXD has the potential to transform educational practices, making them more adaptive, personalized, and effective in meeting the diverse needs of modern learners [58].

3.2.2 Learning Experience Design Techniques

Learning Experience Design (LXD) is a holistic approach to creating educational environments that prioritize the learner's experience [62, 63]. The incorporation of LXD techniques into learning design processes is crucial for developing effective and engaging educational solutions. These techniques can be broadly categorized into three main phases:

A. Identifying User Needs:
The first step in Learning Experience Design (LXD) is identifying user needs, which involves understanding the gap between actual performance and optimal performance [57]. This process typically includes several key activities. First, user research is conducted through methods such as surveys, interviews, and observations to gather insights directly from learners [64]. Additionally, existing performance data and learning outcomes are analyzed to identify patterns and areas for improvement. To ensure that the design addresses diverse user requirements, learner personas are created to represent different user groups, providing a clear picture of their goals, challenges, and preferences [65]. Finally, mapping the learner journey helps uncover pain points and opportunities for enhancing the overall learning experience. Together, these steps form the foundation for designing effective and user-centered learning solutions.

B. Requirements Gathering:
Once user needs are identified, the next step involves generating a set of requirements that define the system capabilities necessary to address those needs [57]. This phase is highly collaborative and typically includes workshops with key stakeholders such as learners, tutors, and subject matter experts. These workshops facilitate open discussions and align diverse perspectives on what the system should achieve. Techniques like brainstorming, data analysis, and use case development are employed to refine ideas and translate user needs into actionable requirements [64]. For example, brainstorming sessions might focus on enhancing feedback mechanisms, improving data visualization, or designing intuitive navigation systems.

Additionally, user stories are developed to capture specific learner needs and expectations in a structured format. These stories help ensure that the system design remains aligned with user goals. Finally, requirements are prioritized based on their potential impact on learning outcomes and their feasibility. This prioritization ensures that critical features are implemented first, optimizing both the learning experience and resource allocation. Together, these activities provide a clear road map for building a system that effectively meets user needs.

C. Prototyping:
In the initial stages of Learning Experience Design (LXD), creating a prototype of the user interface or online learning environment is a crucial step [57]. This process involves several key activities that help designers refine and validate their concept. Designers start by crafting low-fidelity wireframes to establish the fundamental structure of the learning platform. They then progress to developing interactive prototypes that allow for comprehensive testing of user flows and potential interactions [65].

Rapid prototyping techniques play a vital role in this phase, enabling designers to quickly explore and iterate on design concepts with agility. A critical component of this process is conducting usability testing with representative users, which provides invaluable feedback and insights into the user experience [65]. By systematically working through these stages, designers can create a more intuitive, user-centered learning environment that effectively meets the needs of learners.

The use of tools such as personas and prototypes make the learning design process iterative, dynamic, and more responsive to learner needs. This approach aligns with the principles of participatory design and user-centered approaches in LXD [62, 63].

Additional Considerations
When designing learning experiences, designers must adopt a holistic and nuanced approach that goes beyond traditional educational methodologies. Intersectionality emerges as a critical lens, enabling the creation of more inclusive and responsive learning environments that recognize the diverse backgrounds and experiences of learners [66]. By integrating extended reality (XR) technologies and interactive 360º videos, educators can transform online learning into immersive and dynamic

experiences that transcend traditional pedagogical boundaries [67].

The concept of seamless learning further enriches this approach, emphasizing the importance of creating learning experiences that flow smoothly across different contexts and platforms [68]. A multidimensional design strategy becomes crucial, carefully balancing design aesthetics, pedagogical principles, and standardized learning frameworks to maximize learner engagement and educational efficacy [64]. This comprehensive approach ensures that learning experiences are not just informative, but truly transformative, adapting to the complex and varied needs of modern learners.

By incorporating these LXD techniques and considerations, designers can create more effective, engaging, and learner-centered educational experiences. The iterative nature of this process ensures that the final learning environment is well-aligned with user needs and pedagogical goals [62, 69].

3.3 Learning Analytics Dashboards

The table 3.2 presented in this chapter is part of a comprehensive review of recent research studies that focus on the integration of Learning Analytics Dashboards (LADs) in educational environments. The table summarizes key findings from 16 different studies, each of which explores the application of Learning Analytics (LA) in various educational contexts and its impact on stakeholders such as students, teachers, and administrators.

The table 3.2 serves as a valuable resource for understanding how Learning Analytics is being applied across different educational contexts and how it contributes to improving teaching and learning outcomes. By reviewing these studies, we can identify common trends, challenges, and best practices in the design and implementation of Learning Analytics systems.

Table 3.2: Comparison of Research papers used for this study.

S No	**Country and Year**	**Paper Citation**	**Stakeholders**	**Achievements**	**Theory Used**	**Created LAD**
1	Europe and Latin America and 2020	[70]	Teachers	Assists naive advisors in better decision making by comparing two universities' technology adoption levels.	SRL	✗

2	Germany and 2020	[71]	Students	Collected inputs from students and implemented design improvements accordingly. Tracks student performance, predicts final grades, alerts at-risk students.	SRL, SCT, FIT	✗
3	Netherlands and 2020	[72].	Teachers	Grouped indicators into action-related, result-related, social-related indicators to improve retention rates and success rates of students.	SRL, Social Interaction and Collaborative Learning	✗
4	Germany and 2021	[73]	Teachers, Learners, Administrators and Researchers	Conducted a literature review summarizing dashboard visualization techniques and their effectiveness in educational settings.	SRL	✗
5	South Korea and 2021	[74]	Students and Teachers	Developed two dashboards—one for students and one for instructors—providing adaptive feedback during collaborative argumentation sessions.	SRL, CLT, FIT	✓

6	Sweden and 2021	[75]	Teachers	Uses machine learning to provide predictive insights into student behaviors and answers pedagogical questions related to student performance metrics.	SCT, CLT, Constructivism	✔
7	Austria and 2021	[76]	Students	Design and development of TU Graz students' study progress dashboard; tracks study progress compared to peers; official study recommendations; progress in compulsory/optional courses	SRL, SCT, CLT	✔
8	USA and 2022	[77]	Students	Investigates use of LAD named MyLA. Concerned about popularly accessed resources; due assignments; grades;	SRL, CLT	✔

9	France and 2022	[78]	Students	Investigated adaptations requested by students; identified suitable data/visualizations based on profile/course duration. Framed research questions i) Indicators for student LADs? ii) Prevalence of learning indicators by objectives? iii) Links between indicators/need profiles?	SRL, CLT	✗
10	Germany and 2022	[79]	Teachers	Presented design/evaluation of LAD named Learning Analytics Cockpit. Helped teachers understand knowledge levels; monitored students via LA Dashboard	SCT, FIT	✔
11	New Zealand and 2023	[80]	Students	Developed LAD named SensEnablr. Used descriptive/predictive/ prescriptive analytics. Detailed feedback comparing self/peers. Goal: improve self-regulated learning.	SCT, SRL, SCT, Constructivism	✔

12	Sweden and 2024	[7]	Students / Teachers	Reviewed articles on LADs Identified emerging trends Target users; visualization elements; theoretical frameworks; target outcomes/effects	SRL	✗
13	Netherlands and 2020	[81]	Students	Offered a dashboard to freshers group. It shows their progress in the LMS, their predicted chance of passing, their predicted grade and their online intermediate performance compared with the total cohort.	SRL, SCT	✗
14	Germany and 2023	[82]	Teachers	Using this dashboard, popularity of subjects; attendance and completion of learning elements; quick feedback; learning style and student performance, and much more can be studied by the teacher.	SRL, FIT, Constructivism to support Teachers	✗
15	United Kingdom and 2021	[83]	Students	Interviewed students online to know the most useful and the least useful elements of the proposed LAD.	SRL	✗

16	Canada and 2021	[84]	Teachers	Studied how teachers navigate through a LAD. Created a correlation matrix between SRL activities and epistemic emotions among expert and novice teachers.	SRL and emotion theories	✗

A literature research was conducted to know about the studies concerned mainly about Learning Analytic Dashboards that were developed all across the world. This can be seen from the table 3.2. The research papers from the year 2020 to 2024 were considered in this research. As a result a total of 16 papers that found to be relevant were considered. As this research work focuses on developing an LAD for students and tutors primarily, the research papers having students and/or tutors as stakeholders were considered.

Gutierrez et al. [70], proposed a system that assists teachers by comparing universities' technology adoption levels to support decision-making. It highlights the use of LADs in helping naive advisors to make informed choices. Mohseni et al. [75], integrated machine learning into LADs to predict student behaviors. Also to answer pedagogical questions, enhancing teachers' ability to address individual learning needs. Karademir et al. [79], designed a "Learning Analytics Cockpit" for teachers to monitor the knowledge levels and learning progress of students in real-time. Sapsai et al. [82], developed an LAD that allows teachers to analyze learning styles, subject popularity and students' performance to improve the students instructional strategies. Zheng et al. [84], investigated how teachers navigate through the LADs, provided insights into the relationship between self-regulated learning (SRL) activities and emotional states, especially for novice and expert teachers.

Ramaswami et al. [80], developed "SensEnablr," an LAD consisting of descriptive, predictive, and prescriptive learning analytics. It provides detailed feedback comparing a student's performance to his peers, encouraging self-regulated learning practices. Leitner et al. [76], designed a dashboard for tracking study progress, offering official recommendations and comparing individual progress with peers. Han et al. [74], created two separate LADs for students and teachers. Focusing mainly on adaptive feedback during collaborative learning sessions to improve argumentation skills. Eickholt et al. [77], investigated "MyLA," a dashboard emphasizing resource popularity, grades, and due assignments for students. This supports students in prioritizing tasks and improving time management. Hellings et al. [81], focused on first-year students by providing predictions for passing grades, success rates, and progress compared to their peers, improving students' confidence and engagement. Rets et al. [83], collected feedback from students about the most and least useful elements in the LAD created, tailoring the design to student needs.

Zandvliet et al. [72], grouped LAD indicators into action-related, result-related, and social-related categories. These indicators improved student retention and success by providing actionable insights. Han et al. [74], demonstrated how adaptive LADs enhance collaboration and learning during group tasks, improving both engagement and success rates.

Sahin et al. [73], conducted a literature review on LAD visualization techniques and their effectiveness. This study laid the foundation for choosing appropriate visualization methods to increase dashboard usability. Oliver et al. [78], identified student-preferred visualizations and learning indicators, enabling personalized dashboards tailored to specific needs and course profiles.

Many studies applied foundational learning theories to guide LAD design. The popular ones among them are Self Regulated Learning (SRL), Social Cognitive Theory (SCT), Constructivism, Cognitive Load Theory (CLT) etc are explained in detail in the next section.

The creation of intuitive and user-friendly dashboards is crucial for ensuring that both tutors and students can easily interpret and act upon the insights provided by LA. Table 3.2 highlights the growing importance of Learning Analytics in modern education and provides valuable insights into how LA is being applied to enhance teaching and learning experiences. It serves as a foundation for understanding current trends in LA research and offers a road map for future developments in this field. As part of the next section, let use see about the theoretical frameworks used in the LADs.

3.3.1 Theoretical Frameworks used in LADs

The table 3.4 presents a detailed analysis of various theoretical frameworks and their application in Learning Analytics (LA) and Learning Analytics Dashboards (LADs). It highlights key theories such as Self-Regulated Learning (SRL), Social Comparison Theory (SCT), Cognitive Load Theory (CLT), and Feedback Intervention Theory (FIT), each of which plays a significant role in enhancing student learning outcomes. It also outlines the achievements of different studies conducted across various countries, focusing on the stakeholders involved, the use of LA to improve educational processes, and whether a Learning Analytics Dashboard was created as part of the study. This structured overview provides insights into how theoretical frameworks guide the design and implementation of personalized learning interventions.

Each theory addresses specific aspects of learning, offering insights into how students learn, interact, and engage with educational content. These theories are foundational to designing effective learning analytics systems that support both students and instructors in achieving better academic outcomes.

1. Self-Regulated Learning (SRL)

Self-Regulated Learning (SRL) focuses on empowering learners by providing them with performance, progress, and strategy-related information. This information

Table 3.4: Theories used in Learning Analytics

S No.	Theory	Deals With
1	Self Regulated Learning (SRL)	Provides learners with performance, progress, and strategy-related information to help students in self-reflection and learning behavioral modifications [85, 80].
2	Social Comparison Theory (SCT)	Gives motivation to students by making anonymized comparisons with their peers' performance [86, 80].
3	Cognitive Load Theory (CLT)	This concentrates on visualization and measuring of cognitive load learners face. Actionable feedback is provided to students to promote effective learning and prevent information overload, as information overload deteriorates learning. Example: metrics related to the complexity of learning materials [87, 80].
4	Feedback Intervention Theory (FIT)	Supports learners with real-time and personalized feedback in identifying areas of improvement and performance enhancement [88, 80].
5	Constructivism	Active construction of knowledge by learners. Focusses on learners interactions with learning materials, collaborative activities etc [89, 80].
6	Social Interaction and Collaborative Learning	Focusses on social processes in acquiring knowledge. Such dashboards helps in sharing learners thoughts, experiences etc [89, 80].
7	Constructivism to support teachers	Provides teachers with insights into student performance and engagement. Helps teachers in identifying students who need help/support [80].

helps students engage in self-reflection and make behavioral modifications to improve their learning processes. For instance, metrics such as time spent on tasks or progress toward goals can help learners identify areas for improvement and adjust their strategies accordingly. SRL is particularly useful in fostering independent learning habits and encouraging students to take ownership of their academic journey [85, 80].

2. Social Comparison Theory (SCT)

Social Comparison Theory (SCT) emphasizes the importance of motivation through peer comparison. By providing anonymized comparisons of a student's performance with that of their peers, this theory encourages healthy competition and motivation. For example, dashboards can display average class scores alongside individual scores, allowing students to gauge their performance relative to others without feeling judged. This approach fosters a sense of accountability and inspires students to strive for better results [86, 80].

3. Cognitive Load Theory (CLT)
Cognitive Load Theory (CLT) addresses the visualization and measurement of cognitive load experienced by learners. It focuses on actionable feedback that helps students optimize their learning processes by promoting effective learning strategies and avoiding information overload. For example, learning analytics dashboards can provide insights into the complexity of tasks or materials, enabling students to allocate their cognitive resources more efficiently. Metrics related to task difficulty or time spent on complex activities can guide learners in managing their workload effectively [87, 80].

4. Feedback Intervention Theory (FIT) Feedback Intervention Theory (FIT) supports learners by providing real-time and personalized feedback. This feedback helps students identify areas for improvement and enhance their performance. For instance, dashboards can highlight specific weaknesses in a student's understanding of a topic and suggest targeted resources or strategies for improvement. FIT is particularly valuable in adaptive learning environments where timely feedback can significantly impact student engagement and success [90].

5. Constructivism
Constructivism emphasizes the active construction of knowledge by learners through interaction with their environment. In the context of learning analytics, this theory supports tools that encourage collaborative activities such as group discussions, peer reviews, or project-based learning tasks. Dashboards designed with constructivist principles can promote engagement by enabling students to explore concepts interactively and collaboratively [80, 91].

6. Social Interaction and Collaborative Learning
This theory focuses on the social processes involved in acquiring knowledge through collaboration. Learning analytics dashboards designed with this theory in mind facilitate knowledge sharing among peers through features like discussion forums or shared resources. For example, tools that allow students to collaborate on assignments or share insights about course materials foster a sense of community and enhance collective understanding [91].

7. Constructivism Supporting Teachers
In addition to supporting students, constructivist principles also benefit teachers by providing insights into student performance and engagement levels. Dashboards that highlight struggling students or identify patterns in classroom interactions enable instructors to offer timely interventions and personalized support [91].

Significance of These Theories in Learning Analytics
The theories outlined in the table collectively address critical aspects of effective teaching and learning:

- Personalization: By leveraging SRL and FIT principles, learning analytics systems can deliver tailored feedback that meets individual learner needs.
- Engagement: SCT and collaborative learning theories emphasize peer interaction, which fosters motivation and accountability.
- Efficiency: CLT ensures that cognitive resources are used effectively by reducing unnecessary mental strain.
- Collaboration: Constructivist approaches promote active participation and knowledge sharing among both students and teachers.

By integrating these theories into the design of learning analytics dashboards, educational institutions can create tools that not only support academic achievement but also enhance the overall learning experience for both students and instructors.

The table provides a comprehensive overview of how key learning theories are applied in Learning Analytics to improve educational outcomes. By summarizing various studies across different regions, it highlights the effectiveness of LA in supporting both students and tutors. Supports stakeholders with the help of personalized feedback, performance tracking, and predictive insights. The use of Learning Analytics Dashboards further enhances data visualization, and helps tutors to make informed decisions. Overall, the table shows theoretical frameworks used with LA to create adaptive and responsive learning environments.

As educational technologies continue to evolve rapidly post-pandemic era into more personalized digital environments equipped with intelligent systems like Machine Learning algorithms embedded within dashboards—future research will likely focus on refining these systems further while addressing emerging challenges such as data privacy concerns or over-reliance upon automated systems without human oversight.

3.4 Ethical Considerations and Data Privacy

In the domain of Learning Analytics (LA), the collection, processing, and analysis of student data raise significant ethical concerns, particularly regarding data privacy. The "General Data Protection Regulation (GDPR)" provides a framework for ensuring that personal data is handled responsibly and transparently. Below are the key ethical considerations and data privacy measures that have been followed when implementing Learning Analytics systems in compliance with GDPR [92, 93].

Data Collection and Consent: Under GDPR, institutions must obtain explicit consent from students before collecting their data. Following this, students were informed about what data is being collected, the purpose of its collection, and how it will be used. "Transparency" is crucial in ensuring that students understand how their personal information will be processed. Additionally, only the minimum necessary data was collected to comply with GDPR's principle of "data minimization" [94].

Anonymization: To protect student identities, personal data was "anonymized" wherever possible. Anonymization ensures that students cannot be identified from the dataset. This reduces the risk of privacy breaches [95].

Data Access and Security: Access to student data is restricted to authorized personnel only. Implemented robust security measures such as encryption and secure storage to prevent unauthorized access or breaches [96].

Right to Access and Erasure: Students have the right to access their personal data and request its deletion if it is no longer necessary for its original purpose or if they withdraw consent. This aligns with GDPR's principles of "data access" and the "right to be forgotten" [93].

During the development of this tool all the above mentioned measures were followed. This tool is effective and compliant with GDPR regulations

4 Concept

In this chapter, we talk about the characteristics that are considered important and taken from the state of the art papers. Also, we introduce the concept of ARC-Tutoring and explain how it integrates with Learning Analytics (LA) to provide personalized support to students. The ARC-Tutoring framework is designed to assess student performance, offer tailored recommendations, and engage students through conversational agents. The system is data-driven, leveraging Learning Analytics to make informed decisions about when and how to intervene in a student's learning journey.

4.1 Needed Characteristics from the existing LADs

Some of the research papers mentioned in the table 3.2, were very much useful in the development of this ARC Tutoring Workbench. The characteristics taken from those LADs were explained as follows:

I. SensEnablr Dashboard
In the paper [80], the authors developed a Learning Analytics Dashboard (LAD) named SensEnablr. This dashboard was designed to provide real-time feedback to students and instructors by visualizing key performance metrics such as participation, engagement, and academic progress. The goal of SensEnablr was to increase student engagement by offering personalized insights into their learning behaviors and outcomes.

Characteristics taken from SensEnablr:

- Real-Time Data Visualization: SensEnablr provides real-time data on student performance, such as attendance, participation, and test scores [80].
- Engagement Metrics: SensEnablr tracks engagement levels to help identify trends in student behavior [80].
- Personalized Feedback: It offers personalized feedback based on individual student performance, helping students identify areas for improvement [80].
- Instructor Insights: SensEnablr allows instructors to identify at-risk students early by analyzing engagement metrics and performance data [80].

Both SensEnablr and ARC Tutoring Workbench share common goals of improving student engagement through real-time data visualization, personalized feedback, and comprehensive instructor insights. The integration of various data sources in both systems ensures that users—both students and instructors—can access meaningful insights into academic performance.

II. SAVis Dashboard
The SAVis Learning Analytics Dashboard (LAD) had several core functionalities, particularly in its use of interactive visualizations, machine learning techniques, and real-time data analysis. This dashboard aims to enhance the learning experience by providing valuable insights into student performance, engagement, and learning outcomes [75].

Characteristics taken from SAVis:

- Interactive Visualizations: In SAVis, interactive visualizations are used to present complex data in a user-friendly format, allowing both students and instructors to explore performance metrics dynamically [75].
- Machine Learning for Predictive Analytics: SAVis incorporates machine learning algorithms to predict student performance and identify at-risk students based on historical data [75].
- Data Integration: SAVis integrates data from multiple sources to provide a comprehensive view of student progress [75]. SAVis pulls data from various Learning Management Systems (LMS) and other educational tools.
- Real-Time Feedback: SAVis provides real-time feedback on student engagement and performance through its interactive dashboard [75].

SAVis emphasizes the importance of interactive visualizations, predictive analytics using machine learning techniques, real-time feedback, and comprehensive data integration. These features enable this dashboard to provide actionable insights for improving student learning outcomes while offering instructors tools for early intervention [75]. Offering real-time feedback and using machine learning techniques are a pre-requisite for ARC Tutoring Workbench. So this paper is a countable asset in developing this dashboard.

III. Learning Anayltics Cockpit
The Learning Analytics Cockpit aims to provide actionable insights into student performance and engagement, enabling timely interventions by instructors to improve learning outcomes [79].

Characteristics taken from Learning Analytics Cockpit

- Data Visualization: The Learning Analytics Cockpit emphasizes real-time data visualization to monitor student performance. In the Cockpit, instructors can view real-time data on student engagement, participation, and academic progress [79].
- Instructor-Centric Design for Interventions: One of the primary goals of the dashboard is to empower instructors with tools for timely interventions. The Learning Analytics Cockpit is specifically designed to alert instructors when students are at risk of under-performing or disengaging from the course. It provides detailed insights into student behavior, allowing educators to take proactive steps to support struggling students [79].

In summary, the Learning Analytics Cockpit focuses on real-time data visualization, instructor-centric tools for interventions. These features enables it to support proactive educational strategies that enhance student learning outcomes. Enhancing student performance is one of the goals of the ARC Tutoring Workbench. So the above mentioned characteristics are very important in developing ARC Tutoring Workbench.

IV. Learning Analytics Dashboards for Collaborative Argumentation
This research paper focuses on the development of learning analytics dashboards to provide adaptive support during collaborative argumentation in face-to-face learning environments [74]. The ARC Tutoring Workbench, on the other hand, is a comprehensive platform designed to integrate multiple learning analytics functionalities for both students and instructors, offering predictive, diagnostic, and descriptive insights. While both systems aim to enhance learning through data-driven insights, there are key similarities in their goals, functionalities, and implementation.

Characteristics taken from Collaborative Argumentation

- Purpose and Focus: The primary focus of the Collaborative Argumentation system is to provide real-time adaptive support during collaborative learning activities, specifically argumentation. The dashboard helps instructors monitor group dynamics, participation levels, and the quality of arguments being made by students. The system is designed to enhance face-to-face interactions by providing immediate feedback that can guide the learning process [74].
- Real-Time Feedback: It provides real-time feedback during face-to-face interactions. Instructors can intervene immediately based on the data presented in the dashboard, such as participation levels or the quality of student arguments [74]. This real-time aspect is crucial for guiding the collaborative discussions as they happen [74].

- Data Sources and Integration: This system primarily uses data from live discussions and collaborative activities. It tracks metrics such as speaking time, argument quality, and group dynamics in real-time [74].

In conclusion, while both systems aim to improve educational outcomes through learning analytics dashboards, their focus areas differ significantly. The Collaborative Argumentation Dashboard is tailored for real-time intervention during live discussions in collaborative settings, whereas the ARC Tutoring Workbench focuses more broadly on long-term performance tracking and prediction across various assessments. Despite these differences, both systems share a common goal of empowering instructors with actionable insights that can enhance the learning experience for students.

4.2 Concept of LA Integration

The concept involved in the LA integration is depicted through the figure 4.1.

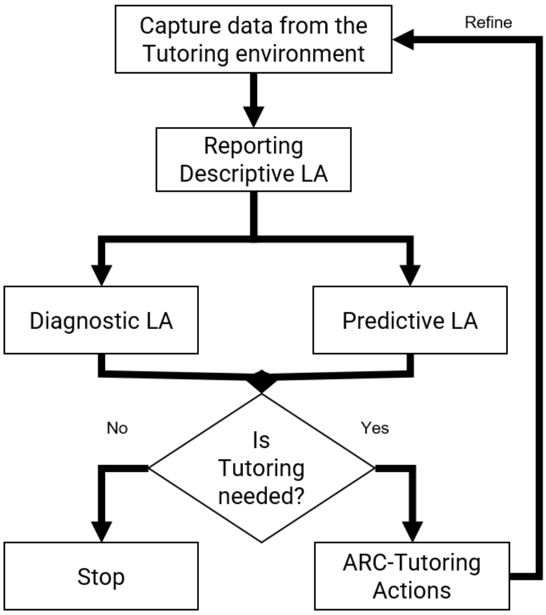

Figure 4.1: Depicting the conceptual flow of data among the LA [4]

1. Capture Data from the Tutoring Environment:
The process begins by collecting data from the tutoring environment. This data can include student performance metrics, engagement levels, attendance, and other relevant information that can help in understanding the learning progress of students.
Data Sources: Learning Management Systems (LMS), assessments, quizzes, and student interactions.

2. Reporting Descriptive Learning Analytics (LA):
Once the data is captured, it is processed using 'Descriptive Learning Analytics'. Descriptive LA focuses on summarizing what has already happened. It provides insights into past events such as student performance trends, engagement patterns, completion rates etc. This step helps in identifying general patterns and understanding the current state of student learning.

3. Diagnostic Learning Analytics (LA):
After descriptive reporting, the process branches into two types of analysis: Diagnostic and Predictive LA. It involves analyzing reasons behind certain learning outcomes, identifying factors that contributed to student success or failure. It uses techniques like data mining and correlation analysis to understand the underlying causes.

4. Predictive Learning Analytics (LA):
The second branch of analysis is "Predictive LA", which answers the question: "What can happen in the future?"
Predictive LA uses statistical models and forecasting techniques to predict future learning outcomes based on historical data. For example, it can predict which students are at risk of under-performing and future performance trends based on current behaviors.

5. Decision Point: Is Tutoring Needed?
After analyzing the data using Diagnostic and Predictive LA, a decision point is reached: "Is tutoring needed?"
This decision is based on whether students are identified as struggling or at risk of falling behind in their learning.

6. If Tutoring is Not Needed:
If the analysis shows that tutoring is not required, the process stops at this point. No further action is taken.

7. If Tutoring is Needed: ARC-Tutoring Actions
If tutoring is determined to be necessary, "ARC-Tutoring Actions" are initiated.ARC-Tutoring refers to personalized interventions designed to support students who need additional help. These actions could include Assessment related assistance, Recommendation of topics based on the score obtained, Assistance through generative chat-bot.

8. Refine Loop:
The flowchart includes a feedback loop labeled "Refine," which indicates that after ARC-Tutoring actions are taken, the system goes back to capturing more data from the tutoring environment. This creates an iterative cycle where data is continuously

collected, analyzed, and used for further interventions if necessary. The refinement loop ensures continuous improvement by adapting tutoring strategies based on updated data.

4.3 Concept of ARC Tutoring Workbench

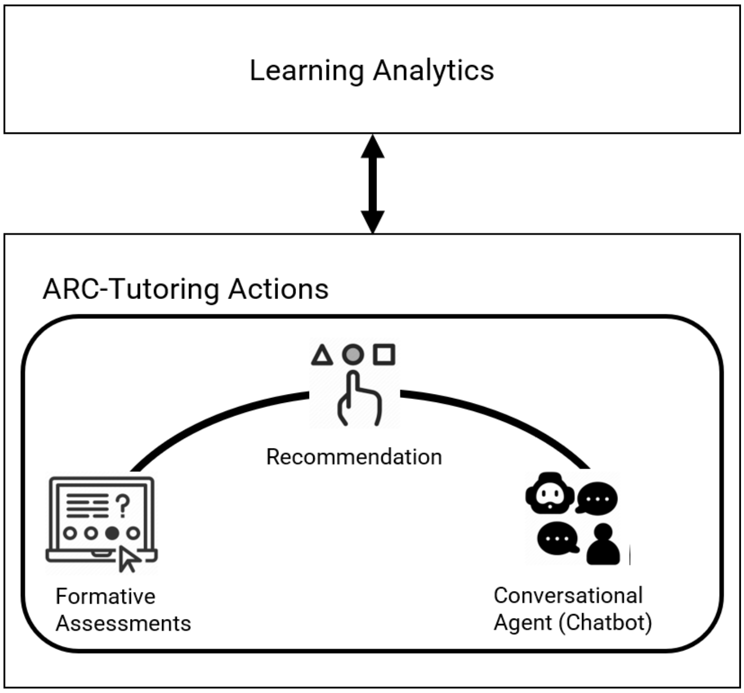

Figure 4.2: Concept of ARC Tutoring Workbench [4]

The ARC-Tutoring Actions are a central part of the decision-making process in the tutoring environment, as depicted in the figure 4.2. ARC stands for Assessment, Recommendation/Reminders, and Conversational Workbench. This system is designed to provide personalized support to students based on Learning Analytics (LA). Below is a detailed explanation of each component:

1. Assessment
The Assessment component focuses on evaluating the student's current performance and progress. This is done through formative assessments, quizzes, and other evaluative tools that provide real-time feedback on the student's learning journey. Formative Assessments assessments are designed to monitor student learning and provide ongoing feedback that can be used by both students and tutors to improve learning outcomes.
- Example: A quiz or assignment that evaluates a student's understanding of a topic and provides immediate feedback on areas where improvement is needed.

The data from these assessments is captured and analyzed through Learning Analytics, which helps in identifying patterns in student performance.

2. Recommendation/Reminders Based on the assessment results and Learning Analytics data, the system generates Recommendations or Reminders for students.

- **Recommendations:** These are personalized suggestions aimed at helping students find a topic that better suits their interests.
  - Example: If a student is struggling to choose a particular topic, the system may recommend top five topics from the list of available topics based on the scores obtained in individual topics.

- **Reminders:** These are timely notifications that help students stay on track with their learning goals. They can include reminders about upcoming assignments, deadlines, or study goals.
  - Example: A reminder to complete an assignment before its due date or to review certain topics before an exam.

3. Conversational Agent/(Chatbot)
This Workbench integrates a conversational agent (e.g., a chat-bot) that interacts with students in real-time to provide additional support. The chat-bot acts as an intelligent tutor that can answer student queries, provide explanations, and guide learners through their study process.
- Example: A student struggling with a concept can ask the chat-bot for clarification, and the chat-bot will respond with explanations or direct them to relevant resources.
This conversational agent can also engage in dialogue with students to keep them motivated and focused on their learning objectives. It enables semi-automated tutoring support by providing responses to student inquiries.

How ARC-Tutoring Actions Work in the Flowchart
1.Learning Analytics Input:
The flowchart begins with capturing data from the tutoring environment through Learning Analytics. This data includes student performance metrics, engagement levels, and other indicators of student progress.

2. ARC-Tutoring Actions:
If it is determined that tutoring is needed (based on diagnostic and predictive analytics), ARC-Tutoring Actions are initiated. These actions consist of three main components:

- Formative Assessments: Continuous evaluation of student performance.
- Recommendations/Reminders: Personalized suggestions and timely reminders to keep students on track.

- Conversational Support via Chat-bot: Real-time interaction with students through a conversational agent that provides guidance and answers questions.

3. Refinement Loop:
After ARC-Tutoring Actions are taken, there is a feedback loop labeled "Refine." This loop ensures continuous improvement by feeding updated data back into the Learning Analytics system for further analysis. Based on this new data, additional ARC-Tutoring Actions may be triggered if necessary.

The ARC-Tutoring system leverages Learning Analytics to offer personalized tutoring support through three key components: Assessment, Recommendation/Reminders, and Conversational Workbench (Chat-bot). By continuously assessing student performance, providing tailored recommendations, and offering real-time conversational support, ARC-Tutoring Workbench aims to enhance student success in their learning journey.

4.4 Use Case

These use cases demonstrate how data-driven decisions can be made to provide personalized interventions to students based on their academic needs. The possible use cases are depicted in the figure 4.3.

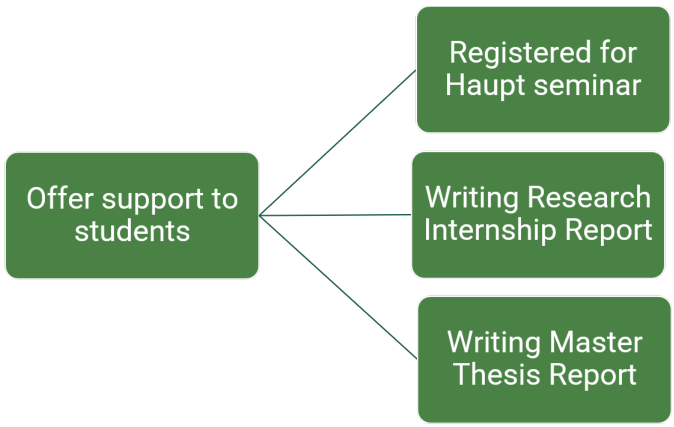

Figure 4.3: Possible Use cases

This ARC Tutoring Workbench tool is mainly useful for

1. **Students registered for the Hauptseminar subject**
   Students enrolled in advanced seminars (Haupt Seminar) often face challenges in managing complex topics and workloads. Learning Analytics can track their progress and provide insights into areas where they may need additional support.

2. **Students writing their report as part of their Research Internship**
   Students working on research internships may require guidance on structuring their reports, managing deadlines, or receiving feedback on drafts. Learning Analytics can help identify students who are struggling with report writing and provide targeted assistance.

3. **Students writing their report as part of their Master Thesis**
   For students writing their master's thesis, personalized support could include feedback on drafts, recommendations for resources, or guidance on research methodologies. Learning Analytics can monitor their progress and ensure they stay on track with their thesis deadlines.

This chapter highlighted the main research papers considered in the development of ARC Tutoring Workbench from the research papers mentioned in the table 3.2. The characteristics taken from those research papers were explained in detail. This chapter also provided a comprehensive overview of the key concepts behind the integration of Learning Analytics (LA) and the development of the ARC Tutoring Workbench. The Concept of LA Integration highlights how data-driven insights can enhance personalized learning experiences by tracking student performance and providing timely interventions. The ARC Tutoring Workbench concept demonstrates how various tools, such as assessments, recommendations, and conversational agents, work together to support both students and educators. Finally, the Use Case section illustrates practical applications of these concepts, showcasing how they improve learning outcomes in real-world scenarios.

5 Implementation

Implementation chapter gives information on how the proposed dashboard is realized in practice. This chapter serves as a bridge between the theoretical concepts outlined in earlier sections and their practical application. It also explains the procedure followed in integrating the existing modules into a single dashboard. The aim is to translate design principles, methodologies, and conceptual frameworks into a working solution that addresses the research objectives.

Here the step-by-step process of implementing the system, including the tools, technologies, and platforms used is discussed. Also, the system architecture, database design, user interface development will be explained. This chapter also demonstrates how theoretical concepts are used to create a functional system that meets the needs of the end users.

5.1 System Architecture

Diagram 5.1 represents a three-layered system architecture designed to support Learning Analytics (LA) and personalized tutoring actions. The system is divided into three main layers: Presentation Layer, Application Layer, and Data Layer. Each layer has specific components that interact to provide a seamless user experience while leveraging Learning Analytics to support decision-making.

1. Presentation Layer
The Presentation Layer is responsible for the user interface(UI) and interaction with the system. It is concerned mainly about the display of data and visualizations to users such as students, tutors, or administrators. Key tasks of this layer are it sends HTTP requests to the Application Layer to fetch data. Then receives responses in JSON format, which are then processed and displayed visually using charts and graphs. It also ensures that users can easily interact with the system and access relevant information through a user-friendly interface.

2. Application Layer
The Application Layer is the core processing unit of the system. It manages user authentication, processes Learning Analytics data, and provides personalized support through various tools. The green color rectangle in the figure 5.1 shows the modules that were part of the ARC Tutoring Workbench.

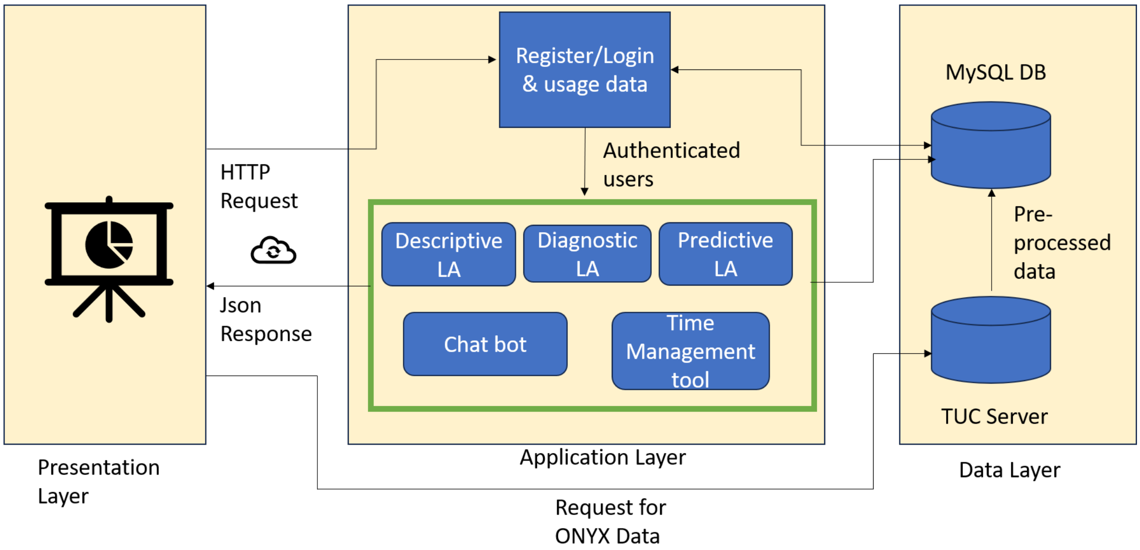

Figure 5.1: Proposed System Architecture

Key Components:

- Register/Login: This component handles user authentication and tracks user activity within the system. It ensures that only authenticated users can access the system's features.
- Descriptive Learning Analytics (LA): This component provides insights into past events by summarizing student performance, engagement levels, and other historical data.
- Diagnostic Learning Analytics (LA): This component analyzes patterns in student behavior and performance to identify reasons behind specific learning outcomes (e.g., why a student is under-performing).
- Predictive Learning Analytics (LA): This component forecasts future learning outcomes based on historical data, helping tutors identify at-risk students or predict future trends in student performance.
- Chat-bot: A conversational agent that interacts with users in real-time, answering queries, providing guidance, and offering personalized feedback based on Learning Analytics data.
- Time Management Tool: This tool helps students manage their time effectively by providing reminders, tracking deadlines, and suggesting optimal study schedules based on their learning patterns.

The Application Layer handles requests received from the Presentation Layer and communicates with the Data Layer to retrieve or store data.

3. Data Layer
The Data Layer is responsible for managing all data storage and retrieval operations. It ensures that data is securely stored and efficiently accessed when needed by the Application Layer.

Key Components involved are the TUC server and the MySQL database. The TUC server has student performance related data that can be accessed by the Application Layer for analysis or reporting. For the sake of backup when the server is down, this student performance data is stored in the excel sheets and used as input source. Extract, Transform, Load (ETL) operations are performed on the data obtained from the input. First data is extracted from the source. It is then cleansed and transformed into the required format. Finally, it was stored in the database required for the elements of the dashboard.

The Application Layer communicates with the Data Layer to fetch or store data. Requests for ONYX Data (ONYX is a test-suite and it has students performance related data) are processed here before being sent back to the Application Layer.

Flow of data in the architecture diagram 5.1
1. Users interact with the system via the Presentation Layer by sending HTTP requests.
2. These requests are processed by the Application Layer, where various Learning Analytics tools (Descriptive LA, Diagnostic LA, Predictive LA) analyze the data.
3. The Application Layer retrieves necessary data from the MySQL Database or TUC Server in the Data Layer.
4. After processing, results are sent back to the Presentation Layer as JSON responses, which are then displayed visually to the users.

This architecture efficiently integrates Learning Analytics tools with user interaction components such as chatbots and time management tools. By dividing responsibilities across three layers(Presentation, Application, and Data layers) the system ensures scalability, security, and ease of use while providing valuable insights into student performance and engagement.

5.2 Data Preparation

Data from the TUC Server is provided as input. Can connect to this server only through VPN when outside the university network. When the server is down, for the sake of backup, this data is stored in the excel file. Different sheets for different types of data. This input data is masked to maintain data privacy. This input data needs to be cleansed. USed python as programming language in Visual Studio code IDE to pre-process the given input data. Anomalies such as null values, unnecessary characters were removed and then stored in the MySQL database. This database server is a computer within the campus network that is accessible to all authorized users with their application. In this way, Extract, Transform and Load operations were carried-on on the given input data.

The data preparation process is a critical step in ensuring that the input data is clean, accurate, and ready for analysis. In this project, data from the TUC Server serves as the primary input source. The TUC Server contains various types of data essential for Learning Analytics (LA) and is accessible only through a secure VPN connection when outside the university network. To ensure continuous access to the data, especially when the server is down or unavailable, a backup system has been implemented using Excel files. These files are structured with different sheets for different types of data, providing a well-organized fallback solution.

Data Privacy and Masking

Given the sensitive nature of the input data, privacy is a top priority. The input data is masked to maintain compliance with data privacy regulations such as the General Data Protection Regulation (GDPR) [97]. Masking ensures that personally identifiable information (PII) is protected throughout the data handling process. The same can be seen in the below screenshot of the masked input data 5.2.

Student_I	Matriculation_Numbe	Last_Nam	First_Nam	Email	UID
1	221958	LN1	SN1	LN1-SN1@tuchemnitz.de	ZCoQh8uM5swkkx0JNxcv0bxRDLb7uG
2	284779	LN2	SN2	LN2-SN2@tuchemnitz.de	5v8RyWA6PB7po99U9YR2Z4cKYguiMr
3	765987	LN3	SN3	LN3-SN3@tuchemnitz.de	y7nX5iz0btBU7gR8hUInqJXNdb9f1P
4	415139	LN4	SN4	LN4-SN4@tuchemnitz.de	1CrzRHj9JnEVohnw749NupSrU0xzy7
5	532315	LN5	SN5	LN5-SN5@tuchemnitz.de	7SOCoSaygixaRhxkYqhIIG6ePM5OBg
6	202433	LN6	SN6	LN6-SN6@tuchemnitz.de	hlHGV2w9xKIRNvAIaIKUncaez2nMAi
7	318126	LN7	SN7	LN7-SN7@tuchemnitz.de	zP7mYFMWZ8FdDKwMSoQCJmFg6YvBbP
8	164044	LN8	SN8	LN8-SN8@tuchemnitz.de	940fBBRRtD9k52ByM9Ga4A94ZmOcMf
9	121959	LN9	SN9	LN9-SN9@tuchemnitz.de	Ai9KGYPRxo61FFxOZW9W5Zl9MbcWKW
10	881239	LN10	SN10	LN10-SN10@tuchemnitz.de	q6WbbB1wKFGas9b0RzFfF2d2k36qLx
11	468452	LN11	SN11	LN11-SN11@tuchemnitz.de	ZHfvkVpGi76fpCct76JszcstFgZOGN
12	171295	LN12	SN12	LN12-SN12@tuchemnitz.de	MrNakB4yXwEDPlg7pzV4Z7WbaV9leK
13	391999	LN13	SN13	LN13-SN13@tuchemnitz.de	62iOIsVpctx13GxZkWhJLNt9IV9yIy
14	867836	LN14	SN14	LN14-SN14@tuchemnitz.de	rTrb1Ip8OJGdGnuVt3hgcqGVl6Yv2v
15	500109	LN15	SN15	LN15-SN15@tuchemnitz.de	DLLSY1hAAHuDGBUGe78VsdIWqRBDCT
16	398356	LN16	SN16	LN16-SN16@tuchemnitz.de	fI7OKxCWT0EI7GZ8uFwGcryPE15cNT
17	202425	LN17	SN17	LN17-SN17@tuchemnitz.de	XFUvwbAPbzqNGiQ1VMCP2zIXyxm75g
18	320984	LN18	SN18	LN18-SN18@tuchemnitz.de	JStahTpnlYwoBHbFwvYy598v55vWZP
19	489957	LN19	SN19	LN19-SN19@tuchemnitz.de	o1Q7KGh07RReMdfSFZDUI2NZpmX7PD
20	322866	LN20	SN20	LN20-SN20@tuchemnitz.de	q3sB52zKz09wil0a55aUHF91VyNS0a
21	234415	LN21	SN21	LN21-SN21@tuchemnitz.de	8MevC2clzpYKv4CnBeUWDT9ZelpzzV

Figure 5.2: Screenshot showing example masked input data

Data Cleansing and Pre-processing

The raw input data from both the TUC Server and Excel backups often contains anomalies such as null values, unnecessary characters, and other inconsistencies that could affect subsequent analysis [98]. To address these issues, a comprehensive data cleansing process was carried out using Python as the programming language within the Visual Studio Code IDE. Python's robust libraries such as Pandas and NumPy were employed to handle missing values, remove unwanted characters, and standardize formats across different datasets.

The key steps in the pre-processing phase included:

- Handling Missing Values: Null values were identified and either filled with appropriate default values or removed, depending on their impact on the dataset [98].
- Removing Unnecessary Characters: Any extraneous characters or symbols that could interfere with data analysis were stripped from the dataset.
- Data Standardization: Formats across different datasets were standardized to ensure consistency in further processing stages.

Once cleaned, the preprocessed data was stored in a MySQL database, which serves as a centralized repository for all subsequent analyses [99].

Excel vs MySQL Database
The table 5.2, provides a comparison between Excel and MySQL databases based on several key factors, highlighting their respective advantages and disadvantages.

Table 5.1: Comparison of Excel and MySQL

Criteria	**Excel**	**MySQL**
Data Volume Handling	Well-suited for small to medium datasets (up to 1 million rows per sheet). Performance decreases as the datasets grow [100].	Optimized for large datasets, capable of managing millions of rows efficiently. Ideal for large-scale applications without significant performance loss [101].
Data Integrity and Security	Basic password protection is available but lacks advanced security features like encryption or user-level access control. Prone to data corruption when used in multi-user environments [101].	Provides robust security features such as encryption, user authentication, and role-based access control. Ensures data integrity in multi-user environments [102].

Collaboration and Multi-User Access	Limited real-time collaboration support. Cloud-based versions like Excel Online offer basic collaboration features but are limited in simultaneous editing [101].	Supports real-time multi-user access with strong concurrency controls. Suitable for collaborative environments where multiple users require real-time data access [102].
Querying Capabilities	Basic querying options available through built-in functions (e.g., VLOOKUP, Pivot-tables). Not designed for complex queries involving multiple datasets or tables [102].	Supports advanced querying using SQL (Structured Query Language). Efficient indexing and complex queries across multiple tables are supported [102].
Automation and Integration	Offers basic automation through macros (VBA), but scalability is limited. Requires third-party tools for more advanced integration with other systems [100].	Provides advanced automation options through stored procedures and triggers. Seamlessly integrates with programming languages (Python, Java) and analytics platforms [102].

Database Setup and Accessibility
The MySQL database server is hosted on a computer within the university campus network, ensuring secure access for all authorized users [99]. Access to this database is restricted to users who have been granted permission through their application credentials. This setup guarantees that only authorized personnel can interact with the stored data while maintaining security protocols.

Extract, Transform, Load (ETL) Process
The entire workflow follows an Extract, Transform, Load (ETL) process. First the data is extracted from either the TUC Server or Excel backup files. As part of the Transform step, the extracted data undergoes transformation through cleansing operations such as handling missing values, removing unnecessary characters, and standardizing formats. Finally in the Load step, the transformed data is loaded into the MySQL database for further analysis [103].

This ETL process ensures that all input data is consistently prepared and stored in a structured format that can be easily accessed for subsequent reporting and analysis.

Further Analysis and Reporting
Once stored in the MySQL database, the cleaned and structured data becomes available for further analysis using Learning Analytics tools. This includes generating reports on student performance, engagement levels, and other key metrics that support decision-making processes within educational environments [98]. By implementing this rigorous data preparation process, we ensure that all input data is reliable, secure, and ready for meaningful analysis.

5.3 Technologies and Tools

The Technologies and Tools section provides a comprehensive overview of the key technologies used in the development of the project. This section covers various tools and programming languages that were essential for building the system, including the IDE used for development.

5.3.1 IDE Used

In this project, Visual Studio Code (VS Code) was utilized as the Integrated Development Environment (IDE) for developing and running the application. VS Code, developed by Microsoft, is a lightweight yet powerful IDE that supports a wide range of programming languages and frameworks, making it highly versatile for modern software development. Its modular architecture allows developers to extend its functionality through various plugins and extensions, which significantly enhance productivity [104].

VS Code provides features such as syntax highlighting, intelligent code completion, debugging tools, and version control integration (e.g., Git), all of which streamline the development process. The IDE also supports multiple programming languages, including Python, JavaScript, and C++, making it an ideal choice for multi-language projects. Moreover, VS Code's seamless integration with cloud platforms like Azure enables developers to deploy applications directly from the development environment, further accelerating the deployment process [105].

One of the key reasons for choosing VS Code in this project was its strong support for Python development. With extensions like Python for VS Code, developers can easily manage virtual environments, run scripts, and debug Python code efficiently. Additionally, its built-in terminal allows developers to execute commands without leaving the IDE, improving workflow efficiency [106].

In summary, Visual Studio Code was selected due to its rich feature set, ease of use, and extensive plugin ecosystem that supports a wide variety of programming languages and technologies. Its ability to integrate with cloud services and version control systems makes it an indispensable tool for modern software development.

5.3.2 Python as Programming Language

In this project, Python was selected as the primary programming language due to its robust ecosystem of libraries that facilitate efficient data handling and processing. Several Python libraries were employed to streamline the data preparation process. For instance, Pandas and NumPy were used for data manipulation and numerical computations, respectively. These libraries provide powerful tools to handle large datasets, perform complex calculations, and ensure efficient data transformation [107].

The "pathlib" library was utilized to navigate through file directories and access input files stored in different locations. This library offers an object-oriented approach to file system paths, making it easier to manage file operations across different platforms [108]. Additionally, the "re" library was used for regular expression operations, which helped identify and extract patterns from the input data. This was particularly useful for cleaning and validating the data by removing unwanted characters or formatting inconsistencies.

For building the web interface and handling HTTP requests, "Flask", a lightweight web framework, was employed. Flask is known for its simplicity and flexibility, making it ideal for developing small to medium-sized applications that require minimal overhead [109]. Furthermore, other libraries such as "Seaborn" were used for data visualization, providing high-level interfaces for creating informative statistical graphics [110].

The integration of these libraries into the development environment was facilitated by Visual Studio Code (VS Code), a powerful Integrated Development Environment (IDE) that supports Python development through various extensions. VS Code's built-in terminal allowed seamless execution of Python scripts, while its debugging tools helped identify and resolve issues during the development process [105].

5.3.3 Flask

Flask is a lightweight and flexible web framework for Python, designed to build web applications quickly and efficiently. Despite its simplicity, Flask is highly versatile and can be extended with various plugins and libraries to meet the needs of complex applications [111].

Key Features of Flask:

1. Minimalistic Design:

Flask follows a minimalist approach, providing only the essential tools required to build web applications, such as routing, request handling, and templating. This design philosophy allows developers to choose specific components and libraries based on their project requirements, without being forced into a particular development structure. The flexibility provided by Flask is one of its biggest advantages, as it allows developers to build applications that are tailored to their needs without unnecessary overhead [112].

2. Routing System:
One of the core features of Flask is its simple routing mechanism. Developers can define URL routes using Python decorators, which link specific URLs to corresponding functions in the application. This makes it easy to handle different HTTP methods (GET, POST, PUT, DELETE) and create dynamic web pages that respond to user input [112].

3. Templating System:
Flask uses Jinja2, a powerful templating engine that allows developers to embed Python code within HTML files. This feature makes it easier to separate the presentation layer from the application logic, leading to cleaner and more maintainable codebases. Jinja2 also supports template inheritance, which helps in creating reusable templates across different pages of an application [112].

4. Extensibility:
Although Flask is minimal out-of-the-box, it is highly extensible through third-party libraries and plugins. Developers can easily add functionality such as database integration (e.g., SQLAlchemy), authentication systems (e.g., Flask-Login), or form handling (e.g., WTForms). This modularity allows developers to pick and choose the tools they need while keeping the application lightweight [112].

5. Built-in Development Server:
Flask comes with a built-in development server that allows developers to test their applications locally during development. The server automatically reloads when code changes are detected, making the development process more efficient by eliminating the need for manual restarts [112].

6. Error Handling and Debugging:
Flask provides robust error handling mechanisms and includes an interactive debugger that helps developers identify and fix issues quickly. When an error occurs during development, Flask displays detailed error messages along with a stack trace, making it easier to pinpoint the source of the problem [111, 112].

7. RESTful API Support:
Flask is well-suited for building RESTful APIs due to its simplicity and flexibility in handling HTTP requests and responses. Developers can easily create APIs that serve JSON data or interact with external services using Flask's request handling capabilities [112].

Use Cases of Flask
Flask is widely used in various domains due to its adaptability and ease of use. Due to its minimal setup requirements, Flask is often used for prototyping web applications or APIs before scaling them into larger projects. Its lightweight nature

makes Flask ideal for building micro-services that perform specific tasks within larger systems [112].

With support for numerous data science libraries like Pandas and NumPy, Flask is often used in data-driven applications that require integration with machine learning models or data visualization tools. The framework's simplicity makes it a popular choice for educational purposes where students are learning web development concepts [112].

In summary, Flask provides a flexible yet powerful platform for developing web applications in Python. Its minimalist design philosophy gives developers complete control over their projects while offering extensive customization through third-party extensions. Whether building simple websites or complex APIs, Flask's ease of use combined with its scalability makes it a go-to choice for many developers [112].

5.3.4 Blueprints

Flask Blueprints are an essential feature of the Flask framework that allow developers to structure and organize their web applications in a modular way. Blueprints enable the creation of reusable components, which can be integrated into a larger application without duplicating code. This modular approach is particularly useful when building complex applications, such as dashboards that require the integration of multiple functionalities or microservices into a single cohesive interface [111].

In the context of integrating different applications into one dashboard, Flask Blueprints provide a scalable solution. Each application or module can be developed independently as a blueprint, with its own routes, templates, static files, and logic. Once these blueprints are created, they can be registered to the main Flask application, allowing all modules to coexist within a single dashboard. This approach not only promotes code re-usability but also simplifies maintenance and updates, as individual components can be modified without affecting the entire system [111, 112].

For example, in a dashboard that integrates multiple data sources or services (such as user management, data visualization, and reporting), each service can be encapsulated within its own blueprint. The user management module might handle authentication and user roles, while the data visualization module could focus on rendering charts and graphs using libraries like Seaborn or Matplotlib. By separating these concerns into blueprints, developers can maintain clear boundaries between different parts of the application while still allowing them to interact seamlessly within the dashboard [113, 112].

Flask Blueprints also support URL prefixing, which allows each module to have its own distinct URL path. This feature is particularly useful when integrating multiple applications into one dashboard because it ensures that routes from different blueprints do not conflict with each other. For instance, the user management module could have routes prefixed with '/auth', while the data visualization module could have routes prefixed with '/charts'. This makes it easier to manage routing and ensures that each component remains isolated from others [111, 112].

Another advantage of using Flask Blueprints is that they facilitate team collaboration. In larger projects where multiple developers are working on different parts of an application, each team member can focus on developing a specific blueprint without worrying about conflicts in other parts of the system. Once all blueprints are complete, they can be easily integrated into the main application by registering them with the Flask app object [112]. As a summary, Flask Blueprints provide a powerful mechanism for integrating multiple applications or services into a single dashboard. By promoting modularity, reusability, and maintainability, they simplify the development process and ensure that complex applications remain scalable and easy to manage.

5.3.5 Virtual Environments

In Python, virtual environments provide a way to create isolated spaces for managing project-specific dependencies. This allows developers to install and use packages and libraries unique to each project, without affecting the global Python environment or other projects. By using virtual environments, developers can ensure that the package versions required for one project remain separate from those used in another. This is especially beneficial when working on multiple projects with differing dependency requirements [114].

The venv module, included in Python's standard library, is commonly used to create virtual environments. It allows developers to install specific versions of packages required for a project without affecting the system-wide Python installation. For instance, a project might require Flask 1.1.2, while another project might need Flask 2.0.0—virtual environments make it possible to use both versions simultaneously in separate environments [114].

Once a virtual environment is activated, any package installed using pip (Python's package installer) will be isolated to that environment. This setup provides flexibility and control over package management and ensures that projects remain stable and reproducible over time [115].

Virtual environments are especially useful in collaborative settings where different team members may be using different operating systems or package versions. By sharing the configuration file (e.g., requirements.txt), all team members can recreate the same environment on their local machines [114].

5.3.6 Node Package Manager (NPM)

The Node Package Manager (NPM) is an essential tool within the JavaScript ecosystem, specifically designed for managing and distributing packages in Node.js applications. As the default package manager for Node.js, NPM enables developers to efficiently install, update, and handle the dependencies needed for their projects. It plays a vital role in modern web development by offering access to an extensive library of reusable code modules, which can be seamlessly integrated into applications. This not only speeds up the development process but also enhances code quality [116].

When used alongside Python frameworks like Flask, NPM helps in managing front-end dependencies such as JavaScript libraries and CSS frameworks that complement back-end services built using Flask. For instance, in a Flask-based web application, developers may use NPM to install front-end libraries like React, Vue.js, or Bootstrap to enhance user interfaces. This integration allows Flask to handle the back-end logic, while NPM manages the client-side dependencies, ensuring a seamless full-stack development experience [117].

One of the primary advantages of NPM is its ability to manage version control for packages, ensuring that projects are not disrupted by updates or changes in external libraries. Developers can specify exact versions of packages in the package.json file, which NPM uses to install the correct versions during deployment or when setting up new environments. This feature is crucial when working on applications that require stability across different environments [118].

Additionally, NPM's modular approach promotes code reuse and maintainability by allowing developers to break down complex applications into smaller, manageable components. These components can be reused across various projects or shared with the broader developer community through the NPM registry. This modularity is particularly beneficial when building scalable web applications that need to integrate multiple services or microservices [119].

In conclusion, NPM is a vital tool for managing front-end dependencies alongside back-end frameworks such as Flask. Its capabilities in handling package management, version control, and modularity make it a cornerstone of contemporary full-stack development workflows [120].

5.3.7 React JS

ReactJS is a popular JavaScript library used for building dynamic user interfaces, particularly in single-page applications (SPAs). In this project, ReactJS was utilized for the frontend, while Flask and Python managed the backend. React's component-based architecture allows developers to create reusable UI components, making the code more modular and maintainable. This approach enhances scalability, especially in large applications [112].

One of React's key features is its virtual DOM, which updates only the parts of the UI that change, leading to faster rendering and a smoother user experience [121]. This makes ReactJS ideal for applications requiring frequent updates or real-time data.

On the back-end, Flask was used to build RESTful APIs that interact with Python logic. These APIs are consumed by ReactJS using libraries like Axios or JavaScript's fetch API, allowing dynamic updates based on user interactions. Flask's lightweight nature complements React's flexibility, creating a seamless full-stack development experience [112].Combining ReactJS for front-end development with Flask and Python for back-end services provides a scalable and efficient solution for modern web applications [122].

This section outlines the critical technologies and tools that were integral to the successful development of the project. By leveraging Python and Flask for back-end development, along with ReactJS and NPM for front-end management, the system was built to be scalable and maintainable. The use of blueprints ensured modularity, while virtual environments helped manage dependencies efficiently. Together, these technologies contributed to a robust and flexible architecture that supports both back-end functionality and a seamless user experience.

5.4 User Interface Design

Data Exchange Between Available Components in the Front-End
In this project, ReactJS was used for the front-end development, while Flask and Python handled the back-end. ReactJS, being a component-based library, allows efficient data exchange between components using props and state. Props are used to pass data from parent components to child components, while state manages local data within components. This modular architecture ensures that data flows seamlessly between different parts of the application [123].

For communication with the back-end, Axios or JavaScript's native fetch API was used to send HTTP requests to Flask's RESTful APIs. Flask processes these requests on the server-side and returns the necessary data, which is then rendered dynamically by ReactJS components. This integration allows for real-time updates and smooth interaction between the front-end and back-end systems [123].

Applying User Experience (UX) Principles for an Intuitive Interface
Several key User Experience (UX) principles were applied to ensure that the interface is intuitive and user-friendly:

- Simplicity: The interface design follows a minimalistic approach, ensuring that users can navigate effortlessly without being overwhelmed by unnecessary elements. Clear navigation menus and call-to-action buttons guide users through the application efficiently.
- Consistency: Consistent use of colors, typography, and layout across different sections of the application provides a cohesive experience. This consistency helps users predict how elements will behave, reducing cognitive load.
- Responsiveness: The interface was designed to be responsive across different devices, ensuring optimal usability on desktops, tablets, and mobile phones.
- Visual Hierarchy: Important actions such as submitting forms or saving settings are highlighted using bold colors and larger buttons to ensure users can easily identify key actions [124].
- Feedback Mechanisms: Feedback mechanisms such as loading indicators and success/error messages were integrated into the interface to provide real-time feedback on user actions (e.g., form submissions or data loading).

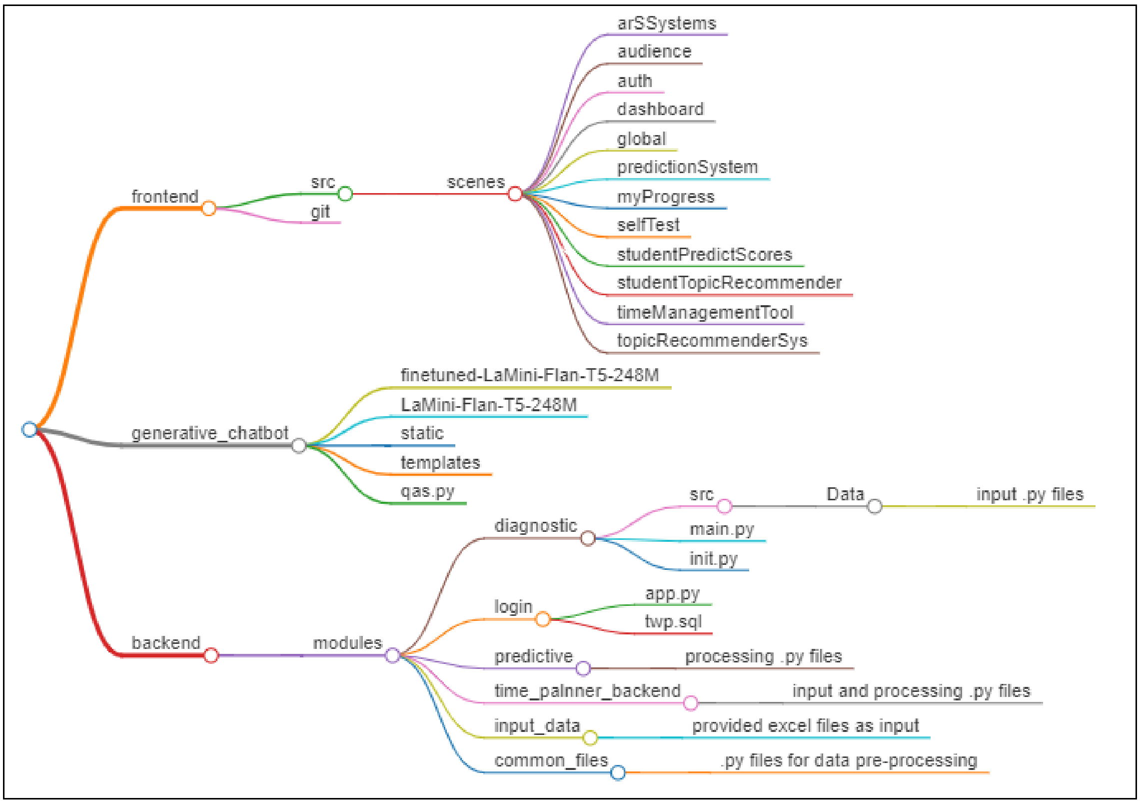

Figure 5.3: Folder structure of the code files of the integrated components

By applying these UX principles alongside robust front-end technologies like ReactJS and NPM, we were able to create an interface that is not only functional but also intuitive for users.

5.5 Learning Analytics Integration

As part of my master's thesis, I focused on the development and integration of various modules into a unified system known as the ARC Tutoring Workbench. This workbench serves as a comprehensive platform for both students and instructors, providing real-time insights into student performance and engagement through various Learning Analytics (LA) functionalities. The key modules developed include Descriptive LA, Diagnostic LA and the Predictive LA.

Each of these components was developed as an individual module to ensure modularity, scalability, and ease of integration. The modular approach allows each component to function independently while still contributing to the overall functionality of the ARC Tutoring Workbench. Example folder structure of the code files is shown in the figure 5.3.

Majority of the modules are developed using Flask and Python for the backend, and ReactJS and NPM for the frontend. These technologies were chosen due to their flexibility, ease of use, and compatibility with modern web development practices.

The only exception was the Predictive LA module, which was initially developed using C# and MVC .Net technologies. However, due to incompatibility with Flask's Blueprint architecture—used across all other modules—I re-implemented this module using Python and Flask to ensure consistency across the entire system.

Predictive Learning Analytics (Re-Implementation)

The input data for the Predictive LA module was provided in Excel sheets. This data underwent pre-processing using Python libraries such as Pandas, after which it was stored in a MySQL database. The relevant database tables include Student, Semester, Student_Test_Data, Formative_Data, Topic, and Student_Topic. These tables store essential student data such as test scores, semester details, and topics selected by students. Below, I will outline the detailed implementation procedure, including the mathematical models and formulas used to predict student scores for presentations and reports.

Multiple Linear Regression (MLR) Model

The Multiple Linear Regression (MLR) model is used to predict a dependent variable (e.g., student performance) based on multiple independent variables (e.g., test scores). The general formula for MLR is:

$$y = a + b_1x_1 + b_2x_2 + b_3x_3 + ...b_nx_n + e \tag{5.1}$$

where,

- y is the predicted score,
- a is the intercept,
- $b_1, b_2, ...b_n$ are the coefficients for independent variables $x_1, x_2, ...x_n$,
- e is the mean square error.

In this context, the independent variables used for prediction are ARS scores, Self-Test scores, and Topic Recommender scores. Each time a student completes an ARS test or Self-Test, their scores are stored in the database for future use in predictions.

In this study, predictions are done using two models: one for calculating presentation score prediction, another for report score prediction. The presentation score prediction and the report score prediction are used to predict the overall score. These ARS test and Self-test scores are used in calculating the presentation score, report score, and overall score.

Presentation Prediction Model:

The formula for predicting a student's presentation score (y_{ppt}) is:

$$\begin{aligned} y_{ppt} = (b_1 * x_{ars\_ppt}) + (b_2 * x_{st\_ppt}) + \\ (b_3 * x_{tr}) + e \end{aligned} \tag{5.2}$$

Where:
- x_{ars_ppt} is the normalized ARS presentation score,
- $x_{self\_test\_ppt}$ is the normalized self-test presentation score,
- x_{tr} is the normalized Topic Recommender score

The independent variables are normalized by calculating percentages based on achieved and maximum possible scores. In the equation 5.2, the terms x_{ars_ppt}, x_{st_ppt} and x_{tr} are calculated using the following formulae:

$$x_{ars\_ppt} = \frac{x_{ars\_ppt\_achieved\_score}}{x_{ars\_ppt\_max\_score}} * 100 \tag{5.3}$$

$$x_{st\_ppt} = \frac{x_{st\_ppt\_achieved\_score}}{x_{st\_ppt\_max\_score}} * 100 \tag{5.4}$$

$$x_{tr} = \frac{x_{topic\_recommender\_achieved\_score}}{x_{topic\_recommender\_max\_score}} * 100 \tag{5.5}$$

$$\begin{aligned} x_{ars\_ppt\_achieved\_score} = (ars\_search\_achieved\_score+ \\ ars\_discussion\_achieved\_score + ars\_ppt\_achieved\_score) \end{aligned} \tag{5.6}$$

$$\begin{aligned} x_{ars\_ppt\_max\_score} = (ars\_search\_max\_score+ \\ ars\_discussion\_max\_score + ars\_ppt\_max\_score) \end{aligned} \tag{5.7}$$

$$\begin{aligned} x_{st\_ppt\_achieved\_score} = (st\_search\_achieved\_score+ \\ st\_discussion\_achieved\_score + st\_ppt\_achieved\_score) \end{aligned} \tag{5.8}$$

$$\begin{aligned} x_{st\_ppt\_max\_score} = (st\_search\_max\_score+ \\ st\_discussion\_max\_score + st\_ppt\_max\_score) \end{aligned} \tag{5.9}$$

$x_{ars\_ppt\_achieved\_score}$ is the points obtained in the ars_search, ars_discussion and the ars_presentation modules. Where as the $x_{ars\_ppt\_max\_score}$ is the sum of the maximum scores possible in the modules search, discussion and presentation. x_{ars_ppt} is the variable containing the percentage obtained in presentation module of the ars test. These variables calculations are shown in the equations 5.3, 5.6, 5.7.

Similarly, x_{st_ppt} variable in the equation 5.4, is calculated using the equations 5.8 and 5.9. Here the achieved scores and the maximum scores of the modules search, discussion and the presentation related to the self-test are considered.

Report Prediction Model:
The formula for predicting a student's presentation score (y_{ppt}) is:

$$y_{report} = (b_1 * x_{ars\_report}) + (b_2 * x_{st\_report}) + (b_3 * x_{tr}) + e \tag{5.10}$$

where:
- x_{ars_ppt} is the normalized ARS presentation score,
- $x_{self\_test\_ppt}$ is the normalized self-test presentation score,
- x_{tr} is the normalized Topic Recommender score

In the equation 5.10, the terms x_{ars_report}, x_{st_report} and x_{tr} are calculated using the following formulae:

$$x_{ars\_report} = \frac{x_{ars\_report\_achieved\_score}}{x_{ars\_report\_max\_score}} * 100 \tag{5.11}$$

$$x_{st\_report} = \frac{x_{st\_report\_achieved\_score}}{x_{st\_report\_max\_score}} * 100 \tag{5.12}$$

$$x_{tr} = \frac{x_{topic\_recommender\_achieved\_score}}{x_{topic\_recommender\_max\_score}} * 100 \tag{5.13}$$

$$x_{ars\_report\_achieved\_score} = (ars\_search\_achieved\_score + ars\_discussion\_achieved\_score + ars\_report\_achieved\_score) \tag{5.14}$$

$$x_{ars\_report\_max\_score} = (ars\_search\_max\_score + ars\_discussion\_max\_score + ars\_report\_max\_score) \tag{5.15}$$

$$x_{st\_report\_achieved\_score} = (st\_search\_achieved\_score + st\_discussion\_achieved\_score + st\_report\_achieved\_score) \tag{5.16}$$

$$x_{st\_report\_max\_score} = (st\_search\_max\_score + st\_discussion\_max\_score + st\_report\_max\_score) \tag{5.17}$$

$x_{ars\_report\_achieved\_score}$ is the points obtained in the ars_search, ars_discussion and the ars_report modules. Where as the $x_{ars\_report\_max\_score}$ is the sum of the maximum scores possible in the modules search, discussion and report. x_{ars_report} is the variable containing the percentage obtained in report module of the ars test. These variables calculations are shown in the equations 5.14, 5.15, 5.11 respectively.

The equations 5.5, 5.13 are same. So the variable, x_{tr} is calculated only once and used in both the equations 5.2 and 5.10. Calculation of x_{tr} variable is clearly explained in the equations 5.18, 5.19 and 5.20.

In the courses like Hauptseminar in the Automotive Software Engineering branch, students must select a topic from a list of available topics. Students have to do research on the selected topic and then finally write a report and present their work. To assist students in choosing topics aligned with their interests, a quiz called Topic_Recommender is provided [43]. When students complete this quiz, their scores are stored in the database.

$$x_{tr\_achieved\_score} = \sum_{i=1}^{17} (points\_achieved\_for\_task_i * weight\_of\_task_i) \tag{5.18}$$

$$x_{tr\_max\_score} = \sum_{i=1}^{17} weight\_of\_tasks_i \tag{5.19}$$

$$x_{tr\_score\_for\_topic\_i} = \frac{x_{tr\_achieved\_score\_for\_topic\_i}}{x_{tr\_max\_score\_for\_topic\_i}} * 100 \tag{5.20}$$

The Topic_Recommender score is calculated by assigning marks to each option based on the options selected by the student during the quiz. The max marks for each question is '1'. Each option is awarded the mark of '0.33'. So based on the options chosen, the score for that particular question is calculated. Topics have modules and each module has a predefined weight in calculating the topic_recommender score. Score obtained for each question is multiplied with the corresponding module's weight as shown in the equation 5.18.

The maximum score that can be achieved for each question is '1'. When this '1' is multiplied by the corresponding weights, the maximum score that can be obtained is the weight of the module itself. So the maximum possible score for topic recommnder is sum of the weights and its calculation is shown in the equation 5.19.

Score for each topic is obtained by taking the percentage between the variables $x_{tr\_achieved\_score}$ and the $x_{tr\_max\_score}$. This calculation is shown in the equation 5.20.

After calculating the percentage scores of each topic, the top five topics with the highest percentages are then suggested to the student. Along with the future percentage prediction, the task of recommending top five topics to students is also done.

But in the percentage prediction process, the percentage obtained for the topic selected by the student is only considered. All the other topic scores were omitted for future percentage prediction.

In this way, the ARS score, the Self-Test score, and the Topic-Recommender score—serve as independent variables in the MLR algorithm used to predict future performance.

Equation for final score prediction:
By the above mentioned formulae in the equations, 5.2 and 5.10 the variable $y_{overall}$

is calculated. As shown in the equation 5.21 $y_{overall}$ is the average of the variables y_{ppt} and the y_{report}.

$$y_{overall} = avg(y_{ppt} + y_{report}) \tag{5.21}$$

Using the above mentioned formulae for each student - presentation, report and overall scores were predicted and presented.

Training Data
To train the MLR models effectively, a training dataset was provided containing historical data on student performance across multiple semesters and tests. This dataset allowed me to fine-tune the model before applying it to real-time prediction tasks within ARC Tutoring Workbench.

Integration into ARC Tutoring Workbench
As part of this research work, all modules were integrated into a single dashboard called the ARC Tutoring Workbench. For seamless integration, each module was registered as a Flask blueprint, ensuring modularity while allowing them to function cohesively within the larger system architecture. The user interface (UI) was designed using ReactJS to provide an interactive experience for both students and tutors. React's component-based architecture allowed me to create reusable UI elements such as charts, graphs, and forms that dynamically update based on real-time data from Flask APIs.

In conclusion, this research work demonstrates how various Learning Analytics modules can be integrated into one cohesive system—the ARC Tutoring Workbench—using modern web development technologies like Python/Flask for backend logic and ReactJS/NPM for frontend design. By combining descriptive, diagnostic, and predictive analytics into one platform alongside additional functionalities like chatbots and time management tools, this workbench provides comprehensive support for both students and instructors.

The modular design ensures scalability while maintaining flexibility through Flask blueprints—allowing future enhancements without disrupting existing functionalities—and React's dynamic UI ensures an intuitive user experience across all devices

5.6 Data management

In this project, MySQL was used as the primary database for storing and managing preprocessed data. The database consists of a total of eight tables, each serving a specific functionality within the system. These tables are designed to handle various aspects of user management, learning analytics, and student performance tracking.

5.6.1 User and Session Management

The Users and Sessions tables store information related to user authentication and session handling. When a new user registers on the dashboard, their information, such as username, email, and password (encrypted), is stored in the Users table. The Sessions table is responsible for managing active sessions by storing session-related data such as the user's email ID, session ID, and expiration time. This structure ensures that user login and session management are handled securely and efficiently.

5.6.2 Learning Analytics Data

For the learning analytics functionality, which includes Descriptive, Diagnostic, and Predictive analytics, several tables are used to store student-related data. These include:
Students: Stores personal information about each student.
Semester: Contains data about the list of semesters.
Topic: Stores information about the list of available topics.
Student_Topic: Contains the information about the topics selected by the students.
TR_Data, Formative_Data, and Student_Test_Data: These tables store data about student performance, including formative assessments and test scores.
All personal data stored in these tables is masked to ensure compliance with privacy regulations such as GDPR. This ensures that sensitive information is protected while still allowing for meaningful analysis of student performance.

5.6.3 Data Integration

To perform learning analytics effectively, required tables are merged based on common columns that serve as primary keys. For example, student IDs are used to link tables like Students, Student_Topic, and Formative_Data, enabling comprehensive analysis across multiple datasets.

Additional Functionalities
A separate local database is used for the "time management tool" functionality, which was implemented by another team member. This tool operates independently of the main MySQL database but integrates with it when necessary.

This structured approach to data management ensures that all functionalities—user management, learning analytics, and additional tools—are efficiently handled while maintaining data security and integrity.

5.6.4 Data exchange between front-end and back-end

The communication between the frontend and the backend components is facilitated through RESTful APIs, which allow for seamless data exchange between the client-side (ReactJS) and the server-side (Flask/Python). Here's a detailed explanation of how this communication works:

1. Frontend (ReactJS and NPM)
The ReactJS frontend interacts with the backend by making HTTP requests to Flask's RESTful APIs. These requests are typically made using JavaScript libraries such as Axios or the native Fetch API. React components send requests to the backend to either retrieve data from the MySQL database or send new data to be stored.

For example, when a user submits a form (e.g., registration), React captures the input data and sends it to Flask via a POST request. Similarly, when displaying data (e.g., student performance), React makes a GET request to fetch the relevant information from the backend.

2. Backend (Python and Flask)
On the backend, Flask handles these HTTP requests and processes them using Python logic. Flask routes are defined to handle different types of requests (GET, POST, PUT, DELETE) based on the needs of the application. For instance:
- A GET request from React might trigger Flask to query specific tables in the MySQL database (e.g., fetching student scores).
- A POST request might involve inserting new user registration details into the MySQL database.
Flask connects to the MySQL database using libraries like SQLAlchemy or PyMySQL, which allow for executing SQL queries in Python. The data retrieved from MySQL is then processed and sent back to React in JSON format.

3. Database (MySQL)
The MySQL database stores all persistent data, such as user information, student performance records, and session data. When Flask receives a request from React, it interacts with MySQL by executing SQL queries to either retrieve or update information. For example, in response to a GET request from React, Flask might query tables like Students, Formative_Data, or Sessions to retrieve relevant records.Whereas for a POST request, Flask may insert new rows into tables like Users or Student_Topic.

4. Data Flow
The entire process follows this flow:

- The user interacts with the ReactJS front-end.
- React sends an HTTP request (GET/POST) to Flask via Axios or Fetch API.
- Flask processes the request and interacts with MySQL to fetch or update data.
- The response is sent back from Flask to React in JSON format.

- React updates the UI based on the received data.

This architecture ensures smooth communication between the front-end and back-end while maintaining data integrity in MySQL.

The Data Management section provides a comprehensive overview of how data is handled throughout the system. It covers User and Session Management, detailing how user information and session data are securely stored and managed. The Learning Analytics Data subsection explains the management of student data, including performance metrics and course selections. Data Integration focuses on how different tables are merged using primary keys to enable seamless analysis. Finally, the section on Data Exchange between Front-end and Back-end highlights the communication between ReactJS (front-end) and Flask/Python (back-end), ensuring efficient data flow between the user interface and the MySQL database.

Finally, the Implementation chapter provides a detailed breakdown of the technologies, tools, and methodologies used to develop the system. It begins by outlining the System Architecture and Data Preparation processes, which lay the foundation for the subsequent sections. The chapter then delves into the various technologies employed, including Python and Flask for back-end development, ReactJS for front-end development, and tools like Node Package Manager (NPM) and Virtual Environments for managing dependencies. User Interface Design section highlights how UX principles were applied to create an intuitive interface. The chapter also covers the integration of Learning Analytics, as well as comprehensive data management strategies, including user session management and learning analytics data handling. Overall, this chapter demonstrates how these technologies were integrated to build a robust, scalable system that supports both back-end functionality and a seamless user experience.

6 Results and Evaluation

The Results and Evaluation chapter provides a detailed analysis of the performance and usability of the ARC Tutoring Workbench. This chapter presents the outcomes of the predictive models developed using Multiple Linear Regression (MLR) to forecast student performance in both presentation and report scores. The results are visualized through 3D scatter plots, comparing actual data with predicted values for both training and test datasets. Additionally, this chapter evaluates the dashboard's effectiveness based on feedback collected from both students and tutors who interacted with the system.

The evaluation section focuses on gathering insights from users regarding the User Experience (UX) design, ease of navigation, clarity of visualizations, and overall usefulness of the dashboard in supporting academic progress. By analyzing these results, we aim to assess how well the ARC Tutoring Workbench meets its objectives of improving student learning outcomes and aiding instructors in monitoring and intervening in student performance.

In this chapter, we will first present the results obtained from the MLR algorithm for predicting presentation and report scores, followed by an evaluation of user feedback to understand the practical impact of the dashboard on both students and tutors.

6.1 Results

This Results section is divided into different sections for explaining how various components are integrated in the different layers. For example, how are the Descriptive, Diagnostic LA, Predictive LA, Chatbot, Time Management tool are integrated in the Presentation layer, then integration of components in the Application layer and finally the integration in the Data layer are discussed. These arechitectural views are displayed in the figure 6.1. This a common figure for all the layers(Presentation, Application and Data layers).

In the later part of this section, the results obtained in the Predictive LA reimplementation in terms of presentation prediction and report prediction were also discussed.

6.1.1 Integration of components in Presentation layer

The Presentation Layer in the architecture diagram 6.2, plays a critical role in enabling user interaction with the ARC Tutoring Workbench. It acts as the interface

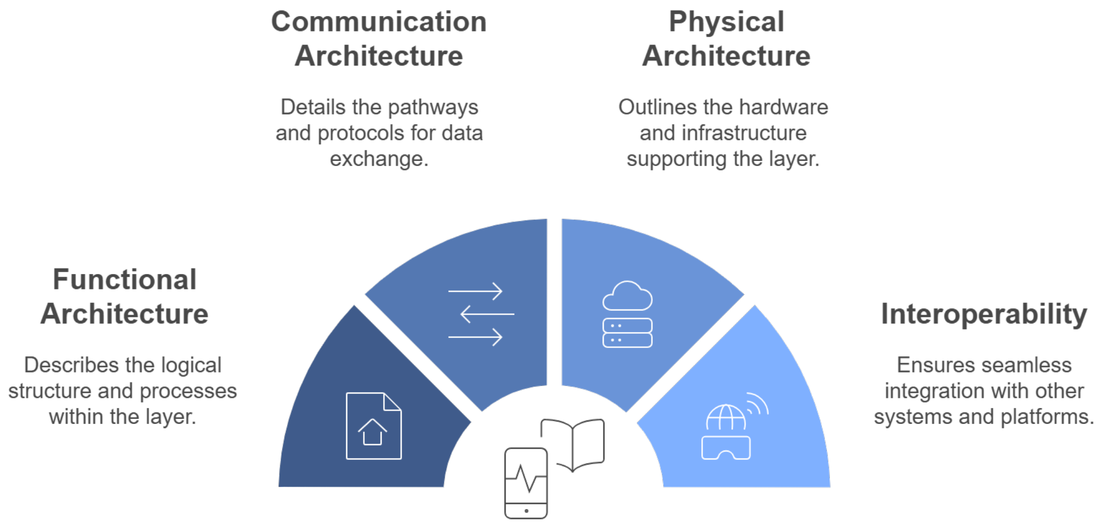

Figure 6.1: Different architectural views in each layer

where students and tutors access various functionalities like Descriptive, Diagnostic, and Predictive Learning Analytics (LA), the Chatbot, and the Time Management Tool. Below, we explain how the Presentation Layer processes HTTP Requests and JSON Responses in terms of Functional Architecture, Communication Architecture, Physical Architecture, and Interoperability.

1. Functional Architecture

The functional architecture of the Presentation Layer focuses on its role in translating user inputs into actionable requests and presenting processed data from the Application Layer back to users. The key functionalities include:

- User Interaction: The Presentation Layer allows users to interact with tools such as dashboards for Learning Analytics, the Chatbot, and the Time Management Tool.
- Request Handling: When a user performs an action (e.g., viewing performance trends or updating tasks), an HTTP Request is generated and sent to the Application Layer.
- Data Visualization: The Presentation Layer receives processed data as a JSON Response from the Application Layer. This data is rendered into user-friendly formats like graphs, charts, or tables using front-end technologies like ReactJS.

For example:
A student requesting their academic progress triggers an HTTP Request to fetch Descriptive LA data. The Application Layer processes this request, retrieves data from the Data Layer, and sends it back as a JSON Response for visualization.

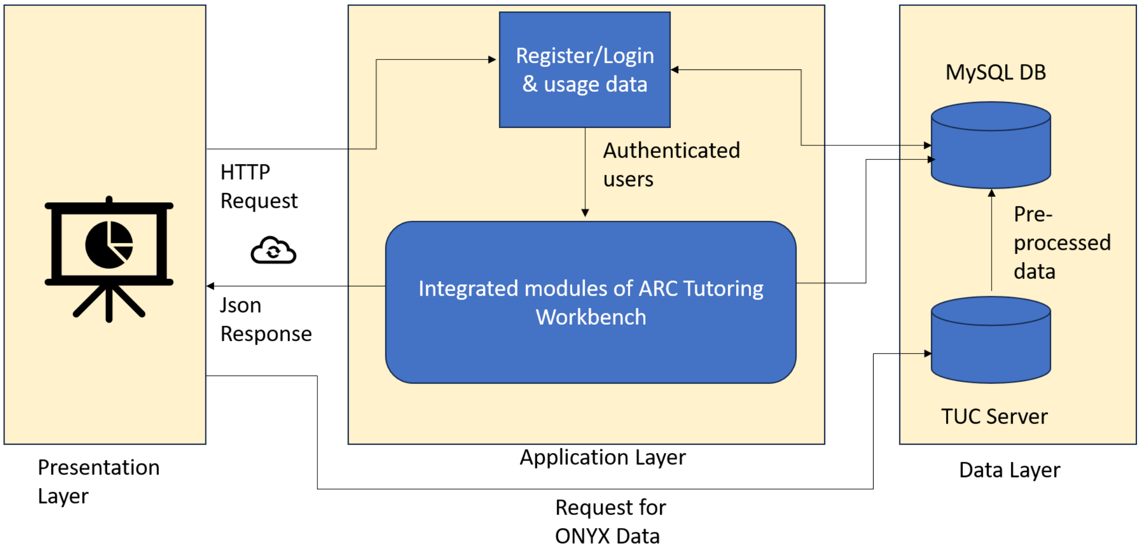

Figure 6.2: Integrated Architecture Diagram

2. Communication Architecture
The communication architecture defines how data flows between the Presentation Layer and other layers:

- HTTP Requests: The Presentation Layer communicates with the Application Layer via RESTful APIs. These requests encapsulate user actions (e.g., fetching predictive analytics or adding a task).
- JSON Responses: The Application Layer processes these requests, interacts with the Data Layer for required information, and sends structured JSON responses back to the Presentation Layer.
- Real-Time Updates: Features like the Chatbot rely on real-time communication protocols to provide instant responses to user queries.

For example:
When a tutor queries student performance trends, an HTTP Request is sent to retrieve diagnostic insights. The JSON Response includes correlation matrices or at-risk student indicators, which are displayed on the dashboard.

3. Physical Architecture
The physical architecture outlines how hardware and software components interact to support system functionality:

- The Presentation Layer operates on client-side devices such as laptops or mobile phones. It uses web technologies like ReactJS to provide an interactive interface.

- The Application Layer resides on a server that processes requests from multiple clients simultaneously.
- The Data Layer is hosted on a database server (e.g., MySQL) that stores pre-processed data and interacts with external systems like the TUC server for ONYX assessments.

This distributed architecture ensures scalability, allowing multiple users to access the system concurrently without performance degradation.

4. Interoperability Between Components
The interoperability of components ensures seamless integration across layers:

- Descriptive and Diagnostic LA Module: Retrieves historical performance data from the Data Layer via the Application Layer and visualizes it in graphs or tables.
- Predictive LA Module: Uses machine learning models stored in pre-processed datasets to forecast future outcomes. Predictions are fetched via APIs and displayed interactively.
- Chatbot: Communicates with both layers to fetch FAQs or user-specific responses while ensuring secure interactions using JWT tokens.
- Time Management Tool: Synchronizes task-related updates between users and database tables through RESTful APIs.

Each component communicates efficiently using standardized protocols (e.g., HTTP/JSON), ensuring real-time data exchange without compatibility issues.

The Presentation Layer acts as a bridge between users and backend systems by sending HTTP Requests and receiving JSON Responses. Its functional architecture ensures smooth user interaction, while its communication architecture facilitates efficient data exchange with other layers. The physical architecture supports scalability by distributing workloads across client devices, servers, and databases. Finally, interoperability between components ensures that all modules—Descriptive, Diagnostic, Predictive LA, Chatbot, and Time Planner—work cohesively to deliver a seamless user experience.

6.1.2 Integration of components in Application layer

The Application Layer in the system architecture of the ARC Tutoring Workbench is designed to act as the central processing unit, enabling seamless communication between the Presentation Layer and the Data Layer. It integrates components such as Descriptive Learning Analytics (LA), Diagnostic LA, Predictive LA, the Chatbot, and the Time Management Tool. Below, we explain its design and functionality in terms of Functional Architecture, Communication Architecture, Physical Architecture, and Interoperability.

1. Functional Architecture
The functional architecture of the Application Layer focuses on processing user requests from the Presentation Layer and retrieving or storing data from the Data Layer. Each component is designed to handle specific tasks:

- Descriptive LA: Processes historical data to generate summaries of student performance, such as progress trends and test scores.
- Diagnostic LA: Analyzes patterns in student performance to identify factors affecting learning outcomes.
- Predictive LA: Uses machine learning models to forecast future performance based on current data. This component retrieves pre-processed datasets from the Data Layer for predictions.
- Chatbot: Handles real-time user queries by processing inputs from the Presentation Layer and fetching appropriate responses from the database.
- Time Management Tool: Manages user tasks, deadlines, and milestones by interacting with task-related data stored in the Data Layer.

Each component operates independently but contributes to a unified system by sending processed data back to the Presentation Layer for visualization.

2. Communication Architecture
The communication architecture defines how data flows between layers:

- The Application Layer communicates with the Presentation Layer via RESTful APIs using HTTP Requests and JSON Responses.
  - For example, when a student requests their academic progress, an HTTP Request is sent to retrieve descriptive analytics data. The Application Layer processes this request and sends a JSON Response containing summarized information.
- The Application Layer interacts with the Data Layer using SQL queries for data retrieval or updates.
  - For instance, when a tutor uses predictive analytics, the Application Layer fetches pre-processed datasets from the database and applies machine learning models to generate predictions.

This architecture ensures efficient data exchange between layers while maintaining high performance.

3. Physical Architecture
The physical architecture outlines how hardware and software components interact:

- The Application Layer is hosted on a server that processes multiple user requests concurrently.
- It leverages frameworks like Python Flask for backend development, ensuring modularity and scalability.
- The Data Layer resides on a separate database server (e.g., MySQL), which stores all user data, including test scores, task lists, and pre-processed datasets.
- External systems like the TUC server provide additional resources (e.g., ONYX assessments) required for advanced analytics.

This distributed setup ensures that workloads are balanced across servers, enabling multiple users to access the system simultaneously without performance degradation.

4. Interoperability Between Components
The interoperability of components within the Application Layer ensures seamless integration across modules:

- The Descriptive and Diagnostic LA module retrieves historical performance data from the Data Layer via SQL queries and formats it into JSON responses for visualization.
- The Predictive LA module applies machine learning models stored in preprocessed datasets to forecast outcomes such as final grades.
- The Chatbot module interacts with both layers to fetch user-specific responses while ensuring secure communication using JWT tokens.
- The Time Management Tool synchronizes task updates between users and database tables through RESTful APIs.

By using standardized protocols (e.g., HTTP/JSON) and modular design principles, these components work cohesively within an integrated system.

The Application Layer of the ARC Tutoring Workbench is designed as a robust intermediary that processes user inputs from the Presentation Layer and interacts with the Data Layer to retrieve or store information. Its functional architecture supports independent operations of components like Descriptive, Diagnostic, Predictive LA, Chatbot, and Time Planner while ensuring they contribute to a unified system. The communication architecture enables efficient data exchange through RESTful APIs, while its physical architecture ensures scalability and reliability. Finally, interoperability between components ensures that all modules work seamlessly together to deliver a cohesive user experience.

6.1.3 Integration of components in Data layer

The Data Layer in the ARC Tutoring Workbench is a critical component that manages the storage, retrieval, and processing of data for all integrated modules, including Descriptive, Diagnostic, Predictive Learning Analytics (LA), Chatbot, and Time Management Tool. Below is an explanation of how the Data Layer is designed and created in terms of Functional Architecture, Communication Architecture, Physical Architecture, and its Interoperability between databases.

1. Functional Architecture The functional architecture of the Data Layer is designed to handle all backend operations related to data management. Its primary responsibilities include:

- Data Storage: The Data Layer uses a MySQL database to store structured data such as user credentials, test scores, task lists, and pre-processed datasets for predictive analytics.
- Pre-Processing: Data required for Predictive LA is pre-processed before storage. This involves cleaning raw data, normalizing scores, and structuring datasets into optimized tables for machine learning models.
- Querying and Retrieval: The Data Layer supports efficient querying for modules like Descriptive LA (to fetch historical performance data) and Diagnostic LA (to analyze patterns or correlations).
- Chatbot Integration: It stores FAQs, user-specific queries, and academic resources to provide real-time responses.
- Task Management: Task-related information such as deadlines, milestones, and completion statuses is stored in dedicated tables linked to user accounts.

This architecture ensures that all modules have access to consistent and accurate datasets while maintaining high performance.

2. Communication Architecture
The communication architecture defines how the Data Layer interacts with other layers:

- Application Layer Communication:
  - The Application Layer sends SQL queries to the Data Layer to retrieve or update data.
  - The responses are sent back to the Application Layer in structured formats (e.g., JSON).
- External Systems Communication:
  - The Data Layer integrates with external systems like the TUC server to fetch ONYX assessment data or other academic resources required for advanced analytics.

This architecture ensures seamless data flow between layers while maintaining security and efficiency.

3. Physical Architecture
The physical architecture outlines how hardware and software components interact within the Data Layer:

- A dedicated MySQL database server hosts all user-related data.
- The database is optimized for handling large-scale queries from multiple users simultaneously.
- Pre-processed datasets required for predictive modeling are stored in separate tables within the database. These datasets include normalized scores and feature-engineered variables.
- The TUC server provides supplementary data such as ONYX assessments. This data is fetched periodically and stored in the database for use by analytics modules.

This distributed setup ensures scalability, allowing the system to handle multiple users without performance degradation.

4. Interoperability Between Datasets
Interoperability ensures that different components can effectively share and access data within the system. By using standardized SQL queries and structured table designs, these components work cohesively within an integrated system.

The Data Layer of the ARC Tutoring Workbench is designed as a robust foundation that supports all backend operations related to data storage, retrieval, and processing. Its functional architecture ensures efficient management of user information, while its communication architecture facilitates seamless interaction with the Application Layer and external systems. The physical architecture provides scalability and reliability through dedicated servers and pre-processed datasets. Finally, interoperability between datasets ensures that all modules—Descriptive LA, Diagnostic LA, Predictive LA, Chatbot, and Time Planner—can access consistent datasets without compatibility issues.

Also, to get a better understanding of the developed dashboard, corresponding pages and their screenshots were attached in the Appendix chapter of the report. Students and tutors have different views. So, the Student View section of the Appendix contains screenshots of the student dashboard. Where as the Tutor view of the Appendix has screenshots of the tutors dashboard.

Example
To provide a better understanding of the process flow involved when a user is interacting with the dashboard, is provided through the figure 6.3.

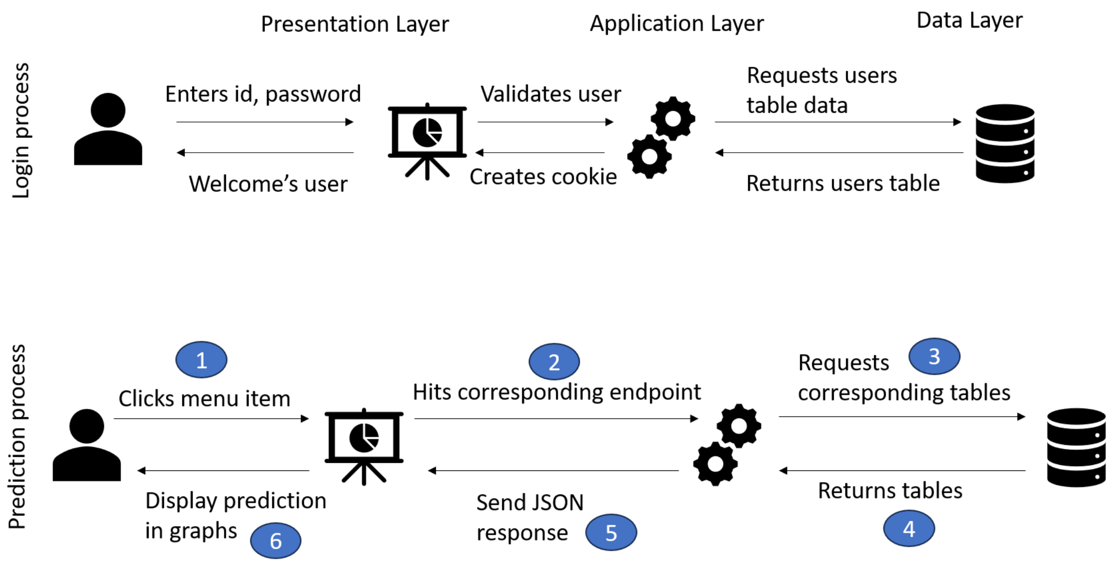

Figure 6.3: Flow of data in login and prediction processes

Two processes have been represented in the figure 6.3, The one above explains the process of login and the below one explains the process of predictive data generation and displaying it to the user. It can be seen in the figure that the process is divided into three layers namely Presentation layer, Application layer and the Data layer. The user and the dashboard comes under Presentation layer, processing of the data comes under the Application layer and here the database comes under the data layer.

Login process:
When a user tries to login to the dashboard, he enters his mail_id and password used during the profile creation. After clicking on the "login" button, a HTTP request with the entered id and password is triggered. This is forwarded to the Application layer. The application layer i.e., the processing unit has to validate the user's credentials. So, it requests "Users" table from the database. As all the registered users data will be stored in the "Users" table of the database.

Now the credentials entered by the user are compared with the id and password columns existing in the "Users" database table. When there is a match, a cookie is being created with the user's information such as "matriculation_number", "mail_id", user type i.e., student or tutor. Finally user will be successfully logged in. In case there is no match of user's credentials with the entries in the database, the user will not be able to login to the dashboard.

Prediction Process:
As shown in the figure 6.3, the below process shows the interaction of a user with

the dashboard to view his prediction results. The numbers in the diagram will be explained in-detail as follows:
① User clicks on the menu item named "My Progress Prognosis" in the dashboard.
② It hits the endpoint

```
base_url/pred_host/predict_score_per_stud?mat_id=284779
```

This mat_id is obtained from the information stored in the cookie. "pred_host" is the url_prefix given to the predictive LA related endpoints. In this integration I have used the concept of Blueprints. When registering each module as a blueprint, it can be given a url_prefix so that it can easily identified which endpoint belongs to which module.
③ Now the process logic in the application layer requests the centralized database for the required tables. Those database tables were shown in the table 6.1. The names of columns and rows are the DB table names and the values in the table represent the name of the common column on which those two corresponding tables were merged on.
④ The db returns the requested tables to the processing logic.
⑤ These tables were then merged on the common columns and then the independent variables created were given as input to the MLR algorithms to calculate the prediction %. Here we have "Presentation_Prediction_%","Report_Prediction_%" and the "Overall_Prediction_%". The result is sent back in the form of JSON response.
- Example JSON Response:


```
[{
 "Predicted_Presentation_Score (%)" :84.31,
 "Predicted_Report_Score (%)":82.46,
 "Predicted_Overall_Score (%):83.385
 }]
```


⑥ For providing a better User Experience, this JSON response is displayed in the form of charts and graphs to the user. This can be seen from the Student view of the dashboard (figure 8.6) in the Appendix chapter.

The image 6.4, 6.5 shows the results generated by the Multiple Linear Regression (MLR) prediction algorithm for predicting the presentation scores and the report scores respectively. The results are visualized in two 3D scatter plots: one for the training data and the other for the test data. These plots help in understanding how well the MLR model fits the training data and how accurately it generalizes to unseen test data. These plots are generated using three key independent variables: ARS Scores, Self-Test Scores, and Topic Recommender Scores, which are used to predict the dependent variable—Presentation Score. The dependent variable is either the Presentation Score or the Report Score, depending on the specific model.

Table 6.1: Database Tables and their common columns

Table1	**Table2**	**Common Columns**
Student	Student_Topic	Student_ID
Student	Student_Test_Data	Student_ID
Student	TR_Data	Student_ID
Semester	Student_Topic	Semester_ID
Semester	Student_Test_Data	Semester_ID
Student_Topic	Topic	Topic_ID
Student_Topic	Student_Test_Data	Student_ID, Semester_ID
Student_Test_Data	TR_Data	Test_ID

6.1.4 Presentation Score Prediction Results

The first set of graphs represents the prediction of presentation scores. The left graph shows the results for the training data, while the right graph shows the results for the test data.

a. Training Data Results (Left Graph)
The left side of the image represents the training data results. In this plot, the green dots represent the actual training data points, while the blue line represents the trained MLR model's predicted values. The axes represent ARS Scores on x-axis, Self-Test Scores on y-axis, Topic Recommender Scores on z-axis.

The plot 6.4 shows how well the trained model fits the actual data points. The blue line indicates how the predicted presentation scores vary with respect to changes in ARS scores, Self-Test scores, and Topic Recommender scores.

Observations:
1. Model Fit: The blue line demonstrates how well the MLR model fits the actual training data points. The alignment between the green dots (actual data) and the blue line (predicted values) indicates that the model has successfully captured the relationship between ARS scores, Self-Test scores, and Topic Recommender scores in predicting presentation scores.
2. Data Distribution: Most of the training data points are clustered around lower ARS and Self-Test scores, indicating that many students scored lower on these metrics. However, there are a few outliers where students achieved higher ARS and Self-Test scores.
3. Topic Recommender Influence: The z-axis (Topic Recommender scores) shows a broader range of values, indicating that topic selection plays a significant role in influencing presentation scores.

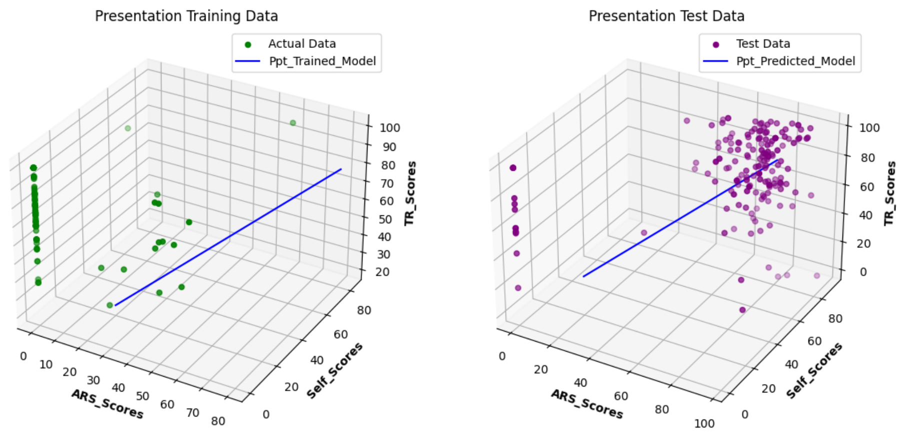

Figure 6.4: Results of presentation prediction

b. Test Data Results(Right Graph)
In this plot, purple dots represent actual test data points, and the blue line represents predictions made by the trained MLR model on unseen test data.

Observations:
1. Prediction Accuracy: The alignment between actual test data points (purple dots) and predicted values (blue line) indicates how well the model generalizes to unseen test data. While there is some alignment, there is also noticeable variance, particularly at higher ARS and Self-Test score ranges.
2. Data Clustering: Similar to the training data, test data points are clustered around lower ARS and Self-Test scores. However, there is more variance in this dataset compared to training data, which may explain some of the discrepancies between actual and predicted values.
3. Generalization: Although there is some deviation between actual test data and predicted values, especially at higher score ranges, the model still captures general trends reasonably well.

6.1.5 Report Score Prediction Results

The second set of graphs represents the prediction of report scores using a similar approach as for presentation scores.

a. Training Data Results (Left Graph)
In this plot, green dots represent actual training data points for report score prediction, while the blue line represents predictions made by the trained MLR model.

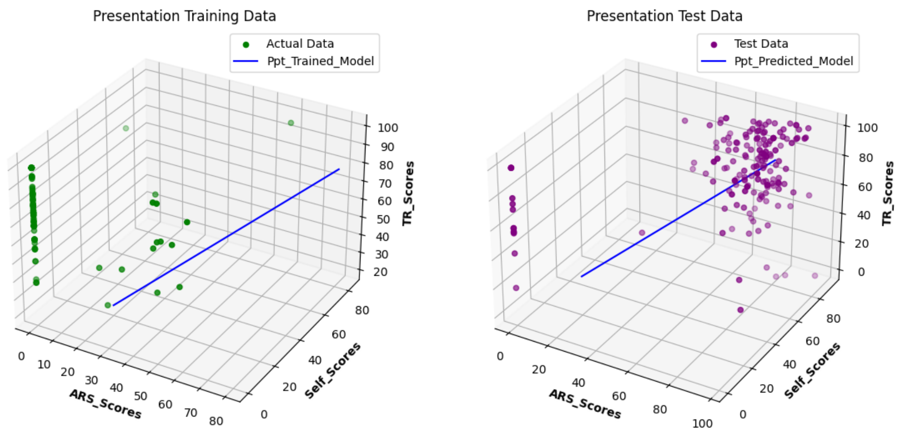

Figure 6.5: Results of report prediction

Observations:
1. Model Fit: The blue line demonstrates how well the MLR model fits report score training data points. The alignment between green dots and the blue line indicates that the model has captured key relationships between ARS scores, Self-Test scores, and Topic Recommender scores in predicting report scores.
2. Data Distribution: Similar to presentation score predictions, most of the report score training data is clustered around lower ARS and Self-Test scores.
3. Topic Recommender Influence: Topic Recommender scores again show a wide range of values, suggesting that topic selection significantly influences report score predictions.

b. Test Data Results (Right Graph)

In this plot, purple dots represent actual test data points for report score prediction, while the blue line represents predictions made by the trained MLR model on unseen test data.

Observations:
1. Prediction Accuracy: The alignment between purple dots (actual test data) and blue line (predicted values) shows how well the model generalizes to unseen report score test data. There is some variance between actual and predicted values at higher ARS and Self-Test score ranges.
2. Data Clustering: Similar to presentation score predictions, report score test data points are clustered around lower ARS and Self-Test scores.
3. Generalization Capability: Although there is some deviation at higher score

ranges, particularly with higher Topic Recommender scores, overall trends are captured reasonably well by the model.

From these visualizations generated by MLR prediction models for both presentation and report scores:

- The MLR models demonstrate a good fit to training data for both presentation and report score predictions.
- The models generalize reasonably well to unseen test data; however, some variance exists at higher ARS and Self-Test ranges.
- Topic Recommender plays a significant role in influencing both presentation and report score predictions.
- The clustering of most students around lower ARS/Self-Test ranges suggests that more emphasis might be needed in improving student performance in these areas.
- Future improvements could involve refining feature engineering or incorporating additional variables to further enhance prediction accuracy.

These results provide valuable insights into student performance trends based on their participation in ARS tests, self-tests, and topic recommendations within courses like Hauptseminar in Automotive Software Engineering.

Along with the Presentation score and the Report score prediction results, this research also found ways and integrated the LA and other available modules into a dashboard. Such results will be discussed as follows:

6.2 Evaluation

The ARC Tutoring Workbench was developed to enhance the learning experience of students and provide tutors with a platform for monitoring and supporting student progress. To evaluate the effectiveness and usability of the dashboard, two surveys were conducted with Master's students who interacted with the tool.These surveys aimed to gather feedback on various aspects of the dashboard, including its user experience, data visualization, chatbot functionality, topic recommender, self-tests, and time management tools.

A. Tool Observation Survey
The Tool Observation Survey was designed to capture students' general impressions of the ARC Tutoring Workbench after observing its features and functionalities. A total of 26 students participated in this survey. The students were asked to rate their level of agreement with various statements about the tool's usefulness and ease of use. The results of this survey were depicted in the figure 6.6.

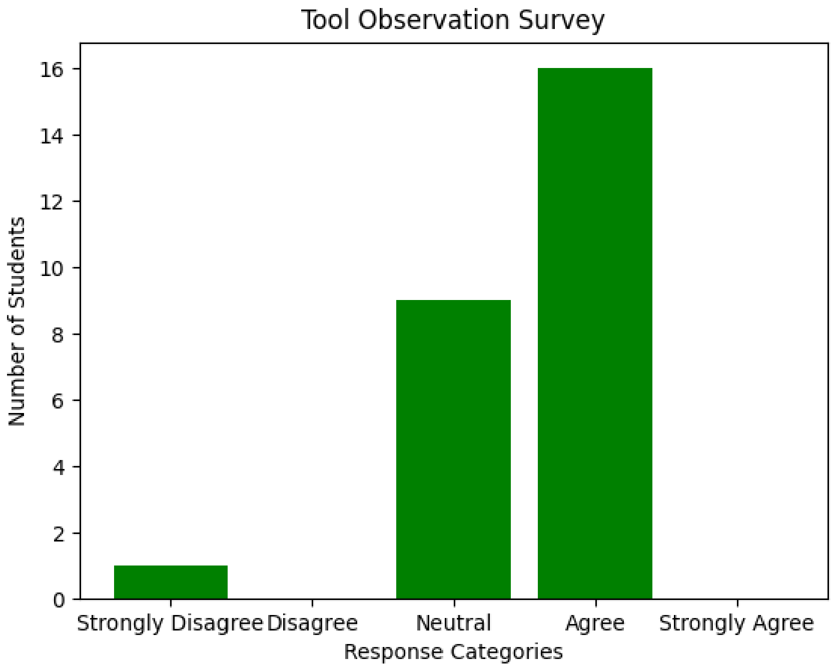

Figure 6.6: Technical Usability of the Tool

The majority of students (16 out of 26) agreed that the ARC Tutoring Workbench was useful and effective in supporting their academic needs. A smaller group (9 students) remained neutral i.e., they chose the option of 'Neither Agree Nor Disagree'. While only 1 student expressed strong disagreement.

The results suggest that most students found the tool beneficial, although there is room for improvement in certain areas to ensure that all users find it equally helpful.

B. Tool Interaction Survey

The Tool Interaction Survey was conducted to gather more detailed feedback on specific aspects of the ARC Tutoring Workbench's functionality. A total of 23 students participated in this survey, with a male-to-female ratio of approximately 4:1. Most participants were between the ages of 21 and 30, representing a typical demographic for Master's students.

The survey included several questions designed to assess satisfaction with different features of the dashboard, including data visualization, chatbot functionality, topic recommender, self-tests, and time management tools. For this survey, 1 stands for strongly disagree and 5 stands for strongly agree.

Key Findings:

1. Data Visualization: Students were asked if data visualization improved their learning experience. The mean satisfaction score was 3.83 (out of 5), indicating that most students found visualizing their progress and predicted performance helpful for understanding their academic standing.

2. Increases Productivity: Students rated whether the tool increased their academic productivity. The mean satisfaction score was 4.00, showing that the majority felt the dashboard helped them stay organized and productive.

3. Easier Study During Thesis/Internship: Students were asked if the tool made it easier to manage studies during research internships or Master's thesis work. The mean satisfaction score was 3.78, reflecting positive feedback but also highlighting areas for potential improvement.

The detailed results for these three questions are summarized in Table 6.2.

Table 6.2: Mean and Standard Deviation for Tool Interaction Survey

S NO.	**Question**	**Mean**	**Standard Deviation (std)**
1.	The ARC-Tutoring Workbench can improve my learning experience.	3.82608	0.49102
2.	This Tutoring Workbench can increase my academic productivity.	4.00000	0.674199
3.	This Tutoring Workbench can make it easier to study during my Research Internship or Master Thesis.	3.78260	0.599736
4.	**Average**	3.869565	0.588320

The survey also evaluated satisfaction with individual modules:

4. Chatbot Functionality:
Among 23 participants, 14 agreed that the chatbot was helpful, while 5 disagreed. Although generally well-received, some students suggested improvements in responsiveness and handling complex queries.

5. Topic Recommender Satisfaction:
For this feature, 10 students agreed it was useful for selecting topics aligned with their interests, while 8 remained neutral, indicating indifference among some users.

6. Self-Test Satisfaction:
The self-tests received overwhelmingly positive feedback, with 18 students agreeing they were helpful for assessing knowledge gaps and tracking progress.

7. Time Management Tool Satisfaction:
This feature was widely appreciated for helping users plan tasks effectively during

critical academic phases like thesis writing or internships. Out of 23 participants, 5 strongly agreed, 11 agreed, and 7 remained neutral about its usefulness.
Table 6.3 provides a detailed breakdown of responses for these modules.

Table 6.3: Survey Results for modules Chatbot, Topic Recommender, Self-Test and Time Planner.

	Chatbot	**Topic Recommender**	**Self-Test**	**Time Planner**
Strongly Disagree / 1	0	1	0	0
Disagree / 2	5	1	1	0
Neither agree nor Disagree / 3	4	8	3	7
Agree / 4	14	10	18	11
Strongly Agree / 5	0	3	1	5
Total	23	23	23	23
Median	4	4	4	4
Mode	4	4	4	4

Overall, the ARC Tutoring Workbench received positive feedback from both surveys. Most users found value in its features such as data visualization for tracking progress, chatbot assistance for academic queries, topic recommender for guiding topic selection during thesis work or internships, self-tests for knowledge assessment, and time management tools for organizing tasks efficiently.

However, there are areas where improvements can be made—particularly in enhancing chatbot responsiveness and providing additional support through topic recommender functionalities.

The Results and Evaluation chapter demonstrates the successful implementation and integration of the ARC Tutoring Workbench's components across its Presentation, Application, and Data Layers. The predictive models for presentation and report scores proved effective in providing actionable insights for students and tutors, enabling better academic planning and timely interventions. This can be seen from the low values of the calculated mean square errors. The lower the mean square error, the greater is the system's accuracy in prediction.

User feedback gathered through surveys highlighted the system's usability, valuing features like predictive analytics, self-tests, and time management tools. Overall, the workbench effectively enhances learning outcomes and teaching efficiency, validating its potential as a robust educational tool.

Limitations and Future Scope

Dashboard load times are significantly impacting user experience, particularly with predictive analytics visualizations. Also, due to long response times of the chatbot, it is affecting student engagement and satisfaction levels. The complexity of learning analytics implementation often creates a gap between research capabilities and practical application.

The ARC Tutoring Workbench shows good potential, but requires key enhancements to improve its performance and user experience. To address current performance issues, implementing sophisticated caching mechanisms and pre-filtered data views could significantly reduce dashboard loading time. The system architecture could benefit from event-driven processing for more efficient handling of real-time data streams and analytics visualization.

The chatbot's performance could be enhanced through optimized Natural Language Processing algorithms and intelligent response caching. A hybrid approach combining pre-defined templates with dynamic processing could better balance response speed and accuracy.

Future improvements include integrating advanced AI models for enhanced predictive analytics and implementing edge computing solutions to reduce latency. These changes, along with continuous performance monitoring, would create a more responsive learning environment. Additionally, bridging the gap between theoretical analytics and practical implementation through streamlined integration methods and adaptive systems would pave the way for more sophisticated learning analytics applications.

7 Conclusion

The ARC Tutoring Workbench is a significant step forward in improving the educational experience for both students and tutors. By integrating multiple tools and learning analytics, the dashboard provides students with a clear understanding of their current performance, helps them identify areas for improvement, and predicts future outcomes. This encourages students to focus their efforts on achieving better grades. The Time Management tool supports students in organizing their tasks and milestones effectively, while the chatbot offers valuable assistance in creating reports and designing presentations.

For tutors, the dashboard provides a comprehensive view of student performance, helping them identify those who need additional support. With features like advanced filtering and detailed data analysis, tutors can deliver personalized assistance and focus their efforts where they are needed most. The secure login ensures that both students and tutors access only their respective dashboards, maintaining privacy and data security.

A survey was conducted to gather feedback from students and tutors. The results showed that the dashboard had a positive impact on academic performance, task organization, and teaching strategies. Both students and tutors appreciated how the workbench simplified their workflows and provided actionable insights.

In conclusion, the ARC Tutoring Workbench successfully combines data-driven analytics with practical tools to support students and tutors alike. Its features empower students to take control of their learning and enable tutors to provide targeted guidance. Feedback from users validates its effectiveness and highlights its potential to transform the way we approach education, paving the way for more personalized and efficient learning experiences in the future.

Bibliography

[1] S. Hantoobi, A. Wahdan, M. Al-Emran, and K. Shaalan, “A review of learning analytics studies,” *Recent advances in technology acceptance models and theories*, pp. 119–134, 2021.

[2] M. S. Islam, “Design and development of a predictive learning analytics system,” Master’s thesis, Chemnitz Technische Universität Chemnitz, Fakultät für Informatik, August 2024.

[3] M. Schmidt, A. A. Tawfik, I. Jahnke, and Y. Earnshaw, “Learner and user experience research: An introduction for the field of learning design & technology,” *EdTech Books*, 2020.

[4] U. Shegupta, “Learning analytics driven arc-tutoring for individual study success.”

[5] O. Viberg, M. Hatakka, O. Bälter, and A. Mavroudi, “The current landscape of learning analytics in higher education,” *Computers in Human Behavior*, vol. 89, pp. 98–110, 2018.

[6] D. Ifenthaler and Y. J.Y.K., “Learning analytics dashboard: Impact on teaching and learning,” *British Journal of Educational Technology*, vol. 51, no. 5, pp. 1434–1448, 2020.

[7] I. Masiello, Z. A. Mohseni, F. Palma, S. Nordmark, H. Augustsson, and R. Rundquist, “A current overview of the use of learning analytics dashboards,” *Education Sciences*, vol. 14, no. 1, 2024. [Online]. Available: https://www.mdpi.com/2227-7102/14/1/82

[8] L. Johnson, S. Adams Becker, M. Cummins, A. Freeman, D. Ifenthaler, and N. Vardaxis, *Technology Outlook for Australian Tertiary Education 2013-2018: An NMC Horizon Project Regional Analysis.* ERIC, 2013.

[9] S. N. Kew and Z. Tasir, “Learning analytics in online learning environment: A systematic review on the focuses and the types of student-related analytics data,” *Technology, Knowledge and Learning*, vol. 27, no. 2, pp. 405–427, 2022.

[10] A. Aristovnik, K. Karampelas, L. Umek, and D. Ravšelj, “Impact of the covid-19 pandemic on online learning in higher education: a bibliometric analysis,” in *Frontiers in Education*, vol. 8. Frontiers Media SA, 2023, p. 1225834.

[11] A. Pfeiffer and D. Uckelmann, "Fostering lab-based learning with learning analytics-a literature review," *International Journal of Online and Biomedical Engineering (iJOE)*, vol. 18, no. 14, pp. 4–27, 2022.

[12] E. Duval, "Attention please! learning analytics for visualization and recommendation," in *Proceedings of the 1st international conference on learning analytics and knowledge*, 2011, pp. 9–17.

[13] D. Ifenthaler and J. Y.-K. Yau, "Utilising learning analytics to support study success in higher education: a systematic review," *Educational Technology Research and Development*, vol. 68, no. 4, pp. 1961–1990, 2020.

[14] D. J. Slutsky, "The effective use of graphs," *Journal of wrist surgery*, vol. 3, no. 02, pp. 067–068, 2014.

[15] A. Bojko, "Informative or misleading? heatmaps deconstructed," in *Human-Computer Interaction. New Trends: 13th International Conference, HCI International 2009, San Diego, CA, USA, July 19-24, 2009, Proceedings, Part I 13.* Springer, 2009, pp. 30–39.

[16] H. Siirtola, "The cost of pie charts," in *2019 23rd International Conference Information Visualisation (IV).* IEEE, 2019, pp. 151–156.

[17] A. Mayorga and M. Gleicher, "Splatterplots: Overcoming overdraw in scatter plots," *IEEE transactions on visualization and computer graphics*, vol. 19, no. 9, pp. 1526–1538, 2013.

[18] A. Sarikaya, M. Correll, L. Bartram, M. Tory, and D. Fisher, "What do we talk about when we talk about dashboards?" *IEEE transactions on visualization and computer graphics*, vol. 25, no. 1, pp. 682–692, 2018.

[19] G. Ramaswami, T. Susnjak, A. Mathrani, and R. Umer, "Use of predictive analytics within learning analytics dashboards: A review of case studies," *Technology, Knowledge and Learning*, vol. 28, no. 3, pp. 959–980, 2023.

[20] J. Klerkx, K. Verbert, and E. Duval, "Learning analytics dashboards," *The handbook of learning analytics*, pp. 143–150, 2017.

[21] C. Lang, G. Siemens, A. Wise, and D. Gasevic, *Handbook of learning analytics.* SOLAR, Society for Learning Analytics and Research New York, 2017.

[22] K. Verbert, X. Ochoa, R. De Croon, R. A. Dourado, and T. De Laet, "Learning analytics dashboards: the past, the present and the future," in *Proceedings of the tenth international conference on learning analytics & knowledge*, 2020, pp. 35–40.

[23] E. U. Sirje Virkus, Sigrid Mandre, *Learning Analytics Dashboards.* TEACH4EDU4, 2023.

[24] N. Bhatti, I. A. Ismaili, and L. D. Dhomeja, "Identifying and resolving disorientation in e-learning systems," in *2017 international conference on communication, Computing and Digital Systems (C-CODE)*. IEEE, 2017, pp. 169–174.

[25] A. Almufarreh, M. Arshad, and S. H. Mohammed, "An efficient utilization of blackboard ally in higher education institution." *Intelligent Automation & Soft Computing*, vol. 29, no. 1, 2021.

[26] F. K. Alzahrani and W. S. Alhalafawy, "Gamification for learning sustainability in the blackboard system: motivators and obstacles from faculty members' perspectives," *Sustainability*, vol. 15, no. 5, p. 4613, 2023.

[27] H. Athaya, R. D. A. Nadir, D. Indra Sensuse, K. Kautsarina, and R. R. Suryono, "Moodle implementation for e-learning: A systematic review," in *Proceedings of the 6th International Conference on Sustainable Information Engineering and Technology*, 2021, pp. 106–112.

[28] A. H. Muhammad, A. Siddique, A. E. Youssef, K. Saleem, B. Shahzad, A. Akram, and A.-B. S. Al-Thnian, "A hierarchical model to evaluate the quality of web-based e-learning systems," *Sustainability*, vol. 12, no. 10, p. 4071, 2020.

[29] I. Bouchrika, N. Harrati, V. Wanick, and G. Wills, "Exploring the impact of gamification on student engagement and involvement with e-learning systems," *Interactive Learning Environments*, vol. 29, no. 8, pp. 1244–1257, 2021.

[30] A. Abid, A. Abdalla, A. Abid, D. Khan, A. Alfozan, and J. Zou, "Gradio: Hassle-free sharing and testing of ml models in the wild," *arXiv preprint arXiv:1906.02569*, 2019.

[31] S. N. Kew and Z. Tasir, "Developing a learning analytics intervention in e-learning to enhance students' learning performance: A case study," *Education and Information Technologies*, vol. 27, no. 5, pp. 7099–7134, 2022.

[32] R. Conijn, C. Snijders, U. Matzat, A. Kleingeld, and W. N. Bijvank, "Exctra-exploiting the click-trail: Assessing the benefits of learning analytics," 2016.

[33] H. M. Judi, Z. Iksan *et al.*, "Engaging students in online data analytics learning: Meaningful learning in higher education," *EDUCATUM Journal of Science, Mathematics and Technology*, vol. 9, no. 2, pp. 16–22, 2022.

[34] S. Islam and H. Mahmud, "Integration of learning analytics into learner management system using machine learning," in *Proceedings of the 2020 2nd International Conference on Modern Educational Technology*, 2020, pp. 1–4.

[35] C. Mahmoud and J. Sørensen, “Artificial intelligence in personalized learning with a focus on current developments and future prospects,” *Research and Advances in Education*, vol. 3, no. 8, pp. 25–31, 2024.

[36] F. Provost and T. Fawcett, “Data science and its relationship to big data and data-driven decision making,” *Big data*, vol. 1, no. 1, pp. 51–59, 2013.

[37] B. Šumak, D. López-de Ipiña, O. Dziabenko, S. D. Correia, L. M. S. de Carvalho, S. Lopes, İ. Şimşek, T. Can, D. I. Kline, and M. Pušnik, “Ai-based education tools for enabling inclusive education: Challenges and benefits,” in *2024 47th MIPRO ICT and Electronics Convention (MIPRO)*. IEEE, 2024, pp. 472–477.

[38] D. R. Thomas, J. Lin, E. Gatz, A. Gurung, S. Gupta, K. Norberg, S. E. Fancsali, V. Aleven, L. Branstetter, E. Brunskill *et al.*, “Improving student learning with hybrid human-ai tutoring: A three-study quasi-experimental investigation,” in *Proceedings of the 14th Learning Analytics and Knowledge Conference*, 2024, pp. 404–415.

[39] A. Sakr and T. Abdullah, “Virtual, augmented reality and learning analytics impact on learners, and educators: A systematic review,” *Education and Information Technologies*, pp. 1–50, 2024.

[40] F. U. Ahmed, “Authentication and verification in diagnostic learning analytics for audience response system (ars) test and self test,” Dept. of Computer Science, Chair of Computer Engineering, Research Internship Report, September 2024.

[41] S. Aggarwal *et al.*, “Modern web-development using reactjs,” *International Journal of Recent Research Aspects*, vol. 5, no. 1, pp. 133–137, 2018.

[42] S. A. Khan, “Design and development of a diagnostic learning analytics system,” Master’s thesis, Chemnitz Technische Universität Chemnitz, Fakultät für Informatik, August 2024.

[43] E. Moser, U. U. Shegupta, K. Ihsberner, O. Jalilov, R. Schmidt, and W. Hardt, “Designing digital self-assessment and feedback tools as mentoring interventions in higher education,” in *Workshop Gemeinschaften in Neuen Medien (GeNeMe) 2022*. TUDpress-Verlag der Wissenschaften, 2022.

[44] S. Ismail, “Development of a generative model based backend of tutoring agent,” Master’s thesis, Chemnitz Technische Universität Chemnitz, Fakultät für Informatik, 2024.

[45] T. K. Shukla, “Time management tool development to support self-regulated learning,” Master’s thesis, Chemnitz Technische Universität Chemnitz, Fakultät für Informatik, 2024.

[46] G. Siemens and P. Long, "Penetrating the fog: Analytics in learning and education." *EDUCAUSE review*, vol. 46, no. 5, p. 30, 2011.

[47] A. M. Abbas, K. I. Ghauth, and C.-Y. Ting, "User experience design using machine learning: a systematic review," *IEEE Access*, vol. 10, pp. 51 501–51 514, 2022.

[48] R. Unger and C. Chandler, *A Project Guide to UX Design: For user experience designers in the field or in the making.* New Riders, 2023.

[49] R. Kaliisa, K. Misiejuk, S. López-Pernas, M. Khalil, and M. Saqr, "Have learning analytics dashboards lived up to the hype? a systematic review of impact on students' achievement, motivation, participation and attitude," in *Proceedings of the 14th Learning Analytics and Knowledge Conference*, 2024, pp. 295–304.

[50] Y. V. Paredes, R. F. Siegle, I.-H. Hsiao, and S. D. Craig, "Educational data mining and learning analytics for improving online learning environments," in *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, vol. 64, no. 1. SAGE Publications Sage CA: Los Angeles, CA, 2020, pp. 500–504.

[51] T. Gallagher, B. Slof, M. van der Schaaf, R. Toyoda, Y. Tehreem, S. Garcia Fracaro, and L. Kester, "Reference frames for learning analytics dashboards: The progress and social reference frame and occupational self-efficacy," *Journal of Computer Assisted Learning*, vol. 40, no. 2, pp. 742–760, 2024.

[52] T. A. B. Tretow-Fish and M. S. Khalid, "Methods for evaluating learning analytics and learning analytics dashboards in adaptive learning platforms: A systematic review," *Electronic Journal of e-Learning*, vol. 21, no. 5, pp. 430–449, 2023.

[53] N. Hamidli, "Introduction to ui/ux design: key concepts and principles," *Academia. URL: https://www.academia.edu/98036432/Introduction_to_UI_UX_Design_Key_Concepts_and_Principles [accessed 2024-04-27]*, 2023.

[54] M. Schmidt and R. Huang, "Defining learning experience design: Voices from the field of learning design & technology," *TechTrends*, vol. 66, no. 2, pp. 141–158, 2022.

[55] A. A. Tawfik, J. Gatewood, J. J. Gish-Lieberman, and A. J. Hampton, "Toward a definition of learning experience design," *Technology, Knowledge and Learning*, vol. 27, no. 1, pp. 309–334, 2022.

[56] K. Finstad, "The usability metric for user experience," *Interacting with computers*, vol. 22, no. 5, pp. 323–327, 2010.

[57] M. Schmidt, Y. Earnshaw, A. Tawfik, and I. Jahnke, *Learning Experience Design*, 10 2023.

[58] K. Slimani, S. Bourekkadi, R. Messoussi, Y. Ruichek, and R. Touahni, "Sharing emotions in the distance education experience: attitudes and motivation of university students," in *2020 International Conference on Intelligent Systems and Computer Vision (ISCV)*. IEEE, 2020, pp. 1–10.

[59] J. Nie *et al.*, "The exploration of quality assurance for student learning experience in blended and online teaching," *Adult and Higher Education*, vol. 5, no. 19, pp. 62–67, 2023.

[60] D. Grant-Smith *et al.*, "Principles and practices for enhanced visual design in virtual learning environments: Do looks matter in student engagement?" in *Research Anthology on Developing Effective Online Learning Courses*, I. R. M. Association, Ed. IGI Global, 2021, pp. 1725–1749. [Online]. Available: https://doi.org/10.4018/978-1-7998-8047-9.ch087

[61] K. Kibithe and P. Naibei, "Unlocking the power of digital libraries: a comprehensive review," *International Research Journal of Modernization in Engineering Technology and Science*, vol. 5, no. 11, pp. 2270–2275, 2023.

[62] S. Hickey and A.-P. Correia, "Centering the learner within instructional design: The evolution of learning design and the emergence of learning experience design (lxd) in workforce training and development," *Journal of Educational Technology Systems*, vol. 52, no. 4, pp. 429–447, 2024. [Online]. Available: https://doi.org/10.1177/00472395231226094

[63] C. M. Gray and E. Boling, "Learning experience design in the light of design knowledge and philosophy," *Journal of Applied Instructional Design*, vol. 12, no. 3, pp. 217–227, 2023.

[64] W. Phommanee, B. Plangsorn, and S. Siripipattanakul, "A systematic review of changing conceptual to practice in learning experience design: Text mining and bibliometric analysis." *Contemporary Educational Technology*, vol. 15, no. 4, 2023.

[65] R. T. Huang, "Designing and evaluating a 3d virtual world game for english language learning: A learning experience design approach."

[66] K. Sims, M. Wegener, L. Nichols, and M. Villanueva, "Using learning experience design (lxd) to promote decreasing stigma in creating a video series about syringe services programs (ssp)."

[67] R. M. Quintana and C. Quintana, "Situating mooc learners within the field of learning experience design through immersion in authentic contexts," *The Journal of Applied Instructional Design*, vol. 12, no. 3, 2023.

[68] H. Hambrock and F. De Villiers, "Proposing a seamless learning experience design (sled) framework based on international perspectives of educators from five higher education institutions," *Electronic Journal of e-Learning*, vol. 21, no. 1, pp. 52–68, 2023.

[69] C. M. Reigeluth and Y. An, "What's the difference between learning experience design and instructional design," *Journal of Applied Instructional Design*, vol. 12, no. 3, p. 89, 2023.

[70] F. Gutiérrez, K. Seipp, X. Ochoa, K. Chiluiza, T. De Laet, and K. Verbert, "Lada: A learning analytics dashboard for academic advising," *Computers in Human Behavior*, vol. 107, p. 105826, 2020.

[71] A. Droit and B. Rieger, "Learning analytics in the flipped classroom–learning dashboards from the students' perspective," 2020.

[72] D. Zandvliet, "Towards effective learning analytics for higher education: returning meaningful dashboards to teachers," Ph.D. dissertation, Vrije Universteit Amsterdam, The Netherlands, 2020.

[73] M. Sahin and D. Ifenthaler, "Visualizations and dashboards for learning analytics: A systematic literature review," *Visualizations and dashboards for learning analytics*, pp. 3–22, 2021.

[74] J. Han, K. H. Kim, W. Rhee, and Y. H. Cho, "Learning analytics dashboards for adaptive support in face-to-face collaborative argumentation," *Computers & Education*, vol. 163, p. 104041, 2021.

[75] Z. Mohseni, R. M. Martins, and I. Masiello, "Savis: a learning analytics dashboard with interactive visualization and machine learning," in *Nordic Learning Analytics (Summer) Institute 2021, KTH Royal Institute of Technology, Stockholm, 23 August 2021*, vol. 2985. ceur-ws. org, 2021.

[76] P. Leitner, M. Ebner, H. Geisswinkler, and S. Schön, "Visualization of learning for students: A dashboard for study progress–development, design details, implementation, and user feedback," in *Visualizations and dashboards for learning analytics*. Springer, 2021, pp. 423–437.

[77] J. Eickholt, J. L. Weible, and S. D. Teasley, "Student-facing learning analytics dashboard: Profiles of student use," in *2022 IEEE Frontiers in Education Conference (FIE)*. IEEE, 2022, pp. 1–9.

[78] K. Oliver-Quelennec, F. Bouchet, T. Carron, K. Fronton Casalino, and C. Pinçon, "Adapting learning analytics dashboards by and for university students," in *European Conference on Technology Enhanced Learning.* Springer, 2022, pp. 299–309.

[79] O. Karademir, A. Ahmad, J. Schneider, D. Di Mitri, I. Jivet, and H. Drachsler, "Designing the learning analytics cockpit-a dashboard that enables interventions," in *Methodologies and Intelligent Systems for Technology Enhanced Learning, 11th International Conference 11.* Springer, 2022, pp. 95–104.

[80] G. Ramaswami, T. Susnjak, and A. Mathrani, "Effectiveness of a learning analytics dashboard for increasing student engagement levels," *Journal of Learning Analytics*, vol. 10, no. 3, pp. 115–134, 2023.

[81] J. Hellings and C. Haelermans, "The effect of providing learning analytics on student behaviour and performance in programming: a randomised controlled experiment," *Higher Education*, vol. 83, no. 1, pp. 1–18, 2022.

[82] I. Sapsai, Y. P. Valencia Usme, and J. Abke, "Learning analytics dashboard for educators: Proposed project to design with pedagogical background," in *Proceedings of the 5th European Conference on Software Engineering Education*, 2023, pp. 38–47.

[83] I. Rets, C. Herodotou, V. Bayer, M. Hlosta, and B. Rienties, "Exploring critical factors of the perceived usefulness of a learning analytics dashboard for distance university students," *International Journal of Educational Technology in Higher Education*, vol. 18, pp. 1–23, 2021.

[84] J. Zheng, L. Huang, S. Li, S. P. Lajoie, Y. Chen, and C. E. Hmelo-Silver, "Self-regulation and emotion matter: A case study of instructor interactions with a learning analytics dashboard," *Computers & Education*, vol. 161, p. 104061, 2021.

[85] E. Panadero, "A review of self-regulated learning: Six models and four directions for research," *Frontiers in psychology*, vol. 8, p. 422, 2017.

[86] J. Suls and L. Wheeler, "Social comparison theory," *Handbook of theories of social psychology*, vol. 1, pp. 460–482, 2012.

[87] J. Sweller, "Cognitive load theory and educational technology," *Educational technology research and development*, vol. 68, no. 1, pp. 1–16, 2020.

[88] A. A. Lipnevich and E. Panadero, "A review of feedback models and theories: Descriptions, definitions, and conclusions," in *Frontiers in Education*, vol. 6. Frontiers, 2021, p. 720195.

[89] M. A. Qureshi, A. Khaskheli, J. A. Qureshi, S. A. Raza, and S. Q. Yousufi, "Factors affecting students' learning performance through collaborative learning and engagement," *Interactive Learning Environments*, vol. 31, no. 4, pp. 2371–2391, 2023.

[90] B. Brown, W. T. Gude, T. Blakeman, S. N. van der Veer, N. Ivers, J. J. Francis, F. Lorencatto, J. Presseau, N. Peek, and G. Daker-White, "Clinical performance feedback intervention theory (cp-fit): a new theory for designing, implementing, and evaluating feedback in health care based on a systematic review and meta-synthesis of qualitative research," *Implementation Science*, vol. 14, pp. 1–25, 2019.

[91] K. Till, B. Muir, A. Abraham, D. Piggott, and J. Tee, "A framework for decision-making within strength and conditioning coaching," *Strength & Conditioning Journal*, vol. 41, no. 1, pp. 14–26, 2019.

[92] N. Bergdahl, J. Nouri, T. Karunaratne, M. Afzaal, and M. Saqr, "Learning analytics for blended learning: A systematic review of theory, methodology, and ethical considerations," *International Journal of Learning Analytics and Artificial Intelligence for Education (iJAI)*, vol. 2, no. 2, pp. 46–79, 2020. [Online]. Available: https://doi.org/10.3991/ijai.v2i2.17887

[93] I. Rets, C. Herodotou, and A. Gillespie, "Six practical recommendations enabling ethical use of predictive learning analytics in distance education," *Journal of Learning Analytics*, vol. 10, pp. 1–19, 03 2023.

[94] K. Holloway, "Big data and learning analytics in higher education: Legal and ethical considerations," *Journal of Electronic Resources Librarianship*, vol. 32, no. 4, pp. 276–285, 2020.

[95] M. Vivitsou and M. Saadatmand, "Privacy in the era of big data and learning analytics: ethical considerations and positions," 2010.

[96] J. Okkonen, T. Helle, and H. Lindsten, "Ethical considerations on using learning analytics in finnish higher education," in *Advances in Human Factors in Training, Education, and Learning Sciences: Proceedings of the AHFE 2020 Virtual Conference on Human Factors in Training, Education, and Learning Sciences, July 16-20, 2020, USA*. Springer, 2020, pp. 77–85.

[97] M. Patrizi, M. Vernuccio, A. Pastore *et al.*, ""ok, google!" are my data safe? the mediated effect of perceived privacy risk on brand trust," in *Proceedings of 20th International Marketing Trends Conference 2021 Venice, ed. Jean-Claude Andreani and Umberto Collesei, Paris-Venice Marketing Trends Association*, 2021, pp. 1–5.

[98] B. Rognoni, "Learning analytics as a tool for exploring student learning patterns," Ph.D. dissertation, University of Southern Queensland, 2017.

[99] L. Thammabanvong, H. Yaoui, V. Rajpho, D. Sinthavong, and M. Choummanivong, “Education technology: Center of excellent for integration mysql course to university of health sciences students,” in *2022 14th International Conference on Software, Knowledge, Information Management and Applications (SKIMA).* IEEE, 2022, pp. 39–41.

[100] M. Jannah, J. Rosalina, and E. Yubarda, “Design of waste treatment applications at pt. rifansi dwi putra using visual basic. net and mysql database,” *Knowbase: International Journal of Knowledge in Database*, vol. 1, no. 1, pp. 51–57, 2021.

[101] R. Deari, X. Zenuni, J. Ajdari, F. Ismaili, and B. Raufi, “Analysis and comparison of document-based databases with sql relational databases: Mongodb vs mysql,” in *Proceedings of the International Conference on Information Technologies (InfoTech 2018)*, 2018, pp. 1–10.

[102] T. Capris, P. Melo, N. M. Garcia, I. M. Pires, and E. Zdravevski, “Comparison of sql and nosql databases with different workloads: Mongodb vs mysql evaluation,” in *2022 International Conference on Data Analytics for Business and Industry (ICDABI).* IEEE, 2022, pp. 214–218.

[103] V. A. Latypova, “Preparation of publication data for mining in scientific review process management,” in *2024 IEEE 25th International Conference of Young Professionals in Electron Devices and Materials (EDM).* IEEE, 2024, pp. 2620–2623.

[104] B. Johnson, *Visual Studio Code.* Wiley, 2019.

[105] J. Tan, Y. Chen, and S. Jiao, “Visual studio code in introductory computer science course: An experience report,” *arXiv preprint arXiv:2303.10174*, 2023.

[106] R. Durelli, V. Durelli, R. Bettio, D. Dias, and A. Goldman, “Divinator: A visual studio code extension to source code summarization,” in *Proceedings of the 10th Workshop on Software Visualization, Evolution and Maintenance.* Porto Alegre, RS, Brasil: SBC, 2022, pp. 1–5. [Online]. Available: https://sol.sbc.org.br/index.php/vem/article/view/22320

[107] W. McKinney, *Python for Data Analysis: Data Wrangling with Pandas, NumPy, and IPython.* O’Reilly Media, 2017. [Online]. Available: https://books.google.de/books?id=BCc3DwAAQBAJ

[108] F. Moreno-Pino, E. Sükei, P. M. Olmos, and A. Artés-Rodríguez, “Pyhhmm: A python library for heterogeneous hidden markov models,” *arXiv preprint arXiv:2201.06968*, 2022.

[109] M. Copperwaite and C. Leifer, *Learning flask framework.* Packt Publishing Ltd, 2015.

[110] M. L. Waskom, "Seaborn: statistical data visualization," *Journal of Open Source Software*, vol. 6, no. 60, p. 3021, 2021.

[111] I. P. Kaur, "Technologies being used in web development," *International Journal of Engineering Research in Computer Science and Engineering*, 2022. [Online]. Available: https://ijercse.com/viewabstract.php?id=15848&volume=Volume9&issue=Issue4

[112] M. Grinberg, *Flask Web Development: Developing Web Applications with Python*, 2nd ed. O'Reilly Media, 2018.

[113] S. Pandey, "Integration of ai in web development for online pharmacy store: An effective approach," *INTERANTIONAL JOURNAL OF SCIENTIFIC RESEARCH IN ENGINEERING AND MANAGEMENT*, 2024. [Online]. Available: https://api.semanticscholar.org/CorpusID:269590940

[114] O. A. Mohamad, "New virtual environment based on python programming," *Iraqi Journal For Computer Science and Mathematics*, vol. 3, no. 2, pp. 111–118, 2022.

[115] S. R. Ellis, "What are virtual environments?" *IEEE Computer Graphics and Applications*, vol. 14, no. 1, pp. 17–22, 1994.

[116] B. Chinthanet, B. Reid, C. Treude, M. Wagner, R. G. Kula, T. Ishio, and K. Matsumoto, "What makes a good Node.js package? investigating users, contributors, and runnability," *arXiv preprint arXiv:2106.12239*, 2021.

[117] A. L. S. Pupo, J. Nicolay, and E. G. Boix, "Brigadier: A datalog-based iast framework for node. js applications," in *2023 IEEE International Conference on Software Analysis, Evolution and Reengineering (SANER)*. IEEE, 2023, pp. 509–521.

[118] A. Hafner, A. Mur, and J. Bernard, "Node package manager's dependency network robustness," *arXiv preprint arXiv:2110.11695*, 2021.

[119] D. Moreno-Lumbreras, J. M. González-Barahona, and M. Lanza, "Understanding the npm dependencies ecosystem of a project using virtual reality," in *2023 IEEE Working Conference on Software Visualization (VISSOFT)*. IEEE, 2023, pp. 84–94.

[120] A. Javan Jafari, D. E. Costa, E. Shihab, and R. Abdalkareem, "Dependency update strategies and package characteristics," *ACM Transactions on Software Engineering and Methodology*, vol. 32, no. 6, pp. 1–29, 2023.

[121] T. V. Lakshmi and L. Rakshitha, "The power of react js for business applications," *International Research Journal on Advanced Engineering and Management (IRJAEM)*, vol. 2, no. 05, pp. 1637–1639, 2024.

[122] R. Kelapure, "Canteen management system," *Gurukul International Multidisciplinary Research Journal*, vol. 12, pp. 493–496, 2024. [Online]. Available: https://gurukuljournal.com/canteen-management-system-the-final-project/

[123] C. A. Trianti, B. Kristianto *et al.*, "Integration of flask and python on the face recognition based attendance system," in *2021 2nd International Conference on Innovative and Creative Information Technology (ICITech)*. IEEE, 2021, pp. 164–168.

[124] Y. Li, "The development and application of the second classroom management system," in *Proceedings of the 4th International Conference on Modern Educational Technology*, 2022, pp. 1–6.

8 Appendix

The following appendix contains screenshots of the ARC Tutoring Workbench, showcasing both student and tutor views. These images demonstrate the functionality and interface of the dashboard, highlighting its key features such as Learning Analytics (LA) modules, self-tests, topic recommender, and time management tools.

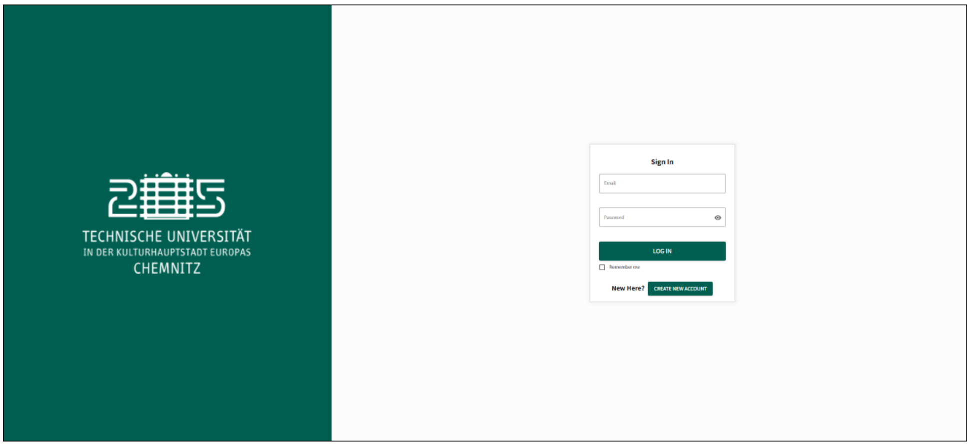

Figure 8.1: Login page of the ARC Tutoring Workbench.

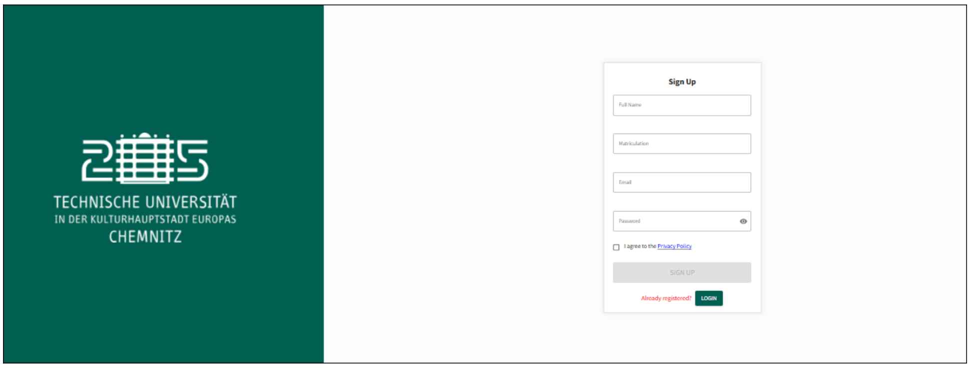

Figure 8.2: Registration page of the ARC Tutoring Workbench.

Login and Registration Pages: The login and the registration pages as shown in figures 8.1, 8.2 allows students and tutors to securely access their personalized dashboard by entering their credentials. During the registration, the users are provided with the privacy policy and only after accepting it they will be navigated to the

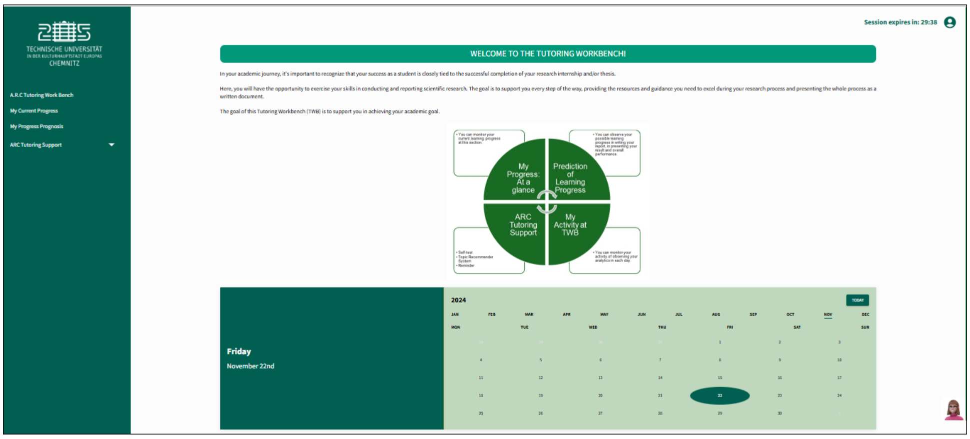

Figure 8.3: Home page after logging-in to the dashboard.

home page. After logging-into the dashboard students and teachers will be navigated to the same home page as shown in the figure 8.3. There they can use the chatbot functionality for their needs.

8.1 Student View

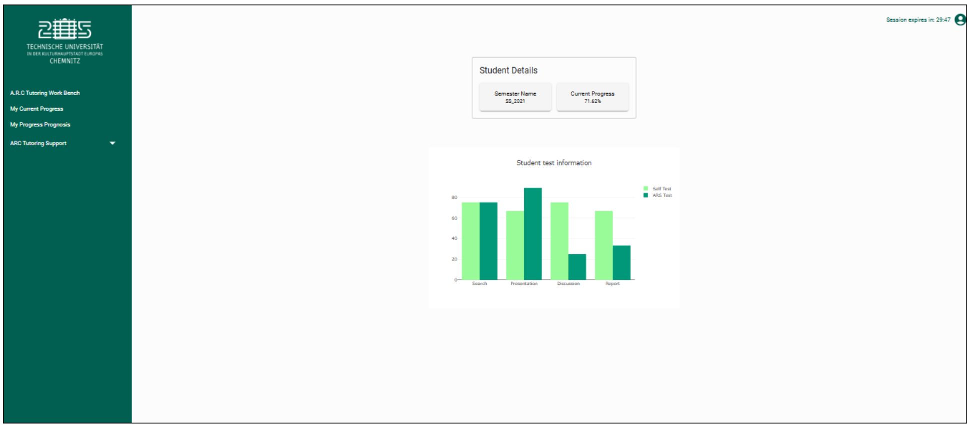

Figure 8.4: Student dashboard showing Descriptive LA page.

Descriptive and Diagnostic LA Pages: These pages as shown in the figures 8.4, 8.5 offer visualizations of student performance across assessments, enabling them to track progress and identify areas for improvement.

Predictive LA Page: As shown in figure 8.6 uses predictive models to forecast future performance based on current data, helping students plan their studies effectively.

Self-Test Page: The self-test module as shown in figure 8.7 allows students to assess their knowledge on specific topics, providing immediate feedback to identify

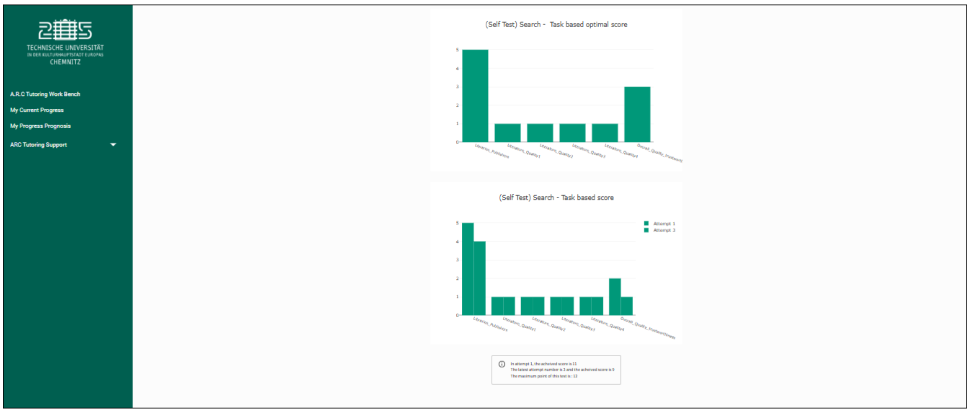

Figure 8.5: Student dashboard showing Diagnostic LA page.

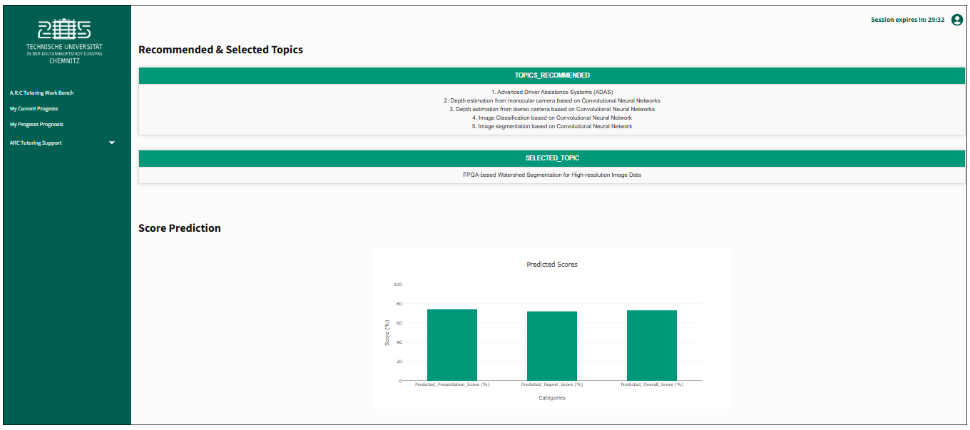

Figure 8.6: Predictive LA page of the student dashboard.

gaps.

Recommendation on Research Focus: This feature as shown in the the figure 8.8 provides personalized recommendations for research topics based on the student's interests and performance.

Time Management Tool: A dedicated tool as shown in figure 8.9, helps students and tutors organize tasks, set milestones, and manage deadlines efficiently.

8.2 Tutor View

The below mentioned pages are different for the student and the tutor.

Descriptive and Diagnostic LA Pages: As shown in the figures 8.10, 8.11 these pages allow tutors to monitor student progress across various assessments and identify at-risk students.

Predictive LA Page: Tutors can use predictive analytics to forecast student outcomes, enabling early interventions for under-performing students. The same is

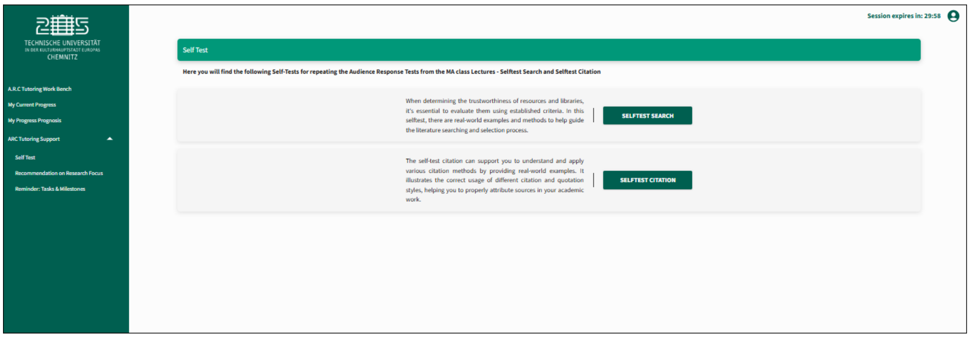

Figure 8.7: Self Test page in the student dashboard.

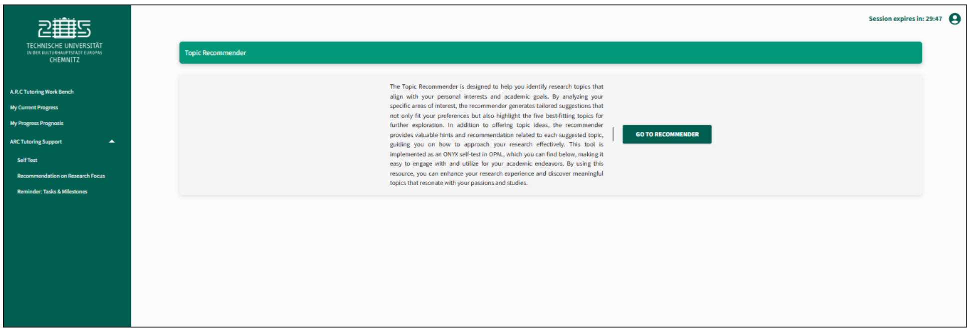

Figure 8.8: Recommendation on Research focus page of student dashboard.

shown in the figure 8.12. Topic Recommender Page: Tutors can view the list of topics recommended to the students. The same is shown in figure 8.13

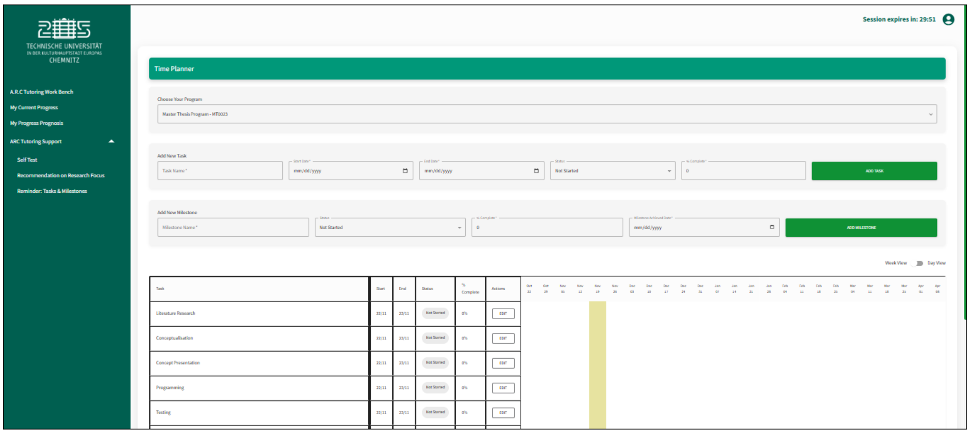

Figure 8.9: Time Management tool page of the student dashboard.

Figure 8.10: Descriptive LA page of the tutor dashboard.

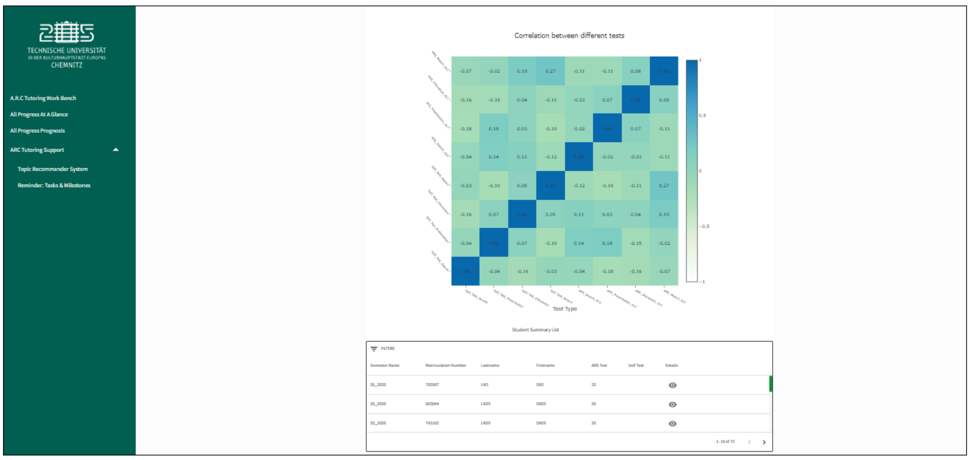

Figure 8.11: Diagnostic LA page of the tutor dashboard.

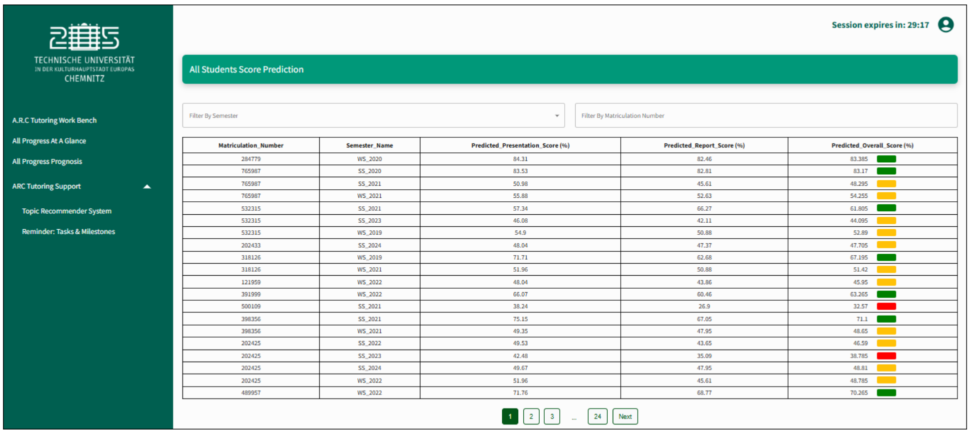

Figure 8.12: Predictive LA page of the tutor dashboard.

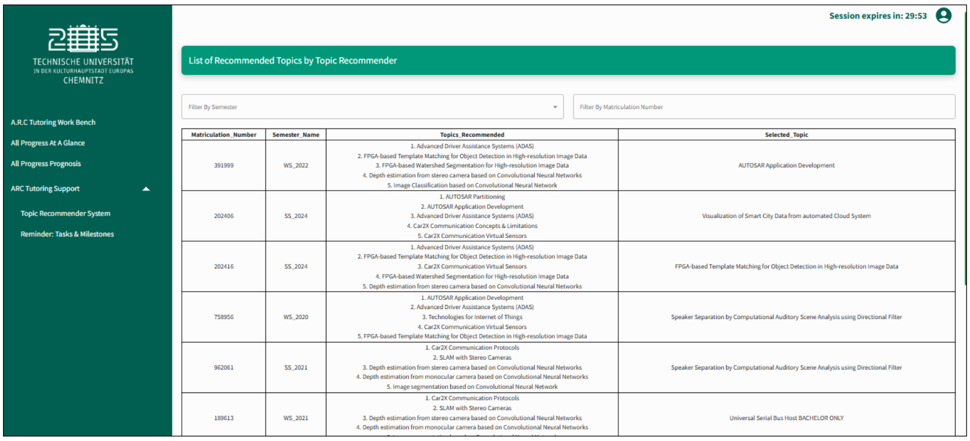

Figure 8.13: Topic Recommender page of the tutor dashboard.

Chemnitzer Informatik-Berichte

In der Reihe der Chemnitzer Informatik-Berichte sind folgende Berichte erschienen:

CSR-22-01 Henrik Zant, Reda Harradi, Wolfram Hardt, Expert System-based Embedded Software Module and Ruleset for Adaptive Flight Missions, September 2022, Chemnitz

CSR-23-01 Stephan Lede, René Schmidt, Wolfram Hardt, Analyse des Ressourcenverbrauchs von Deep Learning Methoden zur Einschlagslokalisierung auf eingebetteten Systemen, Januar 2023, Chemnitz

CSR-23-02 André Böhle, René Schmidt, Wolfram Hardt, Schnittstelle zur Datenakquise von Daten des Lernmanagementsystems unter Berücksichtigung bestehender Datenschutzrichtlinien, Januar 2023, Chemnitz

CSR-23-03 Falk Zaumseil, Sabrina, Bräuer, Thomas L. Milani, Guido Brunnett, Gender Dissimilarities in Body Gait Kinematics at Different Speeds, März 2023, Chemnitz

CSR-23-04 Tom Uhlmann, Sabrina Bräuer, Falk Zaumseil, Guido Brunnett, A Novel Inexpensive Camera-based Photoelectric Barrier System for Accurate Flying Sprint Time Measurement, März 2023, Chemnitz

CSR-23-05 Samer Salamah, Guido Brunnett, Sabrina Bräuer, Tom Uhlmann, Oliver Rehren, Katharina Jahn, Thomas L. Milani, Güunter Daniel Rey, NaturalWalk: An Anatomy-based Synthesizer for Human Walking Motions, März 2023, Chemnitz

CSR-24-01 Seyhmus Akaslan, Ariane Heller, Wolfram Hardt, Hardware-Supported Test Environment Analysis for CAN Message Communication, Juni 2024, Chemnitz

CSR-24-02 S. M. Rizwanur Rahman, Wolfram Hardt, Image Classification for Drone Propeller Inspection using Deep Learning, August 2024, Chemnitz

CSR-24-03 Sebastian Pettke, Wolfram Hardt, Ariane Heller, Comparison of maximum weight clique algorithms, August 2024, Chemnitz

CSR-24-04 Md Shoriful Islam, Ummay Ubaida Shegupta, Wolfram Hardt, Design and Development of a Predictive Learning Analytics System, August 2024, Chemnitz

CSR-24-05 Sopuluchukwu Divine Obi, Ummay Ubaida Shegupta, Wolfram Hardt, Development of a Frontend for Agents in a Virtual Tutoring System, August 2024, Chemnitz

Chemnitzer Informatik-Berichte

CSR-24-06 Saddaf Afrin Khan, Ummay Ubaida Shegupta, Wolfram Hardt, Design and Development of a Diagnostic Learning Analytics System, August 2024, Chemnitz

CSR-24-07 Túlio Gomes Pereira, Wolfram Hardt, Ariane Heller, Development of a Material Classification Model for Multispectral LiDAR Data, August 2024, Chemnitz

CSR-24-08 Sumanth Anugandula, Ummay Ubaida Shegupta, Wolfram Hardt, Design and Development of a Virtual Agent for Interactive Learning Scenarios, September 2024, Chemnitz

CSR-25-01 Md. Ali Awlad, Hasan Saadi Jaber Aljzaere, Wolfram Hardt, AUTOSAR Software Component for Atomic Straight Driving Patterns, März 2025, Chemnitz

CSR-25-02 Billava Vasantha Monisha, Hasan Saadi Jaber Aljzaere, Wolfram Hardt, Automotive Software Component for QT Based Car Status Visualization, März 2025, Chemnitz

CSR-25-03 Zahra Khadivi, Batbayar Battseren, Wolfram Hardt, Acoustic-Based MAV Propeller Inspection, Mai 2025, Chemnitz

CSR-25-04 Tripti Kumari Shukla, Ummay Ubaida Shegupta, Wolfram Hardt, Time Management Tool Development to Support Self-regulated Learning, August 2025, Chemnitz

CSR-25-05 Ambu Babu, Ummay Ubaida Shegupta, Wolfram Hardt, Development of a Retrieval Model based Backend of a Tutoring Agent, August 2025, Chemnitz

CSR-25-06 Shahid Ismail, Ummay Ubaida Shegupta, Wolfram Hardt, Development of a Generative Model based Backend of Tutoring Agent, August 2025, Chemnitz

CSR-25-07 Chaitanya Sravanthi Akula, Ummay Ubaida Shegupta, Wolfram Hardt, Integration of Learning Analytics into the ARC-Tutoring Workbench, August 2025, Chemnitz

Chemnitzer Informatik-Berichte

ISSN 0947-5125

Herausgeber: Fakultät für Informatik, TU Chemnitz
Straße der Nationen 62, D-09111 Chemnitz