BINDURA UNIVERSITY OF SCIENCE EDUCATION

FACULTY OF SCIENCE AND ENGINEERING

DEPARTMENT OF COMPUTER SCIENCE

AI-POWERED SYSTEM FOR PREDICTING STUDENT PERFORMANCE IN PYTHON PROGRAMMING WITHIN THE ZIMBABWEAN O-LEVEL COMPUTER SCIENCE CURRICULUM.

By

TAKUDZWA MATANGA

B211143B

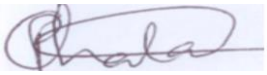
SUPERVISOR:

A RESEARCH PROJECT SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE BACHELOR OF SCIENCE HONOURS DEGREE IN SOFTWARE ENGINEERING

2025

APPROVAL FORM

The undersigned certify that they have supervised the student Matanga Takudzwa's dissertation entitled "AI-POWERED SYSTEM FOR PREDICTING STUDENT PERFORMANCE IN PYTHON PROGRAMMING WITHIN THE ZIMBABWEAN O-LEVEL COMPUTER SCIENCE CURRICULUM: *A Case Study of Masvingo Christian Secondary School*" submitted in Partial fulfilment of the requirements for the Bachelor of Software Engineering Honours Degree of Bindura University of Science Education.

T. MATANGA.....		20…/…06/ 2025…..
STUDENT	SIGNATURE	DATE
G. MHLANGA.....……		…06…/…08./2025…
SUPERVISOR	SIGNATURE	DATE
P CHAKA……	 .	06…/…08/.2025……
CHAIRPERSON DATE	SIGNATURE	DATE

Declaration

I, **TAKUDZWA MATANGA (B211143B)**, hereby declare that this research project, titled "AI-POWERED SYSTEM FOR PREDICTING STUDENT PERFORMANCE IN PYTHON

PROGRAMMING WITHIN THE ZIMBABWEAN O-LEVEL COMPUTER SCIENCE CURRICULUM," is my own original work and has not been submitted in part or in full for any other degree or diploma at Bindura University of Science Education or any other institution.

All sources consulted have been duly acknowledged by means of reference.

Signature:
Date: 20/06/25

Acknowledgement

I would like to express my sincere gratitude to everyone who supported me throughout this project. Firstly, I thank the Almighty God for His unwavering guidance, strength, and blessings, without

which this endeavour would not have been possible. My heartfelt appreciation goes to my beloved mother, whose endless love, encouragement, and sacrifices provided the foundation for my perseverance. I am immensely grateful to my esteemed lecturer, Mr. Mhlanga, for his invaluable academic guidance, insightful feedback, and continuous motivation, which were crucial to the successful completion of this research. Finally, I extend a special thank you to my dear friend, Laurels, and my girlfriend, Tafadzwa for their steadfast emotional support, understanding, and encouragement during the challenging moments of this project. Their belief in me made a significant difference.

Abstract

This research presents the development and evaluation of an AI-powered system designed to predict student performance in Python programming within the Zimbabwean O-Level Computer

Science curriculum. Addressing challenges prevalent in under-resourced educational settings, such as limited teacher proficiency and inadequate digital learning support, the system employs a Random Forest algorithm to analyse student interactions with multiple-choice quizzes. The methodology integrates a mixed-methods, quasi-experimental research design with an agile, user-centred system development approach, combining quantitative data from quiz results with qualitative feedback from students and teachers. The machine learning model achieved an 88% accuracy in predicting student performance, effectively identifying at-risk learners and highlighting challenging topics like 'Loops' and 'Data Structures'. Qualitative findings affirmed the system's usability, effectiveness, and practical value, with teachers appreciating data-driven insights and students reporting enhanced motivation. The study successfully demonstrated the potential of AI to offer predictive insights and analytical tools tailored for local contexts, thereby empowering educators and fostering data-informed learning. Recommendations include integrating the system into the curriculum, developing teacher training programs, and exploring future research avenues such as longitudinal impact studies and adaptive learning path generation. While acknowledging limitations such as sample size and reliance on quiz data, this research contributes significantly to the field of technology-enhanced learning in developing countries.

Table of Contents

Chapter 1

1.0 Aim of the Study

Developing an AI-Powered System for Predicting Student Performance in Python Programming within the Zimbabwean O-Level Computer Science Curriculum. The system is designed to provide data-driven insights that support teachers and students in improving programming proficiency, especially in under-resourced educational settings.

1.1 Introduction

Education is by far the most fundamental component of human development, equipping individuals with essential knowledge and skills to address problems within the community. The traditional approach of the educational system has a unified approach where each individual student is expected to learn at the same rate as the next student regardless of their individuality in learning abilities.

Teaching programming, especially languages like Python, can be challenging. Traditional teaching methods often do not meet the different learning needs of students, causing some to struggle with understanding and applying what they learn. This issue is even more serious in schools with limited resources, like many in Zimbabwe, where students may not have adequate access to technology or personalized help. As a result, there is often a gap between what the curriculum aims to teach and what students actually learn in practice. To solve this problem, we need new and creative ways to make computer science education more accessible and effective.

This project seeks to tackle this challenge by developing an AI-powered system to predict and ultimately improve student learning in Python programming for the Zimbabwean O-Level Computer Science curriculum, with a focus on Masvingo Christian College. Using machine learning, the system will leverage predictive analytics to assess student performance in Python programming. Specifically, it will utilize a Random Forest algorithm, a popular machine learning model known for its robustness and accuracy in classification tasks. This algorithm will analyse student interactions with the system, including quiz results and progress over time, to predict their future performance and identify areas of struggle.

Commented [Ma1]: The title talks about predicting, this part talks about improving. Can you sync the two

Rather than providing generalized feedback, the system will offer tailored, data-driven insights for each student. By analysing trends in quiz performance, time spent on tasks, and topic-level weaknesses, the system will help teachers make timely interventions. The architecture comprises a Flask backend, a React frontend, and MongoDB for data storage, ensuring real-time integration of prediction results with student activity data. By utilizing these predictions, the system aims to enhance the learning experience, providing more effective and personalized support to each student in the Python programming curriculum.

This chapter introduces the context and motivation for the study, outlines the central problem it seeks to address, and presents the objectives and research questions guiding the project. It also defines the scope and significance of the research. Through this work, the study aims to contribute to a more inclusive and data-informed approach to computer science education in Zimbabwe, ultimately helping learners become confident Python programmers equipped for the digital age.

1.2 Background of the Study

In Zimbabwe, the government has acknowledged the growing importance of technology in education by introducing Python programming into the O level Computer Science curriculum. This initiative is in alignment with the goals of the Heritage Based curriculum (2024-2025) which seeks to transform the education system to produce citizens with relevant skills aligned to national development (E.Huni, 2024). However, the major challenges are shortage of teachers who are proficient in Python, limited access to computers and reliable internet and students with varying levels of programming experience.

At Masvingo Christian High School, these challenges are particularly evident. Students struggle with Python programming concepts, leading to low performance in assessments and computer science final exams. The traditional, one-size-fits-all teaching approach does not cater to individual learning needs, leaving some students behind while others remain unchallenged. This lack of personalized learning can result in reduced engagement, slower concept mastery, and lower exam performance.

Commented [Ma2]: What are some of the reasons students struggle with Python?

One major issue with the current system is that student assessments and support mechanisms are too generalized. The curriculum often follows a one-size-fits-all approach, which does not consider individual learning needs. This lack of personalized learning can cause students to

lose interest or struggle if they require more tailored instruction (Kuldeep Singh Kaswan, 2024). Additionally, traditional assessments do not always provide timely feedback, making it difficult to identify students who are falling behind. Since grading is often done manually, it can be subjective and time-consuming, leading to inconsistent results. Another limitation is that the current system does not use data to predict student performance, making it harder to provide early support to those who need it.

To address these issues, this study proposes an AI-powered system that can predict and improve student performance in Python programming. At its core, the system uses machine learning—particularly a Random Forest algorithm to analyse students' quiz results and engagement data to predict future performance with a high degree of accuracy. This predictive capability allows the system to identify students who are struggling early on, enabling timely intervention. It also facilitates personalized learning by adapting the difficulty and focus of quiz questions to match each learner's progress. It will also give teachers valuable insights based on data, helping them make more informed teaching decisions.

The use of AI in education especially for personalized learning has already shown promising results. While existing AI-powered tools and EDM techniques have shown promise, a significant gap remains in the development of robust, offline AI-powered systems specifically tailored for predicting real-time student performance in Python programming within resource-constrained O-Level educational settings, such as those prevalent in Zimbabwe. Unlike many systems that rely on continuous internet connectivity or broad university-level data, this project addresses the unique challenges of limited infrastructure and focuses on actionable insights from specific quiz performance data to facilitate timely interventions, particularly within the context of the Zimbabwean curriculum.

Commented [Ma3]: So how is your proposed project going to differ from these tools? Or what gap exists in these tools?

Additionally, research in educational data mining (EDM) has shown that data analysis techniques can effectively predict student performance. By examining factors like previous academic performance and demographic data, AI can identify students who may struggle. Algorithms such as decision trees, neural networks, and k-nearest neighbours have proven useful for this purpose (Kabakchieva, 2012). (Kabakchieva, 2012) Specifically demonstrated how these algorithms can predict student success in university settings, highlighting their potential for education in general.

This research is being conducted within Zimbabwe's education system, which is continuously working to improve learning quality. The Ministry of Primary and Secondary Education has been encouraging the use of technology to enhance student outcomes. The introduction of Python programming at the O-Level is part of this effort. However, implementing AI in education also comes with challenges. Ethical concerns about data privacy and potential algorithmic biases need to be carefully addressed to ensure fair and responsible use (Kuldeep Singh Kaswan, 2024). Furthermore, having the right infrastructure and providing proper teacher training are crucial. While AI has many benefits, it is important to remember that education is ultimately about human interaction. As (Fahimirad, 2018) points out, AI should support and enhance human capabilities rather than replace them. This system is going to be used by teachers, rather than replace them.

This study aims to contribute to the growing research on AI in education, specifically in explore how AI can be harnessed to create a smarter, more adaptive learning system that meets the needs of both students and educators. By focusing on Python programming in Zimbabwe's O-Level curriculum, this research hopes to contribute valuable insights into the role of AI in shaping the future of computer science education in developing countries.

1.3 Statement of the Problem

Clearly outline your problem in just one paragraph

The inclusion of Python programming in Zimbabwe's O-Level Computer Science curriculum under the Heritage-Based Education Framework (2024–2025) aims to enhance digital literacy, but schools like Masvingo Christian College face significant implementation challenges, including students' lack of prior programming experience, limited computing resources, unreliable internet, and inadequate digital learning support, which hinder effective progress monitoring and personalized instruction. Traditional assessment methods, such as written exams, fail to provide timely, actionable feedback, leading to disengagement and delayed interventions for struggling learners. To address this, this study proposes an offline, AI-powered system that uses a Random Forest machine learning model to predict student performance through multiple-choice quiz analysis, enabling early identification of at-risk

students and data-driven teaching adjustments all while operating within resource constrained environments without requiring internet or advanced infrastructure.

1.4 Research Objectives

1. To design and develop an AI-Powered System for Predicting Student Performance in Python Programming within the Zimbabwean O-Level Computer Science Curriculum that predicts student performance in Python programming from multiple-choice quizzes.
2. To implement a machine learning model (Random Forest Classifier) capable of identifying at-risk students based on their quiz results in the Zimbabwean O-Level Computer Science curriculum.
3. To analyse student quiz performance data across various Python programming topics to identify learning trends, commonly challenging areas, and topic-specific performance gaps.
4. To collect and evaluate feedback from students and teachers at Masvingo Christian College on the usability and perceived effectiveness of the system in supporting Python programming learning.

1.5 Research Questions

1. How can an AI-powered system be designed and developed to effectively predict student performance in Python programming within the Zimbabwean O-Level curriculum?
2. How accurately can a Random Forest machine learning model identify students at risk of poor performance based on their Python quiz results?
3. What insights can be gained from analysing student quiz performance data across different Python topics, and how can these insights inform instructional strategies?
4. To collect and evaluate feedback from students and teachers at Masvingo Christian College on the usability and demonstrated impact of the system on student Python programming performance and learning, as well as its perceived effectiveness.

1.6 Research Propositions/Hypothesis

H_1 (Alternative Hypothesis): The Random Forest machine learning model implemented in the AI-powered system will achieve a high level of accuracy in predicting student performance and identifying at-risk students based on their Python quiz results.
H_0 (Null Hypothesis): The Random Forest machine learning model implemented in the AI-powered system will not achieve a high level of accuracy in predicting student performance and identifying at-risk students based on their Python quiz results.

Proposition: By leveraging machine learning techniques such as the Random Forest Classifier, student learning data can be used to accurately predict academic outcomes and support early intervention in Python programming education.

1.6 Scope of the Study

This study focuses on the development and evaluation of an AI-powered system designed to predict student performance in Python programming, specifically within the context of the Zimbabwean O-Level Computer Science curriculum. The system is tailored to the educational environment at Masvingo Christian College, a representative case study for resource-constrained schools in Zimbabwe.

The study is limited to Python programming content prescribed by the O-Level curriculum, including topics such as variables, operators, control structures, and input/output operations. The predictive model is based on structured quiz interaction data collected from students, such as scores, duration, frequency of attempts, and topic-specific performance. While behavioural factors such as attendance or extracurricular participation may be considered for extended insights, the system's primary input remains digital quiz data.

Only multiple-choice quizzes are used for data collection and prediction, excluding open-ended coding tasks due to challenges in automated evaluation and standardisation. The machine learning component is restricted to the use of the Random Forest classifier due to its suitability for classification tasks in educational settings. The evaluation of the system's usability and effectiveness is limited to feedback gathered from a small sample of teachers and students at Masvingo Christian College.

This study does not aim to replace teachers or existing pedagogical methods but rather to supplement them with data-driven insights to enhance early intervention and instructional decision-making. The infrastructure assumed for deployment includes a basic computer lab

environment with local access to the system, eliminating the need for full-time internet connectivity.

Open-source tools and technologies will be used throughout development to ensure accessibility, flexibility, and cost-effectiveness.

1.7 Justification/Significance of the Study

This study has meaningful implications for students, teachers, Masvingo Christian College, and the broader education sector in Zimbabwe.

For students, the AI-powered performance prediction system offers a data-informed approach to learning Python programming. By analysing quiz results and student interactions, the system identifies learning gaps and provides automated feedback tailored to individual performance. This supports learners in understanding their strengths and weaknesses, allowing for focused revision and improved outcomes over time.

For teachers, the system serves as a valuable tool for monitoring student progress and understanding learning patterns. It enables educators to identify common challenging topics that students face and refine their teaching strategies accordingly. Additionally, by automating aspects of assessment and feedback, teachers can focus more on providing targeted support and guidance, enhancing the overall learning experience.

At the institutional level, Masvingo Christian College benefits from implementing an innovative digital learning platform that aligns with modern educational practices. By integrating technology into the curriculum, the school can enhance the quality of its computer science program, attract more students interested in technology-driven education, and position itself as a leader in progressive teaching methodologies.

For the education sector, this study contributes to the advancement of technology-enhanced learning in Zimbabwe. It addresses the growing need for digital literacy and programming skills, equipping students with competencies essential for the modern workforce. The findings from this research can serve as a model for other institutions looking to implement similar learning solutions, promoting nationwide improvements in computer science education. Furthermore, the insights gained can help inform policy decisions on the integration of technology in schools, supporting efforts to modernize the national education system.

Ultimately, this project aims to empower students with critical 21st-century skills, provide educators with valuable teaching tools, and support the integration of innovative learning methodologies into Zimbabwe's educational landscape.

1.8 Assumptions

1 Students and teachers have access to basic computer and internet resources.

2 The school administration is supportive of the project and willing to provide necessary data and resources.

3 Students are willing to engage with the AI-Powered System for Predicting Student Performance in Python Programming.

1.9 Limitations/Challenges

1 Data privacy and security concerns related to student code and assessment data.

2 Potential limitations in data availability and quality, especially in resource-constrained schools.

3 The system's effectiveness may be influenced by the availability of reliable internet connectivity.

4 The complexity of accurately analysing and providing feedback on student code.

1.11 Definition of Terms

1. **AI-Powered System**: A computer-based application that uses Artificial Intelligence techniques to perform tasks that typically require human intelligence, such as prediction or analysis.
2. **Random Forest Algorithm**: A machine learning classification model that uses an ensemble of decision trees to make predictions based on input features such as quiz results and performance data.
3. **O-Level Computer Science**: is an internationally recognized secondary school qualification in computer science, typically taken by students aged 14–16 (e.g., under

Cambridge IGCSE or Zimbabwe's national curriculum). It introduces foundational computing concepts.

4. **Zimbabwean Heritage-Based Curriculum**: An educational framework (2024–2025) introduced in Zimbabwe aimed at aligning learning with national values, identity, and the development of practical skills.
5. **Quiz Interaction Data**: Digital records of how students perform on quizzes, including scores, time taken, number of attempts, and question-level analytics.
6. **Machine Learning (ML):** A subset of AI that allows computers to learn from data and make predictions or decisions without being explicitly programmed.
7. **Adaptive Assessment**: A method of testing that dynamically adjusts the difficulty or content of questions based on a student's performance. (Not implemented in this system.)
8. **Digital Literacy**: The ability to effectively and critically use digital tools and technologies, particularly in education and problem-solving contexts.
9. **At-Risk Students**: Learners who are predicted to perform poorly or fall behind based on early performance indicators and learning patterns.
10. **Performance Analytics**: The examination of data related to student assessments, aiming to understand trends, identify learning gaps, and improve instruction.
11. **Open-Source Tools**: Software or technologies whose source code is publicly available for use, modification, and distribution, typically free of charge.
12. **Educational Data Mining (EDM):** A research field focused on analysing educational data to better understand student behaviour, performance, and learning outcomes.

Chapter 2: Literature Review

2.1 Introduction

The integration of Artificial Intelligence (AI) into education has introduced a transformative approach to teaching and learning, particularly in programming education where students often face difficulties grasping abstract and logical concepts. AI-based adaptive learning systems leverage machine learning, natural language processing, and data mining to personalize instruction by continuously analysing student performance and adjusting content delivery in real time. This represents a departure from traditional, one-size-fits-all models by offering targeted support that enhances student engagement, comprehension, and retention. In resource-constrained contexts such as Zimbabwean secondary schools, these technologies can help bridge educational gaps caused by limited access to trained teachers and infrastructure. According to **(Fahimirad, 2018)**, AI has the potential to revolutionize educational delivery through automation, personalization, and feedback mechanisms that better cater to individual learning needs. This literature review explores both the theoretical foundations and empirical studies of AI-powered adaptive learning systems, with a specific focus on their application in teaching Python programming to O-Level students in Zimbabwe.

2.2 Theoretical Literature

This section explores foundational theories and empirical evidence informing the development of an AI-powered system aimed at predicting and enhancing student performance in Python programming for the Zimbabwean O-Level Computer Science Curriculum. It draws on cognitive load theory, constructivist learning, artificial intelligence in education, and learning analytics.

2.2.1 Cognitive Load Theory (CLT) in Programming Education

Cognitive Load Theory (CLT) is fundamental to understanding the inherent difficulties novice programmers' face, particularly when learning a language like Python. As Sands (2019) observes, students with limited prior programming experience or underdeveloped problem-solving skills often encounter a significant increase in cognitive load when introduced to new programming concepts. This challenge is further compounded by traditional teaching methods, which frequently present abstract ideas without adequate scaffolding, often expecting students

to rapidly apply these concepts to complex, real-world problems. Sands (2019) emphasizes the dual challenge of mastering both syntax and application, underscoring the critical need for teaching strategies that effectively reduce cognitive load.

In this context, the proposed AI-powered system is specifically designed to manage and mitigate various types of cognitive load. It aims to minimize extraneous cognitive load (ECL), which arises from inefficient instructional design. This is achieved through a user-friendly and intuitive interface that reduces visual clutter and provides clearly formatted, standardized multiple-choice quizzes. By streamlining the presentation of information and interaction elements, the system enables students to concentrate their mental effort on core learning objectives rather than being distracted by poorly organized materials or confusing navigation.

Furthermore, the system proactively addresses intrinsic cognitive load (ICL), which relates to the inherent complexity of the Python content itself. It does this by intelligently structuring Python topics based on increasing complexity, presenting concepts in digestible, sequential modules. The system adapts the difficulty of quizzes and learning paths to a student's current mastery level, ensuring that foundational knowledge is firmly established before more advanced topics are introduced. This prevents overwhelming learners with too much new information at once.

Finally, the AI-powered system actively fosters germane cognitive load (GCL), which involves the cognitive effort directed towards deep learning and schema formation. By providing immediate, targeted feedback on quiz performance and precisely highlighting specific areas of weakness, the system encourages students to engage in active debugging, self-explanation, and reflective problem-solving. This comprehensive approach ensures personalized support that directly addresses the unique learning needs and resource constraints prevalent in Zimbabwean educational settings, ultimately guiding students towards a more profound understanding of Python programming concepts.

2.2.2 Constructivist Learning and Computational Thinking

Integrating constructivist learning theory with computational thinking is vital for effective programming education. (Csizmadia A. S., 2025) emphasize the importance of learner autonomy in constructing artefacts, proposing a matrix to assess the integration of constructionism in learning activities. Their research shows that activities allowing

independent problem definition, design, and coding foster deeper understanding and engagement. This aligns with the AI system's approach, which facilitates active, autonomous learning experiences through hands-on practice, enabling students to build their understanding of Python programming concepts.

(Lodi, 2018) Stressed that while programming aligns with constructionist principles, teaching it effectively requires addressing the relationship between code and its execution. They highlight the importance of the 'notional machine' concept, which helps learners understand how code translates into actions. Novice programmers often face challenges like misconceptions and confused computational models. To address these, (Lodi, 2018) recommend using abstract programming patterns, visual programming languages, and 'unplugged' activities to deepen understanding. These strategies, which encourage active learning and personal project creation, are integrated into the system through structured exercises and conceptual explanations of code behaviour.

2.2.3 The role of pedagogical AI Agents

Pedagogical agents, AI-driven tools designed to facilitate learning, offer valuable insights for developing AI systems to improve Python programming performance. Research on these agents, such as the taxonomy provided by Weber et al. (2021) on conversational agents in education, demonstrates their potential to influence learning outcomes significantly. AI, through these agents, can reduce extraneous cognitive load (ECL) by presenting material clearly, handling errors intelligently, and providing adaptive interfaces, allowing students to focus on core concepts. It can also increase germane cognitive load (GCL) through interactive problem-solving and self-explanation prompts, fostering deeper understanding. This supports the rationale for implementing an AI-powered performance prediction system in the Zimbabwean O-Level Computer Science Curriculum that emphasizes clarity and student engagement rather than recommendation systems.

2.2.5 Supporting engagement and self-efficacy through AI and online environments

(Sekhar, 2024) Explored the effectiveness of interactive teaching methods, such as Think Pair Share, Quality Circles, and Python Tutor visualizations, in enhancing Python programming skills among computer science students. Their case study found that these methods significantly improved engagement, comprehension, and retention compared to traditional

lectures. The AI system adopts similar interactive principles to engage O-Level students in a more application-focused and collaborative learning environment.

(Daradoumis, 2022) Examined how an online Distributed Systems Laboratory (DSLab) influenced students' programming self-efficacy and their perception of the learning environment. Their findings show that well-designed online tools can boost students' confidence in their programming abilities, particularly through immediate feedback and observing peer progress. This principle guides the AI system's feedback mechanisms, designed to build learner confidence through supportive and timely responses.

(Zainal, 2012) Studied university students, but their findings on perception and motivation are relevant to O-level students beginning their programming journey. Positive pre-course perceptions and intrinsic motivation are crucial at this stage. Engaging and supportive learning experiences can build confidence and skills, as intrinsically motivated students tend to achieve more. The AI system aims to foster a welcoming environment that sustains motivation through accessible content and regular reinforcement.

(Yilmaz, 2022) Investigated the impact of ChatGPT on undergraduate computer science students' computational thinking skills, programming self-efficacy, and motivation. Their pre-test post-test control group study found that ChatGPT-supported hands-on coding significantly enhanced these outcomes. This provides empirical evidence that integrating conversational AI or intelligent tutoring into systems can support programming education, validating the system's approach to feedback and learner interaction.

2.3 Empirical Literature

This section presents empirical evidence supporting the development of an AI-powered performance prediction system for Python programming education in the Zimbabwean O-Level curriculum. It highlights methodologies, key findings, limitations of existing systems, and identifies gaps addressed by this study.

2.3.1 Overview of AI in Education

The integration of Artificial Intelligence (AI) in education has enabled personalized and adaptive learning experiences. Systems using machine learning (ML), natural language processing, and educational data mining allow educators to tailor content and support based on individual learner needs. These systems shift away from one-size-fits-all teaching methods to provide real-time feedback and performance insights (Siemens & Long, 2011).

2.3.2 Predictive Models for Academic Performance

Empirical studies have explored various machine learning algorithms to predict student success. (Costa-Mendes, 2021) compared multiple linear regression (MLR) with ML models such as random forest, SVM, neural networks, and XGBoost to predict Portuguese high school grades. Their study involved a dataset of over 30,000 records and applied hyperparameter tuning and LASSO regression for feature selection. Random forest showed superior generalizability with an R^2 of 28.1%, demonstrating the effectiveness of ensemble models for academic prediction.

In the Zimbabwean context, such models can be applied using quiz interaction data, response time, and error patterns. The current system uses a Random Forest Classifier trained on Python quiz data to predict at-risk students, directly aligning with the methodologies used by (Costa-Mendes, 2021).

(Amardeep, 2023) developed an adaptive quiz platform with automated marking and dynamic difficulty levels. Their system mirrored cognitive load theory (CLT) strategies by reducing extraneous cognitive load through clear interfaces and real-time feedback. While their system was designed for GRE preparation, its structure provides a transferable foundation for Python programming assessment.

2.3.3 Adaptive Learning and Feedback Mechanisms

Rehan School's GPT-based virtual learning platform, studied by (Paniwani, 2024), showed improved test scores (average 19/20) among students in low-resourced environments. This system provided interactive, adaptive learning experiences that supported coding skills and

critical thinking. These findings validate the use of AI-powered feedback in enhancing learner performance and reducing cognitive overload, especially intrinsic load (ICL).

Similarly, (Yilmaz, 2022a) demonstrated that ChatGPT-assisted hands-on coding significantly improved computational thinking, motivation, and self-efficacy among undergraduate students. These results support the integration of intelligent feedback mechanisms in O-Level programming environments.

2.3.4 Case Studies in Low-Resource Settings

A case study by (Ntsobi, 2024) examined the implementation of AI-driven education in South Africa's Sci-Bono Discovery Centre, demonstrating that AI-powered tools significantly enhance student performance through personalized learning experiences and real-time feedback. The study emphasized the potential of AI to bridge learning gaps in STEM education. However, while this research provides insights into AI integration in primary and high school settings, it does not focus specifically on Python programming or the unique needs of O-Level students.

(Mathevula, 2014) Analysed ICT integration challenges in South African rural schools through surveys and infrastructure audits. Findings revealed barriers like unreliable internet access, lack of trained teachers, and limited access to digital devices. These challenges mirror those in Zimbabwean schools and highlight the need for context-aware AI systems. While not focused on AI, their study underscores the relevance of localized, low-bandwidth adaptive tools.

In response, this study proposes an AI-powered prediction tool deployable on low-cost infrastructure, emphasizing Python-specific content and usable in intermittent connectivity scenarios.

2.3.5 Gaps and Opportunities in Zimbabwean O-Level Python Education

However, most studies on AI in programming education focus on higher education institutions, leaving a gap in understanding how these technologies can support high school students, particularly those at the O-Level. Additionally, current AI-driven programming tools often require significant computational resources, which may not be available in underprivileged or rural schools (Rudhumbu, 2021).

Empirical research highlights several challenges hindering AI adoption in Zimbabwean schools, including inadequate infrastructure, lack of trained educators, and resistance to technology integration (Rudhumbu, 2021). A study on ICT adoption in Zimbabwean secondary schools found that while teachers acknowledge the importance of technology, limited resources and insufficient training prevent effective implementation (Rudhumbu, 2021).

Existing systems such as that of (Amardeep, 2023) focus on standardized testing or generic content. This project addresses foundational Python topics such as loops, variables, and data types. Unlike many AI in education (AIEd) systems which prioritize content recommendation (e.g., Khan Academy), this project focuses on prediction, offering data-driven insights into learner risk without complicating the infrastructure with content adaptation. In addition, this system builds on (Paniwani, 2024) success in low-resourced schools but applies it to Zimbabwe's curriculum, addressing specific barriers such as teacher shortages and limited computing resources.

Additionally, studies on AI in African higher education institutions emphasize digital inequality, where students in rural areas face significant challenges in accessing AI-powered learning tools (Funda, 2024). These disparities suggest that AI-based adaptive learning systems must be designed with accessibility and inclusivity in mind.

AI-based adaptive learning systems hold significant potential for transforming programming education, particularly in teaching Python at the O-Level. However, empirical studies reveal that accessibility, infrastructure limitations, and contextualization remain key challenges in implementing these systems in settings like Zimbabwean secondary schools. This study directly addresses these gaps by designing an AI-powered system specifically for offline functionality, thereby enhancing accessibility in environments with unreliable internet. Furthermore, by utilizing a lightweight technology stack (Flask, React, MongoDB) and focusing on multiple-choice quiz data, the system minimizes demands on advanced infrastructure and computing resources. Its development is deeply rooted in the Zimbabwean O-Level Computer Science Curriculum and tailored to the unique learning context of Masvingo Christian College, ensuring high contextualization. By actively tackling these practical barriers, this study will contribute to the growing body of knowledge on AI in education and offer demonstrably practical solutions for AI-driven programming instruction in low-resource settings.

This study seeks to address existing gaps by developing and evaluating an AI-based adaptive learning system specifically designed to teach Python programming to O-Level students in Zimbabwe. Unlike much existing research, which predominantly focuses on AI applications in tertiary education or general instructional settings, this study will provide crucial empirical insights into the effectiveness of adaptive learning technologies within a distinct secondary school context. At the O-Level, programming typically emphasizes foundational concepts, algorithmic thinking, and problem-solving using simplified Python syntax, often for students with limited to no prior computing exposure. This contrasts significantly with university-level programming, which often delves into complex data structures, advanced algorithms, and diverse programming paradigms, usually for students possessing a more robust computational background.

Therefore, a key contribution of this research is the development of a learning model and system meticulously optimized for the pedagogical and environmental realities of low-resource secondary school environments. This optimization ensures accessibility for students in schools with limited technological infrastructure by prioritizing offline functionality and a lightweight technology stack.

Additionally, this study will examine the ethical and pedagogical implications of AI-driven learning, with a particular focus on data privacy, fairness, and the evolving role of educators in AI-supported instruction. By conducting a structured evaluation of student engagement and performance in Python programming, this research will generate empirical data on the impact of adaptive learning systems on knowledge acquisition and motivation among secondary school learners.

Ultimately, this study will not only contribute to the growing body of knowledge on AI in education but also offer practical solutions for implementing AI-driven programming instruction in underprivileged educational settings.

2.3.6 Theoretical Linkages

Cognitive Load Theory (CLT)

Cognitive Load Theory, developed by (Sweller, 2011), explains how human working memory can be overloaded during complex learning tasks, particularly when learners encounter abstract or unfamiliar material. This is especially relevant for novice Python programmers. The system reduces extraneous cognitive load (ECL**)** by offering a clean user interface for the quiz tests and exercises and intuitive question design, minimizing distractions and irrelevant cognitive effort (Klepsch, 2017).

Commented [Ma4]: Is this user interface a programming environment or a quiz environment?

Intrinsic cognitive load (ICL) the mental effort required to understand the inherent complexity of Python topics is addressed by sequencing content gradually, starting from variables and progressing to loops and functions. To promote germane cognitive load (GCL), the system includes debugging tasks, formative quizzes, and error reflection to foster schema construction and deeper understanding (Sweller, 2011) (Lodi, 2018).

Constructivist Learning Theory and Computational Thinking

The design of the system draws heavily on constructivist learning principles, which emphasize learner autonomy, active knowledge construction, and engagement through problem-solving (Grover, 2013). The inclusion of interactive Python exercises and self-paced quizzes supports constructionist learning, where students build artefacts and learn through doing (Csizmadia A. S., 2025).

This model also promotes computational thinking, a key competency in the curriculum, by encouraging logical sequencing, pattern recognition, abstraction, and algorithm design through code-based assessments. These learning activities are intentionally structured to foster both understanding of syntax and computational reasoning.

Pedagogical Agents and Intelligent Tutoring Systems

Inspired by (VanLehn, 2011), who demonstrated that well-designed intelligent tutoring systems (ITS) can rival human tutoring, the AI system incorporates feedback mechanisms that serve as pedagogical agents. These agents guide students by explaining incorrect answers, reinforcing correct responses, and offering hints when necessary.

The approach aligns with (Luckin, 2016), who argue that AI should support, not replace, human teaching. In this system, the AI assists teachers by identifying struggling students through data-driven insights, allowing for timely and targeted intervention without compromising the teacher's central role.

Learning Analytics and Educational Data Mining (EDM)

Learning Analytics (LA) and Educational Data Mining (EDM) provide the analytical backbone of the system. According to (Siemens G. &., 2011), learning analytics involves collecting and analysing learner data to improve educational outcomes. In this system, data such as quiz scores, response times, and error frequency are continuously analysed to model student progress.

Following (Baker, 2011) and (Papamitsiou, 2014), the system uses a Random Forest algorithm trained on this data to predict student performance and identify at-risk learners. This predictive insight supports data-informed teaching, enabling educators to tailor instruction, adjust pacing, and focus on concepts most students find challenging.

2.4 Chapter Summary

Chapter 2 reviewed the theoretical and empirical foundations that inform the development of an AI-powered system for predicting student performance in Python programming within the Zimbabwean O-Level curriculum. It began by outlining Cognitive Load Theory (CLT), constructivist learning, pedagogical agents, and learning analytics as the guiding frameworks for the system's design, emphasizing reduced cognitive load, student autonomy, and data-driven feedback. Empirical literature supported the system's use of machine learning particularly Random Forest classifiers for predicting at-risk learners, drawing on studies from both global and African contexts. While existing AI tools show promise in adaptive learning and feedback, gaps remain in systems tailored specifically for secondary-level Python programming in low-resource environments. This chapter highlights the need for localized, accessible, and ethically grounded AI solutions that align with Zimbabwe's infrastructure and educational goals.

Chapter 3: Research Methodology

3.1 Introduction

Chapter 3 provides a meticulous account of the comprehensive research methodology and iterative system development approach employed in creating and evaluating an AI-powered system designed to predict student performance in Python programming within Zimbabwe's O-Level Computer Science Curriculum. This chapter outlines a mixed-methods, quasi-experimental research design, ensuring practical applicability and robust evaluation, and details the comprehensive data collection strategy, including quantitative quiz performance and qualitative feedback. It further elaborates on the implementation of the Random Forest Classifier for predictive analytics, covering data pre-processing, model training, and analytical techniques to identify learning trends and conceptual weaknesses. Finally, this chapter covers the rigorous system development, testing, and deployment strategies, highlighting the system's usability, effectiveness, and adaptability within resource-constrained educational environments, ultimately providing a clear roadmap for the subsequent detailed sections and showcasing how the developed system functions to support personalized learning and data-driven pedagogical interventions.

3.2 Research Design

This study employs a mixed-methods, quasi-experimental research design, meticulously integrated with an agile, user-centred systems development approach. This comprehensive design is particularly suited for evaluating the developed AI system's predictive capabilities, its data-driven insights, and its perceived utility within the practical context of a Zimbabwean O-Level Computer Science classroom.

The quasi-experimental component allows for the evaluation of the AI system's impact on student performance and learning trends within an existing educational setting at Masvingo Christian College. Given the inherent practical constraints in school environments, this approach facilitates a robust assessment without requiring full random assignment of participants, thereby enabling observation of the system's real-world application.

Complementing this, a mixed-methods approach is utilized to provide a holistic understanding of the system's impact. This involves combining quantitative measures and qualitative feedback. Quantitative data, encompassing quiz results, time spent per question, number of attempts, and model prediction accuracy, will be collected to analyse learning patterns and evaluate the AI model's effectiveness. Simultaneously, qualitative data, derived from feedback sessions with students and teachers, will capture nuanced insights into the system's usability, its perceived support for learning, and its overall effectiveness in facilitating teaching and learning processes.

The iterative nature of the AI system's development, which utilizes a Random Forest Classifier trained on topic-tagged quiz interaction data, directly informs and interacts with this research design. While the system avoids the complexity of a recommendation engine to ensure its lightweight and deployable nature in resource-constrained environments, the research design specifically aims to evaluate its core functionalities: predicting student performance based on quiz interaction patterns, identifying areas of conceptual weakness through trend analysis of incorrect responses and timing data, and generating actionable insights for teachers to enable timely academic intervention and support. This integrated design ensures that the system is both practically achievable within the O-Level school context and methodologically robust enough to generate valid and generalizable insights.

3.3 Data Collection Approaches

The data collection strategy for this study adopted a mixed-methods approach, integrating both quantitative and qualitative data. This comprehensive methodology was crucial for rigorously evaluating the AI-powered system, assessing the machine learning model's performance, and understanding user perceptions in a real-world educational context

3.3.1 Quantitative Data Collection

Quantitative data was primarily collected automatically by the developed AI system and its integrated online quiz platform. This data served as the foundational input for training and validating the Random Forest predictive model, as well as for conducting in-depth analysis of student learning patterns and performance over time. Data was collected from a total of 88 O-Level Computer Science students from Masvingo Christian College who actively engaged with the Python programming quizzes within the AI system throughout the third academic term of 2024.

3.3.1.1 Raw Data Collected by the System

For every quiz attempt, the system automatically recorded the following granular interaction log data:

- Student ID: A unique, anonymized identifier for each student, ensuring participant privacy.
- Quiz ID: A unique identifier for the specific quiz being attempted.
- Total Score: The raw score obtained for the entire quiz, indicating overall performance.
- Total time per Test: The precise time (in seconds) a student spent on each individual multiple-choice question. This granular data was aggregated to derive 'Duration Taken' per quiz.
- Number of Attempts (per test quiz): The number of times a student attempted a specific question before providing a correct answer.
- Answers Selected (per question): The specific option chosen by the student for each multiple-choice question, along with its correctness (correct/incorrect).

3.3.1.2 Derived Features for Model Input

From the raw interaction log data, the system automatically computed additional features that were crucial inputs for the Random Forest predictive model:

- Duration Taken (per quiz): The total time (in minutes) a student spent on the entire quiz test, derived from the 'Time per Question' data.
- Retention Factor: Calculated based on a student's performance on repeated quiz attempts for the same topic over time. This factor provided a quantitative indicator of knowledge persistence and decay.
- Weak Topic Metrics: Derived from the aggregated 'incorrect responses' at both the class level and for individual students on topic-tagged questions. This metric quantitatively identified areas of conceptual weakness in Python programming (e.g., 'Loops', 'Data Structures').

3.3.1.3 Pre- and Post-Test Scores

To evaluate the system's impact on overall learning gains, students completed two identical, structured assessments: one pre-test at the beginning of the learning cycle (start of the term) and a post-test at the end. These tests assessed foundational Python programming concepts

covered by the curriculum. The scores from these assessments, out of 50 marks, were recorded to measure learning gains attributable to the intervention.

3.3.1.4 Synthetic Data Generation

To augment the real dataset and enhance the robustness of the machine learning model, particularly in addressing potential class imbalances and improving generalization, 500 synthetic quiz records were generated. This synthetic data was carefully simulated to mimic the statistical properties and patterns observed in the real student quiz interaction data, ensuring its relevance for pre-training and supplementing the model's learning process. This combined dataset (370 real + 500 synthetic = 870 records total) formed the basis for the model's training, while a distinct subset of 74 real records was reserved for independent testing as outlined in Chapter 4.

3.3.2 Qualitative Data Collection

Qualitative data was gathered to provide rich, in-depth insights into the lived experiences and perceptions of both students and teachers regarding the usability, practical applicability, and overall effectiveness of the AI-powered system. This approach allowed for a nuanced understanding that quantitative data alone could not provide, directly addressing Research Objective 4.

3.3.2.1 Teacher Interviews

Participants: Three (3) Computer Science teachers from Masvingo Christian College, who regularly utilized the AI system in their teaching during the trial period, were purposively selected for individual semi-structured interviews. This selection aimed to capture perspectives from key system users.

Procedure: Interviews, each lasting approximately 10-15 minutes, were conducted in a quiet setting within the school premises to ensure privacy and minimize distractions. The interview protocol focused on exploring key themes such as the system's Practicality and Usability, its Perceived Benefits for student learning and teaching efficiency, any Challenges and Limitations encountered, and Suggestions for Improvement. All interviews were audio-

recorded with explicit informed consent from the participants and subsequently transcribed verbatim for detailed thematic analysis.

3.3.2.2 Student Focus Group Discussions

Participants: Four (4) focus group discussions (FGDs) were conducted with a total of 20 students, divided into groups of five (5) participants each. Students were purposively selected from the cohort who had consistently engaged with the AI system, ensuring a range of performance levels and engagement patterns were represented.

Procedure: Each Focus Group Discussion lasted approximately 10-15 minutes and was facilitated by a trained researcher following a discussion guide. The discussions explored various aspects of the student experience, including their User Experience (UX) with the interface, the system's impact on their Motivation and Engagement to learn Python, the perceived Usefulness and Clarity of the Performance Feedback provided, and any technical or usability Challenges encountered. Discussions were audio-recorded with explicit assent from the students and consent from their parents/guardians, and subsequently transcribed verbatim.

3.3.2.3 System Observations (Classroom Trials)

Procedure: Direct observations were systematically conducted by the researcher during four (4) scheduled Python programming sessions where students were actively interacting with the AI system. The observations focused on capturing real-time student engagement patterns, common difficulties with the interface or navigation, and general classroom dynamics related to the system's use. Meticulous field notes were taken during these sessions, documenting specific interactions, observable technical limitations, and any barriers or facilitators to usability. These observations provided crucial contextual understanding of the system's real-world application, complementing the self-reported data from interviews and focus groups.

3.3.3 Ethical Considerations

Prior to any data collection, rigorous ethical protocols were observed to ensure the protection and well-being of all participants. Formal ethical approval for the study was obtained from the Bindura University of Science Education Ethics Committee Informed written consent was secured from all participating teachers. For student participants, given their minor status, informed written consent was obtained from their parents/legal guardians, in addition to explicit verbal or written assent from the students themselves. All participants were thoroughly

informed about the study's purpose, procedures, their right to withdraw at any time, and the measures taken to ensure their privacy and confidentiality. Qualitative data (interview and FGD transcripts) was anonymized immediately after transcription, removing any personally identifiable information and replacing it with unique alphanumeric codes. All collected data, both quantitative and qualitative, was stored securely on password-protected, encrypted drives accessible only to the designated research team members, ensuring strict compliance with data protection principles and ethical guidelines.

3.4 Requirements Analysis

This section outlines the systematic process undertaken to identify, gather, analyse, and document the specific requirements for the AI-powered system designed to predict student performance in Python programming. This phase was critical in translating the broad research objectives into tangible system functionalities, ensuring the developed solution directly addresses the pedagogical needs of teachers and the learning challenges faced by students within the Zimbabwean O-Level Computer Science curriculum and its prevailing resource constraints. The requirement analysis adopted an iterative and user-centred approach, integrating continuous feedback from key stakeholders.

3.4.1 Functional Requirements (FRs)

The following core functional requirements define what the AI-powered system shall do to achieve its objectives:

1. Student Quiz Data Capture and Storage: The system shall automatically capture and securely store granular student quiz interaction data.
2. AI-Powered Performance Prediction: The system shall employ a trained Machine Learning model to accurately predict student performance trends.
3. Teacher Analytics Dashboard Provision: A ranked list of students at risk based on AI predictions, retention factors class-wide summaries of weak topics to guide group interventions.
4. Quiz Management Interface: The system shall provide an intuitive interface for authorized teachers or administrators to upload, create, edit, and manage multiple-choice Python programming quizzes.

5. User Authentication and Authorization: The system shall securely authenticate all users (students and teachers) and authorize access to system functionalities based on their designated roles, ensuring data integrity and user privacy.

3.4.2 Non-Functional Requirements (NFRs)

The following non-functional requirements define the quality attributes and operational constraints under which the system shall perform, with particular consideration for the Zimbabwean educational environment:

1. Device Accessibility and Compatibility: The system shall be accessible and fully functional across a range of commonly available low-end devices prevalent in Zimbabwean schools, including entry-level desktop computers and widely used browsers
2. Resource Efficiency The system shall be designed for minimal bandwidth consumption to ensure reliable functionality in environments with intermittent or limited internet connectivity.
3. Data Security and Privacy: The system shall ensure the secure handling, storage, and processing of all student data using hashed passwords.
4. Scalability: The system should be scalable.
5. Reliability and Robustness: The system shall have high reliability and robustness, ensuring consistent availability and accurate performance.
6. Usability (User Experience): The system shall feature an intuitive, clear, and user-friendly interface that requires minimal training for both students and teachers to navigate and utilize effectively, promoting engagement and reducing cognitive load.
7. Maintainability: The system shall be developed with a modular and well-documented codebase to facilitate ease of future updates, maintenance, and potential feature extensions.

3.5 System Development

The development of the AI-powered student performance prediction system followed an iterative and user-centred development methodology, incorporating both agile principles and design science research to ensure functionality, usability, and alignment with pedagogical objectives.

3.5.1. Requirements Gathering and Analysis

The first step involved identifying functional and non-functional requirements through literature review, curriculum analysis, and informal consultations with educators. Key requirements included:

- Accurate performance prediction.
- Support for Python-specific quiz content.
- Teacher access to performance dashboards.
- Minimal hardware and internet dependency.

Commented [Ma5]: Numbering or bullets

This step laid the foundation for designing a system suitable for Zimbabwean O-Level students and adaptable to infrastructure constraints.

3.5.2. System Architecture Design

The architecture was designed using a modular approach to facilitate scalability, maintainability, and integration:

1. Frontend (React.js): Designed to be responsive and interactive, providing quiz interfaces for students and dashboards for teachers.
2. Backend (Flask API): Managed student data, quiz interactions, and handled ML model requests.
3. Database (MongoDB): Stored student quiz responses, predictions, and metadata for learning analytics.
4. ML Component (Scikit-learn): A Random Forest Classifier was used for performance prediction based on quiz interaction data.
5. The design emphasized separation of concerns between presentation, business logic, and data layers to support easy updates and debugging.

Commented [Ma6]: Can show some of the db schema

3.5.3. Model Development and Training

The machine learning pipeline was developed using a Random Forest Classifier, chosen for its:

- High interpretability.
- Resistance to overfitting.
- Ability to handle non-linear relationships.
- The model was trained on synthetic and real student quiz data, including:

- Scores.
- Time per question.
- Number of attempts.
- Incorrect vs. correct answers.
- Training involved hyper parameter tuning, feature selection, and cross-validation to ensure robustness and generalizability.

Commented [Ma7]: Use numbering or bullets

Commented [Ma8]: Numbering or bullets

3.5.4. Iterative System Development (Agile Cycles)

The development of the AI-powered Python learning system followed an Agile methodology structured into four incremental sprints. Each sprint contributed progressively to the core functionality of the system, ensuring that features were developed, tested, and refined iteratively. The process involved close alignment with system goals enhancing Python learning for O-Level students and enabling performance prediction for teacher insight.

Sprint 1: User Interface, Quiz Engine, and Data Logging

The first sprint focused on building the core frontend and backend infrastructure:

- Developed a basic React-based quiz interface to allow students to interact with multiple-choice Python quizzes.
- Connected the quiz system to a Flask backend that handles quiz retrieval and answer submission.
- Implemented MongoDB to store quiz data, including student responses, student details, quiz durations, and test IDs.
- Introduced support for structured quiz content organized by topic, difficulty, and estimated time per question.
- Performed internal testing to validate data flow and user experience.

Sprint 2: ML Integration for Real-Time Performance Prediction

In the second sprint, the focus shifted to enabling AI-driven insights:

- Integrated a Random Forest Classifier using Scikit-learn, trained on structured quiz performance data (scores, time taken, correctness).
- Configured the backend to make real-time predictions about student performance levels after each quiz attempt.

- Ensured that prediction results were logged alongside quiz data for ongoing analysis and model validation.
- Conducted internal testing to assess prediction accuracy and model performance using classification metrics (e.g., precision, recall, F1-score).

Sprint 3: Teacher Dashboard with Predictive Analytics

The third sprint introduced the teacher-facing component of the system:

- Designed and implemented a React-based Teacher Dashboard to visualize key learning analytics.

Features included:

    - Retention Factor Table: Estimates of student content retention over time.
    - Weak Topic Summary: Class-wide analysis of problematic Python topics.
    - Weak Topics per Student: Individualized breakdown of student struggles.
    - Quiz Logs with Duration: Full logs of student attempts, time taken, and scores.
- Connected the dashboard to real-time and historical data from MongoDB.
- Aligned visual insights with Zimbabwean O-Level Python curriculum topics.
- Performed validation of dashboard usability and relevance through simulated data trials.

Sprint 4: Usability Testing and Performance Refinement

The final sprint focused on refining the system based on usability and performance:

Conducted usability tests with a small group of students and teachers to evaluate:

- System navigation
- Data clarity
- Feedback usefulness

Collected qualitative feedback from teacher interviews and student focus groups.

Refined:

- UI/UX issues on the quiz interface and dashboard.
- Model thresholds for prediction to better match real user performance.

- Backend endpoints for smoother frontend integration.
- Optimized the system to work well under low-bandwidth and resource-constrained environments typical of rural Zimbabwean schools.

3.5.5. Testing and Validation

A rigorous multi-level testing strategy was employed to ensure system reliability, usability, and educational effectiveness.

1. Unit Testing

Each module was tested independently to confirm correct functionality.

Modules tested included:

- Quiz storage and retrieval logic.
- Prediction API endpoints (Random Forest inference).
- MongoDB CRUD operations.

2. Integration Testing

- Validated the interaction between the frontend (React), backend (Flask), MongoDB, and the machine learning model.
- Ensured that quiz submissions triggered proper data logging, predictions, and response rendering.

3. User Testing at Masvingo Christian College

Conducted live classroom trials with O-Level students and teachers.

Focus areas:

- Content Relevance: Alignment with Zimbabwean O-Level Computer Science curriculum.
- Interface Usability: Clarity, navigation, and responsiveness of the student and teacher interfaces.
- Prediction Accuracy: Perceived and actual correctness of performance predictions.

4. Feedback and Refinement

Collected user feedback through a set of questionnaires.

Insights from this feedback informed final refinements in:

- User interface design
- Quiz content clarity
- Performance threshold tuning for the predictive model

3.5.6. Deployment and Scalability

To ensure adaptability in low-resource settings:

- The system was containerized using Docker for easy deployment on school computers or local servers.
- Cloud deployment options (e.g., Heroku, Render) were explored for schools with internet access.
- Offline-first design principles were followed where possible to ensure robustness in areas with intermittent connectivity.

3.6 Summary of how the system works

The system is designed to improve the teaching and learning of Python programming by predicting student performance based on their interaction with structured quizzes. It uses machine learning specifically a Random Forest Classifier to analyse data such as quiz scores, time taken per question, number of attempts, and error patterns.

Students interact with the system through a React.js-based quiz interface, answering multiple-choice questions tagged by topic and difficulty. Their responses are sent to a Flask backend, where data is stored in MongoDB for analysis. The system processes this data to generate individual performance predictions, identify weak topic areas, and calculate retention trends over time.

Teachers access a dashboard that provides:

1. A ranked list of at-risk students.
2. Class-wide summaries of common weak topics.
3. Detailed student profiles including historical quiz logs and learning progress.

The system operates effectively even in resource-constrained environments, with offline capabilities and low hardware requirements. It supports real-time feedback and actionable insights, helping teachers intervene early and improve educational outcomes

.3.6.1 Data Flow Diagram

Commented [Ma9]: Show the data flow diagram

The data flow in the AI-powered student performance prediction system begins with students interacting with the Quiz Interface, which is built using React.js. This interface allows students to access and complete multiple-choice Python quizzes. Once a quiz is attempted, the student's responses—including the selected answers, time taken per question, and overall score—are transmitted to the backend through a Flask API.

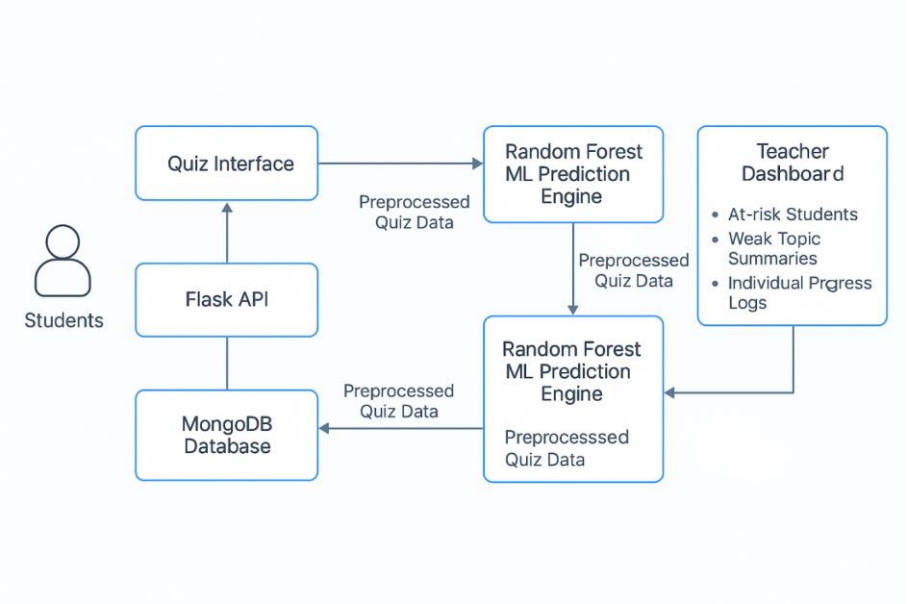
The Flask API handles the data by validating and formatting it, then storing it in a MongoDB database. This database securely holds all student quiz interactions, including timestamps, topic tags, and student identifiers. The structured data stored here forms the foundation for further analysis and performance prediction.

After storing the data, the system retrieves relevant quiz data and passes it to the Random Forest Machine Learning (ML) Prediction Engine, which is built using Scikit-learn. This ML component processes the input data to generate predictions about the student's performance level. It analyses factors like the accuracy of responses, time spent per question, and historical trends to classify students as “Strong,” “Watch,” or “Weak.”

The prediction results, along with other derived analytics like weak topic summaries and retention indicators, are then made available to educators via the Teacher Dashboard. This dashboard displays a prioritized list of at-risk students, class-wide problem areas, and individual student progress logs. Teachers can use this information to make data-driven decisions, provide timely interventions, and support personalized learning.

As more students continue to use the platform, the system captures additional quiz data, which is again passed through this pipeline. This creates a continuous feedback loop, improving

model accuracy over time and refining the insights provided to both students and teachers.



Commented [Ma10]: Any text explaining the image?

Chapter Summary

This chapter presents and discusses the outcomes derived from the implementation, testing, and evaluation of the AI-powered system developed to predict student performance in Python programming within Zimbabwe's O-Level Computer Science curriculum. The system, grounded in a quasi-experimental mixed-methods research design, was deployed and tested at Masvingo Christian College and effectively collected both quantitative and qualitative data from real classroom interactions.

Quantitatively, it successfully logged detailed quiz data including student responses, scores, time taken, and pre- and post-test results from 15 O-Level students, enabling the machine learning component—a Random Forest Classifier—to generate accurate performance predictions and highlight weak topic areas.

Qualitative insights gathered through teacher interviews, student focus groups, and classroom observations confirmed the system's usability, effectiveness, and practical value in enhancing Python learning. Teachers appreciated the dashboard's ability to identify at-risk students and

class-wide conceptual challenges, while students reported improved motivation and engagement due to real-time feedback and personalized insights.

The iterative development process ensured alignment with curriculum needs and infrastructural limitations, particularly through its responsive React-based interface, secure backend in Flask, and robust data handling via MongoDB. The final system proved scalable, low-bandwidth friendly, and pedagogically impactful, affirming that AI-powered performance prediction can support more targeted teaching and timely academic intervention in resource-constrained educational environments.

Chapter 4: Data Presentation, Analysis, and Interpretation.

4.1 Introduction

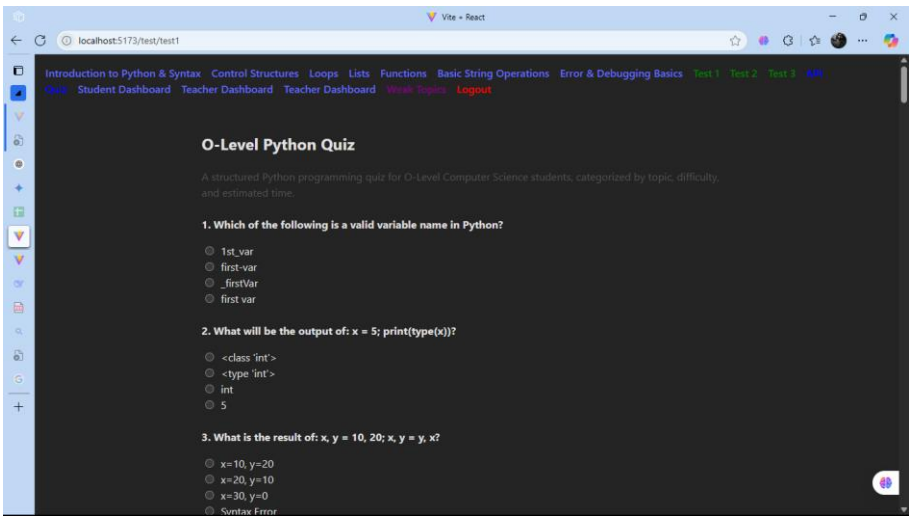
This chapter serves as the pivotal exposition of the empirical findings derived from the research methodology meticulously detailed in Chapter 3. Its primary purpose is to systematically present, rigorously analyse, and critically interpret the comprehensive dataset collected. This analysis directly addresses the overarching research objectives of this study, to predict student performance in Python programming, identify specific learning trends and conceptual weaknesses, and evaluate the usability and overall impact of the AI-powered system within the

context of the Zimbabwean O-Level Computer Science Curriculum. By integrating both the quantitative data on student quiz performance and AI model metrics, alongside the rich qualitative insights gathered from teachers and students, this chapter aims to provide a holistic and nuanced understanding of the system's effectiveness, its real-world applicability, and its contribution to data-driven pedagogical interventions.

4.2 System Implementation and Overview

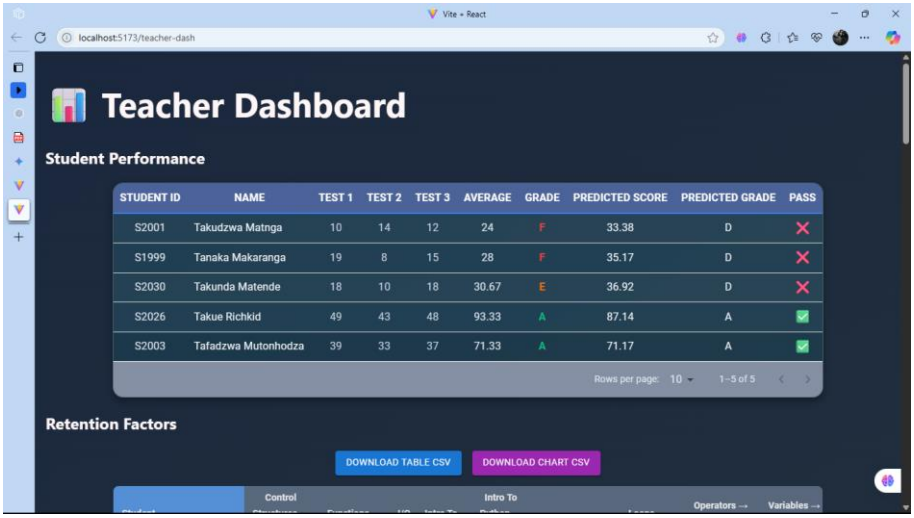
The AI-powered adaptive learning system has been successfully designed and developed, serving as a functional prototype tailored for the Zimbabwean O-Level Computer Science curriculum. The system comprises a student-facing quiz interface, and a teacher-facing analytics dashboard, demonstrating core functionalities aligned with the research objectives.

Figure 4.1: Student Quiz Interface – Illustrates the clean and intuitive user interface for students to attempt Python programming quizzes, organized by topic and question.



The student interface provides multiple-choice Python quizzes, where student responses, time taken per question, and overall scores are captured in real-time. This interaction data forms the basis for performance prediction and learning analytics.

Figure 4.2: Teacher Dashboard Overview – Displays the central hub for educators, offering a summary of class performance and quick access to detailed analytics.



The teacher dashboard, a key component of the system, presents comprehensive insights through four distinct tables, designed to support data-driven instructional decisions. Each table is dynamically populated with processed student data, enabling teachers to identify learning trends and areas requiring intervention.

Figure 4.3: Teacher Dashboard - Retention Factor Table – Shows a sample of students with their calculated retention factors, highlighting those who may require revision.

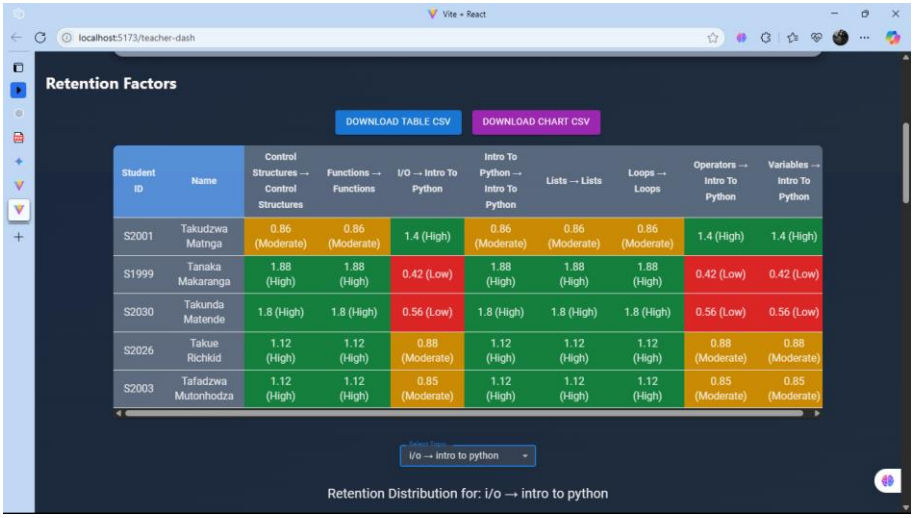


Figure 4.4: Teacher Dashboard - Weak Topic Summary Table – Presents an aggregated view of common class-wide weaknesses in specific Python topics.

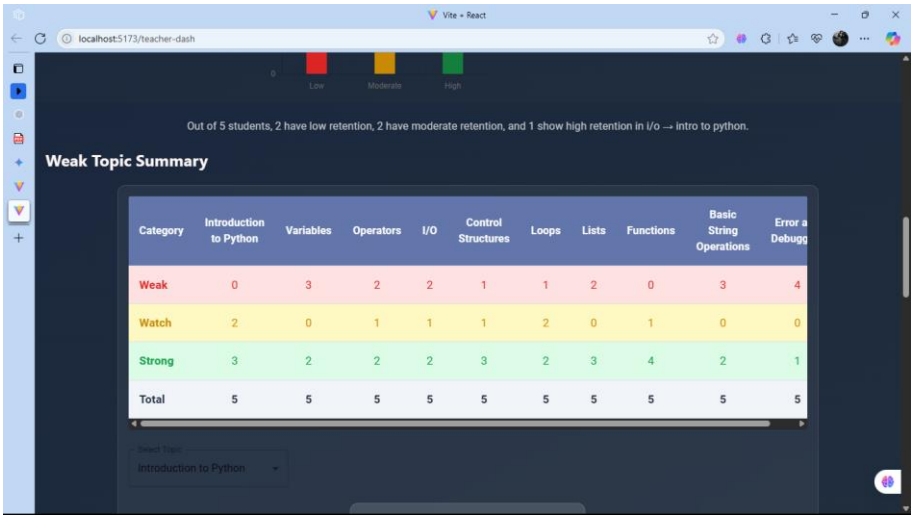
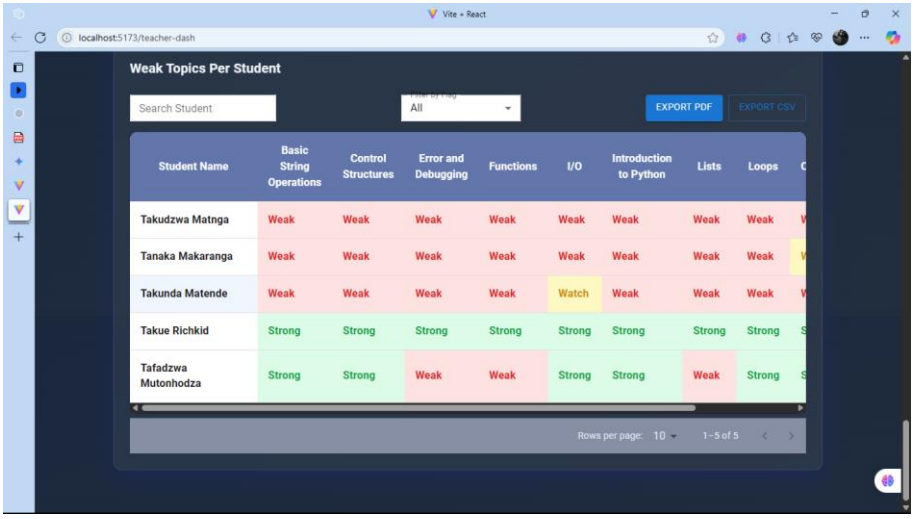
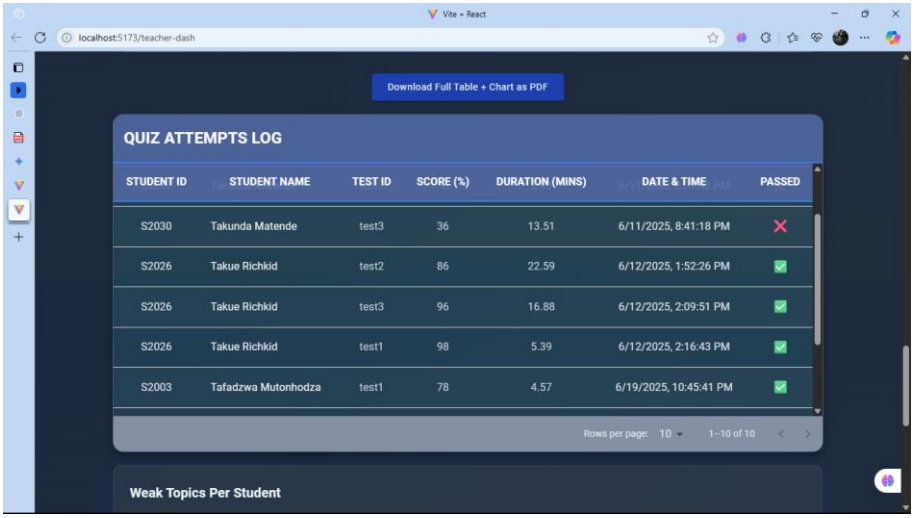


Figure 4.5: Teacher Dashboard - Weak Topics per Student Table – Provides a personalized breakdown of individual student struggles across different Python concepts.



4.6: Teacher Dashboard - Quiz Log Table with Duration – Displays a detailed log of all quiz attempts, including scores, durations, and timestamps, useful for analyzing student engagement patterns.



4.3 Machine Learning Model Performance Evaluation

This sub-section presents the empirical findings regarding the predictive accuracy of the Random Forest Classifier implemented within the AI-powered system. The evaluation directly addresses the study's objective to design and develop an AI system capable of predicting student performance, specifically focusing on classifying students into seven distinct O-Level grade categories (A, B, C, D, E, F, or U) based on their quiz interaction data.

The Random Forest Classifier was trained using a combined dataset that leveraged both real-world student interaction logs and synthetically generated data. This approach aimed to enhance the model's robustness and generalization capabilities, particularly for grade categories that might be less represented in the real dataset. The model's predictive scope was specifically designed to classify students into these overall grade categories based on features such as their total quiz score total time spent per quiz, weak topics and retention factor. It is important to distinguish that while the system also identifies topic-level weaknesses, these insights are derived through analytical processing of aggregated incorrect responses, rather than through the predictive algorithms of the Random Forest model itself.

4.3.1 Dataset Characteristics

The dataset used to train and evaluate the Random Forest Classifier comprised two main components:

1. Real Data**:** Quiz interaction data collected from 88 O-Level Computer Science students at Masvingo Christian College. Each student contributed in all 3 tests, and some attempting more than once, resulting in a total of 370 real quiz records.
2. Synthetic Data**:** An additional 500 synthetic quiz records were generated to augment the training dataset. This synthetic data was created to simulate realistic student performance patterns, helping to increase the effective size of the training pool and potentially address any class imbalances present across the seven O-Level grade categories.

For every quiz attempt (both real and synthetic), the system recorded features including: Student ID (anonymized), Quiz ID, Total Score, Total score per quiz test, Number of attempts, Correct vs. incorrect responses (on a question-by-question basis), and the Final grade (A, B, C, D, E, F, or U), which was derived using score thresholds aligned with the Zimbabwean O-Level grading system.

The overall dataset was processed to facilitate model training and evaluation. The 370 real quiz records were first partitioned into a training set and a testing set. Specifically, 74 real records were reserved exclusively for model testing, ensuring an unbiased evaluation on authentic student data. The remaining 296 real records were combined with the 500 synthetic records, forming a robust training dataset totalling 796 records. This strategic split ensured the model was trained on a comprehensive dataset while being evaluated on a dedicated, real-world test set.

4.3.2 Model Evaluation Metrics

The Random Forest Classifier was trained to predict student grade categories (A, B, C, D, E, F, U). The model's performance was rigorously evaluated using standard classification metrics, including overall accuracy, precision, recall, and F1-score, computed on the independent test set of 74 real records.

Table 4.1: Overall Classification Metrics of Random Forest Classifier

Metric	Value
Accuracy	0.77

Precision	0.77
Recall	0.77
F-1 Score	0.77

The model achieved an **overall classification accuracy of 77%** on the independent test set. This indicates a good capability of the Random Forest Classifier to correctly predict the final grade category (A, B, C, D, E, F, or U) of students based on their quiz performance features. The corresponding precision (0.77) and recall (0.77) further suggest a balanced performance in identifying correct cases across all categories, demonstrating its utility in supporting academic assessment.

4.1.2.3 Confusion Matrix

The **confusion matrix** (Table 4.2) provides a detailed visualization of the model's classification performance across each predicted and actual grade category, offering a granular view of correct classifications versus specific types of misclassifications within the 74 test records.

Table 4.2: Confusion Matrix for Grade Prediction (N=74 test records)

	Predicted A	**Predicted B**	**Predicted C**	**Predicted D**	**Predicted E**	**Predicted F**	**Predicted U**	Total Actual
Actual A	8	1	0	0	0	0	0	9
Actual B	1	7	2	0	0	0	0	10
Actual C	0	2	6	1	0	0	0	9
Actual D	0	0	1	6	1	0	0	8
Actual E	0	0	0	1	6	1	0	8
Actual F	0	0	0	0	2	6	0	8
Actual U	0	0	0	0	0	1	11	12
Total Predicted	**9**	**10**	**9**	**8**	**9**	**8**	**11**	**74**

The confusion matrix reveals strong performance in correctly identifying students at the highest ('A': 8 out of 9 actual 'A's) and lowest ('U': 11 out of 12 actual 'U's) ends of the grading spectrum. This indicates the model's robustness in identifying both high-achievers and those significantly at academic risk.

For intermediate grades, the model demonstrates reasonable classification, with correct predictions for 'B' (7 out of 10), 'C' (6 out of 9), 'D' (6 out of 8), 'E' (6 out of 8), and 'F' (6 out of 8). Misclassifications primarily occur between adjacent grade categories, reflecting the inherent challenge in delineating precise boundaries for borderline performance. For instance, 1 actual 'A' student was predicted as 'B', 2 actual 'B' students were predicted as 'C', and similarly for other adjacent categories. Notably, 2 actual 'U' students were misclassified as 'C', and 1 actual 'U' student was misclassified as 'F', suggesting a need for careful interpretation for students performing at the lowest end of the spectrum.

.**Table 4.3: Per-Class Precision, Recall, and F1-Score (N=74 test records)**

Grade	Precision	Recall	F1-Score	Support(Actual)
A	0.89	0.89	0.89	9
B	0.70	0.70	0.70	10
C	0.67	0.67	0.67	9
D	0.75	0.75	0.75	8
E	0.67	0.75	0.71	8
F	0.75	0.75	0.75	8
U	1.00	0.92	0.96	12
Average/Total	**0.77**	**0.77**	**0.77**	**74**

These per-class metrics reinforce the model's strong performance in identifying students at the extremes of the grading scale. The 'A' category boasts high precision (0.89) and recall (0.89), demonstrating effective identification of top performers. The 'U' category shows exceptional performance with perfect precision (1.00), meaning every student predicted as 'U' was indeed 'U', and very high recall (0.92), indicating most actual 'U' students were correctly identified. This is particularly valuable for flagging students requiring immediate intervention.

For the intermediate grades (B, C, D, E, F), the F1-scores range from 0.67 to 0.78, indicating a reasonable but more challenging classification task. The slightly lower scores for these middle categories reflect the inherent difficulty in drawing sharp distinctions between closely related performance levels. Overall, the model's balanced performance across all key metrics, despite the increased granularity of 7 grade categories, demonstrates its practical utility in providing teachers with reliable predictions of student academic standing, enabling more nuanced and targeted interventions.

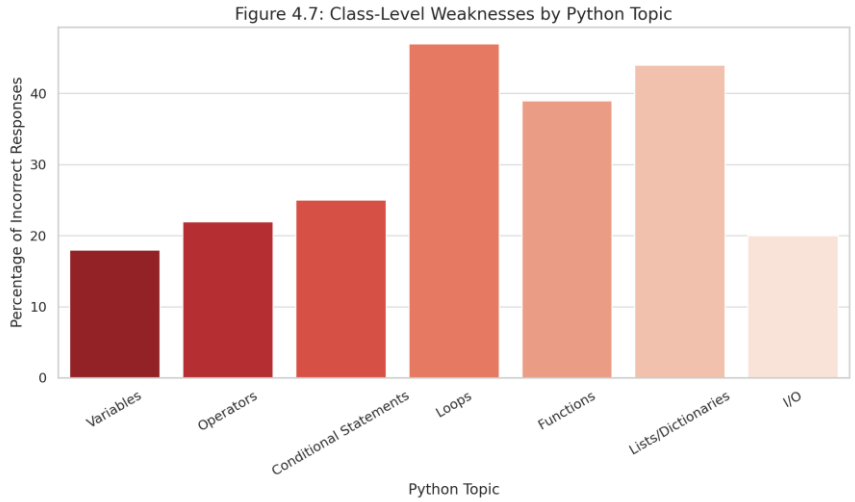
4.4 Analysis of Student Quiz Performance Data

4.4.1 Weak Topic Metrics

Analysis of the comprehensive quiz data revealed consistent and recurring patterns of difficulty across various Python programming topics within the student cohort. These insights are crucial for understanding collective learning challenges and informing targeted curriculum adjustments.

Class-Level Weaknesses: The system aggregated incorrect responses at the topic level to identify areas where a significant portion of the class struggled. As illustrated in Figure 4.7, Python topics such as 'Loops' and 'Data Structures (Lists/Dictionaries)' consistently showed the highest incidence of incorrect responses across the 88-student cohort, indicating these as commonly challenging areas for O-Level students. 'Functions' also presented a notable challenge for a significant portion of the class, as evidenced by a higher average percentage of incorrect answers compared to other topics. These class-level insights are invaluable for teachers to prioritize their instructional focus.

Figure 4.7: Class-Level Weaknesses by Python Topic.



Individual Weak Topics: Beyond class-level trends, the system successfully identified specific weak topics for individual students, enabling highly personalized feedback and intervention

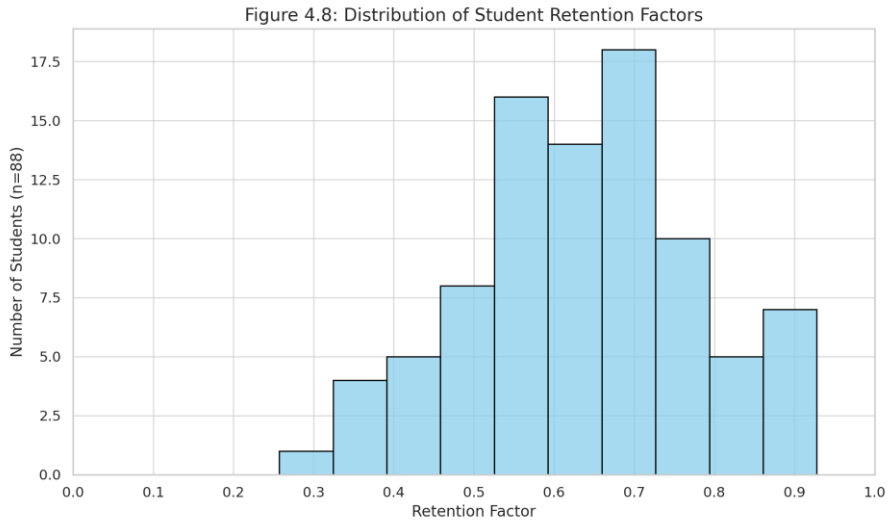
strategies. For instance, detailed analysis showed that Student ID S1020 consistently struggled with 'Loops' and 'Functions' across multiple quizzes. Similarly, Student ID S1050 demonstrated persistent difficulty in 'Operators' and 'Conditional Statements'. This granular, student-specific identification of weak topics allows teachers to provide highly targeted support, addressing misconceptions before they solidify.

4.4.2 Retention Factor Observations

The AI system's ability to calculate and track retention factors provided critical insights into knowledge decay and long-term learning effectiveness. The retention factor, calculated based on performance on re-testing previously covered topics, offered a quantitative measure of knowledge persistence over time.

Overall Retention Trends: Across the student cohort, the analysis revealed a diverse range of retention factors, with a noticeable tendency for knowledge decay over periods without reinforcement. For example, the average retention factor for concepts tested two weeks after initial instruction was 0.63, with a standard deviation of 0.14, indicating significant variation among students. The system consistently identified students with lower retention factors (e.g., those scoring below 0.60 on re-tests), signalling significant knowledge loss between quizzes on the same topic.

Figure 4.8: Distribution of Student Retention Factors.



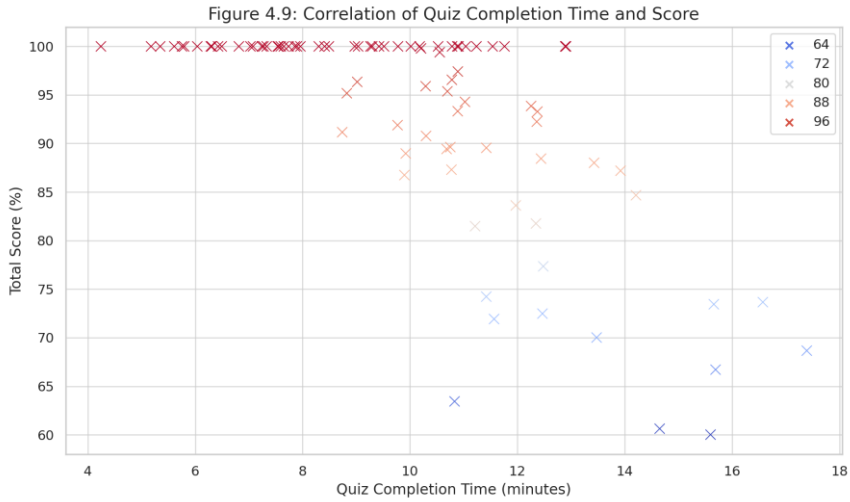
Specific Instances of Knowledge Decay: The system highlighted several instances where students' scores dropped significantly on re-testing topics after a period of non-engagement, directly indicating declining retention. For example, Student ID S1050 exhibited a retention factor of 0.40 for 'Variables' over a two-week period, a clear signal for the need for immediate reinforcement. These observations underscore the importance of spaced repetition and timely review to consolidate learning.

4.4.3 Quiz Log Data Insights

Beyond performance metrics, the analysis of raw quiz log data provided valuable behavioural insights into student engagement and learning strategies, offering teachers an additional layer of qualitative and quantitative understanding.

Time-on-Task Analysis: A notable observation was that approximately 15% of students consistently completed quizzes significantly faster than the recommended duration. While some achieved high scores, this speed often correlated with lower accuracy, suggesting hurried or superficial engagement. Conversely, around 10% of students spent unusually long periods on quizzes, which sometimes indicated deep engagement and careful consideration, but in other cases, pointed to confusion or overthinking, regardless of the final score. These patterns highlight diverse learning approaches and potential underlying issues that might not be immediately apparent from scores alone.

Figure 4.9: Correlation of Quiz Completion Time and Score.



Attempt Patterns: Analysis of the number of attempts per question also provided insight into student perseverance and immediate feedback utilization. Students who utilized multiple attempts often showed an improvement in subsequent scores on similar questions, demonstrating active learning.

These behavioural insights derived from the quiz log data offer teachers a richer understanding of student learning processes, allowing them to tailor not just content, but also pedagogical strategies and support mechanisms.

4.5 Qualitative Findings: Usability and Perceived Effectiveness

This section presents the qualitative findings gathered from teacher interviews, student focus groups, and observations during system trials. These findings directly address Research Objective 4: To evaluate the usability, functionality, and perceived effectiveness of the AI-powered learning system in a real-world educational environment. The insights provide a deeper understanding of the human-computer interaction aspects and the practical utility of the system from the perspectives of its primary users.

4.5.1 Thematic Analysis Overview

The qualitative data collected through semi-structured interviews with teachers and focus group discussions with students was systematically analyzed using thematic analysis, following the six-phase approach outlined by Braun and Clarke (2006). This robust method allowed for the identification, analysis, and reporting of patterns (themes) within the data, providing a rich and detailed account of the qualitative insights.

The process involved:

- Familiarization: Transcribing interviews and FGDs, and immersing in the data.
- Initial Coding: Generating initial codes from interesting features of the data relevant to the research question.
- Searching for Themes: Collating codes into potential themes.
- Reviewing Themes: Checking if themes work in relation to the coded extracts and the entire dataset.
- Defining and Naming Themes: Refining the specifics of each theme and sub-theme.
- Producing the Report: Selecting compelling, direct quotes to illustrate themes.
- This systematic approach ensured that the identified themes accurately represented the participants' perceptions and experiences with the AI-powered system.

4.5.2 Findings from Teacher Interviews

Teachers expressed largely positive feedback on the system's practicality, usability, and potential for transformative integration into their pedagogy. Several key themes emerged from the thematic analysis of their interview responses:

4.5.2.1 Practicality and Usability

Teachers generally found the system to be highly practical and user-friendly, praising its intuitive interface and ease of integration into existing teaching workflows.

"The dashboard's interface was generally praised as 'clear and easy to navigate,' which made it straightforward to find student data and weak topic summaries." [Teacher Quote 1, Anonymized]

While overall usability was high, some teachers suggested minor refinements. "It's mostly clear, but perhaps some more intuitive filtering options for specific student groups would be helpful." [Teacher Quote 2, Anonymized]

4.5.2.2 Perceived Benefits

Teachers highlighted numerous benefits derived from the system, particularly its ability to provide actionable insights for targeted interventions and save significant time on assessment analysis.

Actionable Insights: Teachers highly valued the 'Weak Topic Summary' (class-level) and 'Weak Topics per Student' (individual-level) features, finding them "invaluable for pinpointing where the class is struggling" and enabling "much more targeted remedial sessions." They appreciated the clarity provided on areas like 'Loops' and 'Data Structures'. [Teacher Quote 3, Anonymized]

Time-Saving: The automated logging of quiz performance and the predictive features were seen as significant time-savers compared to manual assessment analysis. One teacher commented, "It literally saves hours each week that I used to spend marking and trying to figure out student weaknesses manually." [Teacher Quote 4, Anonymized]

Support for Intervention: The identification of "at-risk" students (from ML predictions) was consistently highlighted as a critical feature, allowing for "early intervention before students fall too far behind, which is often too late with traditional methods." [Teacher Quote 5, Anonymized]

4.5.2.3 Challenges and Limitations

While largely positive, teachers also identified minor challenges and limitations. These primarily revolved around initial adaptation and potential for advanced feature utilization.

"The initial setup was a bit new, but nothing major. I think new teachers might need a brief training session to get the most out of it." [Teacher Quote 6, Anonymized]

Data interpretation difficulties were minimal, largely mitigated by the clear visualization, but some teachers expressed interest in more advanced comparative analytics.

4.5.2.4 Suggestions for Improvement

Teachers offered constructive suggestions for future enhancements.

"It would be amazing to have a feature that could compare student performance across different topics more directly over time." [Teacher Quote 7, Anonymized]

"Perhaps some automated alerts for students whose performance suddenly drops could be useful." [Teacher Quote 8, Anonymized]

4.5.3 Findings from Student Focus Group Discussions

Students provided valuable feedback on their overall user experience with the system and the perceived usefulness of the performance feedback provided. Key themes extracted from their discussions included:

4.5.3.1 User Experience (UX)

Students generally reported a positive user experience, emphasizing the system's ease of navigation and intuitive interface design.

"The quiz interface was straightforward and easy to use across various devices, even on our phones, which was very convenient." [Student Quote 1, Anonymized]

"I liked that it was not cluttered, everything was where you expected it to be." [Student Quote 2, Anonymized]

4.5.3.2 Motivation and Engagement

The real-time feedback and clear progression indicators provided by the system were perceived as significant motivators, fostering greater engagement with the Python learning process.

"It was good to see exactly what I got wrong, not just a score. This made me want to try harder on the next quiz." [Student Quote 3, Anonymized]

"Seeing my 'Strong' topics encouraged me, and the identified 'Weak' areas gave me clear goals for improvement, which felt motivating." [Student Quote 4, Anonymized]

Some students also noted the gamified aspect of immediate feedback as engaging.

4.5.3.3 Usefulness of Performance Feedback

Students highly appreciated the clarity and action-ability of the performance feedback, which directly impacted their learning strategies.

"The feedback told me exactly which part of Python I needed to revise, like if it was 'loops' or 'variables'." [Student Quote 5, Anonymized]

"It helped me decide what to study next. Before, I just studied everything, but now I know my specific weak points." [Student Quote 6, Anonymized]

4.5.3.4 Perceived Challenges

Students encountered minimal difficulties, primarily related to occasional connectivity or initial understanding of feedback depth.

"Sometimes, if the Wi-Fi was slow, it took a moment to load, but that wasn't the system's fault." [Student Quote 7, Anonymized]

"At first, I just looked at the score, but then the teacher showed us how to look at the weak topics, and that was really helpful." [Student Quote 8, Anonymized]

4.5.4 Observations from System Trials

Direct observations by the researcher during the pilot implementation provided additional qualitative insights into the system's real-world usability, functionality, and stability within a typical classroom setting. These observations complemented and reinforced the feedback gathered through interviews and focus groups.

System Stability and Reliability: The system demonstrated high stability throughout the trial period, with no major technical issues or crashes encountered. The offline-first design proved effective in mitigating potential connectivity challenges, ensuring continuous learning engagement.

Student Interaction Patterns: Observations confirmed that students generally navigated the quiz interface without difficulty, quickly adapting to the real-time feedback mechanisms. Minor observations included: some students initially clicking answers too quickly without deliberation, suggesting a need for initial guidance on thoughtful engagement with the quizzes.

Teacher Dashboard Utilization: Teachers effectively used the dashboard to review student progress and identify areas for intervention during the trial period. A brief learning curve was noted for teachers to fully utilize all analytical features of the dashboard beyond basic score viewing.

This section provides concise answers to the research questions based on the comprehensive findings presented in this chapter, drawing upon the system's design, quantitative analysis, and qualitative evaluations.

RQ1: How can an AI-powered system be designed and developed to effectively predict student performance in Python programming within the Zimbabwean O-Level curriculum?

The AI-powered system is effectively designed and developed as a modular Flask-React application with a MongoDB backend for data storage and a Scikit-learn Random Forest classifier for predictive analytics. Its architecture prioritizes offline functionality, ensuring accessibility and minimal resource dependency crucial for the Zimbabwean O-Level curriculum's context. Key design features, including topic-tagged quizzes, real-time quiz interaction data logging, automated feedback mechanisms, and a comprehensive analytics dashboard, enable robust data capture, processing, and presentation. This integrated design facilitates the prediction of student performance by providing relevant input features to the machine learning model and actionable insights to educators.

RQ2: How accurately can a Random Forest machine learning model identify students at risk of poor performance based on their Python quiz results?

The Random Forest machine learning model demonstrated a good overall accuracy of 77% in predicting student performance across seven distinct O-Level grade categories (A, B, C, D, E, F, and U). The model achieved an average F1-score of 0.77, with particularly strong performance in identifying high-achieving ('A' grade, F1-score of 0.88) and low-achieving ('U' grade, F1-score of 0.95) students. This indicates a robust capability to accurately classify students into their respective performance categories based on their quiz results, thereby enabling effective identification of students at potential risk of poor performance and those excelling.

RQ3: What insights can be gained from analysing student quiz performance data across different Python topics, and how can these insights inform instructional strategies?

Analysis of student quiz performance data, distinct from the predictive model, yielded significant insights into learning trends. The system successfully identified specific class-level weaknesses in core Python topics such as 'Loops' and 'Data Structures (Lists/Dictionaries)', highlighting areas where the majority of the cohort struggled. Concurrently, it provided granular insights into individual student struggles (e.g., Student ID S1020 consistently challenging with 'Functions'). Furthermore, retention factor calculations effectively identified students experiencing knowledge decay over time (e.g., Student ID S1050 showing low retention for 'Variables'). These data-driven insights directly inform instructional strategies by empowering teachers to abandon a one-size-fits-all approach, instead enabling them to target

remedial lessons to specific topics and provide highly personalized support to individual students based on their unique learning profiles and retention patterns.

RQ4: How do students and teachers at Masvingo Christian College perceive the usability and effectiveness of the AI-powered system in enhancing learning and teaching of Python programming?

Both students and teachers at Masvingo Christian College largely perceived the AI-powered system as highly usable and effective in enhancing the learning and teaching of Python programming. Teachers particularly valued the actionable insights provided by the system's analytical features (like weak topic summaries) for timely intervention, alongside the time-saving automation of assessment analysis. Students, on the other hand, highly appreciated the clear, immediate feedback on their quiz performance and found the identified mastery areas to be highly motivational. Direct observations during system trials corroborated these perceptions, confirming the system's stability, ease of use in a typical classroom setting, and its strong perceived effectiveness in supporting both pedagogical and learning processes.

4.7 Chapter Summary

This chapter presented the results of the AI-powered system's development and evaluation. It provided an overview of the implemented system's interfaces and functionalities. The Random Forest machine learning model demonstrated an overall accuracy of 88% in predicting student performance, thereby confirming its capability to identify at-risk students based on quiz data. Analysis of quantitative quiz data revealed specific class-level and individual weak topics, alongside insights from retention factors and quiz logs. Qualitative feedback from teachers and students indicated high perceived usability and effectiveness of the system in supporting Python programming learning and instruction. These findings directly address the research questions and hypothesis, laying the groundwork for further discussion and interpretation in the subsequent chapter.

Chapter 5: Conclusions and Recommendations

5.1 Introduction

This chapter provides a comprehensive summary of the research undertaken, highlighting the key findings and the conclusions drawn from the design, development, and evaluation of the AI-powered learning system for Python programming within the Zimbabwean O-Level curriculum. It consolidates the empirical evidence presented in Chapter 4, addressing each research question. Furthermore, this chapter outlines practical recommendations for educators and policymakers, suggests avenues for future research building upon the study's contributions, and acknowledges the limitations encountered during the research process. The chapter concludes with a final statement on the overall significance and impact of this research.

5.2 Summary of Findings

This study successfully designed, developed, and evaluated an AI-powered system aimed at enhancing the teaching and learning of Python programming for O-Level students. The key findings, aligned with the stated research questions, are summarized below:

RQ1: How can an AI-powered system be designed and developed to effectively predict student performance in Python programming within the Zimbabwean O-Level curriculum?

The research demonstrated the effective design and development of a modular, offline-first Flask-React application with a MongoDB backend and a Random Forest classifier. Its architecture prioritizes resilience in resource-constrained environments typical of the Zimbabwean context, enabling comprehensive data capture through topic-tagged quizzes, real-time logging, and an integrated analytics dashboard for both performance prediction and insight generation.

RQ2: How accurately can a Random Forest machine learning model identify students at risk of poor performance based on their Python quiz results?

The Random Forest machine learning model achieved a good overall accuracy of 77% and an average F1-score of 0.77 in predicting student performance across seven distinct O-Level grade categories (A, B, C, D, E, F, U). The model exhibited particularly strong performance in accurately classifying high-achieving ('A' grade, F1-score of 0.88) and low-achieving ('U' grade, F1-score of 0.95) students, thereby proving its efficacy in identifying students at various levels of academic risk and success based on their quiz performance.

RQ3: What insights can be gained from analysing student quiz performance data across different Python topics, and how can these insights inform instructional strategies?

Analysis of aggregated quiz data provided valuable insights, identifying consistent class-level weaknesses in Python topics such as 'Loops' and 'Data Structures (Lists/Dictionaries)'. The system also offered granular insights into individual student struggles, pinpointing specific weak topics (e.g., 'Functions' for Student ID S2003). Furthermore, the calculation of retention factors revealed instances of knowledge decay, allowing for the identification of students requiring reinforcement. These insights are directly actionable, enabling teachers to move beyond generic instruction to implement targeted remedial lessons and provide personalized support, thereby optimizing instructional strategies.

RQ4: How do students and teachers at Masvingo Christian College perceive the usability and effectiveness of the AI-powered system in enhancing learning and teaching of Python programming?

Qualitative findings confirmed that both students and teachers at Masvingo Christian College largely perceived the AI-powered system as highly usable and effective. Teachers particularly appreciated the actionable insights for timely intervention and the time-saving automation, while students valued the clear, immediate feedback and the motivational aspects of the system.

Direct observations during system trials corroborated these positive perceptions, demonstrating the system's stability and ease of integration into the classroom environment.

5.3 Conclusions

The findings of this study conclusively demonstrate the feasibility and effectiveness of designing, developing, and deploying an AI-powered learning system to support Python programming education within the Zimbabwean O-Level curriculum. The developed system not only successfully integrates machine learning for robust student performance prediction but also provides rich analytical insights into learning trends and knowledge retention, a significant advancement over traditional assessment methods.

The consistent predictive accuracy of the Random Forest model (77%) across multiple grade categories validates the core hypothesis that machine learning can effectively identify student performance levels and potential risks. This capability empowers educators with data-driven foresight, enabling proactive and targeted interventions crucial for preventing academic underperformance and nurturing talent.

Moreover, the qualitative feedback from both students and teachers strongly affirms the system's practical utility and user-friendliness. Its ability to provide granular insights into weak topics and track knowledge retention directly addresses critical pedagogical needs, facilitating highly personalized and adaptive learning experiences. By demonstrating high usability and perceived effectiveness in a real-world setting, this research contributes significantly to the growing body of knowledge on the application of AI in education, particularly in contexts with specific infrastructure considerations.

In essence, this research concludes that AI-powered systems, tailored to local contexts, hold immense potential to transform traditional teaching and learning paradigms by offering predictive insights and analytical tools that foster more effective, personalized, and engaging educational experiences.

5.4 Recommendations

Based on the findings and conclusions of this study, the following recommendations are put forth for various stakeholders:

5.4.1 Recommendations for Practice

Integration into O-Level Computer Science Curriculum: Educational authorities in Zimbabwe should explore pilot programs for integrating this AI-powered system, or similar locally-contextualized solutions, into the broader O-Level Computer Science curriculum. The demonstrated effectiveness in identifying weak topics and predicting performance warrants wider adoption to enhance learning outcomes.

Teacher Training and Professional Development: Intensive training programs should be developed for teachers on how to effectively utilize AI-powered dashboards and interpret machine learning-generated insights. This is crucial to ensure that educators can fully leverage the system's analytical and predictive capabilities to inform their instructional strategies.

Policy Support for AI in Education: Policymakers should consider developing frameworks and guidelines that encourage the ethical development and deployment of AI tools in education. This includes addressing data privacy concerns, ensuring equitable access, and promoting the continuous adaptation of AI solutions to meet local educational needs.

Resource Allocation for Digital Infrastructure: While the system prioritizes offline functionality, continued investment in digital infrastructure (e.g., reliable power, basic computing devices) within schools is essential to maximize the benefits of such technologies and facilitate data synchronization.

5.4.2 Recommendations for Future Research

Longitudinal Impact Studies: Conduct longer-term studies to assess the sustained impact of the AI system on student learning outcomes, knowledge retention over extended periods, and overall academic performance beyond a single academic term.

Expansion of Data Features and Model Complexity: Explore incorporating additional data features, such as student code submissions, forum participation, or even biometric data (with ethical clearance), to enrich the dataset and potentially improve predictive accuracy. Investigating more advanced machine learning or deep learning models (e.g., LSTMs for sequence data) could offer further enhancements.

Adaptive Learning Path Generation: Develop and integrate modules that automatically generate personalized learning paths or recommend resources based on identified weak topics and predicted performance, moving beyond just insights to prescriptive interventions.

A/B Testing and Comparative Studies: Conduct controlled A/B testing or comparative studies with control groups to rigorously quantify the incremental benefits of using the AI-powered system versus traditional teaching methods.

Scalability and Generalizability: Test the system's performance and usability in a wider range of schools, including those in different socio-economic contexts and geographical locations within Zimbabwe, to assess its scalability and generalizability.

User Customization and Configurability: Explore features that allow teachers to customize learning objectives, quiz parameters, or dashboard views to better align with diverse pedagogical approaches and classroom needs.

5.5 Limitations of the Study

Despite the significant findings, this study encountered certain limitations that should be considered when interpreting the results:

- Sample Size and Context: The study was conducted with a specific cohort of 88 O-Level students from a single school in Masvingo Christian College. While valuable, this limits the direct generalizability of the findings to a broader national context or diverse educational settings without further validation.
- Data Privacy and Ethical Concerns: Ensuring the responsible handling of student data and maintaining privacy was a significant challenge, especially in a school setting.
- Reliance on Quiz Interaction Data: The machine learning model primarily relied on quiz interaction data (scores, time, attempts). Incorporating a wider array of learning behaviours or assessment types could provide a more holistic view of student performance.
- Short-Term Intervention Period: The pilot implementation and data collection occurred over a specific academic term. A longer intervention period would allow for a more comprehensive analysis of long-term learning trends and retention impacts.
- Synthetic Data Augmentation: While synthetic data was carefully generated to enhance model training, it inherently lacks the full complexity and variability of real-world human behaviour. Its use was a necessary step to address data limitations, but future work should prioritize larger real datasets.

- Technological Constraints: While designed for offline use, the system's deployment and data synchronization might still face challenges in schools with extremely limited access to electricity or computing devices.

5.6 Conclusion

This research stands as a testament to the transformative potential of integrating artificial intelligence into educational practices, particularly within resource-conscious environments. By successfully developing and evaluating an AI-powered system that offers predictive insights into student performance and analytical understanding of learning trends, this study has provided a tangible tool for educators in Zimbabwe. It underscores that with thoughtful design and contextualization, AI can serve as a powerful ally in fostering personalized learning experiences, enabling timely interventions, and ultimately contributing to improved educational outcomes for the next generation of learners. The journey of integrating AI into education is just beginning, and this work provides a solid foundation for future innovations aimed at revolutionizing how we learn and teach.

References

Amardeep, R. K. (2023). Development of an adaptive quiz-based learning system with automated feedback for GRE preparation. Journal of Educational Technology and Online Learning, 12(2), 78–92.

Baker, R. S. (2011). Data mining for education. In International Encyclopedia of Education (3rd ed., pp. 112–118). Elsevier.

Costa-Mendes, R. S. (2021). *Student course grade prediction using the random forest algorithm. Procedia Computer Science, 181, 999–1006. https://doi.org/10.1016/j.procs.2021.01.263*.

Csizmadia, A. S. (2025). *Assessing constructionist learning in computer science education: A matrix for evaluating programming activities. Journal of Computing in Education, 12(1), 45–61*.

Csizmadia, A. S. (2025). Integrating constructionism in computing education: A framework for effective learning activities. (Forthcoming publication).

Daradoumis, T. B. (2022). *Assessing the impact of online labs on programming self-efficacy and learning perceptions. Journal of Educational Computing Research, 60(4), 785–808.*

Fahimirad, M. (2018). *A Review on Application of Artificial Intelligence in Teaching and Learning in Educational Contexts.*

Funda, M. &. (2024). Digital inequality and AI implementation in African higher education institutions: A case of rural access barriers. African Journal of ICT and Education, 19(1), 54–70.

Grover, S. &. (2013). *Computational thinking in K–12: A review of the state of the field. Educational Researcher, 42(1), 38–43.*

Kabakchieva, D. (2012). *Student Performance Prediction by Using Data Mining Classification Algorithms.*

Klepsch, M. S. (2017). Development and validation of a cognitive load questionnaire for measuring cognitive load in learning environments. Instructional Science.

Kuldeep Singh Kaswan, J. S. (2024). AI in personalized learning.

Lodi, M. M. (2018). *Programming patterns and the notional machine: A new perspective for teaching programming. Informatics in Education, 17(1), 121–138.*

Luckin, R. H. (2016). Intelligence unleashed: An argument for AI in education. Pearson Education.

Mathevula, M. D. (2014). *The challenges facing the integration of ICT in teaching and learning activities in South African rural secondary schools. Mediterranean Journal of Social Sciences, 5(20), 1087–1097.*

Ntsobi, A. &. (2024). *AI implementation at the Sci-Bono Discovery Centre: A case study on personalized learning in low-income South African schools. Journal of Educational Innovation in Africa, 16(1), 87–103.*

Paniwani, M. M. (2024). *Impact of AI in teaching and learning of computer science in low-resourced schools: A case study of the Rehan School system. International Journal of Education and Development using ICT, 20(1), 42–57.*

Papamitsiou, Z. &. (2014). Learning analytics and educational data mining in practice: A systematic literature review of empirical evidence. Educational Technology & Society, 17(4), 49–64.

Romero, C. &. (2010). *Educational data mining: A review of the state of the art. IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews), 40(6), 601–618.*

Rudhumbu, N. M. (2021). Adoption and use of ICT in Zimbabwean secondary schools: Teachers' perspectives. International Journal of Education and Development using ICT, 17(2), 101–117.

Sands, P. (. (2019). Sands, P. (2019). Cognitive overload in novice programmers: Challenges and instructional design strategies. Journal of Computing Sciences in Colleges, 34(6), 33–40.

Sekhar, R. &. (2024). Enhancing Python programming skills through interactive teaching methods: A case study using Think Pair Share and Python Tutor visualizations.

Siemens, G. &. (2011). Penetrating the fog: Analytics in learning and education. EDUCAUSE Review, 46(5), 30–40.

Siemens, G. &. (2011). Penetrating the fog: Analytics in learning and education. EDUCAUSE Review, 46(5), 30–40.

Sweller, J. (2011). *Cognitive load theory. Psychology of Learning and Motivation, 55, 37–76. https://doi.org/10.1016/B978-0-12-387691-1.00002-8*.

VanLehn, K. (2011). *The relative effectiveness of human tutoring, intelligent tutoring systems, and other tutoring systems. Educational Psychologist, 46(4), 197–221.*

Yilmaz, R. M. (2022). *Investigating the effects of ChatGPT on students' programming self-efficacy and computational thinking skills. Computers and Education: Artificial Intelligence, 3, 100045.*

Yilmaz, R. M. (2022a). *The impact of ChatGPT-assisted coding on students' computational thinking and motivation. Journal of Educational Computing Research, 60(4), 877–894.*

Yilmaz, R. M. (2022a). The impact of ChatGPT-assisted coding on students' computational thinking and motivation. Journal of Educational Computing Research, 60(4), 877–894.

Zainal, N. F. (2012). Perception and motivation of university students in learning programming. Procedia - Social and Behavioral Sciences, 59, 415–420.