

Virtual Reality-Based Programming Learning System Using Random Forest for Assessing and Enhancing Programming Skills Among Computer Science Students

Lee Matthew Alar¹, Mark Ashley Cortez², Pauleen Faith David³, Foemie Garcia⁴, Lyka Mae Marzan⁵

College of Computing and Information Sciences, Systems Plus College Foundation, Angeles City, Philippines

Private, Non-Sectarian Higher Education Institution, Philippines

**Corresponding Author Email: davidpauleenfaith@gmail.com*

ABSTRACT

Programming is one of the skills that Computer Science students should acquire, although there are students who have challenges comprehending programming knowledge and solving problems by applying it. There may also be limitations in conventional programming lessons in terms of providing chances for the students to program interactively. One technology that is capable of offering immersive and interactive learning opportunities is virtual reality (VR), which gives chances to the learner to participate in learning activities within the simulation environment. Prior research studies how that immersive virtual reality helps in learning, especially if the activities are interactive (Hamilton et al., 2020). This study proposes a Virtual Reality-Based Programming Learning System Using Random Forest for Assessing and

Enhancing Programming Skills Among Computer Science Students. This system will offer programming lessons, exercises, coding exercises, and assessments in a virtual environment. The performance parameters chosen for analysis using the Random Forest classification technique include the student's assessment score, coding efficiency, errors, time taken, and task completion. Classification of the students into groups based on the level of their programming skills will be carried out using the identified performance parameters. From the assessment, the students will receive feedback and learning activities that will improve their skills. The purpose of this system is to provide a system of interactive learning and performance evaluation based on students' assessment.

SECTION I — INTRODUCTION (THE PROBLEM AND ITS SETTING)

1.1 Introduction

Programming is an important skill in Computer Science because it requires students to understand programming concepts, write code, solve problems, and apply logical thinking. However, many students find programming difficult because they need to understand abstract concepts while practicing them through coding activities. This study proposes a Virtual Reality-Based Programming Learning System that provides an interactive environment where Computer Science students can learn and practice programming concepts. The system will also use the Random Forest algorithm to assess students' programming performance based on their activities and results. Through the integration of VR and Random Forest, the study aims to provide an interactive learning environment while supporting the assessment of students' programming skills.

1.2 Background of the Study

Programming is one of the essential skills in Computer Science since it involves the understanding of concepts, coding, problem-solving, and algorithmic thinking. However, programming can be difficult for beginners since students should be able to comprehend abstract ideas and apply them simultaneously through programming exercises. Recent research has examined the application of immersive and interactive technologies in programming education. For instance, Srimadhaven et al. (2020) examined the application of Virtual Reality (VR) in a programming course in higher education institutions. Their study used VR and learning analytics to support programming education and analyze learners' performance.

Virtual reality is an engaging and immersive technology that enables students to take part in learning activities in a simulated setting. A systematic review conducted by Conrad et al. (2024) found that immersive virtual reality provides opportunities for action-oriented learning and can support the development of knowledge and skills. The study also emphasized that effective learning through VR depends not only on the technology itself but also on appropriate instructional design and methodological implementation. In addition, Yang et al. (2024) reported a positive overall effect of VR on the practical skills of students in science and engineering education based on a meta-analysis of previous studies.

The use of VR has also been specifically applied to programming education. Wee et al. (2022) developed iProgVR, a VR environment designed to improve introductory programming learning. Their study found that the VR intervention improved programming concept comprehension and helped address programming misconceptions. Students also rated the VR intervention as more engaging than video-based instruction. In addition, Dewi et al. (2024) examined VR in programming algorithm courses and found that VR-based learning improved students' learning outcomes compared with traditional learning. These findings suggest that VR can provide an interactive approach for students learning programming and algorithms.

In addition to improving the learning environment, there is also a need to identify and assess the

programming performance of learners. Machine learning can be used to predict or classify student performance based on learner data. One machine learning model that can be used for this purpose is Random Forest. Nachouki and Abou Naaj (2022) used the Random Forest algorithm to predict student performance and support academic advising by identifying students who may need additional support. Similarly, Nachouki et al. (2023) developed a Random Forest model for predicting student course grades using data from undergraduate computing students. Their study demonstrated the use of Random Forest for analyzing student performance and identifying important predictors of course outcomes.

Despite these studies, a methodological gap can be identified in the integration of VR-based programming instruction and machine-learning-based assessment. Existing studies have investigated VR as a learning environment for programming, while other studies have used Random Forest to assess or predict student academic performance. However, these approaches have generally been examined separately rather than being integrated into a single programming learning system. For example, Wee et al. (2022) focused on developing and evaluating a VR environment for introductory programming, while Nachouki et al. (2023) focused on using Random Forest to predict course performance.

This presents a methodological gap because there is a need for an approach that combines interactive VR-based programming activities with Random Forest-based assessment within one learning system. Such an approach can allow students to learn and practice programming in an immersive environment while their performance data, such as activity results and other learning indicators, are analyzed to determine their programming skill level. This is also consistent with Conrad et al. (2024), who emphasized that the effectiveness of immersive VR depends on appropriate instructional and methodological implementation.

Therefore, this study intends to develop a Virtual Reality-Based Programming Learning System Using Random Forest for Assessing and Improving Programming Skills of Computer Science Students. The study will provide learners with a virtual environment where they can practice programming through interactive activities while the Random Forest algorithm analyzes their performance. Through this integration, the study aims to provide both an interactive programming learning environment and a data-based method for assessing students' programming skills.

1.3 Objectives of the Study

- **General Objective:** Develop and evaluate a Virtual Reality-based programming learning system that uses the Random Forest algorithm to assess and help improve the programming skills of Computer Science students.
- **Specific Objectives**
 1. To develop a simple VR-based learning system that will allow students to learn and practice basic

programming concepts through interactive activities.

2. To use the Random Forest algorithm to assess the programming performance and skill level of the students based on their results and activities.
3. To determine if the VR-based learning system can help improve the students' programming skills.
4. To evaluate the system in terms of its usability, effectiveness, and ease of use.
5. To know the students' level of engagement and overall experience while using the VRbased programming learning system.

1.4 Scope and Limitation of the Study

- **Scope:** The study aims at designing a VR-based programming learning system intended for students of computer science. This system will be an interactive virtual environment that offers a platform to learn and practice the concepts of programming through lessons, activities, exercises, and evaluations. The Random Forest algorithm will be used to evaluate the programming capabilities of students based on their performance in the programming activities provided by the system.
- **Limitation:** The research will be limited to the Computer Science students as the target users of the system. Only the programming-related issues and activities that are available within the system will be the subject matter of the study. The output of the random forest algorithm will be based on the data of the student's performance collected through the process of the study. The effectiveness of the system could be influenced by the availability of the necessary Virtual Reality hardware and computer devices.

1.5 Significance of the Study

The significance of this research is that it presents a Virtual Reality-Based Programming Learning System Using Random Forest that might offer Computer Science students an interactive way of learning and practicing programming. The proposed system will be used to merge Virtual Reality and machine learning to facilitate programming learning, evaluation of student performance, and gathering information on the programming skills of students.

For Computer Science Students this study may be helpful for Computer Science students to learn and practice the basics of programming interactively. The proposed system may offer an alternative way of learning programming for the students, and let them practice their programming skills by using the system. The results of the evaluation might also help the students to know their programming skills and find ways to improve them.

For Programming Instructors this research might give programming instructors additional information on the programming performance of their students. The data collected from students' activities and evaluations might be useful for identifying areas in which the students have

difficulties. The instructors may use this information to help their students.

For the Computer Science Department this system may present an extra option for computer science programming education based on the combination of Virtual Reality and machine learning. It may show how new technologies can be used to develop interactive learning activities and programming skills evaluation. Also, this system may become a good learning tool for further programming education improvement.

For Future Researchers this study may become a good point of reference for future researchers interested in Virtual Reality, programming education, machine learning or assessment of students' skills. This study may become a starting point for further studies where programming concepts will be expanded, system's characteristics improved, machine learning algorithms modified or another way of assessment will be considered.

For The Researchers this study may become a good chance for researchers to develop and improve their knowledge and skills related to Computer Science, programming, software development, Virtual Reality, data gathering and machine learning.

1.6 Definition of Terms

Computer Science Students - These are the students who will use the proposed Virtual Reality-Based Programming Learning System. The performance, programming skills, engagement, and overall learning of the students will be determined during the study.

Programming Concepts - These are the basic programming concepts chosen to be incorporated in the learning activities of the proposed system.

Programming Performance - This is the output of the students when doing programming activities using the proposed system. This could be the answers of the students, their mistakes, accuracy and other outputs from the activities which could be used as a basis to evaluate their programming skills.

Programming Skills - This is the capability of Computer Science students to understand and apply programming concepts, logic, and skills in doing the programming activities using the proposed system.

Random Forest - This is an algorithm in machine learning which utilizes several decision trees for classification or prediction. Random Forest will be used in this study to evaluate the programming performance of students and aid in determining the programming skill level of the students through their performance and activities.

Skill Level - This is the programming ability level determined from the programming performance and assessment of the students. For the purposes of this study, the performance and activities of the students will be used as data by the Random Forest algorithm to help in assessing their programming skill level.

Student Engagement - This is the extent of participation and involvement of the students while using the VR-based programming learning system.

Usability - Usability is defined as the ease with which students can access and interact with the proposed system. Usability will be among the factors evaluated to determine the effectiveness of the system developed.

Virtual Reality (VR) - It is a technology that generates an environment in which users can interact with learning materials and activities. In the current research, VR will be applied to create an interactive environment for the purpose of learning basic programming concepts.

VR-Based Programming Learning System - The VR-based programming learning system will be the proposed system that uses Virtual Reality to create an interactive learning environment for learning and practicing basic programming concepts among computer science students. The system will also employ Random Forest to evaluate the performance and skill level of the students in programming.

SECTION II — LITERATURE REVIEW

2.1 Introduction to the Review

This chapter presents and discusses literature and related studies relevant to the development of a Virtual Reality-Based Programming Learning System using the Random Forest algorithm for assessing and enhancing the programming skills of Computer Science students. The reviewed literature provides a foundation for understanding how Virtual Reality (VR) can be used as an interactive learning environment and how machine learning can be applied to evaluate students' learning performance.

The literature is organized into several themes. The first theme focuses on Virtual Reality in Education, particularly its use in creating immersive and interactive learning experiences for students. The second theme discusses Virtual Reality in Programming and Algorithm Learning, including the use of visualization and interactive activities to help students understand programming

concepts and algorithms. The third theme covers Programming Education and Student Skill Development, focusing on the challenges students may experience when learning programming and the need for learning approaches that support logic, problem-solving, and understanding of code execution. The fourth theme discusses Machine Learning for Student Performance Assessment, particularly the use of the Random Forest algorithm to analyze learning data and classify or assess students' programming skill levels.

The final theme examines the integration of Virtual Reality and machine learning in programming education. Existing literature shows the potential of VR to make programming and algorithm learning more interactive, while machine learning can provide information about student performance and learning patterns. However, the integration of these technologies within a specialized VR-based programming learning environment remains an area that requires further exploration. This provides the basis for the proposed study, which aims to combine interactive VR programming activities with Random Forest-based assessment to provide students with a more engaging and data-informed learning experience.

2.2 Review of Related Literature and Studies

Global Literature and Studies Virtual Reality in Education

Virtual Reality (VR) has increasingly been explored as an educational technology because it can provide immersive, interactive, and experiential learning environments. Radianti et al. (2020) reviewed immersive VR applications in higher education and found that VR environments commonly support visualization, interaction, simulation, and experiential learning. Their review emphasized that the effectiveness of VR depends not only on the technology but also on the design of learning activities and the objectives of the instruction. Similarly, Di Natale et al. (2020) reported that immersive VR provides affordances that can support learning through interaction and presence.

Hamilton et al. (2021) systematically reviewed experimental studies comparing immersive VR with less immersive learning approaches. Their findings showed that many studies reported positive learning outcomes from immersive VR, although some studies found no significant difference compared with conventional approaches. This indicates that VR should not automatically be considered superior to traditional instruction; instead, its effectiveness depends on appropriate instructional design and assessment.

Coban et al. (2022), through a meta-analysis, found an overall positive but relatively small effect of immersive VR on learning outcomes. The results also suggested that the effects varied according to educational level and learning environment. Likewise, Conrad et al. (2024) concluded that immersive VR can support action-oriented learning and the development of knowledge and skills, while emphasizing the importance of carefully designed learning environments.

Won et al. (2023) examined different approaches to immersive VR learning and found that VR

learning activities vary considerably in terms of interactivity, openness, and personalization. Their review of 219 empirical studies suggests that technological features alone are insufficient; learning activities must also be designed around appropriate pedagogical objectives.

Cabrera-Duffaut et al. (2024) further examined immersive learning platforms in higher education and reported that VR has been used to support competence development across different academic contexts. The study highlights the potential of immersive environments for active participation and skill development.

Virtual Reality in Computer Science and Programming Education

VR is particularly relevant to Computer Science education because programming involves abstract processes that can be difficult for beginners to visualize. Pirker et al. (2020) conducted a systematic review of VR applications in Computer Science education and identified VR as a potential tool for interactive learning in areas such as algorithms and programming. Their review also identified challenges involving usability, interaction, and the design of appropriate learning activities.

Wee et al. (2022) developed iProgVR, a virtual reality environment specifically designed for introductory programming. Their study addressed the abstract nature of programming by representing programming concepts using concrete objects within a virtual environment. The evaluation reported improvements in programming concept comprehension and misconceptions, while participants also considered the VR intervention more engaging than video-based instruction.

Srimadhaven et al. (2020) investigated the use of VR and learning analytics in a higher-education programming course. Their work demonstrates how VR can be combined with student activity data to examine learning behavior and programming performance. This is particularly relevant to the present study because the proposed system also collects student performance information while students participate in programming activities.

Dewi et al. (2024) examined the use of VR in programming algorithm courses and reported that VR-based learning can support student learning outcomes. The study is relevant to the proposed system because it demonstrates the application of immersive environments specifically to programming and algorithm learning rather than to general education alone.

Jin et al. (2024) developed and evaluated a VR-based drone programming teaching system. Their results indicated that the immersive programming environment promoted student engagement and interaction while supporting STEM literacy and computational thinking. This demonstrates that VR can be extended beyond passive visualization toward interactive programming-related activities.

Agbo et al. (2023) designed and evaluated a VR game-based application for computational thinking. Their study demonstrated how VR can combine interaction and game-based activities to support computational-thinking development.

A recent scoping review by Janani et al. (2026) examined 30 peer-reviewed VR applications in

Computer Science education. The review found that many existing applications focus on conceptual learning involving algorithms, object-oriented programming, and computational thinking, while comparatively fewer applications provide extensive practical programming activities. It also identified usability, collaboration, and theoretical-design issues as continuing challenges. This finding provides a strong basis for developing a system that combines immersive programming activities with performance-based assessment.

Challenges in Learning Programming

Programming education remains challenging for many students because learners need to understand programming concepts while applying logic, problem-solving, and coding skills. Cheah (2020) reviewed factors contributing to difficulties in learning computer programming and identified problem-solving ability, programming concepts, and other learner-related factors as important challenges. Similarly, Qian et al. (2021) reviewed research on introductory programming and discussed different approaches for improving teaching, learning, assessment, and student performance.

Morais et al. (2020) also conducted a systematic literature review on algorithms and programming in higher education and reported that students experience different difficulties when learning programming. More recent research continues to identify challenges in introductory programming, including difficulties with programming concepts, problem-solving, motivation, engagement, and appropriate teaching methods (Efan et al., 2023; Reddy et al., 2024). These findings show the need for learning approaches and tools that can provide students with interactive activities, appropriate feedback, and support for different levels of programming ability.

More recently, Kalelioğlu and Keskinkılıç (2026) reviewed research on difficulties in higher-education programming courses. Their review identified lack of conceptual knowledge, misconceptions regarding syntax and semantics, programming skills, problem-solving skills, and difficulties in developing mental models of computer operation as major challenges. These findings support the need for learning environments that provide interactive practice and feedback.

Prather et al. (2024) also discussed difficulties experienced by novice programmers, particularly in programming problem-solving and metacognitive processes. Their work emphasizes the importance of learning environments that help students understand their errors and improve their problem-solving strategies.

Machine Learning for Student Performance Assessment

Machine learning has increasingly been applied to educational data to identify patterns in student behavior and predict academic performance. Su et al. (2022) investigated programming-course data using machine-learning classification techniques. Their study used quizzes and programming-system logs to identify learning characteristics and predict student performance. Random Forest was

among the algorithms evaluated, demonstrating the feasibility of using programming activity data for performance prediction.

Nachouki and Abou Naaj (2022) applied Random Forest to predict student academic performance. Their study demonstrated that Random Forest can be used to analyze multiple academic variables and identify students who may require additional academic support. This is relevant to the present study because the proposed system similarly uses several performance indicators rather than relying solely on examination scores.

Nachouki et al. (2023) further applied Random Forest to predict course grades using data from 650 undergraduate computing students. Their research demonstrated that Random Forest can analyze multiple predictors and determine the relative importance of different variables in predicting student performance.

A systematic review by a 2025 study on machine learning in higher education found that machine-learning techniques are increasingly being used to predict student performance, engagement, and self-efficacy. Commonly used algorithms include decision trees, Random Forest, support vector machines, and neural networks, with accuracy, precision, recall, and F1-score frequently used to evaluate classification models.

These studies support the use of Random Forest in the proposed research because the system collects multiple indicators, including assessment scores, errors, completion time, completed activities, and coding accuracy. These indicators can be processed to classify students according to predefined programming-skill levels.

National Literature and Studies: Philippine Context

The use of immersive and technology-supported learning has also been investigated in Philippine educational settings. Ante et al. (2022) examined VR in science classes at the University of Southern Philippines Foundation. Their quasi-experimental study reported significantly higher learning performance for students exposed to VR compared with students receiving traditional instruction. This demonstrates the potential applicability of immersive learning technologies within the Philippine educational context.

A more recent Philippine study by Antonio et al. (2026) examined a VR-integrated guided inquiry approach among Grade 7 students. The study reported significant improvement in post-test performance and increased engagement and visualization, although some students experienced mild discomfort such as dizziness and eye strain. These findings demonstrate both the educational potential and implementation considerations associated with VR.

Tolentino and Varela (2025), from the Technological University of the Philippines and University of Santo Tomas, investigated VR-based biology instruction and its effects on conceptual understanding and perceived usefulness. Their work provides additional Philippine evidence that VR can be used to support the understanding of complex concepts.

Figuerola et al. and related University of the Philippines Open University researchers have also explored VR-based learning experiences. A study involving VR photo-based tours reported that immersive technology could contribute to student enjoyment, engagement, and perceived learning, while also identifying implementation challenges.

Almodiel et al. (2024) examined user acceptance and experiences of a VR-enhanced campus tour at the University of the Philippines Open University. The study demonstrates that VR can be implemented within a Philippine open-university environment and highlights the importance of user experience and acceptance when evaluating immersive educational technologies.

Arada (2025) investigated the use of Tinkercad for Arduino programming education among 42 first-year Computer Technology students at a Philippine state university. The study evaluated usability, student engagement, and academic performance. While many students were willing to use the platform, the study also identified infrastructure and hands-on-learning limitations. This supports the importance of evaluating usability and technical constraints when implementing technology-supported programming education in the Philippines.

Garcia (2025) investigated skill mastery among introductory programming students using cognitive diagnostic modeling. The study emphasized that evaluating only overall academic performance may fail to identify students' specific programming weaknesses, supporting the need for more detailed learner profiling and skill assessment.

Philippine researchers have also begun applying machine learning to computing-related student data. Yu et al. (2025) examined the prediction of programming anxiety among Computer Studies students using automated machine learning and student demographic, academic, and behavioral data. The study demonstrates the growing application of data-driven techniques to understand and support students in computing disciplines.

Local and Contextual Studies

Within Philippine higher-education and computing contexts, technology-supported programming environments have increasingly been examined in terms of usability, engagement, and student performance. The Tinkercad study by Arada (2025) demonstrates how a simulated environment can provide programming-related practice when physical laboratory access is limited. However, the study also emphasizes that simulation does not completely eliminate the need for practical experiences.

The VR campus-tour research conducted by Almodiel et al. (2024) similarly demonstrates the relevance of user acceptance and experience when implementing VR in Philippine higher education.

These Philippine studies are relevant to the proposed research because they establish that immersive and simulated learning technologies can be implemented in local educational settings. However, most of the identified Philippine studies focus on science education, campus experiences, simulations, or general technology acceptance rather than an integrated VR environment specifically designed for programming instruction and automated programming-skill classification.

The available Philippine literature also shows a need to consider usability, hardware availability, internet or computer infrastructure, and student experience when developing educational systems. These considerations are consistent with the scope and limitations of the proposed research, which recognizes that VR hardware and computer requirements may influence system implementation.

Focal Literature and Studies

The focal literature and studies provide the most direct basis for the development of the proposed Virtual Reality-Based Programming Learning System Using Random Forest for Assessing and Enhancing Programming Skills Among Computer Science Students. Unlike the broader global, national, and local literature, these studies specifically address one or more of the core components of the proposed system: virtual reality for programming education, programming performance analysis, machine-learning-based assessment, and personalized learning support

Wee et al. (2022) developed iProgVR, a virtual reality environment specifically designed for introductory programming. The study demonstrated how VR can be used to represent programming concepts through interactive activities and reported improvements in students' understanding of programming concepts. This study is closely related to the present research because both systems use VR as an environment for learning and practicing programming.

Srimadhaven et al. (2020) explored the use of VR and learning analytics in a higher-education programming course. Their study is relevant because it demonstrates the possibility of collecting and analyzing student activity data within a VR programming environment. This provides a foundation for the present study's approach of collecting performance indicators while students perform programming activities.

Dewi et al. (2024) investigated the impact of VR on programming algorithm courses and examined its relationship with student learning outcomes. Their findings provide support for applying immersive environments to programming and algorithm instruction. The present study extends this concept by incorporating automated performance assessment through Random Forest.

Su et al. (2022) applied machine-learning techniques to student learning data and performance prediction. Their research is particularly relevant to the present study because programming activity

data can be analyzed to identify learning patterns and predict performance. This supports the proposed collection of data such as errors, accuracy, completion time, and assessment results.

Nachouki and Abou Naaj (2022) used Random Forest to predict student academic performance and identify students who may require additional support. Their study demonstrates the suitability of Random Forest for analyzing multiple student-performance variables. In the proposed system, Random Forest will similarly process multiple indicators to classify students according to predefined programming skill levels.

Nachouki et al. (2023) further demonstrated the application of Random Forest in predicting student course grades and analyzing the importance of different predictors. This is relevant to the present study because the proposed system also uses multiple performance variables rather than relying exclusively on a single test score.

Janani et al. (2026) reviewed applications of VR in Computer Science education and identified programming, algorithms, and computational thinking as important areas where VR has been applied. The review also identified continuing opportunities for developing more practical and interactive VR learning applications. This supports the need for a system that combines immersive programming practice with performance assessment.

Synthesis of Related Literature and Studies

Overall, the reviewed literature demonstrates that VR can support interactive and immersive learning, particularly when learning activities require visualization, practical interaction, or engagement. Research specifically addressing Computer Science education shows that VR can be used to represent algorithms, programming concepts, computational thinking, and other abstract concepts in more interactive ways. However, recent reviews indicate that many VR applications remain focused on conceptual learning rather than comprehensive programming practice.

The literature on programming education further establishes that students may experience difficulties in understanding programming concepts, solving programming problems, developing accurate mental models, and identifying their own errors. These challenges emphasize the importance of providing opportunities for repeated practice, interaction, assessment, and feedback.

Meanwhile, machine-learning research demonstrates that student performance can be analyzed using multiple data sources. Random Forest has been applied to academic-performance prediction and can accommodate multiple predictors, making it suitable for a system that collects different programming-performance indicators.

In the Philippine context, existing studies demonstrate the feasibility of VR, simulation, and technology-supported learning environments. However, the reviewed Philippine studies generally address science education, virtual tours, simulation-based learning, or general student experiences rather than an integrated VR programming-learning and Random Forest assessment system

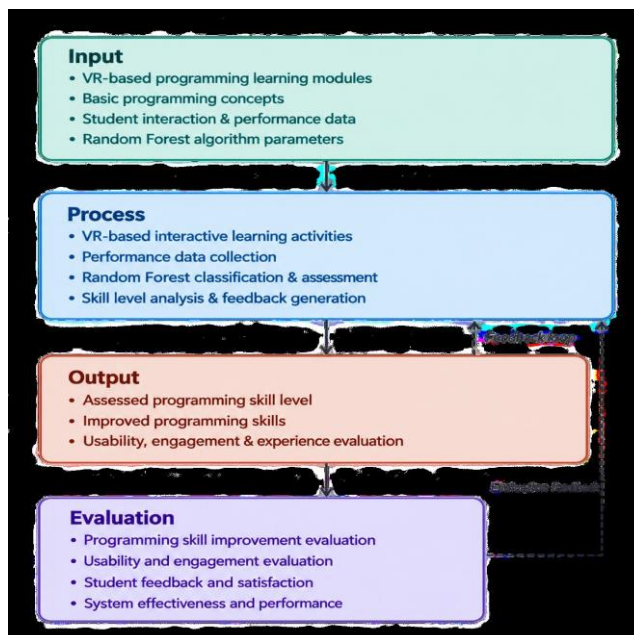
Thus, the identified research gap is the limited integration of immersive VR programming instruction and machine-learning-based programming-skill assessment in a single educational system, particularly within the Philippine Computer Science context. The present research seeks to address this gap by developing a VR environment where Computer Science students can learn and practice programming, while Random Forest analyzes their performance and classifies their programming skill level. This directly supports the conceptual framework already established in the study, where VR activities generate performance data that are processed through Random Forest and converted into skill classification and feedback.

2.3 Conceptual Framework

This study is guided by an Input-Process-Output (IPO) framework, which illustrates how the Virtual Reality-Based Programming Learning System takes in student and learning data, processes it through VR-based instruction and Random Forest analysis, and produces measurable outcomes on students' programming skills.

Figure 1

Conceptual framework of the study



The Input consists of the VR-based programming learning modules covering basic programming concepts, together with the data generated as students interact with the system, such as their performance and activity data. This information, along with the parameters used by the Random

Forest algorithm, forms the basis for the next stage of the framework.

The Process stage covers the core activities of the study. Students engage in interactive VR-based learning activities, during which their performance data, including errors, accuracy, and time spent on tasks, are collected. This data is then analyzed using the Random Forest algorithm to classify students according to their skill level and generate corresponding feedback.

The Output represents the results of the study, consisting of the assessed programming skill level of each student, the observed improvement in their programming skills, and the evaluation of the system in terms of usability, engagement, and overall learning experience.

The Evaluation stage determines the effectiveness of the system by examining changes in students' programming performance, the accuracy of skill-level classification, system usability, and student engagement and satisfaction. The results are used to identify areas for improvement and provide feedback for subsequent learning activities.

A feedback loop connects the evaluation results back to the Process, allowing the system to use the assessment results and evaluation findings to improve subsequent VR-based learning activities. This supports the system's role as an interactive and responsive learning tool for Computer Science students.

2.4 Synthesis

The literature reviewed for this study points to two overlapping developments in educational technology: the growing use of virtual reality (VR) as an interactive learning tool, and the use of machine learning to analyze and assess student performance. Several studies support the value of VR as an immersive tool for helping students engage with course material (Cabrera-Duffaut et al., 2024; Radianti et al., 2020). When applied specifically to computer science education, VR has been found useful for helping students visualize and understand abstract and complex topics such as algorithms and programming logic (Janani et al., 2026; Mukasheva et al., 2023; Dewi et al., 2024). These studies generally agree that VR's interactivity and three-dimensional visualization make it well-suited for topics that are otherwise difficult to grasp through traditional instruction alone.

At the same time, not all findings on VR are uniformly positive. Makransky et al. (2019) found that immersive VR can increase students' sense of presence but does not automatically lead to better learning outcomes compared to non-immersive formats. This suggests that VR's effectiveness depends less on the technology itself and more on how its activities are designed and aligned with specific learning goals (Janani et al., 2026), a consideration this study takes into account by focusing the VR activities on selected, clearly defined basic programming concepts rather than broad, unstructured exploration.

Separately, Su et al. (2022) demonstrated that machine learning techniques, including Random Forest, can be applied to programming-related learning data, such as logs of student activity, to analyze learning behavior and predict performance. This shows that algorithmic assessment can extend beyond simple quiz or exam scores to capture a more complete picture of how students actually perform.

Taken together, the reviewed studies show that VR and machine learning have each been studied and applied on their own, VR for interactive learning and visualization, and machine learning for behavioral analysis and performance prediction, but rarely combined within a single system for programming education (Radianti et al., 2020; Su et al., 2022). This is the specific gap that the present study addresses: an integrated VR-based programming learning environment in which the Random Forest algorithm is used directly to assess and classify students' programming skill level based on their in-system performance, rather than relying on traditional testing methods alone.

SECTION III — METHODOLOGY

3.1 Research Design

In this study, a Developmental Research Design with Quantitative Approach will be used to conduct systematic designing, development, and evaluation of Virtual Reality Based Programming Learning System Using Random Forest for Assessing and Improving Programming Skills among Computer Science Students. In this context, the mentioned research design is most appropriate, because it combines the technological implementation of an interactive Virtual Reality (VR) application with empirical research methodologies perfectly. Specifically, the developmental component will help with designing the architecture of VR application, integrating algorithms into it, and refining it through iterations along with Random Forest skill assessment engine. At the same time, the quantitative approach will provide an empirical base for measuring system performance, analyzing logs of interactions between students and the VR application, as well as assessing the accuracy of the prediction model.

3.2 System Development Methodology

For the development of the proposed Virtual Reality-Based Programming Learning System Using Random Forest, the researchers will employ the Waterfall Model. This is a sequential model of development whereby after completion of one stage, the next one comes into play. It offers the researchers a structured way of developing the proposed system.

To start with, the researchers will undertake requirements analysis. The aim of the process will be to come up with the requirements of the proposed system. The requirements will range from user requirements, programming learning activities, VR requirements, to Random Forest assessment data. Functional and non-functional requirements of the system will also be established.

After identification of the requirements, the next process that the researchers will undertake is that of system design. The focus here will be on coming up with the design of the system. It will range from the user interface of the system, the VR part, programming activities, and Random Forest assessment. The system design will act as a guide to the researchers when developing the proposed system.

The following phase will involve implementation where the researchers will construct the system based on the identified requirements and approved design. All components required for the system construction such as the Virtual Reality learning environment, programming activities and others will be constructed. The Random Forest component will also be incorporated within the system according to the designed plan for the system.

After the construction of the system, it will undergo testing and evaluation. Testing will be done by the researchers to help them detect errors and find out whether the functionalities of the system meet the identified requirements. At the same time, the system will be evaluated in terms of the research objectives.

The following phase will involve implementation where the researchers will construct the system based on the identified requirements and approved design. All components required for the system construction such as the Virtual Reality learning environment, programming activities and others will be constructed. The Random Forest component will also be incorporated within the system according to the designed plan for the system.

After the construction of the system, it will undergo testing and evaluation. Testing will be done by the researchers to help them detect errors and find out whether the functionalities of the system meet the identified requirements. At the same time, the system will be evaluated in terms of the research objectives.

The final phase will be maintenance where all identified errors and the need for improvement of the system based on the results of the testing and evaluation process will be taken into consideration. In general, the Waterfall Model will offer the researchers a step-by-step procedure for constructing the proposed Virtual Reality-Based Programming Learning System Using Random Forest. By following each phase, the researchers will be able to structure the process of development of the system and consider its requirements and design properly.

3.3 Requirement Specification and Analysis

The requirement specification and analysis outline the proposed Virtual Reality-Based Programming Learning System Using Random Forest's functional and non-functional requirements. It is described by the system's main functions that should be installed to achieve the goals of

learning programming, assessing students' performance, and providing feedback. The system is meant for Computer Science students. Instructors or administrators manage the lessons on programming, users' accounts, and assess students' performance. The functional requirements define what activities the system should accomplish. The non-functional requirements determine what requirements the system should fulfill. To illustrate the system's requirements, Data Flow Diagrams (DFD), an Entity-Relationship Diagram (ERD), and flowchart will be applied.

Functional Requirements

The proposed system shall provide the following functional requirements:

Table 1

No.	Functional Requirement	Description
FR1	User Registration and Login	The system shall allow users to create accounts and securely log in to the system.
FR2	Student Profile Management	The system shall store and manage student information such as name, course, year level, and section.
FR3	Programming Lessons	The system shall provide programming lessons covering selected basic programming concepts.
FR4	VR Learning Activities	The system shall provide interactive virtual reality activities related to programming concepts.
FR5	Programming Visualization	The system shall visualize programming processes and concepts to help students understand how programs work.
FR6	Activity Recording	The system shall record student activity, scores, attempts, completion time, and other relevant performance data.
FR7	Performance Assessment	The system shall assess the student's programming performance using collected activity and assessment data.
FR8	Random Forest Assessment	The system shall use the Random Forest algorithm to classify or assess the student's programming skill level based on performance data.
FR9	Results Display	The system shall display assessment results and identified programming skill levels to the student.

FR10	Feedback and Recommendations	The system shall provide feedback and recommendations based on the student's assessment results.
FR11	Administrator/Instructor Management	The system shall allow authorized administrators or instructors to manage user accounts, programming lessons, and assessment-related information.
FR12	Data Storage and Retrieval	The system shall store and retrieve student learning and assessment records for monitoring and reporting purposes.

Non-Functional Requirements

The proposed system shall also satisfy the following non-functional requirements:

Table 2

No.	Non-Functional Requirement	Description
NFR1	Usability	The system should provide a simple and understandable interface that students can easily navigate.
NFR2	Performance Efficiency	The system should respond efficiently when loading lessons, recording activities, processing assessments, and displaying results.
NFR3	Security	The system should protect user accounts, student information, and assessment records from unauthorized access
NFR4	Reliability	The system should consistently record learning activities and assessment results without unnecessary data loss.
NFR5	Maintainability	The system should be organized so that programming lessons, activities, and system components can be updated or maintained.
NFR6	Compatibility	The system should operate with the required hardware and software environment for the VR-based learning activities.

Data Flow Diagram

The Data Flow Diagram (DFD) shows how information flows between users and the proposed system. The DFD Level 0 depicts the system as one main process and depicts major inputs and outputs that flow between external entities and the system.

Figure 2. Data Flow Diagram Level 0 – Context Diagram

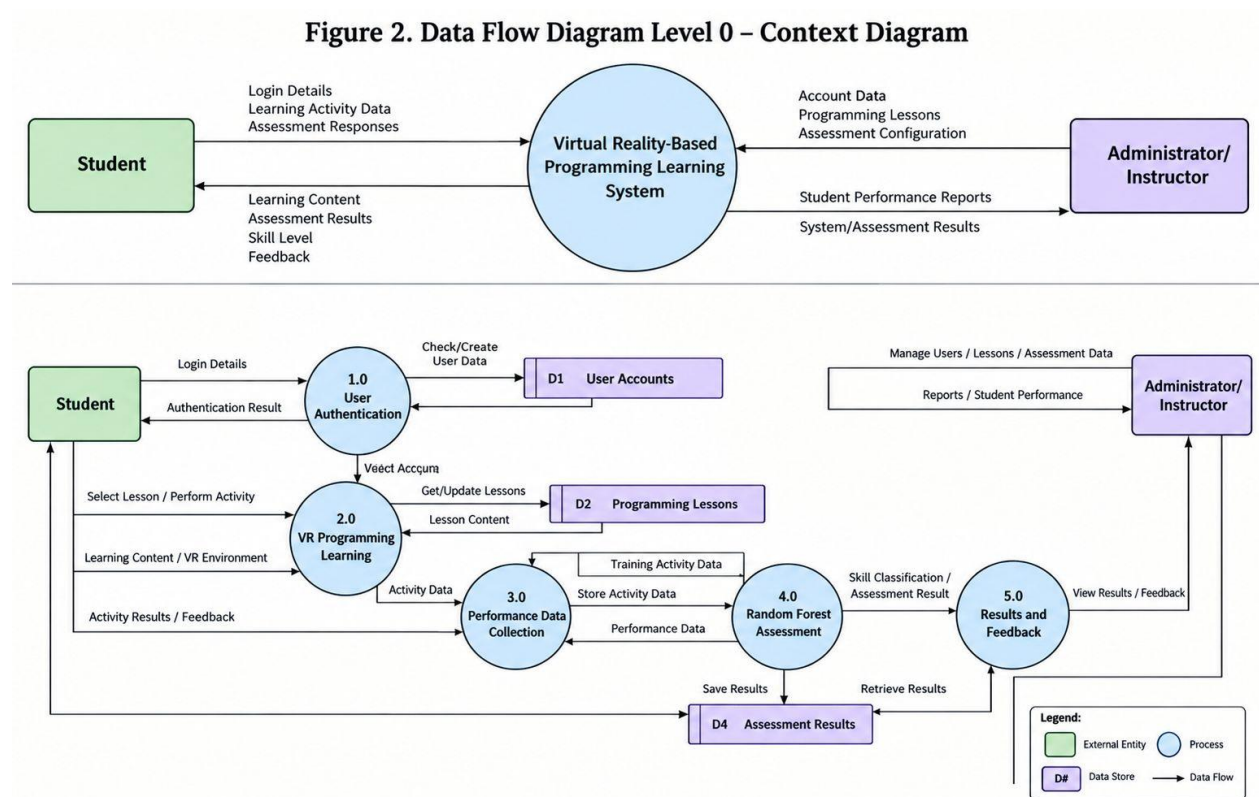


Figure 2.0 represents the overall interaction of the proposed system with its primary users. The student enters the login details, data about learning activities, and responses to assessments. In turn, the system provides learning material, assessment results, information about the level of skills, and feedback. The Administrator/Instructor enters information about accounts, programming lessons, and assessments and receives reports regarding students' performance and the system's results.

Data Flow Diagram Level 1

DFD Level 1 describes the processes of the proposed system in more detail. It shows the inputs and outputs of the main system processes such as user authentication, VR programming learning activities, performance data collection, Random Forest assessment, and results and feedback production. The diagram also shows external entities that are used to store user account data, programming lessons, student activity, and results.

Figure 3. Data Flow Diagram Level 1 - System Overview

Figure 3. Data Flow Diagram Level 1 – System Overview

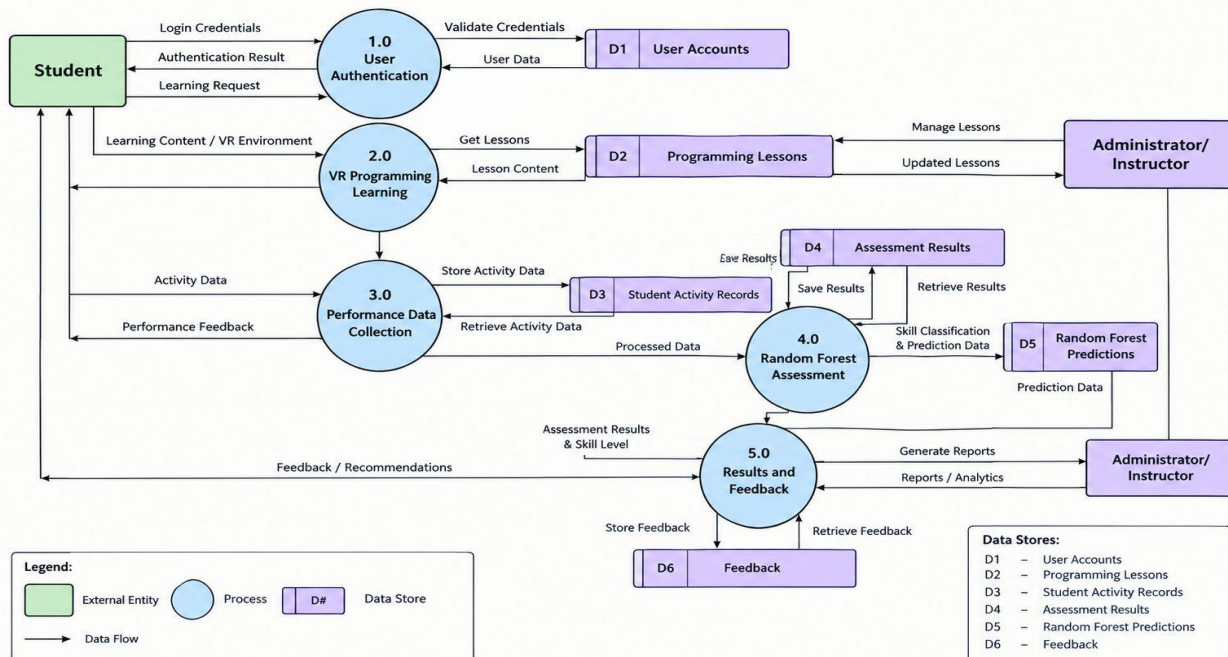


Figure 3.0 represents the main processes and data stores of the proposed system. Thus, the User Authentication process ensures the identification of users, and the VR Programming Learning process provides them with programming courses in virtual reality. The student activity and performance data store contain information about users' behaviors and acquired skills, which the Random Forest Assessment process uses to produce a skill classification. Finally, the Results and Feedback process presents the obtained results and recommendations to the users, while the Administration process allows maintaining accounts, programming courses, and assessments.

Entity-Relationship Diagram

The Entity-Relationship Diagram (ERD) below represents the main entities, their attributes, primary and foreign keys, and relationships established in the proposed system. The ERD below demonstrates the connection between student information, programming lessons, VR activities, activity records, assessment results, random forest predictions, and feedback stored in the system's databases.

Overall, the proposed database contains eight main entities, including user accounts, student profiles, programming lessons, VR activities, activity records, assessment results, random forest predictions, and feedback

Figure 4. Entity-Relationship Diagram (ERD).

Figure 4. Entity-Relationship Diagram (ERD)

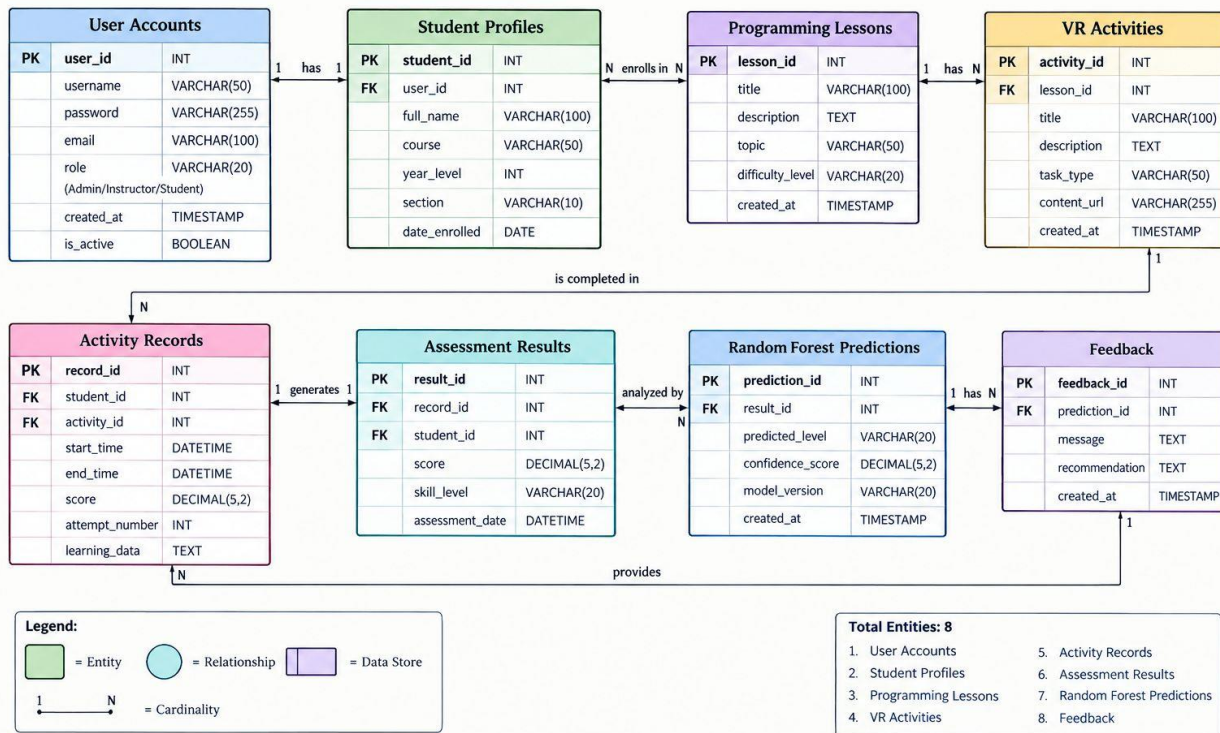


Figure 4.0 represents the eight key entities of the proposed system. The User Accounts entity stores the accounts' credentials and user roles. Student Profiles contain students' personal and academic details. Programming Lessons hosts the lessons on programming, whereas the VR Activities is a repository of activities tied to particular lessons. Similarly, Activity Records holds the details of students' activities and performance. The Assessment Results entity stores the results of students' performance assessments. Random Forest Predictions stores the skill classification from the Random Forest algorithm, whereas the Feedback entity contains the feedback and recommendations.

System Flowchart

The system flowchart below represents the general sequence of activities that the suggested Virtual Reality-Based Programming Learning System design will involve. It starts with users logging in and selecting the desired programming lessons, followed by the VR learning activities, obtaining performance data, Random Forest assessment, and receiving the results and feedback.

Figure 5. System Flowchart of the Proposed VR Programming Learning System

Figure 5. System Flowchart of the Proposed VR Programming Learning System

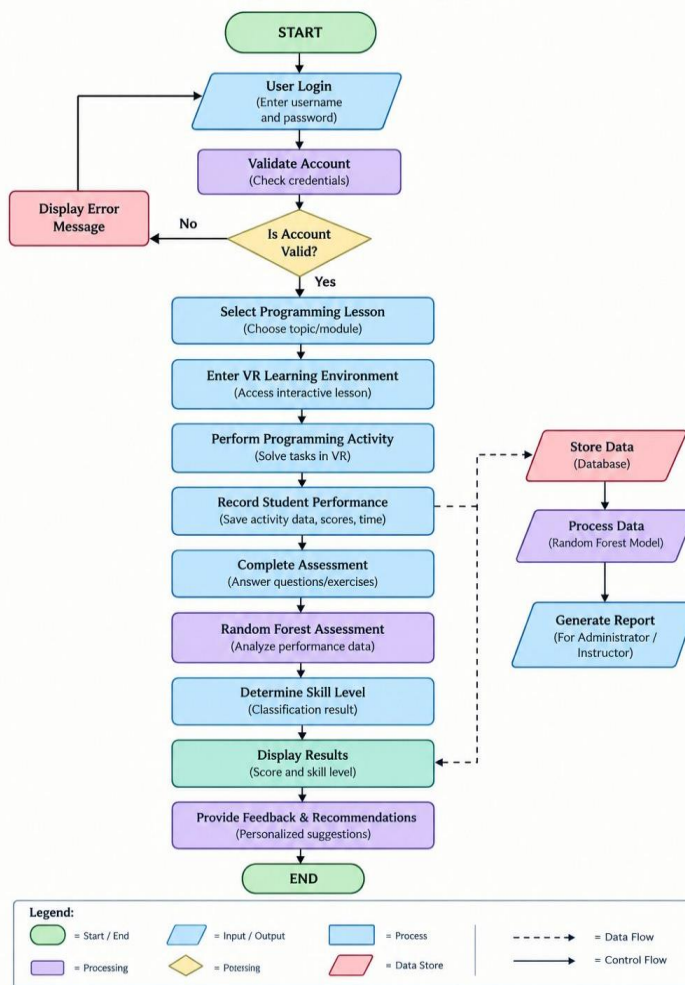


Figure 5.0 System Flowchart of the Proposed VR Programming Learning System. First, the student logs in the system, and their account is verified. Then, the student accesses the desired programming lesson and is immersed in the VR learning space. The student performs the indicated programming tasks, and the system collects the relevant data. Upon the completion of the required assessment, the gathered data are subjected to the Random Forest training model to predict the student's programming skill level. The system outputs the performance evaluation results and provides the individual feedback and recommendations.

Overall, the requirement specification and analysis were completed for the proposed VR programming learning system. The developed DFD Level 0 and Level 1 demonstrate the data flows within the system, whereas the ERD represents the database's entities and relations. Finally, the system flowchart displays the general processes from the user authentication and VR learning session initiation to the performance assessment and Random Forest modeling. All in all, the requirement analysis was achieved, which serves as the foundation for the design and development of the proposed system.

3.4 System Design Specification

The components of the proposed system include a VR Learning Module, a Programming Exercise Module, an Assessment Module, a Random Forest Prediction Module, a Feedback Module, and a Database Module. The system will enable the user to enter a virtual learning environment, learn programming concepts, carry out programming tasks, be assessed, and receive feedback based on their performance.

The design of the system includes logical specifications and physical specifications.

A. Logical Specifications

1. VR Learning Module

The system will have a virtual learning environment where the students will be able to explore various programming learning zones. These zones can include virtual classroom, programming activity zones, instructions, interactive objects, and assessments zones. Unity XR Interaction Toolkit has various components required for VR interactions such as object interaction, controller interactions, user interface, and locomotion, making it ideal for the intended VR environment.

2. Programming Learning Module

Programming lessons and activities related to programming concepts will be incorporated in this module. Based on the extent of the research, some of the programming concepts include variables, conditional, loops, functions, arrays, and problem solving. Lessons will incorporate explanations followed by programming activities.

3. Programming Exercise Module

Programmed activities of students will take place in the system. Their activities will be captured as performance data will include:

- test score;
- number of correct answers;
- number of programming errors;
- task completion time;
- number of completed activities; and
- coding accuracy

The variables will act as inputs for the assessment and machine learning components.

4. Assessments Module

This module will assess the level of programming skills of the student via quizzes, coding exercises, and programming challenges. Data will be saved in the database of the system and used to assess the level of performance of the student. The importance of assessment design comes from past literature that has identified that there is a relationship between outcomes and assessment process in case of BR-based learning. Hamilton et al. (2020) have pointed out that many studies conducted on

VR have paid a lot of attention to test scores.

5. Random Forest Assessments Module

The Random Forest Module will be used to analyze the data set collected on students' performances. The module will be developed using the student-performance data set collected earlier.

The algorithm works as follows

Students Performance Data → Data Preprocessing → Random Forest Model → Skill Classification → Feedback Learning Recommendations.

For instance, the classification may involve categorizing students into groups such as:

- Beginning
- Developing
- Proficient

This categorization should be done based on the criteria selected by the researcher and not the universally accepted programming skills.

Random Forest Assessments method is suitable for this task because it uses several decision trees and combines their decisions for classification. This method is also capable of using more than one input variable.

6. Feedback Module

Once the assessment process is completed, feedback will be provided by the system according to the identified weaknesses in the students' performance. If a student demonstrates low proficiency in conditional statements, the system may suggest extra practice of conditional statements. The role of this module is not only to give a name for the skill but to link assessment outcomes with further exercises. This concept is supported by the notion that VR-based learning environment can be constructed on engagement and practice (Conrad et al., 2024).

7. Database Module

The database will house the required information for the proper functioning of the system. This may comprise:

- Information about student's accounts;
- learning material;
- Programming problems;
- Assessment questions;
- Students' assessment outcomes;
- Information about programming performance;

- Random Forest Assessments Information; and
- Learning Recommendations.

Only such information that is essential for the research and operation of the system will be gathered.

B. Physical Specification

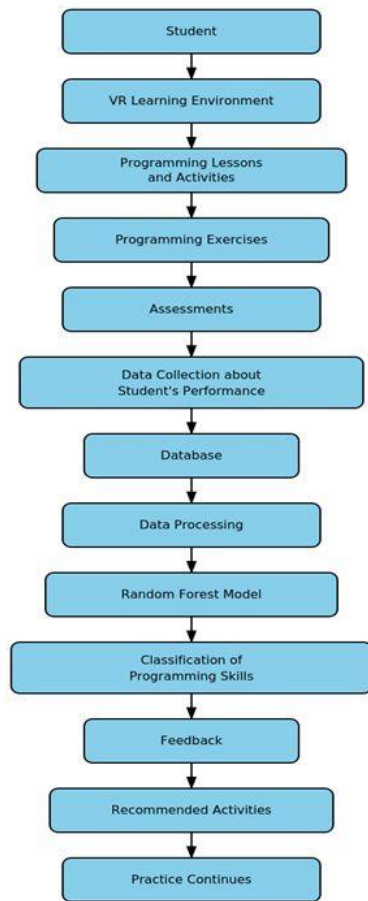
The proposed physical architecture may consist of the following components:

Table 3

Component	Proposed Specification
VR Development Platform	Unity
VR Framework	Unity XR Interaction Toolkit
Programming Language for VR	C#
Machine Learning	Python
Machine Learning Algorithm	Random Forest
ML Library	Scikit-learn
Database	My SQL or PostgreSQL
Development Computer	Computer Capable of running Unity and VR development tools
VR Hardware	Compatible VR Headset and Controllers
Network	Local Network or Internet
Storage	Local Server based storage for system and assessment data

The use of Unity is justified due to its existing tools for XR development. According to the documentation of Unity, its XR Interaction Toolkit contains components for interaction in virtual reality such as controllers, object manipulation, locomotion, and XR user interfaces.

Thus, the architecture can be described as follows:



3.5 Implementation

The proposed Virtual Reality-Based Programming Learning System will combine a VR learning environment with a Random Forest classification model to assess and enhance the programming skills of Computer Science students.

1. VR Learning Environment

Students will enter a virtual environment containing interactive activities focused on selected basic programming concepts. The activities will allow students to practice programming through hands-on interaction.

2. Performance Data Collection

While students perform the activities, the system will record important performance indicators, such as:

- Accuracy
- Number of errors
- Time spent completing tasks

These variables will be used to evaluate the student's programming performance.

3. Random Forest Assessment

The collected performance data will be processed and provided to the Random Forest algorithm. The model will analyze the student's performance and classify their programming skill level. Random Forest is appropriate for classification because it combines multiple decision trees to produce a final prediction.

4. Feedback and Improvement

After classification, the system will provide feedback based on the student's results. The assessment can be used to identify areas that need improvement and guide the student toward appropriate learning activities. This creates a feedback loop between assessment and subsequent VR activities.

5. System Evaluation

The completed system will be evaluated based on:

- Programming skill improvement
- Usability - Effectiveness
- Ease of use
- Student engagement
- Overall learning experience

3.6 Testing and Evaluation

The proposed Virtual Reality-Based Programming Learning System Using Random Forest will undergo three levels of testing: Unit Testing, Integration Testing, and User Acceptance Testing (UAT). These tests will verify the functionality of the VR learning modules, the integration of the Random Forest assessment, and the system's usability and effectiveness. The testing approach is aligned with the study's objectives of evaluating programming skill improvement, usability, effectiveness, ease of use, engagement, and overall learning experience.

Unit Testing

Each major component of the system will be tested independently to ensure that it functions correctly. This includes the VR programming activities, user input and answer checking, performance-data collection, skill-level classification, feedback mechanism, and Random Forest module. Test cases will verify whether each function produces the expected result under normal and incorrect inputs.

Integration Testing

Integration testing will determine whether the different components work correctly when combined. The researchers will test the flow from VR learning activities → collection of errors, accuracy, and

task completion time → Random Forest analysis → programming skill classification → feedback. This follows the study's conceptual framework, where student performance data are processed by Random Forest to classify skill levels and generate feedback.

User Acceptance Testing (UAT)

UAT will involve selected Computer Science students using the completed system and performing the intended programming-learning activities. After using the system, participants will evaluate it based on usability, effectiveness, ease of use, engagement, and overall learning experience, which are specified in the study objectives.

Evaluation Metrics

The system will be evaluated using the following measures:

Table 4

Evaluation Area	Metrics
System Functionality	Successful completion of required functions, error occurrence, and test-case pass rate
Random Forest Performance	Accuracy, precision, recall, and F1-score of skill-level classification
Programming Skill Improvement	Comparison of students' programming performance before and after using the system
Usability	Ease of navigation, ease of use, effectiveness, and user satisfaction
Engagement	Participation, interaction with VR activities, and willingness to continue using the system
Overall Experience	Students' overall assessment of the learning experience

REFERENCES

Agbo, F. J., Oyelere, S. S., Suhonen, J., & Tukiainen, M. (2023). Design, development, and evaluation of a virtual reality game-based application to support computational thinking. *Educational Technology Research and Development*, 71(2), 505–537. <https://doi.org/10.1007/s11423-023-10204-7>

Almodiel, M. C., Cañas-Llamas, A. M. E. F., Hernandez, M. S., Duria, A. M. C., & Villanueva, Z. E. M. (2024). Expanding campus engagement: A study on the user acceptance of a virtual reality-enhanced campus tour in an open university. *International Journal in Information Technology in Governance, Education and Business*, 6(2).

Ante, M. G. L., Balbero, R. V., & Cordesar, E. L. (2022). Virtual reality in science classes of junior high school students. *Paidagogos*, 10(1).

Arada, B. P. (2025). Arduino programming education using Tinkercad: A mixed-method study on usability and student engagement during the COVID-19 pandemic. *International Journal in Information Technology in Governance, Education and Business*, 7(1), 135–150.

Cabrera-Duffaut, A., Pinto-Llorente, A. M., & Iglesias-Rodríguez, A. (2024). Immersive learning platforms: Analyzing virtual reality contribution to competence development in higher education—A systematic literature review. *Frontiers in Education*, 9, Article 1391560. <https://doi.org/10.3389/feduc.2024.1391560>

Cheah, C. S. (2020). Factors contributing to the difficulties in teaching and learning of computer programming: A literature review. *Contemporary Educational Technology*, 12(2), Article ep272. <https://doi.org/10.30935/cedtech/8247>

Coban, M., Bolat, Y. I., & Goksu, I. (2022). The potential of immersive virtual reality to enhance learning: A meta-analysis. *Educational Research Review*, 36, Article 100452. <https://doi.org/10.1016/j.edurev.2022.100452>

Conrad, M., Kablitz, D., & Schumann, S. (2024). Learning effectiveness of immersive virtual reality in education and training: A systematic review of findings. *Computers & Education: X Reality*, 4, Article 100053. <https://doi.org/10.1016/j.cexr.2024.100053>

Dewi, I. P., Ambiyar, A., Effendi, H., Giatman, M., Hanafi, H. F., & Ali, S. K. (2024). The impact of virtual reality on programming algorithm courses on student learning outcomes. *International Journal of Learning, Teaching and Educational Research*, 23(10), 45–61. <https://doi.org/10.26803/ijlter.23.10.3>

Di Natale, A. F., Repetto, C., Riva, G., & Villani, D. (2020). Immersive virtual reality in K-12 and higher education: A 10-year systematic review of empirical research. *British Journal of Educational Technology*, 51(6), 2006–2033. <https://doi.org/10.1111/bjet.13030>

Efan, K., Krismadinata, Jama, J., & Mulya, R. (2023). A systematic literature review of teaching and learning on object-oriented programming course. *International Journal of Information and Education Technology*, 13(2), 302–312. <https://doi.org/10.18178/ijiet.2023.13.2.1804>

Hamilton, D., McKechnie, J., Edgerton, E., & Wilson, C. (2021). Immersive virtual reality as a pedagogical tool in education: A systematic literature review of quantitative learning outcomes and experimental design. *Journal of Computers in Education*, 8(1), 1–32. <https://doi.org/10.1007/s40692-020-00169-2>

Janani, A. S., Yang, Y., Lu, X., Liang, H.-N., & Baghaei, N. (2026). Virtual reality applications in computer science education: Scoping review. *Computers & Education: X Reality*, 8, Article 100159. <https://doi.org/10.1016/j.cexr.2026.100159>

Jin, Z., Bai, Y., Song, W., Yu, Q., Yue, X., & Jia, X. (2024). DVRT: Design and evaluation of a virtual reality drone programming teaching system. *Computers & Graphics*, 125, Article 104114. <https://doi.org/10.1016/j.cag.2024.104114>

Kalelioğlu, F., & Keskinçilic, F. (2026). Difficulties of learning programming in higher education: A systematic literature review of recent research. *Ankara University Journal of Faculty of Educational Sciences*, 59(2).

Lampropoulos, G., & Kinshuk. (2024). Virtual reality and gamification in education: A systematic review. *Educational Technology Research and Development*, 72(4), 1691–1785. <https://doi.org/10.1007/s11423-024-10351-3>

Morais, C. G. B., Mendes Neto, F. M., & Osório, A. J. M. (2020). Difficulties and challenges in the learning process of algorithms and programming in higher education: A systematic literature review. *Research, Society and Development*, 9(10), Article e3389109287. <https://doi.org/10.33448/rsd-v9i10.9287>

Nachouki, M., & Abou Naaj, M. (2022). Predicting student performance to improve academic advising using the Random Forest algorithm. *International Journal of Distance Education Technologies*, 20(1), 1–17. <https://doi.org/10.4018/IJDET.296702>

Nachouki, M., Mohamed, E. A., Mehdi, R., & Abou Naaj, M. (2023). Student course grade prediction using the random forest algorithm: Analysis of predictors' importance. *Trends in*

Neuroscience and Education, 33, Article 100214. <https://doi.org/10.1016/j.tine.2023.100214>

Pirker, J., Dengel, A., Holly, M., & Safikhani, S. (2020). Virtual reality in computer science education: A systematic review. In Proceedings of the 26th ACM Symposium on Virtual Reality Software and Technology (Article 27, pp. 1–8). Association for Computing Machinery. <https://doi.org/10.1145/3384382.3384533>

Prather, J., Reeves, B. N., Leinonen, J., MacNeil, S., Randrianasolo, A. S., Becker, B. A., Kimmel, B., Wright, J., & Briggs, B. (2024). The widening gap: The benefits and harms of generative AI for novice programmers. In Proceedings of the 2024 ACM Conference on International Computing Education Research (Vol. 1, pp. 469–486). Association for Computing Machinery. <https://doi.org/10.1145/3632620.3671100>

Qian, Y., Yan, S., & Pan, X. (2021). Teaching and learning to program: Umbrella review of introductory programming in higher education. Mathematics, 9(15), Article 1737. <https://doi.org/10.3390/math9151737>

Reddy, E. M., Venkateshwarlu, B., Raju, N. Ch., & Sreenivasulu, M. (2024). A systematic literature review on teaching and learning introductory programming in higher education. International Journal of Emerging Technologies in Learning, 9(1).

Srimadhaven, T., Chris Junni, A. V., Naga Harshith, S., Jessenth Ebenezer, S., & Shabari Girish, S. (2020). Learning analytics: Virtual reality for programming course in higher education. Procedia Computer Science, 172, 433–437. <https://doi.org/10.1016/j.procs.2020.05.097>

Su, Y.-S., Lin, Y.-D., & Liu, T.-Q. (2022). Applying machine learning technologies to explore students' learning features and performance prediction. Frontiers in Neuroscience, 16, Article 1018005. <https://doi.org/10.3389/fnins.2022.1018005>

Tolentino, A. N., & Varela, B. V. (2025). Learning biology through virtual reality instructional approach: Effects on conceptual understanding, performance, and perceived usefulness. International Journal of Research and Innovation in Social Science.

Wee, C., Yap, K. M., & Lim, W. N. (2022). iProgVR: Design of a virtual reality environment to improve introductory programming learning. IEEE Access, 10, 100054–100078. <https://doi.org/10.1109/ACCESS.2022.3204392>

Won, M., Ungu, D. A. K., Matovu, H., Tsai, C.-C., & Treagust, D. F. (2023). Diverse approaches to learning with immersive virtual reality identified from a systematic review. *Computers & Education*, 195, Article 104701. <https://doi.org/10.1016/j.compedu.2022.104701>

Yang, C., Zhang, J., Hu, Y., Yang, X., Chen, M., Shan, M., & Li, L. (2024). The impact of virtual reality on practical skills for students in science and engineering education: A meta-analysis. *International Journal of STEM Education*, 11(1), Article 28. <https://doi.org/10.1186/s40594-024-00487-2>

Yu, E. R., Pineda, E. D., Tano, I. M., Lagman, A. C., Victoriano, J. M., Mababa, J. C., & Pulumbarit, J. P. (2025). Predicting programming anxiety among computer studies students using automated machine learning. *International Journal of Latest Technology in Engineering, Management & Applied Science*, 14(5), 314–324.