NATIONAL UNIVERSITY
COLLEGE OF COMPUTING AND
INFORMATION TECHNOLOGIES

DEVELOPMENT OF AN INSIGHT-DRIVEN ONLINE ASSESSMENT PLATFORM UTILIZING DATA ANALYTICS

A Thesis Submitted in Partial Fulfillment of the Requirements

for the degree of Bachelor of Science in Computer Science

with specialization in Machine Learning

Presented by:

Francheska Leonida Olympia

Francis Stephen Regidor Lee

Ian Jay Duazo Yuson

John Victor Gapoy Corpus

Marchuzs Deimar Vicuña Cortez

Adviser: Prof. Armida P. Salazar

Professor: Prof. Susan S. Caluya, MSc

NATIONAL UNIVERSITY
COLLEGE OF COMPUTING AND
INFORMATION TECHNOLOGIES

APPROVAL PAGE

This thesis entitled **"DEVELOPMENT OF AN INSIGHT-DRIVEN ONLINE ASSESSMENT PLATFORM UTILIZING DATA ANALYTICS"** prepared and submitted by **FRANCHESKA LEONIDA OLYMPIA, FRANCIS STEPHEN REGIDOR LEE, IAN JAY DUAZO YUSON, JOHN VICTOR GAPOY CORPUS,** and **MARCHUZS DEIMAR VICUÑA CORTEZ** in partial fulfillment of the requirements for the degree of BACHELOR OF SCIENCE IN COMPUTER SCIENCE with specialization in Machine Learning has been examined, accepted and recommended for oral examination

Armida P. Salazar
Adviser

PANEL OF EXAMINERS

Approved by the committee on oral examination with a grade of ______ on _________.

Susan S. Caluya
Panel Chair

Jensen A. Santillan
Panel Member

Emmanuel A. Ramirez
Panel Member

Eliseo Q. Ramirez
CS Chair

Emeliza R. Yabut
OIC Dean, CCIT

ACKNOWLEDGMENT

We would like to express our deepest gratitude to our thesis advisor, Professor Armida P. Salazar for the continuous support, guidance, and patience throughout the entire research process. Your valuable insights and constructive feedback have greatly contributed to the refinement of this work. We are also thankful for the assistance provided by Professors Elizer Ponio Jr. and Mico Magtira. Their expertise and thoughtful suggestions significantly enhanced the quality of this thesis. We would also like to thank our thesis professor, Professor Susan S. Caluya, for the guidance provided in completing this thesis. The researchers would also like to thank the panel of this research committee for their insightful comments and revisions during the presentation. And lastly special thanks to our family for their unwavering encouragement and understanding during the demanding periods of this academic pursuit. Their belief in our abilities have been a constant source of inspiration and moral support. Their encouragement and camaraderie made this journey more enjoyable.

TABLE OF CONTENTS

LIST OF TABLES

Page

LIST OF FIGURES

Page

ABSTRACT

This study explores the integration of artificial intelligence (AI) and learning analytics into an online assessment platform designed for first-year computer science students. The platform leverages a fine-tuned GPT-4o Mini model, combined with static analysis tools (PMD and Javac), to provide real-time, personalized feedback on Java programming assessments. A structured JSONL dataset was developed to enhance the AI's ability to evaluate student code submissions, ensuring accurate and constructive feedback.

The model's performance was assessed using BLEU, ROUGE-L, CodeBERT, and GLEU metrics, demonstrating that fine-tuning improved feedback specificity without significantly altering response structure. Additionally, static analysis and compilation checks ensured that students received well-rounded evaluations covering syntax, logic, and coding best practices.

A user evaluation survey, conducted via Microsoft Forms, revealed high satisfaction with the platform's usability and effectiveness. The navigation system (4.54) and readability (4.52) received the highest ratings, while AI-generated feedback and score distribution charts were also rated positively (4.36). Educators particularly benefited from the teacher dashboard (4.45), which provided actionable insights into student progress.

The findings confirm that AI-powered assessment tools can enhance programming education by offering instant, structured feedback while reducing the burden on educators. Although the study does not directly measure learning outcomes, its results highlight the potential of AI-driven assessment systems in fostering self-directed learning. Future research should explore long-term impacts on student performance, refine visual analytics, and further enhance AI-driven feedback mechanisms.

Keywords: Artificial Intelligence, Learning Analytics, GPT-4o Mini, AI-Powered Assessment, Programming Education, Static Code Analysis, Java Compiler, PMD, Javac, Fine-Tuning

CHAPTER I: INTRODUCTION

Introduction

Digital technologies have brought changes to the nature and scope of education and led education systems worldwide to adopt strategies and policies for ICT integration. The latter brought about issues regarding the quality of teaching and learning with ICTs, especially concerning the understanding, adaptation, and design of the education systems in accordance with current technological trends (Timotheou et al., 2022). The greatest technological changes in our lives are predicted to be brought about by Artificial Intelligence (AI) (Păvăloaia & Necula, 2023). Artificial intelligence (AI) is an evolving set of technologies used for solving a wide range of applied issues. The core of AI is machine learning (ML)—a complex of algorithms and methods that address the problems of classification, clustering, and forecasting (Mukhamediev et al., 2022). These technologies that are showing a powerful impact are nothing without a certain factor, and it is data. Data serves as fuel on making most modern technologies, making technology evolve into a powerful and more intelligent tool.

Advances in artificial intelligence (AI), including intelligent machines, are opening new possibilities to support teaching and learning in higher education. These technologies can enhance personalized learning experiences by adapting educational content to meet individual student needs, thereby fostering greater engagement and understanding (Wang et al., 2019). Large Language Models (LLM), a type of AI that focuses on text generation, provides a new era of possibilities in the realm of education providing personalized feedback, assisting with automated grading (Wang et al., 2024) real-time insights and support, helping learners grasp complex concepts at their own pace (Aïmeur et al., 2019). Furthermore, AI can assist educators in administrative tasks, allowing them to devote more time to interaction with students and improving overall educational outcomes (Luckin et al., 2016). By analyzing large datasets, AI can also provide insights into student performance and learning patterns, enabling institutions to tailor their curricula more effectively (Siemens & Long, 2011). Overall, the integration of AI in higher education creates a new learning environment, introducing students to innovative learning styles and curricula.

In the modern world, where there is an increasing need for high-quality education, higher education has taken on more significance. Higher education organizations are therefore under pressure to deliver a top-notch education and guarantee that their pupils are adequately prepared for their future employment. Analyzing student performance data to identify areas for improvement and develop targeted interventions is a critical component of enhancing student success. (Krishna, 2023). To improve this situation, recent developments in artificial intelligence (AI) that have increased the capabilities of large language models (LLMs) might be helpful, as they have the potential to automatically create individualized feedback on student performance at low cost for the educators (Kasneci et al., 2023)

One of the ways to measure a student's performance is on collection of data based on the process of online assessments. Online assessment is defined as a systematic method of gathering information about a learner and learning processes to draw inferences about the learner's dispositions. Analyzed student answers from online assessments provide opportunities for meaningful feedback and interactive support for learners as well as possible influences on the engagement of learners and learning outcomes (Heil & Ifenthaler, 2023).

Background of the Study

Education plays a fundamental role in the development of individuals and societies, especially in developing countries like the Philippines, where education is viewed as a pathway to success (de Guzman et al., 2017). However, maintaining student engagement and motivation remains a major concern. A study by Espinosa et al. (2023) indicates that academic persistence is often hindered by various personal and external factors, such as social pressure, mental health issues, and financial difficulties, which impact students' academic achievements. Additionally, the shift to online learning has introduced new challenges, including concerns about academic integrity. Cheating in online assessments has become a growing issue, as students find ways to bypass traditional monitoring methods. This raises the need for more effective mechanisms to ensure fairness in online examinations and uphold the credibility of academic evaluations. Specifically, many schools demonstrated a lack of experience and low digital capacity, which resulted in widening gaps, inequalities, and learning losses. In this context, emerging technologies like AI and LLM's offer promising solutions. For example, AI can adapt instructional content based on students' strengths and weaknesses, allowing for a more responsive and customized learning environment. These technologies support the teachers by reducing the time spent on repetitive tasks and by offering additional resources and explanations to students in real time (Smith et al., 2023). The capability of AI to personalize the instructions enhances the engagement of students and would provide helping factors to academic success by addressing their learning gaps more effectively

However, the transition to digital learning has raised significant concerns about academic integrity, particularly in online assessments. Leslie Ching Ow and HeeJeong

Jasmine Lee (2021) emphasized that remote examinations create new opportunities for misconduct, as students can take advantage of digital tools to engage in dishonest practices. Their study introduced an AI-driven monitoring system that tracks student behavior, such as tab switching and response irregularities, to detect potential cheating. Similarly, Ali M. Duhaim et al. (2022) proposed a multi-layered approach to identifying academic dishonesty by analyzing IP addresses, exam completion times, and answer similarities. Their findings revealed that monitoring response patterns is one of the most effective ways to detect collusion among students. These studies highlight the growing need for AI-powered cheating detection mechanisms to uphold the fairness and credibility of online assessments.

Large Language Models (LLMs) present an opportunity to automate and promote equity in learning assessments, providing rapid valuable feedback to students while reducing the burden on instructors. (Matelsky et al., 2023). In coding evaluation, LLMs trained on text and code have the potential to power next-generation AI-driven educational technologies and drastically improve the landscape of computing evaluation (Phung et al., 2023) By analyzing the submitted code of students, detect errors, and provide feedback that helps learners to provide guidance into a better understanding of their mistakes in the given problem.

One of the foundations of enhancing the effectiveness of AI and LLM is data analysis. One of the foundations of enhancing the effectiveness of AI and LLM is data analysis. Data analysis has become an essential tool in education, enabling institutions to derive meaningful insights from vast amounts of data generated through various educational activities (Job, 2018)., with the help of system analyzation on student data, educational institutions could identify those students who are at-risk and evaluate the effectiveness of teaching methods (Feguson, 2012).

Furthermore, the introduction of learning analytics tools has democratized access to data insights, empowering both professors and students. Learning Analytics has found its niche in enhancing decision-making by providing insights through student learning and engagement data. More specifically, Learning Analytics Tools (LATs) which are distinct applications and software programs are utilized for collecting, analyzing, and reporting this data. These educational tools prove useful for students, teachers, and the overall learning institution. (Mukred et al., 2024).

However, Salido (2023) highlights the limitations of AI data techniques in education, emphasizing the essential role of human educators in providing empathy and contextual understanding. While AI-powered learning tools can personalize learning experiences and enhance student performance, it is crucial to address ethical concerns, such as data privacy and potential biases, as well as infrastructure limitations associated with their implementation. Responsible integration of AI is vital to ensuring fair educational outcomes, as it allows institutions to harness the benefits of technology while safeguarding students' rights. Additionally, ongoing research is necessary to navigate these challenges effectively, ensuring that AI tools are used to complement traditional teaching methods rather than replace the invaluable human element in education. This balanced approach can lead to improved educational practices and better support for both students and educators.

In conclusion, this study will venture into how AI-derived technologies will be transformative through the utilization of LLM fine tuning and learning analytics tool such as an assessment platform in creating a well-specified, in-depth, and attentive insight on the learning struggles of students, with an understanding on the exploration into the multifaceted impact on students and their academic performance as a supplemental intervention for educators. This study will not be beneficial to the students that are struggling within their learning capacities but as well as educators who are adapting to a more structured and specified ways of teaching with a larger scale of students within a class. The insights derived from this study will collectively guide a wider and broader understanding of educators beyond the four-pillars of education and ease the digital gap brought by changes in the education system in the Philippines. The study strives to facilitate a much deeper comprehension on the education implications of AI-derived technologies to make well thought out implications on its usage and integration to supplemental teaching interventions with a collective vision on efficient and effective quality education for all with accessibility throughout generations

Statement of the Problem

The existing assessment systems in educational institutions are limited in providing a comprehensive understanding of student performance, hindering the ability to address learning gaps effectively. Specifically, the problems are as follows:

1. **Lack of Insight into Students Performance**

    - Current assessment system methods do not provide detailed insights into why students fail specific questions, making it difficult to address their learning gaps effectively.

2. **Inability to Identify and Address Student Weaknesses and Strengths**

    - Current assessment systems do not identify individual student's strengths and weaknesses clearly, leaving students unsure about which areas they need to improve.

3. **Overwhelming Teacher Workload**

    - Teachers often have large class sizes, making it difficult for them to give individual attention and detailed feedback to every student.

Objectives of the Study

General Objective

The general objective of the study is to develop an online assessment platform that uses data analysis to offer detailed insights into student performance, provide personalized feedback that highlights individual strengths and weaknesses to improve overall student progress.

Specific Objectives

1. **Develop an online assessment module that analyze student performance using LLM** by designing a system that evaluates student responses in real-time, providing detailed feedback on coding

2. **Fine Tuning of LLM for better performance** by training the chosen API LLM to relevant datasets to assess the students' submissions accurately, enhancing its capability to provide personalized feedback.

3. **Develop a dashboard of students' performance analytics** that visualizes and tracks students' progress over time offering detailed insights into their strengths and weaknesses.

4. **Develop a dashboard for teachers** that provides comprehensive analytics on student performance, enabling educators to monitor progress, identify learning gaps, and assess overall student performance. The dashboard will offer real-time insights and visual reports to help teachers make data-driven decisions and personalize instructions to enhance student learning outcomes.

Scope and Limitations

Scope

The study focuses on the design, development, and implementation of an online assessment platform aimed at improving first year students' performance, specifically in the Introduction to Programming subject. The platform uses model AI-driven analysis, fine-tuning, and static analysis techniques to evaluate students' code-based answers, providing detailed insights into their strengths and weaknesses. Key features include automated grading, AI-powered feedback, and real-time performance tracking, allowing students to quickly identify areas for improvement.

The platform is designed specifically for programming-related assessments, covering seven key topics: Variables and Types, Classes and Objects, Operators, Methods, Variable Scope, Sequential Structure, Conditional Structure, Iteration Structure, Arrays, and Multidimensional Arrays. These topics are grouped into three major categories:

- Fundamentals of Programming (variables and types, classes and objects, operators, methods, and variable scope).
- Control Structures (sequential, conditional, and iteration structures).
- Arrays (one-dimensional and multidimensional arrays).

The platform only supports coding-based assessments and does not include multiple-choice or theoretical questions. To measure students' programming proficiency, it uses a metrics-based grading system, calculating scores for each category and generating an overall programming skills grade to help students and teachers track progress.

For data management, the platform uses Supabase as its backend database to store students' answers, questions, and feedback. This ensures real-time data retrieval and automatic saving of student responses when they submit their assessments, preventing data loss.

To support programming-based assessments, the platform will feature a Java textbox with functionalities like a coding environment, enabling students to write and test their Java code efficiently. However, not all functionalities of a full coding environment will be available, as the textbox is designed primarily for assessment purposes rather than full-scale development.

The platform will implement a user-friendly design, ensuring accessibility and ease of use. It will also support both dark mode and light mode to enhance usability based on user preference.

Additionally, the platform supports teachers by automatically checking and grading assessments, generating detailed performance reports, and reducing their workload. A student dashboard will also be integrated into the platform, providing real-time insights into performance metrics through interactive dashboards.

Furthermore, the platform will incorporate cheating detection mechanisms, where actions such as copying questions and frequently switching tabs can serve as warning signs of misconduct, ensuring a fair and secure assessment environment.

Limitations

Privacy and ethical considerations remain fundamental to the platform's analysis processes, ensuring adherence to relevant regulations and safeguarding student confidentiality. Although the platform does not collect or store sensitive personal data, including grades or identifiable information, maintaining ethical standards in data handling remains a priority. All coding submissions are anonymized, and data usage is strictly limited to research and performance evaluation purposes, minimizing privacy concerns.

Additionally, the platform primarily focuses on qualitative assessments, specifically targeting coding problems related to the subject of Introduction to Programming. The questions are designed around programming tasks rather than essays or theoretical questions. The data used for these assessments is curated, sourced from publicly available resources or websites to ensure privacy and ethical standards. While it can effectively identify areas for improvement and provide targeted, constructive feedback, the system is not designed to offer direct instructional support or tutoring. Instead, it highlights focus areas for students without delivering comprehensive lessons, leaving instructional responsibility to educators.

Furthermore, the platform’s primary focus is on providing an enriched student dashboard, designed to give in-depth feedback on performance and areas for improvement. Meanwhile, the teacher’s dashboard offers comprehensive data and analytics on student progress and performance, enabling educators to monitor learning trends and identify areas where students may need additional support. However, it lacks advance customization options for deeper analysis or personalized interventions.

To ensure privacy and minimize ethical concerns, researchers utilized synthetic data for training and testing the platform.

Another notable limitation is the platform's reliance on the quality and consistency of assessment data. Accurate insights are highly dependent on the robustness of the dataset used for evaluation. Poorly designed assessments, inconsistently formatted submissions, or insufficient data collection can hinder the platform's ability to generate meaningful feedback. Furthermore, without a structured and continuous data collection process such as regular assessments, analyzing performance trends and identifying common areas of difficulty may be challenging.

A significant limitation of the platform is its dependence on stable internet connectivity. Since the platform relies on API and backend implementation, a stable internet connection is necessary for submitting assessments, retrieving feedback, and performing data analysis. Connectivity issues can lead to delays or disruptions in the platform’s functionality.

Lastly, A potential issue within the platform is the possibility of students copying answers during assessments. While basic academic integrity measures are in place to monitor behaviors like tab-switching and question-copying, students may still attempt to copy answers from external sources. The platform currently lacks advanced mechanisms to fully detect or prevent such behavior, which could impact the fairness and accuracy of the assessments. More sophisticated cheating detection tools, such as analyzing response patterns and completion times, are needed to better identify and mitigate these instances, ensuring the platform's effectiveness in maintaining fairness during assessments.

Definition of Terms

AI-Driven Feedback – Personalized responses and insights generated by artificial intelligence to help students understand their mistakes and improve their performance.

API (Application Programming Interface) – A set of rules that allows different software applications to communicate with each other, used in this platform to integrate AI and database functions.

Arrays and Multidimensional Arrays – Data structures used in programming to store multiple values in a single variable. A multidimensional array is an array containing other arrays, used for organizing complex data.

Artificial Intelligence (AI) – A branch of computer science that enables machines to mimic human intelligence, including problem-solving, learning, and decision-making.

Automated Grading – The process of evaluating and scoring student assessments using algorithms instead of manual checking by instructors.

Backend – The underlying system that processes and manages data, handling requests from the frontend.

Backend Database – A system that stores and organizes data, allowing applications to retrieve and manipulate information as needed.

Cheating Detection Mechanisms – Technologies and rules designed to monitor student behavior during assessments to prevent academic dishonesty, such as copying answers or using unauthorized resources.

Code Efficiency – A measure of how well a program performs in terms of speed and resource usage, often optimized by reducing unnecessary computations or improving algorithms.

Control Structures – Programming constructs that control the flow of execution in a program, such as loops (for, while) and conditional statements (if-else, switch).

Dark Mode and Light Mode – Two visual themes available in applications to adjust screen brightness and contrast, improving user experience based on preference.

Data Retrieval – The process of fetching stored information from a database or system, often used to display past assessment records or student performance reports.

Fine-Tuning – A process of training an AI model on a specific dataset to improve its accuracy and relevance for a particular task, such as analyzing student code and generating feedback.

Frontend – The part of a web application that users interact with, such as buttons, forms, and dashboards.

JSONL Format (JSON Lines) – A structured data format used for training AI models, where each line contains a separate JSON object, making it efficient for machine learning applications.

Language Model (LLM) – A type of artificial intelligence model designed to process and generate human-like text, used in this platform to analyze student code and provide feedback.

Logical Errors – Mistakes in program logic that cause incorrect outputs despite having correct syntax, such as using the wrong mathematical operator or condition.

Metrics-Based Grading System – A scoring method that evaluates student performance based on predefined criteria, such as correctness, efficiency, and logical structure in programming assessments.

Personalized Learning Insights – Tailored performance reports generated by the system to help students understand their strengths and areas for improvement.

Performance Analytics Dashboard – A visual interface that displays key metrics and trends related to a student's performance, helping both students and instructors track progress over time.

Programming Fundamentals – The basic concepts and principles of programming, including variables, data types, loops, and functions.

Real-Time Feedback – Instant analysis and suggestions provided to students upon submitting their work, helping them identify mistakes and improve their learning process.

Scalability – The ability of a system to handle an increasing number of users or data efficiently without performance issues.

Static Analysis – A method of examining program code without executing it to detect errors, inefficiencies, or security vulnerabilities.

Student Progress Tracking – The continuous monitoring and recording of student performance to provide insights into their learning journey.

Supabase – An open-source backend platform used for managing databases, authentication, and real-time data updates in web applications.

Syntax Errors – Mistakes in the structure of programming code that prevent it from being compiled or executed correctly, such as missing semicolons or incorrect variable declarations.

Tab-Switching Detection – A mechanism that monitors whether a student frequently switches between different browser tabs during an assessment, which may indicate potential cheating.

User Interface (UI) – The visual layout and interactive components of an application that users interact with, such as buttons, menus, and input fields.

Significance of the Study

The significance of this study is highlighted from three perspectives: Theoretical Significance, Practical Significance, and Future Research.

Theoretical Significance

This study contributes to the academic discourse on AI-driven educational tools, particularly in the domain of programming assessments. By leveraging Large Language Models (LLMs) and fine-tuning techniques, the study highlights the potential of AI in real-time evaluation of student coding performance. It adds to the growing body of knowledge on AI-enhanced education, showcasing how AI-powered analysis can offer personalized feedback and identify patterns in student learning. Additionally, the research underscores the role of AI in improving coding assessments, emphasizing the importance of adaptive learning systems in education.

Practical Significance

The findings of this study offer significant practical benefits to educational institutions, instructors, and students. For students, the platform provides real-time, AI-powered feedback that helps them identify their strengths and weaknesses, enabling them to improve their programming skills efficiently. With features such as automated grading and progress tracking, students gain valuable insights into their learning journey, fostering self-improvement and confidence. For educators, the study aids instructors by reducing their workload through automated assessments and performance analytics. The platform's dashboard enables teachers to track student progress, identify common areas of difficulty, and provide targeted support. The inclusion of academic integrity measures further ensures fair assessments, allowing teachers to focus on deeper instructional strategies rather than manual grading. For educational institutions, the platform enhances the effectiveness of programming education by offering scalable and efficient assessment solutions. By integrating AI into educational practices, institutions can streamline coding evaluations, improve the quality of student feedback, and ensure a data-driven approach to skill development in programming.

Future Research

This study opens avenues for further research on AI-driven educational platforms and their long-term impact on student learning. One key area for future exploration is enhancing cheating detection, as while basic academic integrity measures are implemented, future research could develop more sophisticated mechanisms such as behavioral pattern analysis and anomaly detection in coding submissions. Another avenue is expanding subject coverage, as the platform currently focuses on Introduction to Programming, and future studies could explore its adaptation to other subjects, expanding the AI-driven assessment model to a broader range of disciplines. Additionally, advancing AI feedback mechanisms is an important aspect to consider, as further research could investigate improvements in AI-generated feedback, exploring how reinforcement learning techniques or domain-specific fine-tuning can enhance personalized feedback accuracy. Lastly, investigating long-term learning outcomes is crucial, as future research could assess the long-term impact of AI-driven assessments on student performance and retention, evaluating how personalized feedback influences student learning over extended periods. By addressing these areas, future research can build upon this study's findings, further advancing AI-driven educational assessment tools and enhancing their effectiveness in programming education

CHAPTER II: RELATED LITERATURE AND STUDIES

Introduction

The integration of Artificial Intelligence (AI) and Machine Learning (ML) in education has significantly transformed teaching methodologies, student assessments, and learning experiences. As digital technologies continue to evolve, AI-driven tools such as adaptive learning systems, feedback mechanisms, and automated grading have gained traction in both global and local educational contexts. These advancements have paved the way for personalized learning, real-time performance analysis, and enhanced academic support for students.

This chapter presents a comprehensive review of existing literature and studies related to AI applications in education, particularly in the Philippine setting. It explores key areas such as AI-driven feedback generation, the role of large language models (LLMs) in student assessments, and the use of static analysis tools in programming education. Additionally, it examines learning analytics, online assessment challenges, and backend technologies like Node.js and Supabase, which support modern educational platforms.

By analyzing previous research, this chapter aims to establish a foundation for understanding the potential and limitations of AI in education. The insights gathered will help contextualize the study's objectives and provide a basis for further exploration into the integration of AI-powered feedback mechanisms in student assessment.

Related Literature

The Role of AI and Machine Learning in Education

The future of higher education is intrinsically linked with developments on new technologies and computing capacities of the new intelligent machines (Popenici & Kerr, 2017). Providing innovative solutions for teaching and learning, AI is defined as a computing system that can engage in humanlike processes, such as learning, adapting, synthesizing, and self-correction (Dann et al., 2024). The core functionality of AI is Machine Learning, a core sub-area of artificial intelligence which promotes the reality of being able to give machines the access to data for more ease in human work and just to learn them for themselves (Jagwani & Aloysius, 2019). This connection of Artificial Intelligence and Machine Learning transformed how the data is utilized in the field of education, providing meaningful insights on datasets gained.

Artificial Intelligence Integration in Education in the Philippines

In the Philippine educational context, the integration of AI in the Philippine educational context offers a multitude of opportunities for enhancing the overall learning experience. These AI-driven tools are seen as augmenting educators at large, particularly to better understand students' learning progress and tailor teaching strategies accordingly (Estrellado & Miranda, 2023). While the prospects of AI integration in the Philippine industry are promising it is noteworthy where the progress initially lies, and the deterministic movement goes to the educational context. Still, hanging on the hype are the privacy concerns, data security, and the potential for widening the digital divide which tend to be overlooked and must be initially addressed explicitly. (Holmes et al., 2021, as cited in Estrellado & Miranda, 2023).

Adaptive Learning and Learning Management Systems

As Artificial Intelligence continues to be integrated into educational systems, it denotes the capacity to significantly improve student learning outcomes by providing tailored learning experiences with the help of knowledge tracing (KT) and collaborative filtering. AI technology can be used to customize learning experiences to match and predict specific student needs (Kamalov et al., 2023). One of the most notable applications of AI in the educational field is the Learning Management Systems (LMS). By employing the AI algorithms, learning management systems can analyze student behavior and progress, identify learning patterns, and automatically adapt content and activities to meet the individual needs of each learner (Vergara et al., 2024). This adaptability not only enhances student engagement but also helps educators to respond more effectively to diverse learning needs, providing a path for a data – driven education. One of the data sources to be found in the field of education is assessments. According to Heil and Ifenthaler (2023), online assessment is defined as a systematic method of gathering information about a learner and learning processes to draw inferences about the learner's dispositions. It provides opportunities for meaningful feedback and interactive support for learners, as well as possible influences on the engagement of learners and learning outcomes. Educators can gain valuable insights that support more informed decision – making which results in improved learning strategies and student performance.

Feedback Mechanisms in the Educational Field

The use of forecasting methods serves as a foundation for advancements in the education sector, particularly through the integration of Learning Analytics (LA). As shown by Sharif and Atif (2024), it has significantly transformed feedback mechanisms within the educational institutions. This transformation marks the shift from a broad and irregular form of feedback to a more refined, ongoing, and individualized means of communication, benefiting both students and educators alike. One corresponding learning analytics tool mentioned is the Real - Time Feedback. Sharif and Atif further analyzed that the way time works in feedback has changed in a revolutionary way. Real-time feedback became possible, so students could see right away where they were performing well and where they needed to improve. For example, when using a digital learning module, students could obtain immediate feedback on their answers. One of the advantages of the implementation of Learning Analytics in education is that it can assess teaching quality and identify as well as excavate the endorsement, support and association that are needed to help students in their learning (Viberg et al., 2018, as cited in Mian et al., 2022). With these, educational institutions can strengthen a more responsive learning environment that promotes continuous improvement for both students and instructors.

Large Language Model and Fine Tuning in Feedback Generation

Another vital application of AI in education lies in automated feedback generation. Providing personalized and timely feedback is essential for improving academic performance as well as the growth of the student. However, offering individualized or specific feedback can be challenging for educators, usually when it comes to handling many students in a certain class. LLM, such as OpenAI's GPT series, shows a significant potential in generating human-like, relevant responses. Generating corrective feedback for wrong codes is challenging for open-source code LLMs. These models are incapable of capturing small changes in source codes and providing natural language explanations of the codes (Muennighoff et al., 2023; Miceli Barone et al., 2023). With this, fine - tuning process steps in. It involves training an LLM on a special dataset that reflects the desired outcomes and learning objectives of a specific subjective area. According to Pornprasit and Tantithamthavorn (2024), fine - tuned LLMs significantly improve code review processes by recognizing common student errors, analyzing logic patterns, and offering targeted suggestions for improvement. When aligned with educational rubrics, this allows the automated generation of the personalized feedback more effective in the fields of programming and computer science education.

Static Analysis for Code Checkers

With LLM enhancing feedback processes, improving its effectiveness can be further enhanced by incorporating it with static analytic tools in coding. This plays a crucial role in the programming field of education. Static code analysis facilitates the examination of code for irregularities without program execution, which significantly impacts project quality. Furthermore, tools for static code analysis serve as educational aids, imparting essential lessons on coding practices. Motivated by the growing complexity of software projects and the pivotal role of code quality in academic performance within computing disciplines (Nikolić et al., 2024).

Static code analysis tools, such as Java compiler (Javac) and PMD are used in educational environments to assess code structure, readability, and guidelines. Integrating these tools into LMS, the instructors can automate the evaluation of coding assignments, ensuring that the students follow the proper coding standards. According to NVIDIA Develoeprs (2024), combining static analysis with AI - driven feedback systems enhance the overall criteria of automated evaluations that ensures the students receive the proper feedback in logical and proper styling of their code.

Research by AlOmar et al. (2023) highlights the effectiveness of the integration of PMD into educational courses significantly improving the students' ability to detect and correct code quality issues. According to Kaur and Nayyar (2020), PMD is a tool for statically analyzing the JAVA source code for various programming flaws like unused variables,empty catch blocks, unnecessary object creation among many others. PMD features many built-in checks. Together with PMD, javac also helps with syntax errors.

Incorporating static analysis codes alongside a fine - tuned LLM feedback system creates a robust framework for automated code assessment. While javac serves as an important tool for detecting method structures and compilation of Java code, tools such as PMD focuses on enforcing the coding standards and identifying the structural flaws provides an approach that ensures students receive detailed feedback that is both accurate and aligned with the best programming practice, applying both the conceptual understanding and practical skill development.

Data Analytics Dashboards in Education

The implementation of learning analytics dashboards has been shown to improve both teaching effectiveness and student success rates (Khosravi et al., 2021). A dashboard is one of the essential process components of data analytics. This has become integral in modern educational environments, offering visual representations of key metrics to inform decision-making. Dashboard’s main purpose is to collect, analyze, and visualize the data to provide enhancement on students’ learning outcomes. (Masiello et al., 2024). By providing real-time insights into student performance, attendance, and engagement, dashboards enable educators to identify trends, address issues promptly, and tailor instructional strategies to meet individual needs (Specht, 2024).

Node.js and React for Educational Platforms

Educational platforms during the modern times use a tool with efficient, scalability and real-time capabilities, one of the known website tools that contains these characteristics are Node.js and React. Node.js, a server-side JavaScript runtime, enables asynchronous, event-driven programming, making it ideal for handling multiple student interactions simultaneously (Mardan, 2018). React, a popular front-end library, facilitates the development of interactive and dynamic user interfaces, ensuring a seamless learning experience for students (Boduch & Derks, 2020). Together, these technologies enable the development of highly responsive, data-driven educational applications, integrating seamlessly with backend services like Supabase for real-time updates and secure authentication. Their ability to handle large datasets and enhance user experience makes them invaluable in educational platforms that rely on analytics and interactive content (Specht, 2024).

Supabase as a Backend Solution

Supabase is an open-source alternative to Firebase. It offers a comprehensive backend solution for web applications. Built on PostgreSQL, it provides features such as real-time data synchronization, authentication, and storage, making it a robust choice for developers (Supabase, n.d.). Its open-source nature ensures flexibility and control over application data, which is crucial for educational platforms requiring customization and scalability. Supabase's integration capabilities and ease of use have made it a popular choice among developers aiming to build secure and high-performance backends with minimal configuration (Carnes, 2022).

Related Studies

Fine-Tuning in Large Language Models

The Advancements in large language models particularly GPT along with other LLMs created significant impacts on natural language processing and code generation technologies (Brown et al., 2020; OpenAI, 2023). These models absorb extensive training data before achieving capability to undergo fine-tuning because it improves their ability to execute specific tasks (Wei et al., 2023). Through fine-tuning mechanisms these models acquire specialized skills which enable them to handle complex tasks within specific domains effectively.

The process of fine-tuning is essential for code LLMs to develop greater code generation and debugging along with coding editing abilities. The CoffeePots framework applies fine-tuning techniques to match feedback output with proper code modifications which enhances performance of open-source code LLMs according to Moon et al. (2024). Within this domain practitioners use additional training sessions to enhance pre-trained models through smaller task focused datasets for their ability to solve coding problems and provide correct code corrections.

Enhancing Student Performance through Fine-Tuned LLMs

In recent years, the integration of fine-tuned Large Language Models (LLMs) into educational platforms has shown significant promise in enhancing student performance. A study by Neshaei et al., (2024) explored the application of fine-tuned LLMs (GPT-3) in modeling the performance within tutoring systems and suggested that it can effectively capture complex learning patterns, offering a new avenue of personalized educational insight. Furthermore, a study by Latif and Zhai (2023) states that fine-tuned GPT-3.5 provides an improved assessment accuracy which leads to a more effective in evaluating complex student responses.

These studies collectively show transformative potentials of fine-tuned LLMs used in education. By accurately modeling student performance, providing detailed feedback and enhancing assessments methods, these models contribute to a more specific and effective learning journey.

Cheating Detection in Online Assessments

In addition to these challenges, the rise of online assessments has introduced new concerns regarding academic integrity, prompting the exploration of AI-driven solutions for detecting cheating behaviors. Leslie Ching Ow and HeeJeong Jasmine Lee (2022) developed an e-cheating intelligence agent designed to monitor student behavior during remote exams. Their system integrates an IP detector, which assigns randomized assessment sets to minimize collusion, and a behavior detector that tracks actions such as frequent tab switching and abrupt changes in answering speed. Their findings demonstrated that these monitoring techniques significantly improved the accuracy of detecting misconduct.

Similarly, Ali M. Duhaim et al. (2022) proposed a multi-layered model to enhance the integrity of online examinations by analyzing student behavior and response patterns. Their framework includes an IP-based detection system, answer similarity analysis, and a clustering algorithm to identify potential cheating cases. Their research revealed that answer similarity detection was the most effective method, as high levels of overlap between student responses often indicated collusion.

Further supporting this approach, Wang et al. (2023) introduced a machine learning-based cheating detection system that combines behavioral tracking with real-time anomaly detection. Their model analyzes keystroke dynamics, cursor movements, and response times to identify irregularities that may suggest dishonest practices. Their study highlighted the importance of continuous monitoring and adaptive AI models in strengthening the security of online assessments.

Building on these findings, our study suggests that observing student behavior during online assessments can help identify potential cheating. Actions such as copying questions, frequently switching tabs, or unusual response patterns can serve as warning signs of misconduct. By closely monitoring these behaviors, educational institutions can enhance the security of online assessments and create a fair testing environment, ensuring that all students are evaluated equitably. The collective efforts of researchers in this field underscore the growing need for advanced monitoring systems in online learning environments to uphold academic integrity.

Challenges in Large Language Models

While the continuous advancement of Large Language Models (LLMs) in education shows significant progress, several challenges remain to be addressed. A major concern is that expanding the volume of pre-training data has been a crucial factor in enhancing the educational capabilities of LLMs. As pre-training datasets have rapidly exceeded the number of documents that human teams can manually review for quality, data collection processes have increasingly relied on heuristics for selecting sources and applying filters (Kaddour et al., 2023).

The effectiveness of LLMs is heavily influenced by the quality of the data they are trained on, which can either significantly enhance or hinder their performance.

LLMs are prone to hallucinations when they generate false or inaccurate information. These hallucinations can be categorized as either intrinsic or extrinsic. Intrinsic hallucinations occur when the model produces information that directly contradicts the source text, while extrinsic hallucinations arise when the generated content cannot be confirmed or denied by the source text. Various factors can cause hallucinations during inference, such as misinterpreting facts from the source material. To ensure accuracy, LLMs must possess strong reasoning abilities to correctly interpret and understand the source text (Patil & Gudivada, 2024).

Overcoming these challenges requires continuous monitoring, rigorous quality checks, data cleaning, and significant financial investment to ensure that LLMs provide accurate and reliable results in the educational field.

Challenges in Online Assessments

As the country's educational perspective innovates with the technological advancements, students in the Philippines face challenges on online assessments. As highlighted in the research by Cahapay (2021), there are five major problems encountered by students: The incompatibility of browsers by the version requirements of the assessment. Anxiety over tracking tools, there are students who felt anxious over live reports of the tracking tools of the online test application while they were taking the test. Third is an unstable internet connection. Fourth, is the electric power interruption. And lastly, distractions in the environment of the student. These challenges point out the necessary approaches to improving online assessment experience. By addressing these issues, educators can develop more effective and just online assessment tools that aid in analyzing student performance.

Synthesis

The integration of Artificial Intelligence (AI) and Machine Learning (ML) in education has become a transformative force, reshaping teaching methodologies and enhancing learning outcomes. As defined by Dann et al. (2024), AI involves systems capable of human-like learning, adaptation, and self-correction, with ML serving as its core functionality (Jagwani & Aloysius, 2019). This relationship has significantly influenced how educational data is utilized, allowing for the development of adaptive learning systems that personalize learning experiences based on each student's unique needs (Kamalov et al., 2023). Tools such as Learning Management Systems (LMS) leverage AI algorithms to analyze student behavior and academic performance, enabling educators to identify learning patterns and adapt instructional strategies accordingly (Vergara et al., 2024).

In the Philippine educational context, the integration of AI presents opportunities to enhance learning outcomes by tailoring teaching strategies to student progress (Estrellado & Miranda, 2023). However, concerns such as data privacy, digital divide, and technological infrastructure limitations must be addressed to ensure equitable access to these advanced educational tools (Holmes et al., 2021).

One of the most impactful applications of AI in education is the use of Large Language Models (LLMs) for generating automated feedback. Studies show that fine-tuned LLMs, such as OpenAI's GPT models, offer significant potential in providing real-time, personalized feedback to students—particularly in programming and computer science education (Pornprasit & Tantithamthavorn, 2024). However, LLMs face challenges in recognizing subtle code errors and generating accurate feedback without specialized fine-tuning (Muennighoff et al., 2023; Miceli Barone et al., 2023).

The integration of static analysis tools like PMD and Javac has proven effective in addressing these shortcomings by ensuring code quality and adherence to programming standards (Nikolić et al., 2024; Kaur & Nayyar, 2020).

The use of AI-driven platforms has also emerged as a promising solution for addressing the growing issue of academic dishonesty in online assessments. Systems that monitor student behavior, such as tracking tab-switching and analyzing response patterns, have shown significant success in detecting misconduct (Leslie Ching Ow & HeeJeong Jasmine Lee, 2022; Duhaim et al., 2022). These findings highlight the importance of integrating monitoring systems into online assessments to uphold academic integrity.

Despite these advancements, challenges in online assessments persist, particularly in the Philippine setting. Students face issues such as browser incompatibility, anxiety caused by tracking tools, unstable internet connections, power interruptions, and environmental distractions (Cahapay, 2021). Addressing these barriers is essential for ensuring fair and accurate assessments of student performance.

Recent research highlights the potential of fine-tuned LLMs in improving the quality of feedback and enhancing student engagement. The GPT-4o-mini model has shown reliability in generating meaningful feedback for programming tasks, offering a scalable solution to support student learning (Koutcheme et al., 2024). This allows educators to focus on more complex cases where human expertise is still necessary, creating a balanced learning environment that integrates both technological innovation and human support.

In conclusion, by combining AI-driven tools, static code analysis, and advanced monitoring systems, educational institutions can create more adaptive, personalized, and effective learning environments. Addressing technological, ethical, and infrastructural challenges is essential for maximizing the potential of AI in education. A balanced approach that integrates technological advancements with human expertise can lead to improved academic outcomes, ensuring that both students and educators benefit from an evolving educational landscape.

CHAPTER III: METHODOLOGY

Introduction

With the increasing integration of artificial intelligence (AI) in education, traditional assessment methods are evolving to become more adaptive and insightful. This study explores the development of an AI-driven online assessment platform that leverages Large Language Models (LLMs) to analyze student responses, provide real-time feedback, and generate meaningful insights into student performance. By automating the evaluation process, this platform aims to enhance student learning experiences while offering educators a data-driven approach to monitoring progress.

This chapter presents the methodology used in designing and implementing the platform. It details the processes involved in data collection, preprocessing, model fine-tuning, system deployment, and performance evaluation. The assessment system is structured into two key components: real-time feedback generation based on student responses and data-driven analytics that visualize performance trends using synthetic datasets. The study employs machine learning techniques, natural language processing (NLP), and cloud-based storage solutions to ensure efficiency and scalability.

By integrating AI-powered feedback, static analysis tools, and real-time data processing, the proposed platform seeks to bridge the gap between traditional assessments and intelligent, automated evaluation. The following sections outline the research design, data collection methods, preprocessing techniques, and system architecture, providing a comprehensive view of how the platform is developed and optimized for educational use.

Conceptual Framework

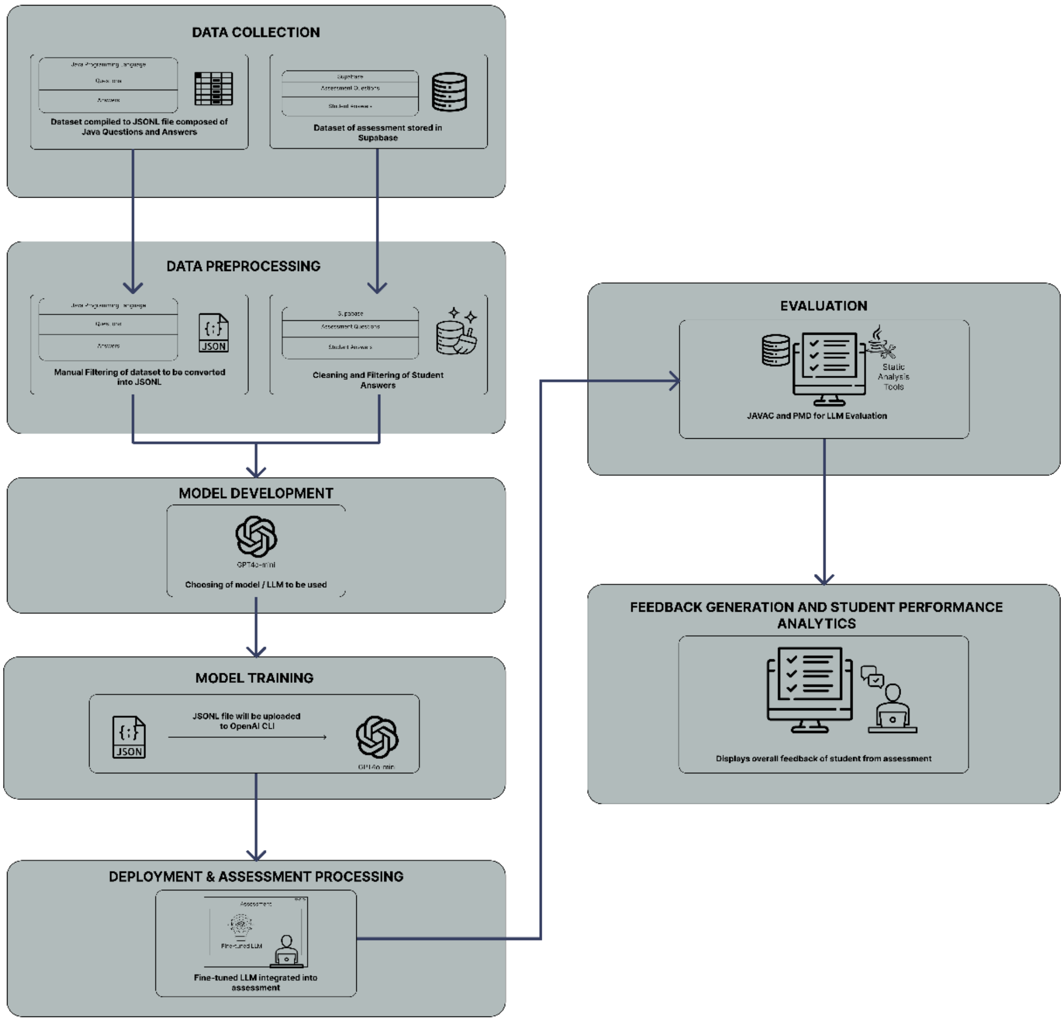


Figure 1: Assessment Page Conceptual Framework

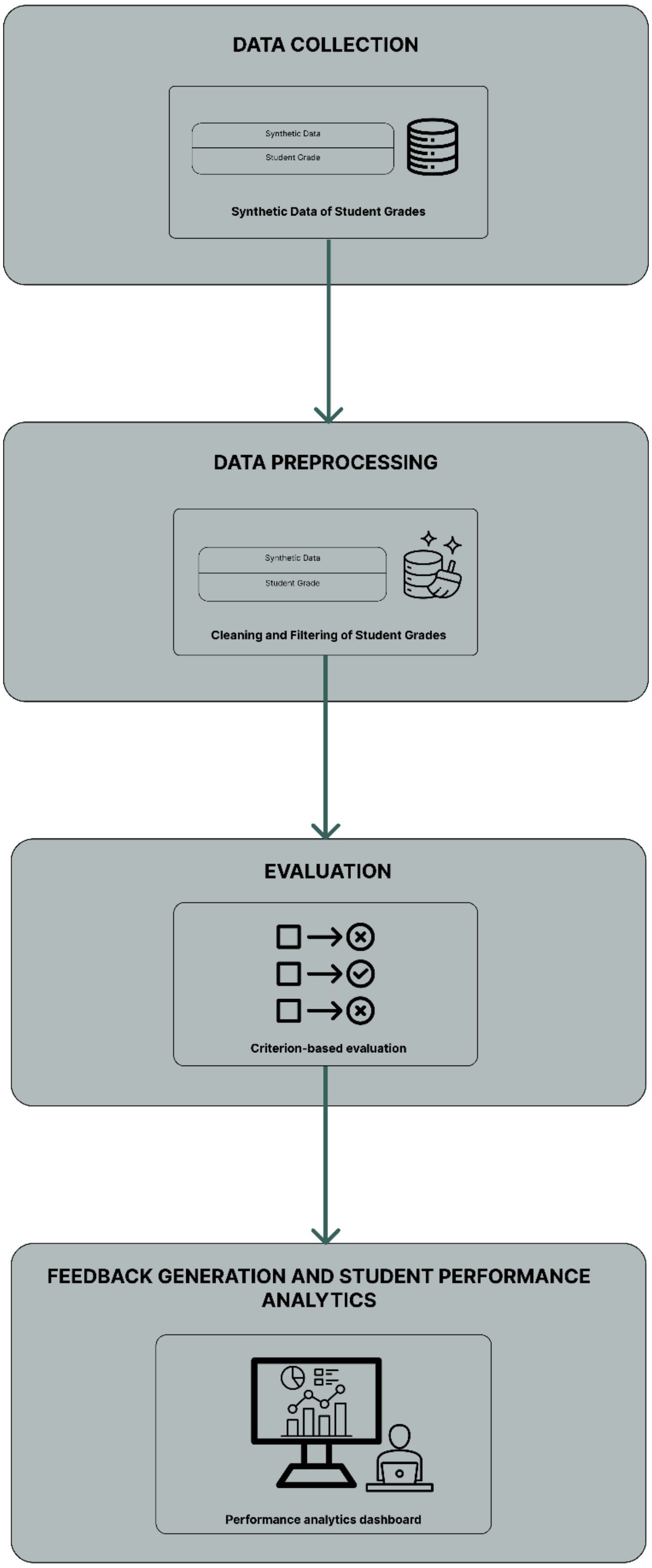


Figure 2: Dashboard Page Conceptual Framework

The framework follows the Machine Learning based conceptual framework which shows the flow for the assessment feedback analyzation and analytic dashboard. The framework is divided into two flows due to the data used for the dashboard, which is synthetic data. On the other hand, assessment data is real-time based and will be updated when answered by the user.

The first component starts with the data collection on the assessment page, gathering of dataset for java question and answers were done through different open-source sites for the fine-tuning process to be done later. Next is during taking of the assessment, the answers of the student are collected with the questions they've answered that will be processed later. On the other hand, the dashboard page data collection is done by gathering synthetic data to provide meaningful analytics that will be later shown in the dashboard.

After data collection, the data preprocessing takes place, this is where the dataset collected from respective sections and tasks are cleaned and fixed to the desired format. The fine-tuning of OpenAI requires a JSONL file format for the process which is why the dataset first was manually filtered and then converted to the proper file format to prepare for the next steps

For the assessment section, model development is done, this is where choosing the desired Large Language Model is properly checked and decided for the whole data fine-tuning process. The researchers chose gpt-4o mini LLM API.

The dataset converted into JSONL files is now ready for the fine-tuning process and this is where model training steps on. OpenAI automatically splits the data into training and validation sets (80-20) to evaluate learning during the training process.

As soon as the model training is finished, the deployment and assessment processing follows for integrating the fine-tuned LLM inside the assessment platform to provide real-time feedback generation based on student responses.

Evaluation processes will be done to provide proper checking of the criteria and metrics. With the assessment branch, LLM-based criteria will be used for checking answers to identify correctness, efficiency, and logic from the answers provided. While on the dashboard branch, criterion-based evaluation will be used for generating structured feedback to identify proper analysis that will be soon visualized.

Lastly, feedback generation and student performance analytics component is where the output visualization is presented to the platform of the student. For the assessment, the overall feedback is shown to the student based on their answers from the assessment platform. While the student dashboard provides performance analytics based on the synthetic data gathered during the data collection component.

Research Design

The study adopts a qualitative research design that focuses on the development and implementation of an insight-driven online assessment platform. This platform utilizes advanced AI algorithms, specifically Large Language Models (LLMs) and machine learning, to analyze student performance data. The focus is on providing personalized feedback and enhancing the evaluation process beyond traditional grading metrics to offer a more comprehensive assessment of student progress. By identifying strengths and weaknesses, the platform aims to provide tailored support that enhances academic performance. The primary focus is to assess how effectively AI-driven insights can support student improvement, particularly regarding critical factors impacting student success, such as engagement and performance metrics, as identified by Namoun and Alshanqiti (2021).

Data Collection

The dataset used for fine-tuning the language model was constructed using questions sourced from reputable internet-based educational platforms, including W3Schools, GeeksforGeeks, Edabit, HackerRank, and W3Resource. These platforms were selected due to their widespread use in programming education and their alignment with fundamental Java programming concepts commonly taught in introductory computer science courses.

Each of these platforms serves a distinct role in programming education:

- **W3Schools** provides structured tutorials and syntax explanations, making it a useful resource for foundational Java concepts.

- **GeeksforGeeks** offers extensive programming examples, including problem-solving approaches and algorithmic techniques commonly encountered in beginner-level programming courses.

- **Edabit** features coding challenges that reinforce problem-solving skills, allowing the dataset to incorporate practical application scenarios.

- **HackerRank** is widely used in coding assessments and competitive programming, offering real-world coding problems that simulate practical programming challenges.

- **W3Resource** provides a collection of Java exercises and programming tasks that cover a wide range of topics, from basic syntax to more advanced programming concepts, making it a valuable supplementary resource.

While these platforms are not peer-reviewed academic sources, they were chosen because they provide widely recognized programming exercises that are commonly used by educators and students for learning Java. Additionally, they offer a diverse set of problems, covering basic syntax, control structures, object-oriented programming, and algorithmic logic—all essential components of an Introduction to Programming curriculum.

To ensure dataset quality and reliability, only well-structured problems with clearly defined solutions were selected. Furthermore, incorrect responses were manually curated to include common student mistakes, allowing the model to better identify misconceptions and provide targeted feedback.

Although educational platforms such as W3Schools, GeeksforGeeks, and W3Resource provide valuable learning resources, future work may incorporate peer-reviewed academic sources and university course materials to further validate the dataset and ensure comprehensive coverage of Java programming principles.

Once the fine-tuning process was completed, the model was integrated into the assessment platform. During assessments, student responses are processed through the fine-tuned model, which analyzes their answers and generates feedback based on predefined evaluation criteria. This system enables real-time feedback generation, ensuring that the insights provided are relevant and tailored to the specific Java programming tasks presented in the assessment. Synthetic data was used to represent the grades of first-year Computer Science students. Due to data privacy concerns, we did not use actual student grades. Instead, we generated artificial datasets that mimic real-world academic performance patterns, ensuring that our analysis remains ethical and compliant with data protection regulations. This data was used for the analytics in the student dashboard to display various performance metrics, providing insights into student progress and areas for improvement.

Data Preprocessing

Filtering for Java-Specific Content

Since our objective was to fine-tune the model for Java-related questions, we ensured that only Java-relevant queries and responses were retained. Non-Java content, including discussions about other programming languages or general administrative topics, was excluded.

Manual Filtering

It was carefully picked that both correct and incorrect Java code implementations to enhance the model's ability to differentiate between valid and erroneous solutions. This selection ensured that the dataset covered a diverse range of coding patterns and potential mistakes.

Expanding the Dataset

Expanding the dataset from 45 to 126 entries, incorporating a diverse range of Java code structures, including both correct and incorrect answers. This expansion aimed to improve the model's ability to differentiate between valid and erroneous solutions.

Data Analysis

Data analysis will focus on evaluating student performance and identifying strategies for improvement through the following methods:

1. **Qualitative Analysis**: Performance data collected through the assessment platform will be analyzed using statistical analysis tools and the fine-tuned LLM. Descriptive statistics will summarize key performance metrics, such as code correctness, input handling, code structure and logic functionality providing an overview of student achievement.

2. **AI-Driven Insights**: AI algorithms specifically LLMs will be employed to evaluate student performance dynamically. By analyzing data trends, the platform will generate recommendations for individualized learning paths, helping students focus on specific areas that require improvement. Techniques such as machine learning and predictive analytics will be utilized to enhance the learning experience and foster academic success, as highlighted by Pardo and Siemens (2014).

3. **Data Storage and Feedback Collection (Supabase Integration)**: All data requests, including user answers, feedback generation, and assessment results, will be stored in a cloud-based Supabase database. Supabase, chosen as the open-source backend provider, will serve as the central data repository for both real-time and historical student performance data. The database will handle the following tasks:

- **Storing User Responses**: Every student input during assessments, including code submissions and answers to Java programming tasks, will be securely stored in Supabase.

- **Feedback Storage**: Feedback generated by the fine-tuned LLM and static analysis tools (like Javac , PDM, and Overall feedback) will be stored in the database for each student, linked to their performance data.

- **Real-Time Data Updates**: Supabase's built-in real-time capabilities will enable the platform to update the dashboard dynamically. As soon as student responses are submitted and feedback is generated, the system will immediately update the dashboard with the latest insights, making the process more interactive and providing instant feedback to students.

- **Data Retrieval for Dashboard**: The collected data will be fetched from Supabase and displayed on the performance dashboard. This dashboard will present a clear and detailed breakdown of the student's performance across the core metrics—Fundamentals of Programming, Control Structures, and Arrays—and visualize their strengths and weaknesses.

The use of real-time updates via Supabase ensures that both students and educators can access up-to-date performance data instantly. The ability to track real-time progress will enhance the learning experience, allowing students to adapt and focus on areas that need the most improvement. For educators, real-time data will enable them to identify common challenges and adjust teaching strategies accordingly.

System Architecture

Description of the Platform

The online assessment platform is designed with a full-stack architecture, integrating a React-based frontend, a Supabase backend, and an AI-driven feedback system. This platform is aimed at providing an adaptive, personalized assessment experience for students while offering detailed insights and performance metrics for educators.

- **Frontend (User Interface)**: The frontend is built using Next.js, a framework built on top of React. This enables server-side rendering (SSR) and static site generation (SSG) for fast, responsive, and SEO-friendly pages. Next.js was chosen because it optimizes performance, improves load times, and enables a smooth user experience with features like real-time updates. The frontend serves as the primary interface where students complete assessments, view feedback, and track their progress on a personalized dashboard.

- **Backend (Data Storage and Processing)**: The backend is powered by Supabase, an open-source backend-as-a-service (BaaS) solution that provides a secure, scalable database and real-time data synchronization. Supabase handles the storage of user data, assessment results, feedback, and more. Using Supabase, the platform leverages PostgreSQL for data storage, ensuring reliable and structured management of student and assessment data. Additionally, Supabase's real-time features allow the platform to instantly update the dashboard whenever new data (e.g., feedback, scores) is generated, providing users with immediate insights.

- **AI Feedback Generation**: The AI-driven component of the platform is powered by GPT-4o Mini, a fine-tuned version of OpenAI's GPT-4. GPT-4o Mini is used to analyze student responses, generate personalized feedback, and identify strengths and weaknesses in their programming tasks. The AI model is fine-tuned using a custom dataset of Java-related programming questions and answers to ensure that the feedback aligns with the core topics of the Intro to Programming course.

- **Performance and Scalability**: The platform is designed to handle multiple users simultaneously, ensuring scalability as the number of students and educators grows. Supabase provides the necessary infrastructure for real-time database updates and supports rapid scaling as the user base expands.

Technology Stack

- **Frontend**: Built using Next.js (React framework), enabling server-side rendering and efficient performance for real-time updates.

- **Backend**: Supabase is used as the database and backend platform. It offers real-time synchronization and data storage capabilities with PostgreSQL.

- **AI Integration**: The platform integrates with GPT-4o Mini to provide AI-powered feedback. This model is fine-tuned on Java-specific programming tasks to ensure the feedback is accurate and relevant to the Intro to Programming curriculum.

- **Feedback and Performance Tools**: Utilizes tools like Javac and PDM for static analysis and validation of Java code submissions, ensuring correct evaluation of student performance.

User Interface Design

The platform's user interface (UI) has been designed to ensure ease of use and a smooth experience for both students and educators. The platform includes two main interfaces: Student Dashboard and Teacher Dashboard, each catering to the needs of the respective users.

- **Student Dashboard**: Students have access to a personalized dashboard where they can view their ongoing assessments, track performance, and receive AI-generated feedback. The interface is designed to be intuitive and user-friendly, with simple navigation and clear, actionable insights. Students can see their overall scores and detailed feedback on coding tasks after taking the assessments. Real-time feedback ensures that students are aware of their performance and can take immediate actions to enhance their learning.

- **Teacher Dashboard**: It provides educators with a clear overview of student performance that can track their progress, create custom questions for assessments, the dashboard also analyzes trends in student data, helping teachers adjust lessons. AI-generated feedback can be reviewed. To ensure academic integrity, the platform includes cheating detection features, flagging behaviors like tab-switching and code-copying.

- **Real-Time Updates:** Both student and teacher dashboards benefit from real-time updates, powered by Supabase. As soon as a student completes an assessment, the platform updates the feedback and displays new insights instantly on the dashboards. This ensures that students receive immediate feedback to support continuous learning.

System Tools Chosen

- **Next.js**: Chosen for its speed, SEO capabilities, and ability to support real-time data rendering, which enhances the overall user experience by providing instant updates on the dashboards.

- **Supabase**: Selected for its ease of integration with PostgreSQL, real-time data synchronization, and open-source nature, making it both cost-effective and scalable.

- **GPT-4o Mini**: Used to provide intelligent, context-aware feedback to students, enhancing the platform's ability to offer personalized, actionable insights for improvement.

- **Javac & PDM**: These tools were integrated for syntax checking and static analysis, ensuring that the platform provides reliable and precise evaluations of student code submissions.

Ethical Consideration

Ethical considerations are utmost in conducting this research, which aims to improve students' educational experiences through technology. The following measures will be implemented to ensure the study is conducted responsibly and ethically:

1. **Confidentiality:** The privacy of participants will be strictly maintained. Personal information, including any identifying data, will be anonymized to protect student identities. Data will be securely stored and only accessible by the research team. Anonymization ensures that results cannot be traced back to individual students, protecting their privacy, as emphasized in the IEEE Code of Ethics (Institute of Electrical and Electronics Engineers, 2020).

2. **Data Protection:** The study will comply with the Data Privacy Act of 2012 (Republic Act No. 10173) of the Philippines, ensuring the secure and ethical handling of students' data. Collected data will be used solely for research purposes, and participants will have the right to access or request the deletion of their data at any time.

CHAPTER IV: RESULTS & DISCUSSION

Introduction

The effectiveness of an AI-powered assessment system relies not only on its ability to generate feedback but also on the accuracy, relevance, and clarity of that feedback. This chapter explores the performance evaluation of the fine-tuned GPT-4o Mini model in assessing Java code submissions. Various quantitative and qualitative metrics were employed to analyze the model's effectiveness, ensuring that it provides structured, meaningful, and pedagogically valuable feedback to students.

Additionally, to enhance the depth of analysis, static code evaluation techniques and compilation checks were incorporated. These tools complement the AI-generated feedback by identifying structural issues, syntax errors, and best practice violations in student submissions. Beyond automated assessment, user feedback was gathered through a structured evaluation process, measuring the platform's usability, clarity, and overall impact on learning.

By examining these multiple dimensions of evaluation, this chapter aims to provide a detailed analysis of the model's strengths, limitations, and contributions to improving programming education.

Presentation of Data

Dataset

The dataset used in this study was constructed using a JSONL (JSON Lines), specifically designed to fine-tune the assessment system's language model for more accurate code evaluation. This dataset played a critical role in enhancing the model's ability to assess java code submissions and provide constructive feedback.

The **JSONL** for fine-tuning contains a format of:

- Messages - a field that represents a structured conversation where multiple exchanges of roles and systems happen inside a single entry
    - **Role** - specifies who is speaking, in the data there are three roles present in the dataset:
        - System - provides instructions or context for the AI
        - User - represents the input from the user
        - Assistant - represents the AI's response to the user input
    - **Content** - contains the actual text that provides the instructions, the user input, and the AI responses

The dataset used in this study was constructed using a synthetic dataset to ensure compliance with student privacy laws and data protection regulations, preventing the use of real student data. Our dataset was carefully designed to simulate real academic performance, incorporating scores from activities, quizzes, and exams to provide a realistic evaluation of student grading trends. The synthetic data was directly based on the National University grading system (Table 1), ensuring alignment with actual academic assessment criteria. This approach allowed us to maintain both ethical standards and data integrity while effectively analyzing and interpreting student performance patterns.

Table 1: NU Grading System

NU Grade Point	Grade Range (%)	Description
4.0	96 - 100	Excellent
3.5	90 – 95	Very Good
3.0	84 – 89	Good
2.5	78 – 83	Above Satisfactory
2.0	72 – 77	Satisfactory
1.5	66 – 71	Fair
1.0	60 – 65	Pass
R	< 60	Repeat
0.0		Fail
Dr		Officially Dropped
Inc		Incomplete
P	>= 60	Pass for Bridging / Intervention Courses
F	< 60	Fail for Bridging / Intervention Courses

Descriptive Statistics

Model Performance and Evaluation

GPT-4o Mini and Custom GPT-4o mini model

The fine-tuning process was conducted using the GPT-4o Mini model via OpenAI's API. A structured JSONL dataset was utilized to enhance the model's ability to assess both correct and incorrect Java code submissions. Prior to fine-tuning, the dataset was tested on the base GPT-4o Mini model to evaluate its initial feedback. Insights from this evaluation were used to identify areas for improvement, which were then integrated into the fine-tuning process to enhance the accuracy and relevance of feedback in our custom GPT-4o Mini model.

The effectiveness of the models was first evaluated using the BLEU score, which measures the similarity between the generated feedback and the expected dataset. As shown in Table 2, all models, including the base GPT-4o Mini and fine-tuned variations, achieved a BLEU score of 0.672. This indicates that, in terms of surface-level similarity, the models consistently aligned with the dataset, producing structured and relevant feedback.

However, BLEU primarily captures lexical similarity, meaning that while the generated feedback closely follows expected responses, it does not necessarily reflect improvements in clarity, correctness, or usefulness. This necessitates further qualitative analysis to determine whether fine-tuning led to better, more educational feedback rather than simply maintaining textual consistency.

To further assess feedback quality, the ROUGE-L metric was applied to analyze key phrases and sequence overlap between the generated responses and expected feedback. The ROUGE-L scores remained consistent across models at 0.6316, suggesting that fine-tuning did not drastically alter the structure of responses in terms of key content inclusion (Table 3).

While this confirms that the fine-tuned model remains aligned with expected phrasing, it does not necessarily indicate improved feedback effectiveness. A key limitation of this metric is that it does not evaluate whether the feedback provides meaningful guidance for students. Thus, additional semantic similarity measures were needed.

To supplement these evaluations, CodeBERT was utilized to measure the semantic similarity of feedback responses (Table 4). The results showed that all models, including the fine-tuned GPT-4o Mini, achieved high similarity scores ranging from 0.9928 to 0.9958, indicating that fine-tuning did not significantly alter the structure of the generated feedback. However, it refined the responses to be more concise and assessment-focused, rather than providing excessive explanations or directly offering correct solutions.

Lastly, the GLEU metric was used to assess the fluency and coherence of the generated feedback (Table 5). The fine-tuned GPT-4o Mini model achieved a GLEU score of 0.1549, while the base GPT-4o Mini model had a higher score of 0.2685. While a higher GLEU score typically indicates greater similarity, in this case, the decrease in the fine-tuned model suggests that it was producing more refined, context-specific feedback rather than generic responses that strictly followed the dataset structure.

Despite iterative refinements to the JSONL dataset, including multiple modifications aimed at improving the training data, the evaluation results remained consistent. This indicates that the fine-tuning process effectively constrained the model’s output within the defined feedback parameters, ensuring that it adhered to the assessment-focused approach without unintended deviations.

Table 2: BLEU Comparison Table

MODEL	BLEU SCORE	MODEL EVALUATION
B3Rjviyf	0.6782	Correct! This program correctly prompts the user to enter a number, reads it from the input, and prints it back to the console.
B4r8Uh7Y	0.6782	Correct. However, there is no input validation. If the user enters a non-integer value, the program will throw an InputMismatchException. A better approach would be to use a try-catch block to handle potential input errors.
B4pnTUrs	0.6782	Correctness: 5. This implementation correctly reads an integer input from the user and prints it. However, it does not handle potential exceptions from invalid inputs.
B71S0VqX	0.6782	### Detailed Analysis of the Java Code Correctness Score: 5/5 Input Handling: The program correctly reads an integer from standard input using nextInt(). Printing Output: It correctly outputs the entered integer to the console. No Errors: Assuming the user inputs a valid integer, there are no logical errors. Potential Improvements: There could be some validation for non-integer inputs to prevent exceptions, but this is beyond the scope of correctness evaluation as it correctly handles valid scenarios. Readability Score: 4/5 Commenting: The code includes comments that describe what each part is doing, which aids newcomers in understanding. Code Structure: The structure is straightforward and follows common Java conventions. Improved Practices: The comments could be slightly more concise. For example, // Create a scanner for user input is shorter and still clear. The variable names are clear and descriptive, which adds to readability. Minor Suggestions: Adding a comment indicating that the program expects an integer input would improve clarity. Efficiency Score: 5/5

		Input Method: Using Scanner is efficient for reading simple input from the console. The performance for this kind of operation is typically not an issue unless in a very constrained environment. Time Complexity: The operations performed (reading an integer and printing it) are O(1), which is optimal for this task. Memory Usage: The code uses minimal memory, just holding an integer and the scanner object. Overall Evaluation Score: 4/5 Implements basic input handling well and outputs results correctly. Could improve input validation to handle edge cases, such as non-integer inputs. Basic usage of Scanner and basic functionality is efficient, following correct coding practices. Final Scores: Correctness: 5/5 Readability: 4/5 Efficiency: 5/5
BASE MODEL GPT4o-mini	0.6782	Model Evaluation: Certainly! Let's evaluate the provided Java code based on three criteria: correctness, readability, and efficiency. Correctness Evaluation: The code correctly uses the Scanner class to read an integer from the user via standard input, and then prints it back to the console. It correctly handles user interaction for getting an integer and has no syntax errors. However, it has no input validation. If a user enters something that isn't an integer, the program will throw an InputMismatchException. This can cause the program to crash, leading to an undesirable user experience. Score: 3/5 Reason: While the primary function works correctly, the lack of input validation is a significant oversight, impacting the robustness of the code. Readability Evaluation: The variable names and comments are clear and informative, which enhances

		overall readability. The use of spaces and indentation conforms to standard Java coding conventions, making it easy to follow the code's flow. The comments preceding the sections of code clearly explain the purpose of each section, which is helpful for someone trying to understand the code quickly. Score: 4/5 Reason: The code is well-structured and easy to read, but it could benefit from slightly more detailed error-handling comments or a brief explanation of exceptions. Efficiency Evaluation: The code is efficient for its purpose. Reading a single integer and printing it back involves minimal computational overhead. However, since input handling lacks exception management, the program can experience inefficiencies when an improper input is provided, leading to program termination and requiring a restart. Score: 3/5 Reason: While the logic is efficient for reading an integer, the overall efficiency is hampered by the lack of error handling for invalid input. Overall Analysis The code fulfills its basic objective but needs enhancements for robustness, such as input validation and exception handling. Adding these elements would improve its correctness and overall user experience. It maintains good readability with well-named variables and adequate comments but could slightly expand on its explanations, particularly concerning potential exceptions. The efficiency is adequate, but improving the error management could prevent inefficiencies in scenarios of invalid input. Final Scores Correctness: 3/5 Readability: 4/5 Efficiency: 3/5

Table 3: Rogue-L Comparison Table

MODEL	ROGUE L-SCORE	MODEL EVALUATION
B3Rjviyf	0.6316	Model Evaluation: Correct. This version explicitly separates the conditions for clarity, making it easier to understand the logic.
B4r8Uh7Y	0.6316	Model Evaluation: Correct. The code accurately identifies December, January, and February as winter months. However, it could be more concise using an array or list.
B4pnTUrs	0.6316	Model Evaluation: Correct. The code correctly identifies winter months (December, January, and February) and the implementation is clear. It explicitly separates conditions for improved readability.
B71S0VqX	0.6316	Model Evaluation: ### Analysis of the Code #### Correctness: The code checks if a given month (represented as a number from 1 to 12) is considered a winter month (December, January, February). The integer numberOfMonth is statically set to 12, which is indeed a winter month. The logical checks correctly determine if the month is winter: December (12), January (1), or February (2). The output will be true since 12 is a winter month. Score: 5/5 #### Readability: The variable names (condition1, condition2, condition3, and isWinterMonth) are descriptive, but the use of separate boolean conditions can be considered verbose.

		This could be simplified for better clarity. The main logic of checking the conditions is clear, but it could be enhanced by directly comparing the month with an array or a set of winter months instead of using individual boolean variables. The lack of comments can detract from understanding for someone unfamiliar with the context. Score: 3/5 #### Efficiency: The implementation checks three conditions separately and uses a logical OR. While this is efficient for small numbers of conditions, it could be simplified by using a data structure such as an array or a set for lookup, reducing code complexity. However, given that this is a single check in an application with a limited number of months (only 3 conditions), the performance impact is negligible. Still, it's not the best practice for scalability if a different approach was warranted. Score: 4/5 Final Scores: Correctness: 5 Readability: 3 Efficiency: 4 Overall, the code successfully identifies winter months, but enhancements in readability and maintainability could lead to a better implementation. Using a more concise format, such as defining winter months in a collection, would streamline the logic further.
BASE MODEL GPT4o-mini	0.6316	

		Model Evaluation: Here’s a detailed evaluation of the provided Java code for the WinterMonthChecker class. Correctness The code is logically correct. It correctly identifies whether the month represented by the variable numberOfMonth (in this case assigned the value of 12) is a "winter month." The winter months in most parts of the northern hemisphere, including December (12), January (1), and February (2), are correctly identified using boolean conditions. Consequently, when numberOfMonth is 12, the output is true, which is the expected result. Score: 5/5 Readability The code is generally quite readable. The class and method names are clear and descriptive, which makes it easy to understand the purpose of the code. However, a few improvements could be made: The variable numberOfMonth could be renamed to a more specific name, such as currentMonth, to better convey its purpose. The use of separate boolean conditions (condition1, condition2, and condition3) is redundant. A simpler check using a single condition could improve clarity. Overall, while the readability is good, the redundancy could be confusing for someone reviewing it and detracts from clear code design. Score: 4/5 Efficiency

		In terms of performance, the code is efficient given its simplicity. The evaluation of three boolean expressions and a logical OR operation is not computationally intensive. However, the use of separate boolean variables for each condition is unnecessary and adds redundant checks which do not modify the time complexity. A more efficient way would be to combine the checks into a single logical condition without intermediate boolean variables.
		For example, instead of:
		boolean isWinterMonth = condition1 || condition2 || condition3;
		You could directly evaluate:
		boolean isWinterMonth = (numberOfMonth == 12 || numberOfMonth == 1 || numberOfMonth == 2);
		This will keep the code compact and possibly make the execution marginally more efficient, though the difference would be negligible for a small number of calculations like this.
		Score: 4/5
		Overall Evaluation
		This code meets all functional requirements with correct logic, demonstrates a fair level of readability, and has minor inefficiencies that could be addressed.

Table 4: CodeBERT Comparison Table

MODEL	CodeBERT Similarity Score
B3Rjviyf	0.9958
B4r8Uh7Y	0.9938
B4pnTUrs	0.9938
B71S0VqX	0.9940
BASE MODEL GPT4o-mini	0.9928

Table 5: GLEU Comparison Table

MODEL	GLEU SCORE
FINE-TUNED GPT 4-o Mini	0.1549
BASE GPT4-o Mini	0.2685

Static Analysis and Compilation Feedback

To complement the feedback generated by the fine-tuned model, static analysis and compilation tools were incorporated to provide a comprehensive and structured evaluation process. These tools worked in conjunction with the model, ensuring a multi-dimensional assessment of student submissions that covered code quality, syntax validity, and adherence to programming best practices.

- **PMD (Programming Mistake Detector):** PMD was employed to analyze the overall structure and style of Java code submissions. It effectively identified common programming issues such as unnecessary object creation, unused variables, and poor formatting. By detecting these issues, PMD encouraged clean, efficient, and maintainable coding practices, reinforcing essential skills for beginner programmers.

- **Static Analysis:** Static analysis techniques were applied to ensure adherence to standard Java programming conventions. This process verified aspects such as code structure, readability, and logical organization, reinforcing best practices that support students in writing well-structured and logically sound code.

- **Java Compiler:** The Java compiler was utilized to detect syntax errors that could prevent code from executing correctly. Ensuring syntactic correctness at this stage prevented unnecessary complications during further assessment and emphasized the importance of proper syntax in Java programming.

By integrating the fine-tuned model's AI-driven feedback with PMD analysis, static code evaluation, and compilation checks, the assessment platform delivered a well-rounded and sound evaluation of student submissions. This multi-faceted approach extended beyond simply determining whether a solution was correct or incorrect—it provided students with meaningful, structured, and targeted guidance to enhance their coding practices and deepen their understanding of fundamental Java programming concepts.

User Evaluation Result

The scaling criteria used in the evaluation metrics is Likert Scale. This scale, typically ranging from 1 to 5, facilitated the measurement of responses in a standardized manner, ensuring consistency and reliability in grading. Each criterion was assigned a corresponding Likert scale value, where lower values indicated weaker grade and higher values signified stronger grade. By leveraging the Likert scale, the online evaluation rating made by the student evaluators were assessed based by the ranges identified (1 to 5).

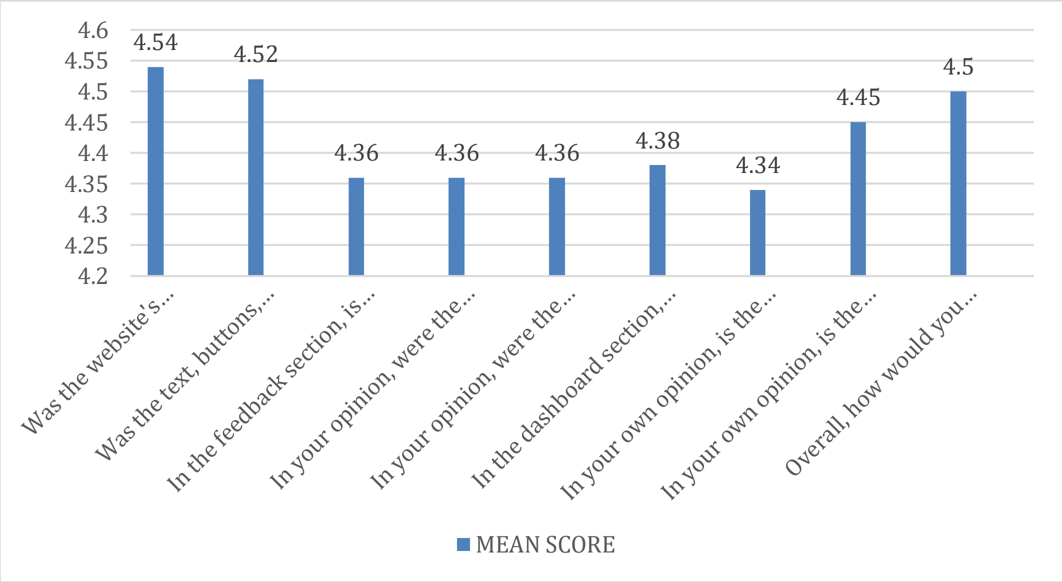
In gathering the user evaluation survey, Microsoft Forms was used wherein the participants rated the questions provided with the Likert Scale ranging 1 – 5. The results were summarized by calculating the average referencing to the mean formula:

$$Mean = \frac{Sum\ of\ all\ Data\ points}{Number\ of\ Data\ points}$$

Table 6: Survey Question Mean Scores

QUESTION	MEAN SCORE
Was the website's navigation clear and intuitive based on the video?	4.54
Was the text, buttons, content easily readable in the video?	4.52
In the feedback section, is the bar graph of score distribution easy to understand?	4.36
In your opinion, were the feedback provided by our Fine - tuned Large Language Model accurate to their respective answers?	4.36
In your opinion, were the feedback provided helpful?	4.36
In the dashboard section, are the charts easy to understand and analyze?	4.38
In your own opinion, is the student dashboard helpful on monitoring the performance of each student?	4.34
In your own opinion, is the teacher dashboard helpful on monitoring their students?	4.45
Overall, how would you rate our platform together with the analysis and feedback?	4.5

Figure 3: Survey Question Mean Score Graph



The survey results indicate a generally positive response toward the platform's usability, clarity, and effectiveness in providing feedback and analysis. The highest-rated aspect, with a mean score of 4.54, is the website's navigation, suggesting that users found it intuitive and easy to use. Closely following this, the readability of text, buttons, and content in the video received a high rating of 4.52, reinforcing the platform's accessibility.

The feedback section's elements, including the bar graph of score distribution (4.36) and the accuracy and helpfulness of AI-generated feedback (both rated 4.36), show that users found the system reliable in assessing responses. However, the charts in the dashboard section received a slightly lower score of 4.38, indicating that while still favorable, there may be minor improvements needed for better clarity.

The student dashboard (4.34) and teacher dashboard (4.45) were both rated positively, with the teacher dashboard receiving a slightly higher rating, suggesting that the platform is particularly useful for educators in monitoring student performance.

Overall, the platform's combined analysis and feedback system received a strong mean score of 4.5, reflecting users' confidence in its overall functionality and value.

CHAPTER V: CONCLUSION AND RECOMMENDATIONS

Conclusion

This study explored the integration of artificial intelligence (AI) and learning analytics into an insight-driven online assessment platform designed to provide real-time, personalized feedback for first-year computer science students. By fine-tuning a Large Language Model (LLM) and incorporating static analysis tools (PMD and Javac), the platform delivered structured, automated evaluations that identified syntax errors, logical flaws, and structural inefficiencies with greater precision than a base AI model.

The findings demonstrate that AI-powered assessment tools can generate detailed and targeted feedback, helping students correct mistakes and improve their programming skills. Unlike traditional assessment methods, this system reduces the burden on educators while providing students with instant, personalized feedback, fostering a more engaging and self-directed learning experience.

While this study successfully implemented AI-driven feedback, it did not directly measure the impact on student learning outcomes. However, the platform's ability to provide real-time insights and structured evaluations highlights its potential as a scalable, cost-effective solution for programming education.

Ultimately, this research validates the feasibility of AI-assisted assessment in programming education, paving the way for future advancements in AI-driven learning environments.

Recommendations

To further enhance the effectiveness of the online assessment platform and ensure sustained improvements in student learning outcomes, several recommendations are proposed for future research and platform development:

1. Optimizing AI Model Efficiency and Cost

   Running an AI-driven assessment system requires significant computational resources, making it essential to analyze cost-efficiency trade-offs between fine-tuning models and using prompt engineering techniques on base models. Future studies should explore lighter-weight models or distillation techniques to reduce latency and enhance scalability for real-time feedback. Additionally, implementing caching strategies for commonly occurring feedback responses could optimize system performance while reducing API costs, ensuring a more efficient and cost-effective assessment platform.

2. Longitudinal Study for Impact Assessment

   Conducting a longitudinal study to track student performance over an extended period could provide valuable insights into the long-term effectiveness of the platform. This would help determine how continuous use of the platform influences learning outcomes, engagement, and overall academic performance.

3. Improve Feedback Accuracy & Adaptability

   To enhance the accuracy and relevance of AI-generated feedback, the fine-tuning dataset should be expanded to include real-world student errors. However, since accurate feedback is highly dependent on structured and wellannotated assessment data, future improvements should focus on maintaining data consistency. One approach is to develop a standardized data collection framework where student submissions are systematically categorized with detailed annotations. Incorporating teacher-verified error labels or automated classification methods can ensure that the dataset remains structured and reliable. Additionally, implementing an adaptive learning approach can improve the system's effectiveness by dynamically adjusting feedback based on a student's progress and common misconceptions.

By implementing these recommendations, the platform can continue to evolve as a comprehensive educational tool, providing both students and educators with meaningful, actionable insights. This will contribute to a more personalized, engaging, and effective learning environment, ultimately improving educational outcomes and fostering academic success.

APPENDICES

A: User Manual

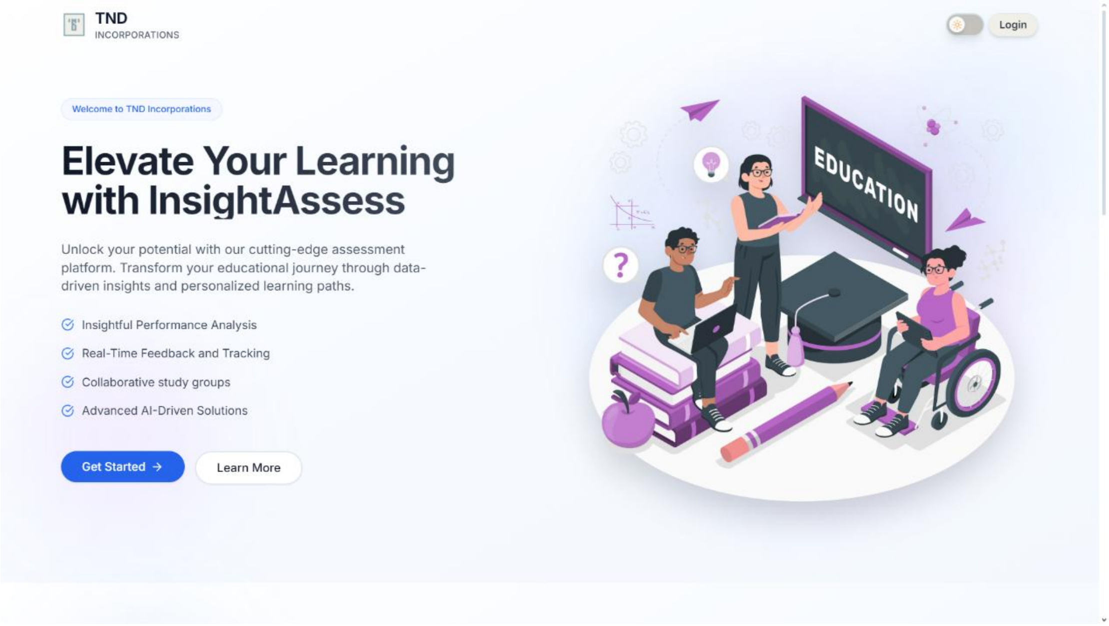


Figure 4: System Landing Page

Displays the default page of the system

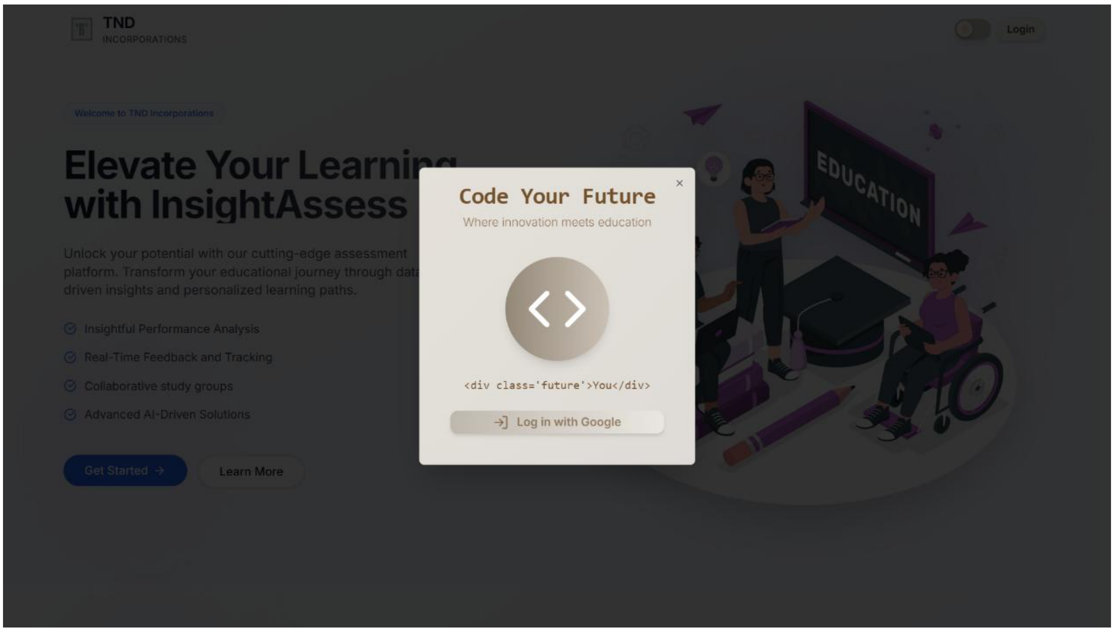


Figure 5: Log-in Page

Page where the students log in to proceed to the dashboard

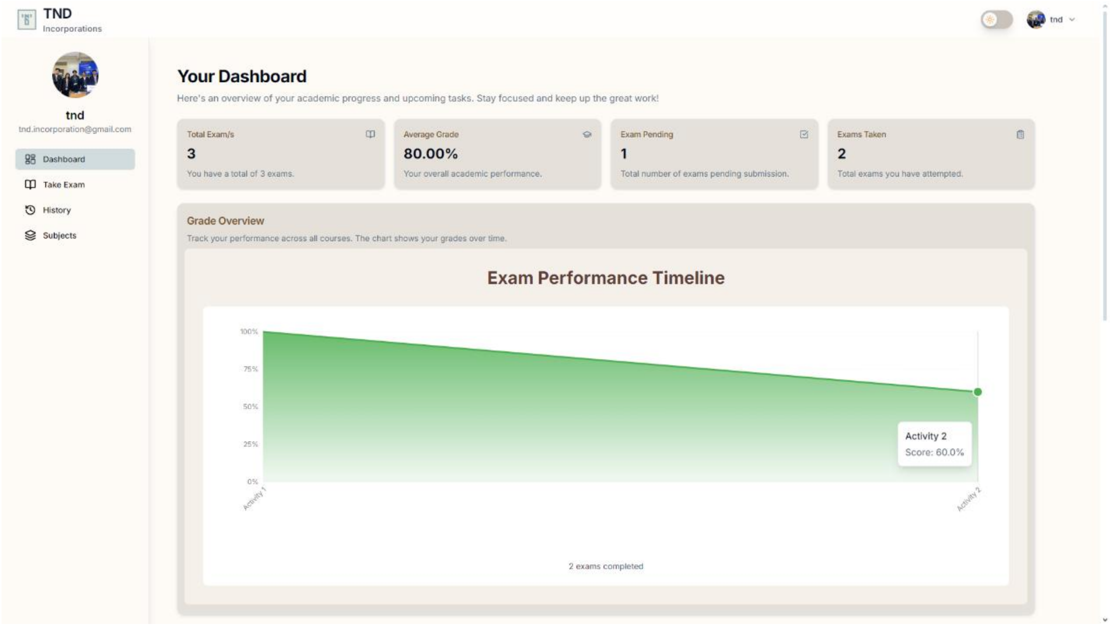


Figure 6: Student Dashboard Overview

The overview of student's dashboard, together with their total of exams, average grade, pending exams, and taken exams. The overview of the grade from their past assessments are also displayed

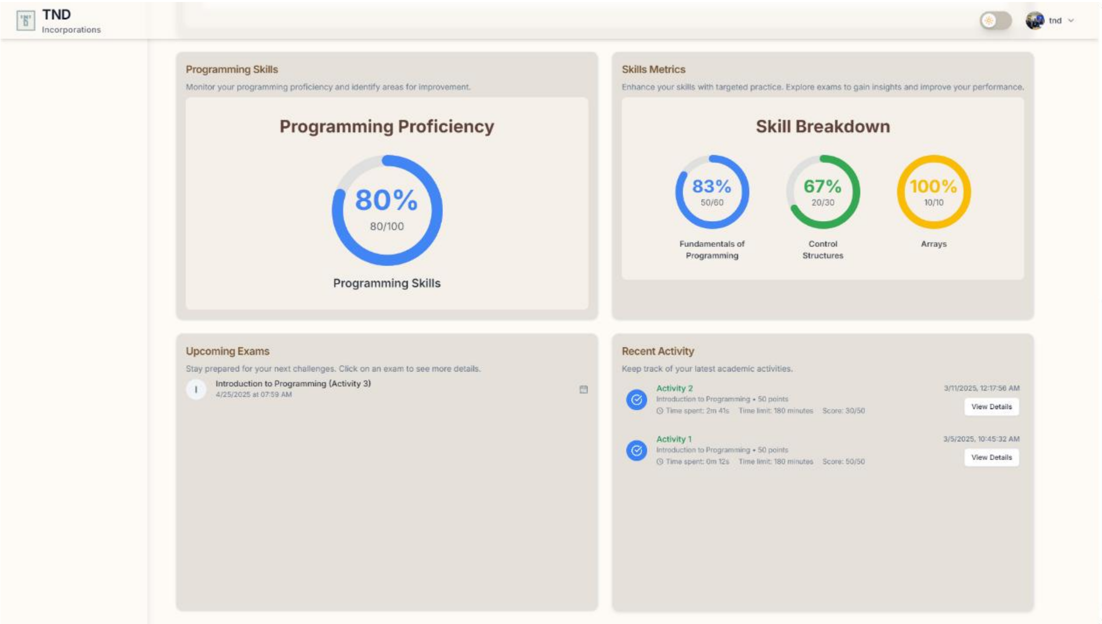


Figure 7: Student Skills Overview

Together with the student dashboard, skills overview of the student based on the assessments could be found and the recent activity or assessments done

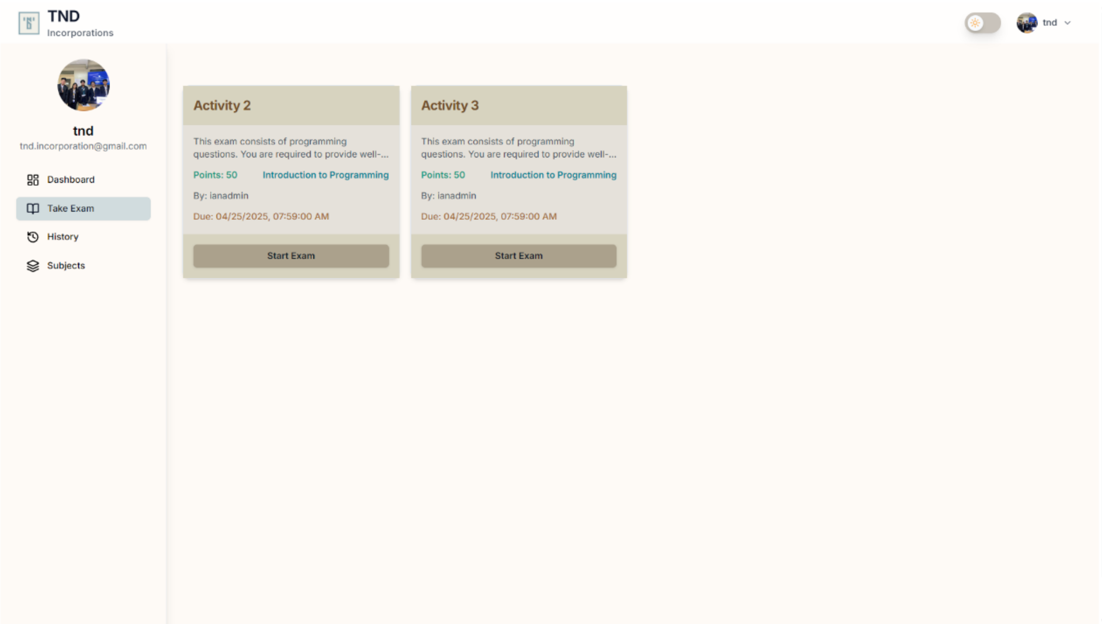


Figure 8: Student Active Exams Navigation

Navigation to identify current exams open for the student and proceed to taking the assessment

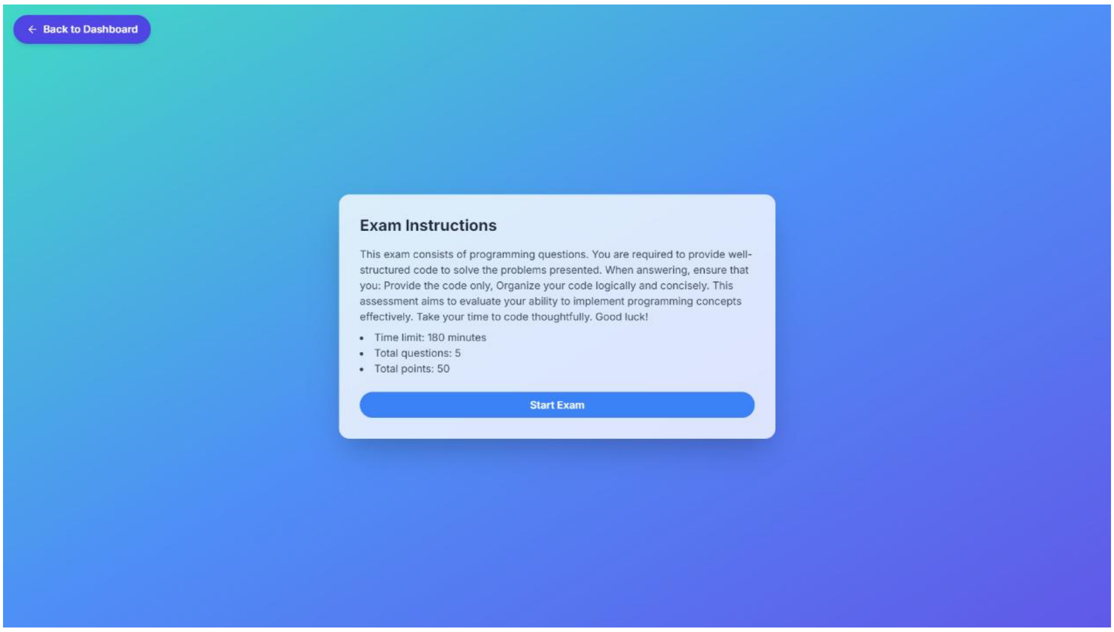


Figure 9: Student Examination Start

Instructions section regarding the assessment to be taken by the student

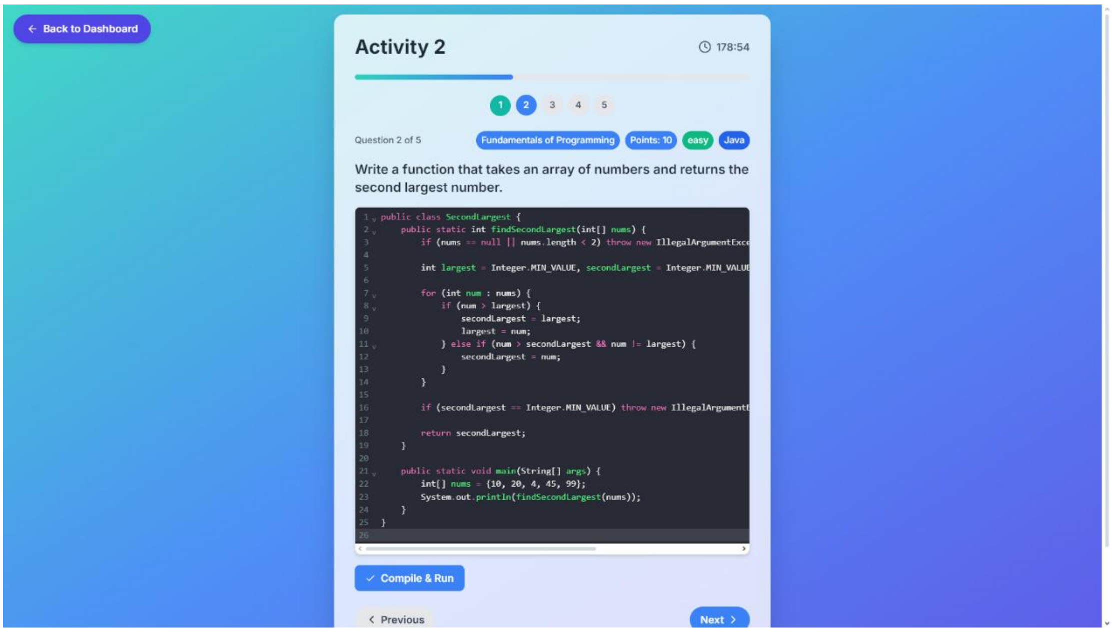


Figure 10: Student Examination Ongoing

The display where the student will be answering the assessment in programming

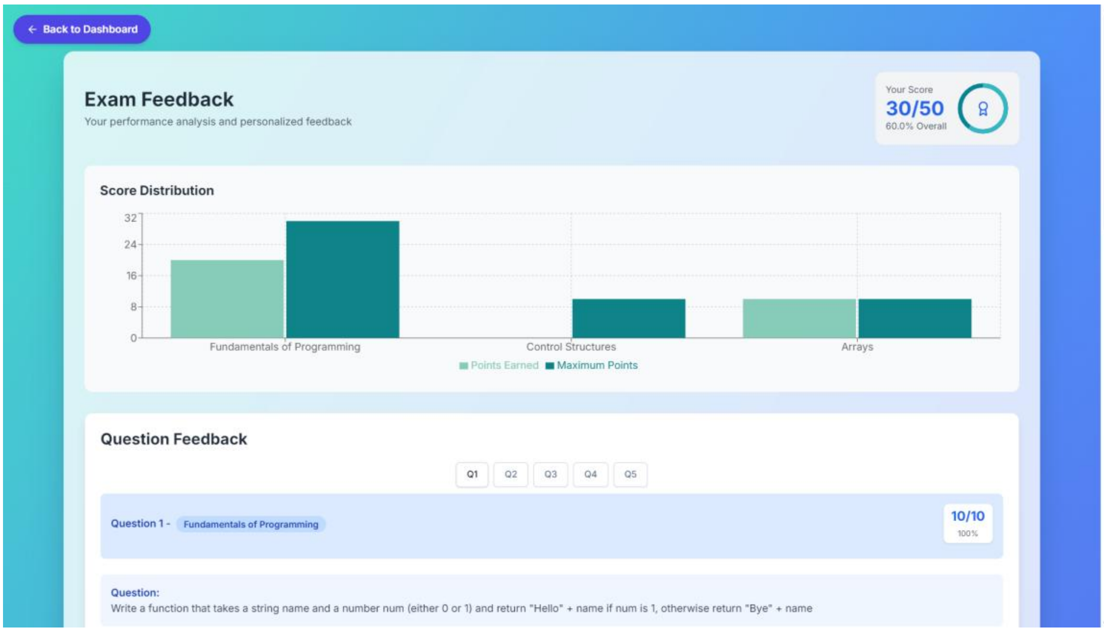


Figure 11: Student Score Distribution Graph

Feedback page shows with a graph of the score distribution from the assessment grade

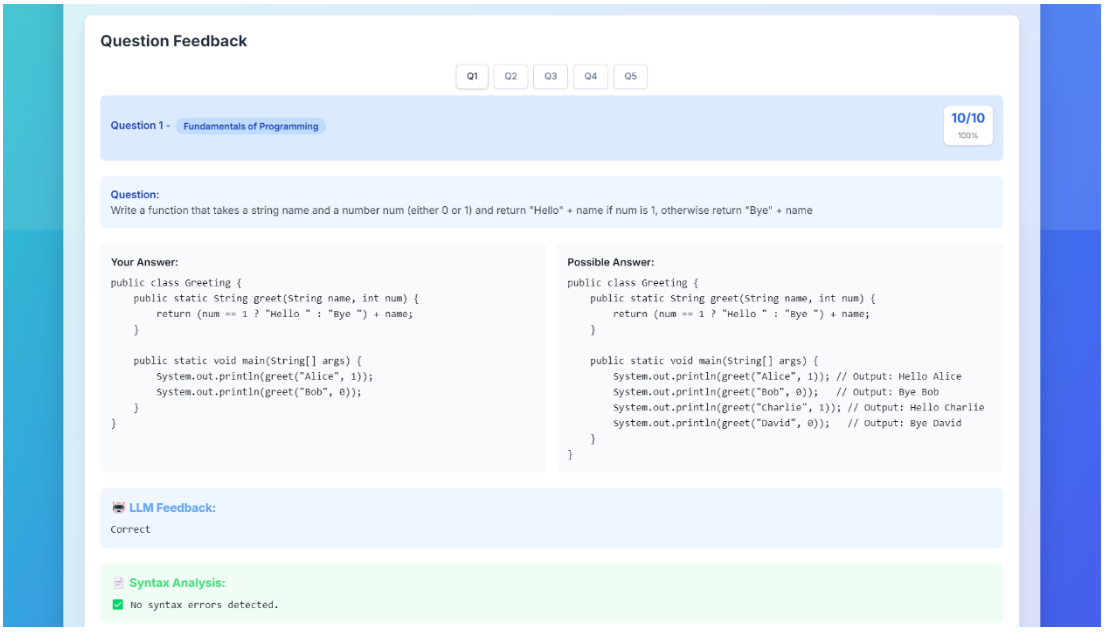


Figure 12: Student Answer Comparison

The feedback page shows the answer of student together with one of the possible programming answers

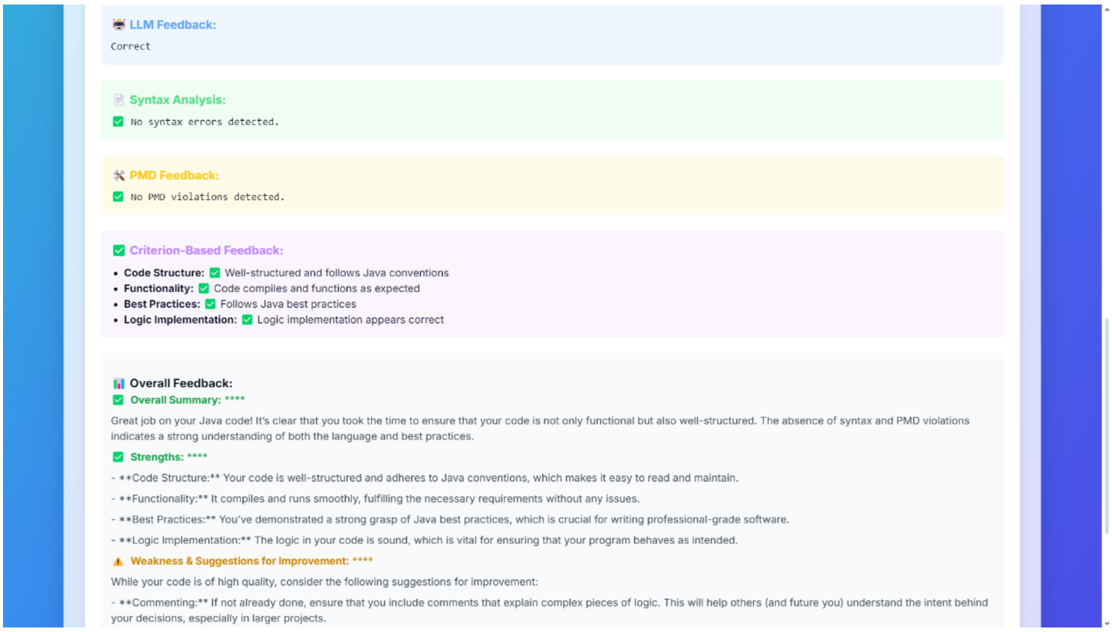


Figure 13: Correct Answer - Feedback from LLM and Analytic Tools

Feedback from the pre-trained LLM model is displayed together with the Static Analysis tools JAVAC and PMD with criterion-based feedback all based on the student’s answer

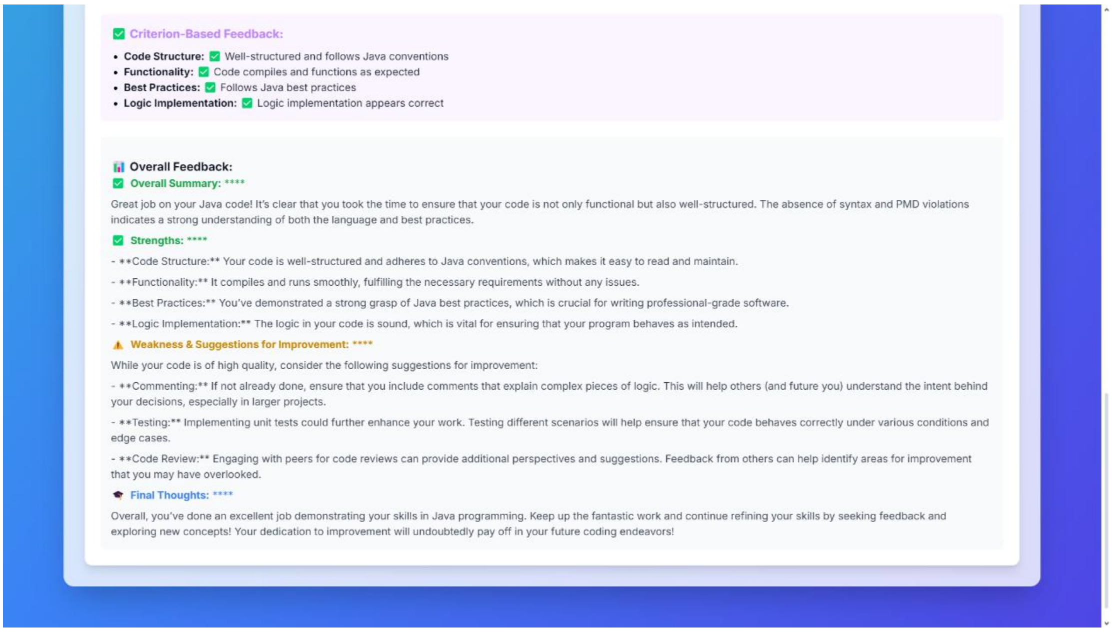


Figure 14: Correct Answer - Overall Feedback

Overall feedback to provide summary and insights for the student

Figure 15: Incorrect Answer - Feedback from LLM

Feedback from pre-trained LLM model displayed from an incorrect answer and providing proper correction

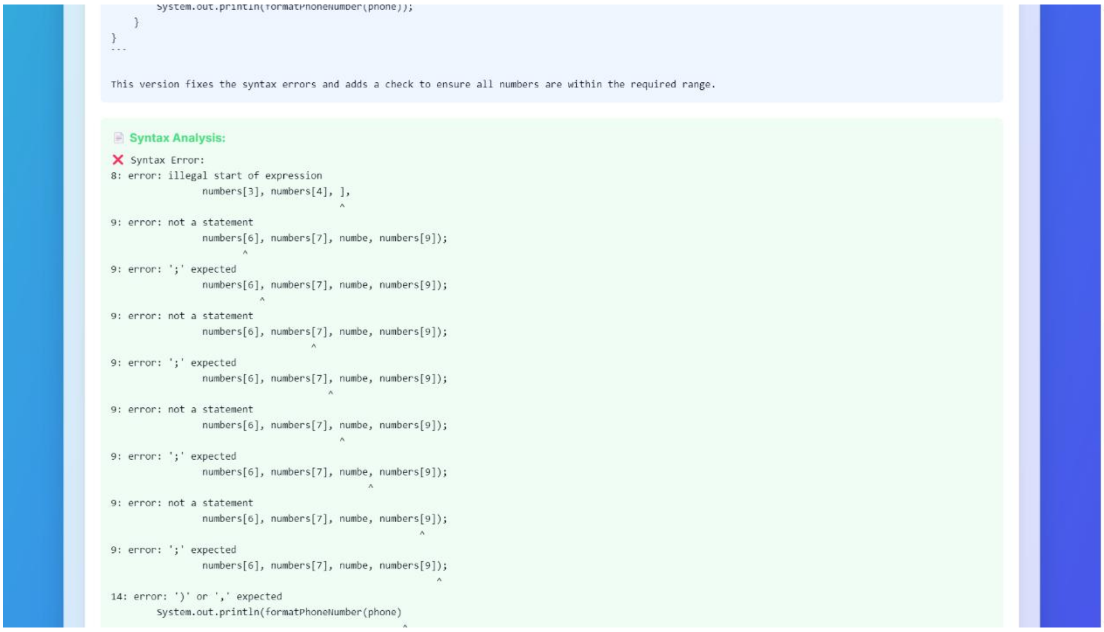


Figure 16: Incorrect Answer - Feedback from JAVAC

Feedback from static analytic tool JAVAC displayed from an incorrect answer and providing proper correction in the line of code

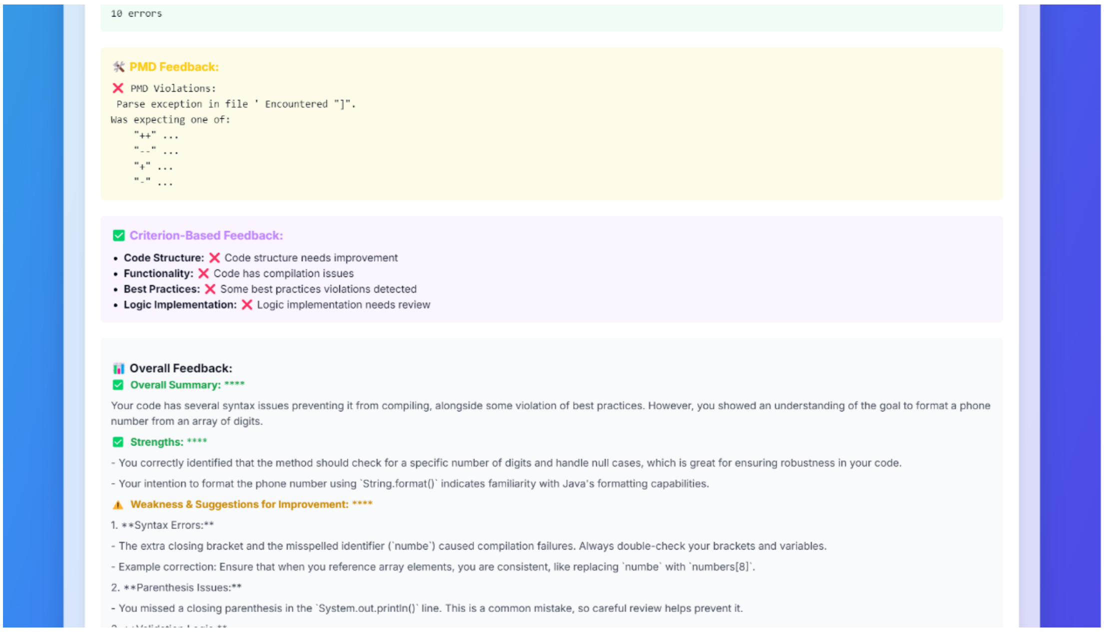


Figure 17: Incorrect Answer - PMD Feedback

Feedback from static analytic tool PMD displayed from an incorrect answer and providing proper correction in the line of code

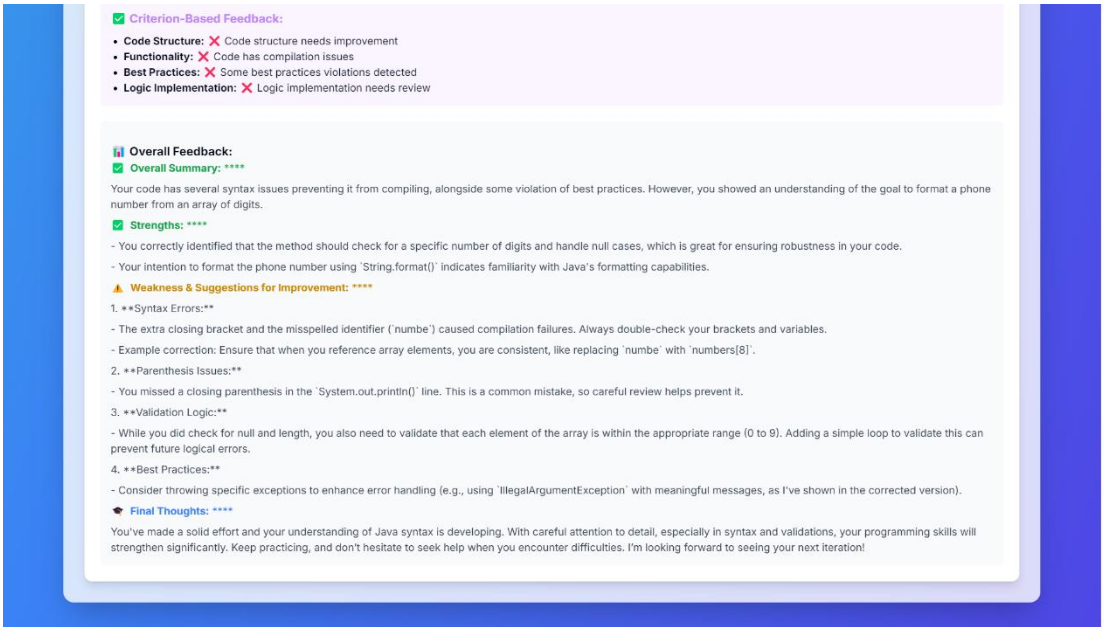


Figure 18: Incorrect Answer - Overall Feedback

Overall feedback to provide summary and insights for the student based on their incorrect answers

B: Admin / Teacher Manual

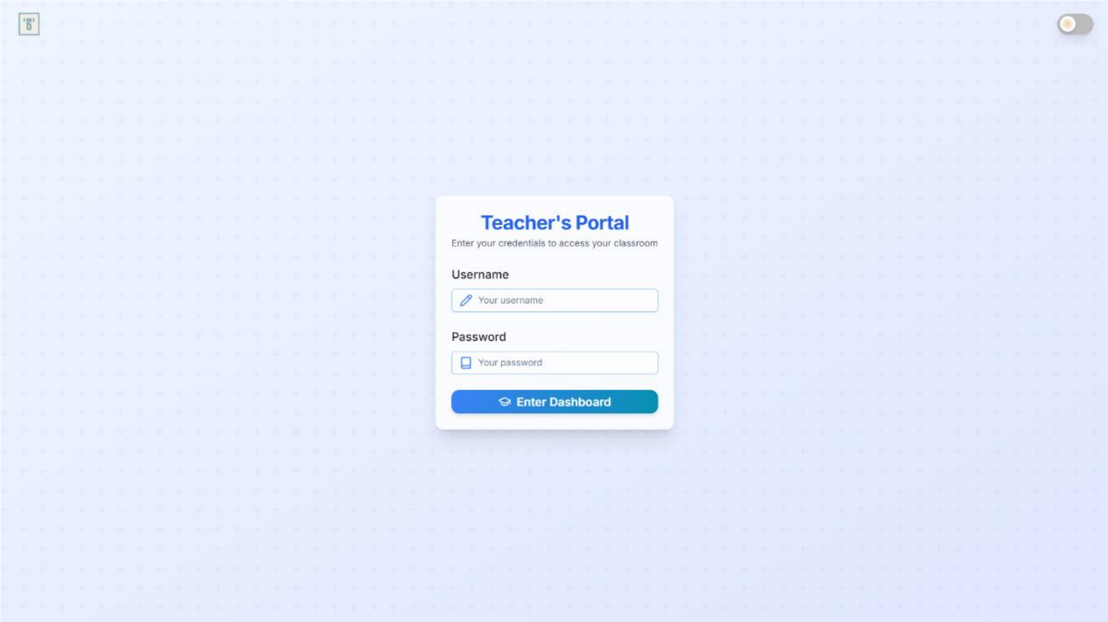


Figure 19: Teacher Login page

Login page for the teacher to proceed to their dashboard

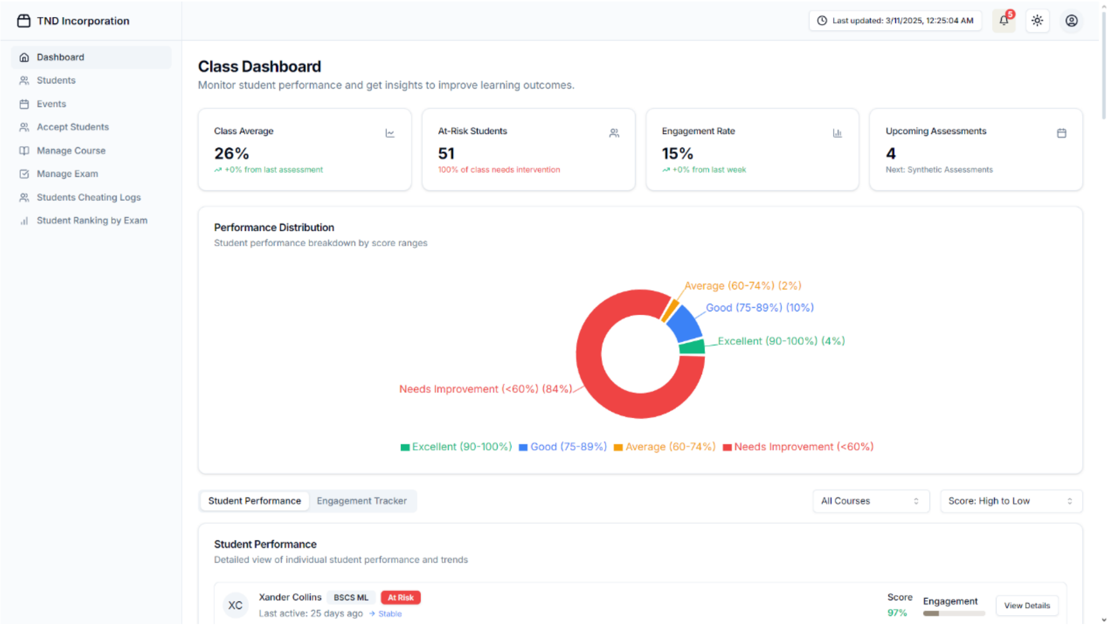


Figure 20: Teacher Dashboard

Teacher class dashboard performance overview that contains the classes' average, at-risk students, engagement rate, and upcoming assessments

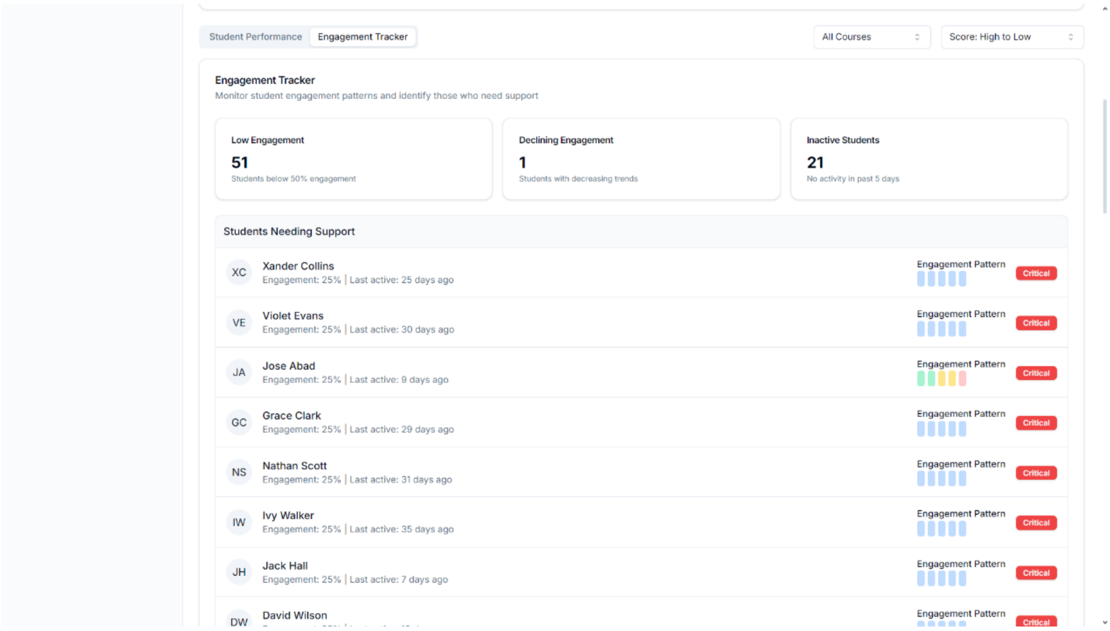


Figure 21: Teacher Dashboard - Engagement Tracker

Teacher engagement tracker to identify and be aware of students in need of support

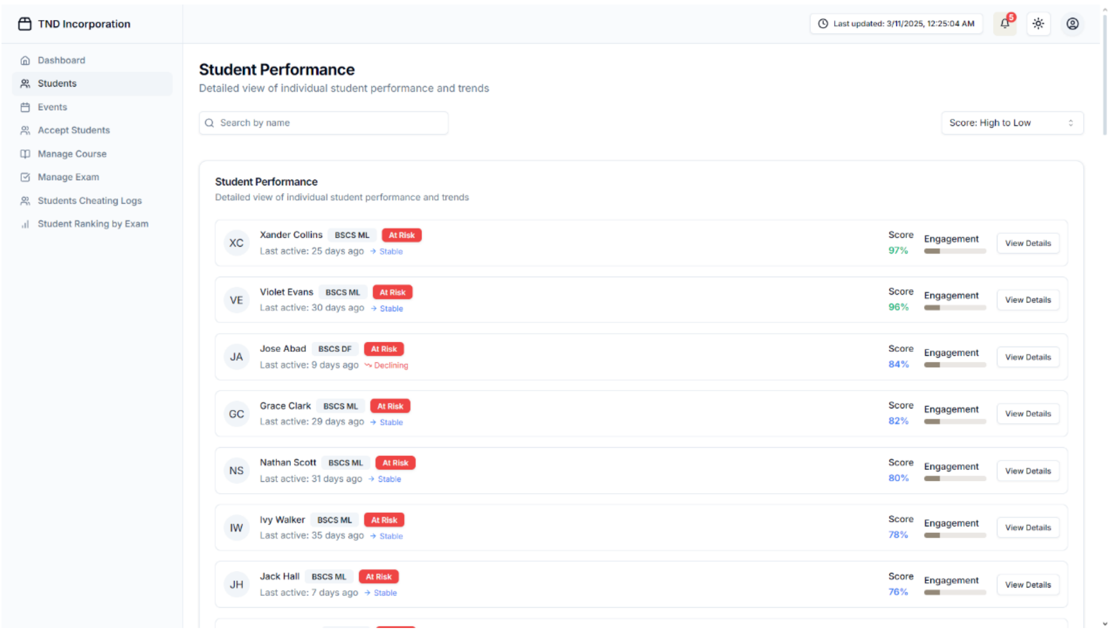


Figure 22: Teacher Dashboard - Student Performance Section

Student overview for teachers to see at risk students and their scores or grades together with their engagement on assessments

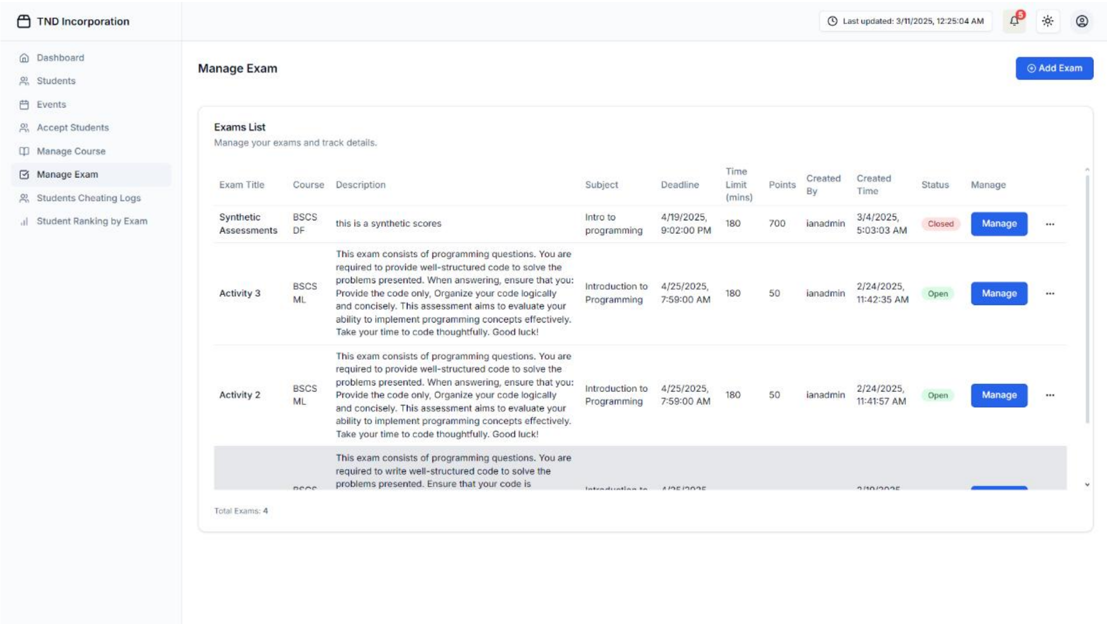


Figure 23: Teacher Managing Exams Section

Section for modifying and creating exams or assessments

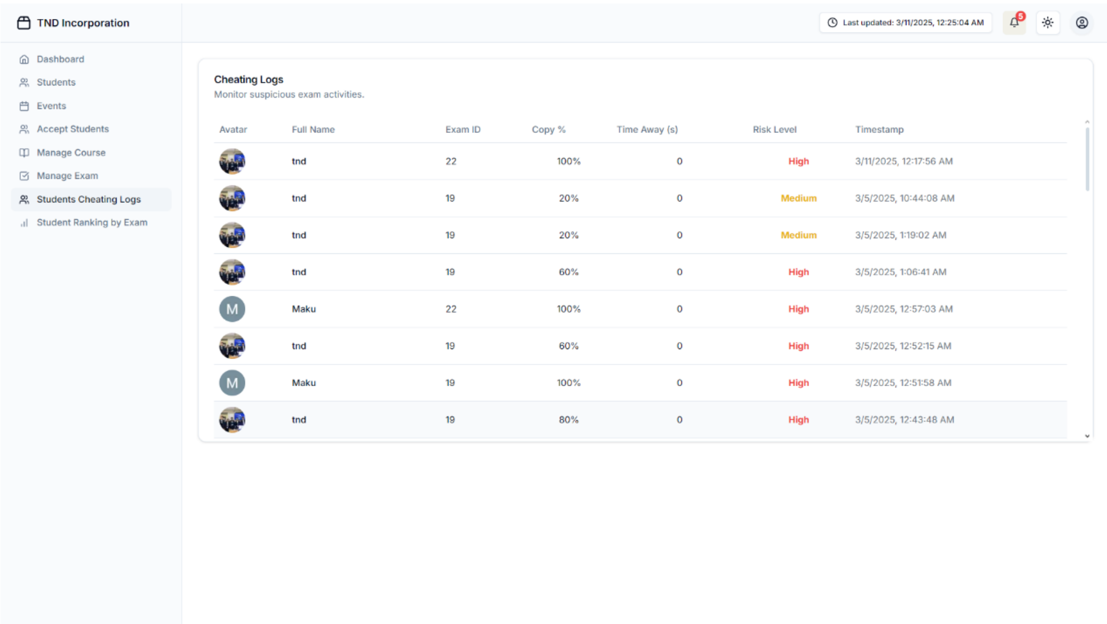


Figure 24: Cheating Logs

Cheating logs section for teachers to identify students who are copying and pasting from other sites

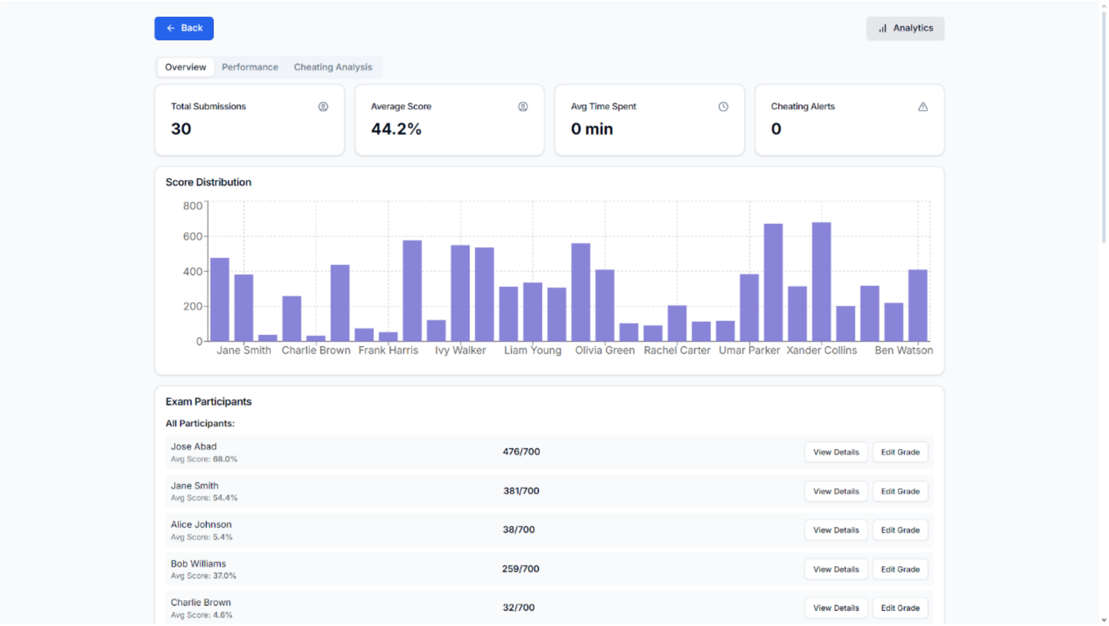
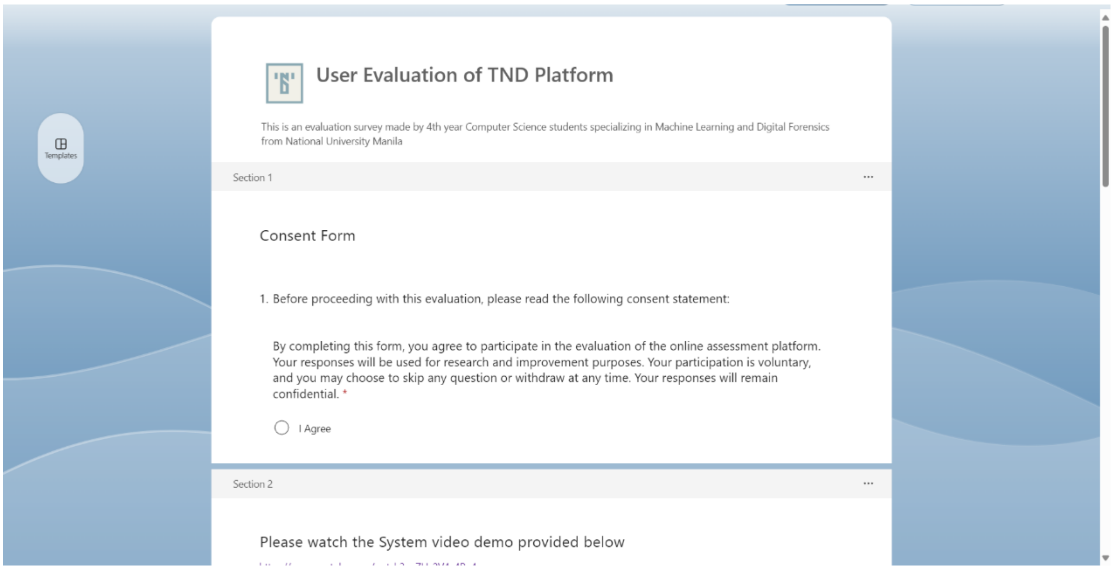
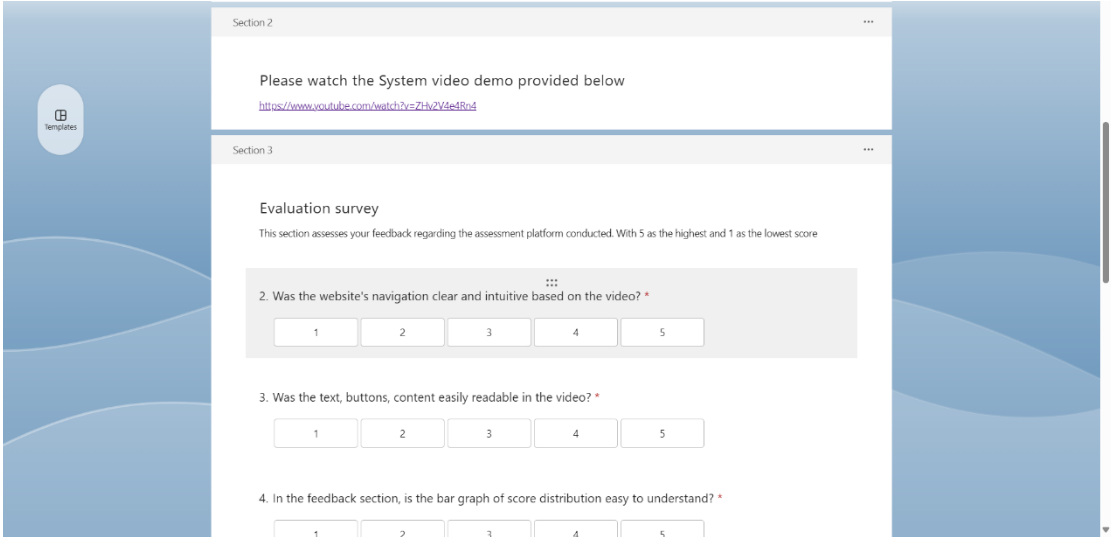


Figure 25: Assessment Analytics

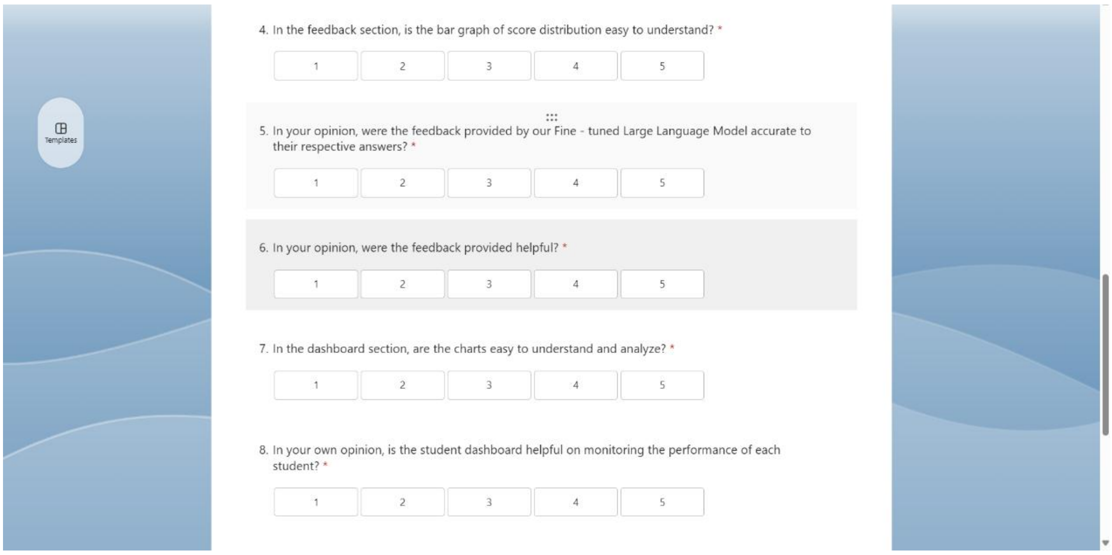
Analytics based on the chosen assessment, displayed here are the total of students who submitted, average score, average time spent, cheating analytics, and the overall score distribution with the participants or students

C: User Evaluation Survey

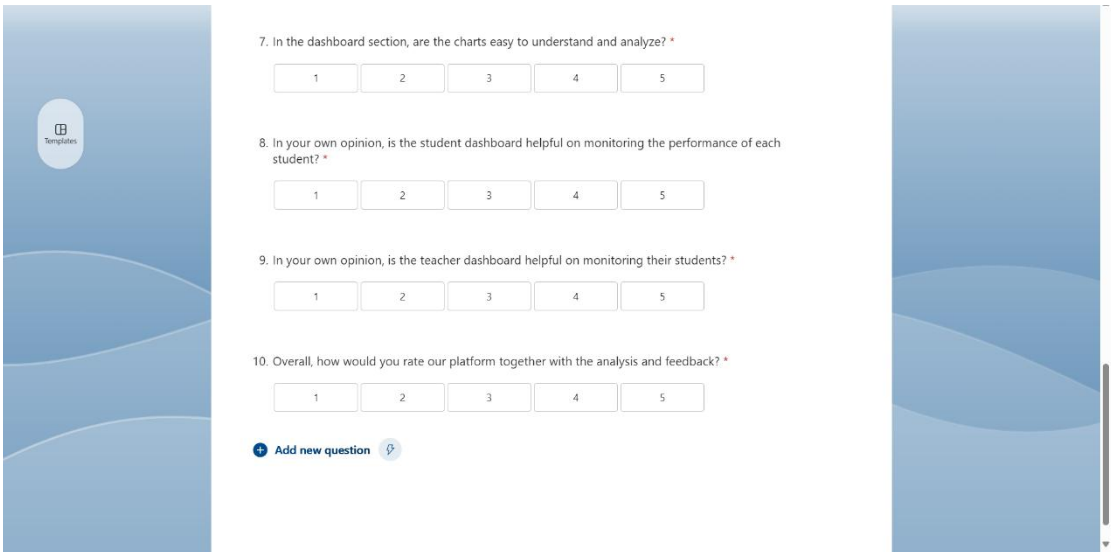




Templates
Section 2
Please watch the System video demo provided below
https://www.youtube.com/watch?v=ZHv2V4e4Rn4
Section 3
Evaluation survey
This section assesses your feedback regarding the assessment platform conducted. With 5 as the highest and 1 as the lowest score
2. Was the website's navigation clear and intuitive based on the video? *
1 2 3 4 5
3. Was the text, buttons, content easily readable in the video? *
1 2 3 4 5
4. In the feedback section, is the bar graph of score distribution easy to understand? *



Templates
4. In the feedback section, is the bar graph of score distribution easy to understand? *
1 2 3 4 5
5. In your opinion, were the feedback provided by our Fine - tuned Large Language Model accurate to their respective answers? *
1 2 3 4 5
6. In your opinion, were the feedback provided helpful? *
1 2 3 4 5
7. In the dashboard section, are the charts easy to understand and analyze? *
1 2 3 4 5
8. In your own opinion, is the student dashboard helpful on monitoring the performance of each student? *
1 2 3 4 5



Templates
7. In the dashboard section, are the charts easy to understand and analyze? *
1 2 3 4 5
8. In your own opinion, is the student dashboard helpful on monitoring the performance of each student? *
1 2 3 4 5
9. In your own opinion, is the teacher dashboard helpful on monitoring their students? *
1 2 3 4 5
10. Overall, how would you rate our platform together with the analysis and feedback? *
1 2 3 4 5
Add new question

E: Researchers Curriculum Vitae

Francheska L. Olympia

🖂 olympiafl@students.national-u.edu.ph

🖂 olympia.francheska.2@gmail.com

EDUCATION

Bachelor of Science in Computer Science specialization in Machine Learning

National University – Manila

TRAININGS / SEMINARS ATTENDED

INTERNSHIP

Stratpoint Global Outsourcing Inc.
Data Engineer Intern
April 2022 – September 2022

ONLINE TRAININGS

Participating in Workplace Communication
TESDA
2021

Receiving and Responding to Workplace Communication
TESDA
2021

Introduction to CSS
TESDA
2021

Installing and Configuring Computer Systems
TESDA
2021

Maintaining Computer Systems and Networks
TESDA
2021

Setting Up Computer Networks
TESDA
2021

Setting Up Computer Servers

TESDA

2021

CERTIFICATES

SQL (Basic) Certification

HackerRank, 2022s

Python (Basic) Certification

HackerRank, 2022

Java (Basic) Certification

HackerRank, 2023

SKILLS

Programming Languages: C, C#, C++, Java, Python

Web Development: HTML, CSS, Tailwind CSS, JavaScript (Node.js)

Databases & Query Languages: SQL, Database Management (Supabase), Amazon Athena

Big Data & Cloud Computing: PySpark (Basic), AWS (Amazon S3, AWS Glue, Amazon Athena)

Machine Learning: Data Processing (Pandas, Data Cleaning, Feature Engineering & Selection), Model Development (Scikit-learn, Supervised & Unsupervised Learning – Regression, Classification, Clustering), Data Visualization (Matplotlib),

UI / UX Design: Figma

Tools & Version Control: Git / GitHub, Jupyter Notebook & Google Colab

Leadership & Organizational Skills: Block Secretary (Documentation, Coordinated Meetings)

Francis Stephen R. Lee

✉ leefr@students.national-u.edu.ph

✉ stephenrflee@gmail.com

EDUCATION

Bachelor of Science in Computer Science specialization in Machine Learning

National University – Manila

SKILLS

Technical skills

Programming Languages: C, Java, Python

Web Development: HTML, CSS, Tailwind CSS, JavaScript (Node.js)

Databases & Query Languages: SQL, Database Management (Supabase)

Spreadsheet Tools: Microsoft Excel (Data Analysis, Pivot Tables, Charts)

Machine Learning & AI:

- **Deep Learning:** Neural Networks (TensorFlow, PyTorch), Convolutional Neural Networks (CNNs), Recurrent Neural Networks (RNNs), Transformers (BERT, GPT)
- **Reinforcement Learning:** Policy Gradient Methods, Q-Learning, Deep Q-Networks (DQN)
- **Natural Language Processing (NLP):** Tokenization, Sentiment Analysis, Text Classification, Text Generation (GPT, T5, XLNet)

Data Science & Analytics:

- **Data Processing:** Pandas (Data Cleaning, Feature Engineering & Selection)
- **Model Development:** Scikit-learn (Supervised & Unsupervised Learning – Regression, Classification, Clustering)

- **Data Visualization:** Heatmap, Matplotlib, Seaborn
- **Statistical Analysis & Predictive Modeling:** Hypothesis Testing, Time Series Analysis, Forecasting

Computer Vision:

- Image Preprocessing (Resizing, Grayscale Conversion, Edge Detection)
- Feature Extraction for Object Detection

Tools & Version Control: Git / GitHub, Jupyter Notebook & Google Colab

Work ethic skills & other skills:

Can work independently and reliably

Has creativity and time-management skills

Fluent in English and Filipino

Can work independently and reliably

Ian Jay D. Yuson

- yusonid@students.national-u.edu.ph
- ianyuson.personal@gmail.com

EDUCATION

Bachelor of Science in Computer Science specialization in Digital Forensics

National University – Manila (2022-2025)

Mobile App and Web Development

STI – Ortigas-Cainta (2018- 2022)

TRAININGS / SEMINARS ATTENDED

WORKSHOP

Web3 Development on Solana Blockchain

Solana Superteam Philippines

May 2024 – Aug 2024

ONLINE TRAININGS

Junior Cybersecurity Analyst Career

Cisco Networking Academy

June 2024 – Feb 2025

Ethical Hacker

Cisco Networking Academy

September 2024 – Feb 2025

Web Application Security

Portswigger

January 2025 – Feb 2025

CERTIFICATES

Junior Cybersecurity Analyst Career Path Exam

Cisco Networking Academy, Feb 4, 2025

3rd Place - Hack the Beat: Capture The Flag 2025

Junior Information Systems Security Association, Feb 15, 2025

Trend University Capture The Flag

Trend Micro, Aug 30 2024

Special Award – Whack A Thon

Blockchain Campus Conference, June 14, 2024

Civil Service Exam Passer

Civil Service Commission, Aug 11, 2024

SKILLS

Cybersecurity: Incident response utilizing skills in digital forensics, mobile forensics, computer forensics, network forensics, and malware analysis. Proficient in identifying system and web vulnerabilities using various tools and analyzing threats.

Capture the Flag (CTF) Competitions: Experience in identifying and exploiting vulnerabilities, solving security challenges, and applying ethical hacking techniques in a competitive environment.

Networking: Understanding of networking protocols acquired through coursework and Cisco materials. Foundational knowledge of network security concepts.

Machine Learning Basics: Introductory experience applying machine learning concepts through academic projects and coursework.

Programming Languages: C, C#, C++, Java, Python

Web Development: Full Stack Developer (Stack: Next JS), Web3 Developer

App Development: Proficient in using Unity to create mobile games and Android Studio for app development.

John Victor G. Corpus

✉ corpusjg@students.national-u.edu.ph

✉ corpusjv9@gmail.com

EDUCATION

Bachelor of Science in Computer Science specialization in Digital Forensics

National University – Manila

TRAININGS / SEMINARS ATTENDED

UX Analysis and Design Thinking Mentorship
Nov 27, 2024

Introduction to Cybersecurity
Cisco
March 2024

Geek Speaks
November 30, 2024

Learnership Series 4.2 Mautax 2.0
AIPO and Academic Services
September 25, 2024

University Capture the Flag
Trend Micro
Aug 30, 2024

SKILLS

Programming Languages: C, C#, C++, Java, Python, PHP

Web Development: HTML, CSS, Tailwind CSS, JavaScript (Node.js)

Databases & Query Languages: SQL, Database Management (Supabase), MySQL,

Cybersecurity: Incident Response, Malware Analysis, Log Analysis, Network Analysis, Threat Detection, Encryption & Cryptography, Mobile Penetration Testing.

Forensic Tools: FTK, Encase, Wireshark, Autopsy, Scalpel, DCode, Recuva.

UI / UX Design: Figma, SceneBuilder

Marchuzs Deimar V. Cortez

✉ cortezmv@students.national-u.edu.ph

✉ cortezmarchuzs@gmail.com

EDUCATION

Bachelor of Science in Computer Science specialization in Machine Learning

National University – Manila

SKILLS

Programming Languages: Java, Python

Web Development: HTML, CSS, JavaScript (Node.js)

Databases & Query Languages: SQL, Database Management (Supabase),

Machine Learning & AI:

- **Deep Learning:** Neural Networks (TensorFlow, PyTorch), Convolutional Neural Networks (CNNs), Recurrent Neural Networks (RNNs), Transformers (BERT, GPT)
- **Reinforcement Learning:** Q-Learning, Deep Q-Networks (DQN)
- **Natural Language Processing (NLP):** Tokenization, Sentiment Analysis, Text Classification, Text Generation (GPT, T5, XLNet)

Data Science & Analytics:

- **Data Processing:** Pandas (Data Cleaning, Feature Engineering & Selection)
- **Model Development:** Scikit-learn (Supervised & Unsupervised Learning – Regression, Classification, Clustering)
- **Data Visualization:** Heatmap, Matplotlib, Seaborn
- **Statistical Analysis & Predictive Modeling:** Hypothesis Testing, Time Series Analysis, Forecasting

UI / UX Design: Figma

Tools & Version Control: Git / GitHub, Jupyter Notebook & Google Colab

REFERENCES

Alsabhan, W. (2023). Student cheating detection in higher education by implementing machine learning and LSTM techniques. Sensors, 23(8), 4149. https://doi.org/10.3390/s23084149

Boduch, A., & Derks, R. (2020). React and React Native: A complete hands-on guide to modern web and mobile development with React.js (3rd ed.). Packt Publishing.

Cahapay, M. B. (2021). Problems encountered by college students in online assessment amid COVID-19 crisis: A case study. International Journal of Computer Science and Information Technology, 6(1), 1-10. https://doi.org/10.21742/ijcsite.2021.6.1.01

Carnes, M. (2022). Supabase: The opensource Firebase alternative. Retrieved from https://supabase.com/

Dann, C., O'Neill, S., Getenet, S., Chakraborty, S., Saleh, K., & Yu, K. (2024). Improving Teaching and Learning in Higher Education through Machine Learning: Proof of Concept' of AI's Ability to Assess the Use of Key Microskills. Education Sciences, 14(8), 886. https://doi.org/10.3390/educsci14080886

De Guzman, M. F. D., Edaño, D. C., & Umayan, Z. D. (2017). Understanding the essence of outcomes-based education (OBE) and knowledge of its implementation in a technological university in the Philippines. Asia Pacific Journal of Multidisciplinary Research, 5(4), 64-71. https://www.apjmr.com/wp-content/uploads/2017/11/APJMR-2017.5.4.08.pdf

Dolcetti, G., Arceri, V., Iotti, E., Maffeis, S., Cortesi, A., & Zaffanella, E. (2024). Helping LLMsimprove code generation using feedback from testing and static analysis. arXiv. https://arxiv.org/abs/2412.14841

Duhaim, A. M., Al-mamory, S. O., & Mahdi, M. S. (2022). Cheating Detection in Online Exams during Covid-19 Pandemic Using Data Mining Techniques. Webology, 19(1), 341-366. https://doi.org/10.14704/WEB/V19I1/WEB19026.

Espinosa, A. A., Gomez, M. A. C., Miranda, P. A., David, A. P., Abulon, E. L. R., Hermosisima,M. V. C., Quinosa Jr., E. A., Soliman, A. A., De Vera, J. L., Claros, I. H. A., Cruz, H. G. M., & Gonzales, N. S. (2023). Technology in education: A case study on the Philippines. Background paper prepared for the 2023 Global Education Monitoring Report: Technology and education, Southeast Asia. https://doi.org/10.54676/XKHI4627

Estrellado, C. J. P., & Miranda, J. C. (2023). Artificial intelligence in the Philippine educational context: Circumspection and future inquiries. International Journal of Scientific and Research Publications, 13(5), 1-6. https://www.ijsrp.org/research-paper-0523/ijsrp-p13704.pdf

Heil, J., & Ifenthaler, D. (2023). Online assessment in higher education: A systematic review. Education and Information Technologies, 28(1), 345-370. https://files.eric.ed.gov/fulltext/EJ1382640.pdf

Institute of Electrical and Electronics Engineers. (n.d.). IEEE Code of Ethics. Retrieved October 3, 2024, from https://www.ieee.org/about/corporate/governance/p7-8.html

Jagwani, A., & Aloysius, S. (2019). A REVIEW OF MACHINE LEARNING IN EDUCATION. ResearchGate.https://www.researchgate.net/publication/352932002_A_REVIEW_OF_MACHINE_LEARNING_IN_EDUCATION

Kaddour, J., Harris, J., Mozes, M., Bradley, H., Raileanu, R., & McHardy, R. (2023). Challenges and applications of large language models. https://arxiv.org/abs/2307.10169

Kamalov, F., Calonge, D. S., & Gurrib, I. (2023). New era of Artificial intelligence in Education: Towards a sustainable Multifaceted Revolution. Sustainability, 15(16), 12451. https://doi.org/10.3390/su151612451

Khosravi, H., Buckingham Shum, S., & Chen, G. (2021). Learning analytics dashboards to support advisor-student dialogue: An evaluation of functionality and usability. International Journal of Artificial Intelligence in Education, 31, 1–28. https://doi.org/10.1007/s40593-020-00210-5

Koutcheme, C., Hellas, A., Dainese, N., Leinonen, J., Sarsa, S., & Denny, P. (2024). Open-source language models can provide feedback: Evaluating LLMs' ability to help students using GPT-4-as-a-judge.arXiv. https://arxiv.org/abs/2405.05253

Leslie Ching Ow Tiong, HeeJeong Jasmine Lee, & Kai Li Lim. (2022). Online\ Assessment Misconduct Detection using Internet Protocol and Behavioural Classification. Computational Science Research Center, Korea Institute of Science and Technology, Seoul, Republic of Korea. Retrieved from https://arxiv.org/pdf/2201.13226

Mardan, A. (2018). Full Stack JavaScript: Learn Backbone.js, Node.js and MongoDB (2nd ed.). Apress.TARGET

Masiello, I., Mohseni, Z., Palma, F., Nordmark, S., Augustsson, H., & Rundquist, R. (2024). A current overview of the use of learning analytics dashboards. Education Sciences, 14(1), 82. https://doi.org/10.3390/educsci14010082

OpenAI. (n.d.). Fine-tuning. OpenAI Platform. Retrieved from https://platform.openai.com/docs/guides/fine-tuning

Meyer, J., Jansen, T., Schiller, R., Liebenow, L. W., Steinbach, M., Horbach, A., & Fleckenstein, J. (2023). Using LLMs to bring evidence-based feedback into the classroom: AI-generated feedback increases secondary students' text revision, motivation, and positive emotions. Computers and Education Artificial Intelligence, 6, 100199. https://doi.org/10.1016/j.caeai.2023.100199

Mian, Y. S., Khalid, F., Wong, A. C. Q., & Ismail, S. S. (2022). Learning analytics in education, advantages and issues: A systematic literature review. Creative Education, 13(09), 2913-2920. https://doi.org/10.4236/ce.2022.139183

Moon, S., Chae, H., Song, Y., Kwon, T., Kang, D., Ong, K. T., Hwang, S.-w., & Yeo, J. (2024). Coffee: Boost your code LLMs by fixing bugs with feedback. Yonsei University; Seoul National University https://arxiv.org/html/2311.07215v2?fbclid=IwY2xjawIoKPdleHRuA2FlbQ MAABH wTZaRVON R5JwoA1ZnSLWAH1Gm98WOg50wPgFCqZdAVaYqx 8BBslyHQ_aem_X_yxKyAWtTWeVEkAnlebyg

Mukhamediev, R. I., Popova, Y., Kuchin, Y., Zaitseva, E., Kalimoldayev, A., Symagulov, A., Levashenko, V., Abdoldina, F., Gopejenko, V., Yakunin, K., Muhamedijeva, E., & Yelis, M. (2022). Review of Artificial Intelligence and Machine Learning Technologies: Classification, Restrictions, opportunities and Challenges. Mathematics, 10(15), 2552. https://doi.org/10.3390/math10152552

Mukred, M., Mokhtar, U. A., Hawash, B., AlSalman, H., & Zohaib, M. (2024). The adoption and use of learning analytics tools to improve decision making in higher learning institutions: An extension of technology acceptance model. Heliyon, 10(4), e26315. https://doi.org/10.1016/j.heliyon.2024.e26315

National Privacy Commission. (n.d.). Data Privacy Act of 2012 (Republic Act No. 10173). Retrieved October 3, 2024, from https://privacy.gov.ph/data-privacy-act/

Pardo, A., & Siemens, G. (2014). Ethical and privacy principles for learning analytics. British Journal of Educational Technology, 45(3), 438–450. https://doi.org/10.1111/bjet.12152

Patil, R., & Gudivada, V. (2024). Review of current trends, techniques, and challenges in large language models (LLMs). https://www.mdpi.com/2076-3417/14/5/2074

Păvăloaia, V., & Necula, S. (2023). Artificial Intelligence as a Disruptive Technology—A Systematic Literature Review. Electronics, 12(5), 1102. https://doi.org/10.3390/electronics12051102

Popenici, S. a. D., & Kerr, S. (2017). Exploring the impact of artificial intelligence on teaching and learning in higher education. Research and Practice in Technology Enhanced Learning, 12(1). https://doi.org/10.1186/s41039-017-0062-8

Salido, V. (2023). Impact of AI-powered learning tools on student understanding and academic performance. Cavite State University. https://doi.org/10.13140/RG.2.2.37666.92483

Sharif, H., & Atif, A. (2024). The Evolving Classroom: How learning analytics is shaping the future of education and feedback mechanisms. Education Sciences, 14(2), 176. https://doi.org/10.3390/educsci14020176

Smith, C. E., Shiekh, K., Cooreman, H., Rahman, S., Zhu, Y., Siam, M. K., Ivanitskiy, M., Ahmed, A. M., Hallinan, M., & Grisak, A. (2023). Early adoption of generative artificial intelligence in computing education: Emergent student use cases and perspectives in 2023. Proceedings of the 2023 ACM Conference on Innovation and Technology in Computer Science Education. https://doi.org/10.1145/3649217.3653575

Specht, M. (2024). Data Analytics Dashboards in Education: Enhancing Learning through Visualization. Educational Technology Journal, 35(2), 45-59.

Specht, M., Jivet, I., Scheffel, M., & Drachsler, H. (2018). License to evaluate: Preparing learning analytics dashboards for educational practice. In Proceedings of the 8th International Conference on Learning Analytics and Knowledge (pp. 31–40). ACM. https://doi.org/10.1145/3170358.3170421

Supabase. (n.d.). Supabase: The open source Firebase alternative. Retrieved from https://supabase.com/

The Code affirms an obligation of computing professionals to use their skills for the benefit of society. (n.d.). https://www.acm.org/code-of-ethics

Timotheou, S., Miliou, O., Dimitriadis, Y., Sobrino, S. V., Giannoutsou, N., Cachia, R., Monés, A. M., & Ioannou, A. (2022). Impacts of digital technologies on education and factorsinfluencing schools' digital capacity and transformation: A literature review. Education and Information Technologies, 28(6), 6695–6726. https://doi.org/10.1007/s10639-022-11431-8

Vergara, D., Lampropoulos, G., Antón-Sancho, Á., & Fernández-Arias, P. (2024). Impact of Artificial intelligence on learning Management Systems: A bibliometric review. Multimodal Technologies and Interaction, 8(9), 75. https://doi.org/10.3390/mti8090075

Wang, S., Xu, T., Li, H., Zhang, C., Liang, J., Tang, J., Yu, P. S., & Wen, Q. (2024). Large language models for education: A survey and outlook. arXiv preprint arXiv:2403.18105. https://arxiv.org/abs/2403.18105

Wei, J., Wang, X., Schuurmans, D., Bosma, M., Xia, F., Misra, V., ... & Zoph, B. (2023). Finetuned language models are zero-shot learners. arXiv https://arxiv.org/pdf/2109.01652