INTERNATIONAL JOURNAL OF LATEST TECHNOLOGY IN ENGINEERING, MANAGEMENT & APPLIED SCIENCE (IJLTEMAS)

ISSN 2278-2540 | DOI: 10.51583/IJLTEMAS | Volume XIV, Issue III, March 2025

# Student Information System for Computer Studies with Integrated Programming Anxiety Level Prediction and Balanced Group Recommendations

**Eduardo R. Yu II., Elmerito D. Pineda., Isagani M. Tano., Ace C. Lagman., Jayson M. Victoriano., Jonilo C. Mababa., Jaime P. Pulumbarit**

**La Consolacion University Philippines**

**DOI : https://doi.org/10.51583/IJLTEMAS.2025.140300060**

**Received: 28 March 2025; Accepted: 09 April 2025; Published: 21 April 2025**

**Abstract—** Programming anxiety is a recognized challenge in computer studies, often affecting students' academic performance and retention. Addressing this issue requires a structured and technology-driven approach that enables faculty to identify at-risk students and implement targeted academic interventions. This study aimed to provide a solution by developing a web-based system that integrates predictive analytics to support decision-making. Specifically, it incorporated a pre-developed machine learning-based prediction model, automate student group formation using a custom heterogeneous algorithm, and featured a data visualization dashboard for faculty analysis. The system was developed using the Spiral Model to ensure iterative improvements and was evaluated based on the ISO/IEC 25010 Software Quality Model, focusing on key software quality attributes. Expert evaluation of the system's performance resulted in a grand mean score of 3.70, indicating strong quality across all metrics. The findings demonstrate that the developed system effectively integrates predictive analytics to assist higher education institutions in addressing programming anxiety. By enabling real-time identification of at-risk students and facilitating structured academic support, the system contributes to the fields of educational technology and learning analytics, offering a scalable solution for improving student outcomes in computing education.

**Keywords:** Spiral Model, Programming Anxiety, Machine Learning, Logistic, Intelligent System, Classification Algorithms

## I. Introduction

Mental health conditions such as anxiety, stress, and depression have become critical global public health concerns. These disorders significantly affect individuals' emotional well-being, cognitive functioning, productivity, interpersonal relationships, and overall quality of life (World Health Organization [WHO], 2023). In the Philippine context, anxiety, which is classified under depressive disorders, has shown a notable increase in prevalence, with a consistent upward trend observed in recent years (Department of Health [Philippines], 2021).

In the domain of computing education, anxiety is particularly salient during programming tasks, where students frequently encounter high cognitive demands, time pressures, and performance expectations (Tomić et al., 2022). Driven primarily by fear of failure and self-doubt, this anxiety undermines students' confidence and performance, leading to a condition known as programming anxiety. While self-report tools such as the Programming Anxiety Scale (PAS) developed by Yildirim and Özdener (2022) provide useful insights, they depend on manual data collection methods that limit their ability to provide timely or real-time support for interventions.

Recent advances in educational data mining and machine learning have enabled the development of predictive models to identify students at risk of academic and psychological difficulties. Several studies have successfully implemented classification algorithms to forecast general anxiety and related mental health conditions using behavioral, demographic, and academic indicators (Khor & Darshan, 2024; (Mahalakshmi & Sujatha, 2023; Mendoza et al., 2024; Mutalib, 2021; Zhai et al., 2024). Despite these contributions, a gap remains in addressing programming-specific anxiety within educational contexts and in operationalizing these predictive insights through integration with institutional systems for timely interventions.

This study aimed to develop and evaluate a student information system for computer studies that integrates a pre-developed prediction model for programming anxiety, utilizing the Logistic Regression algorithm. The system includes features such as a data visualization dashboard and an automated balanced grouping mechanism designed to foster collaborative learning environments by strategically pairing students with varying anxiety levels. The system's overall performance was evaluated using the ISO/IEC 25010 Software Product Quality Model to ensure compliance with established standards across relevant quality attributes.

By integrating an intelligent prediction model into an operational educational system, this study presents a practical framework for supporting data-driven decision-making to mitigate programming anxiety. The developed system enables educators and administrators to implement targeted and timely interventions, thereby promoting student well-being, improving academic performance, and contributing to the growing body of research on intelligent support systems in education.

### Framework of The Study

The framework of this study served as the research paradigm, providing a high-level overview of the development and integration of key system components. As illustrated in Figure 1, the Software Development Life Cycle (SDLC) approach was utilized to ensure

ISSN 2278-2540 | DOI: 10.51583/IJLTEMAS | Volume XIV, Issue III, March 2025

iterative enhancements in system design and functionality. A core feature of the system was the programming anxiety level prediction, which employed a pre-developed predictive model based on the Logistic Regression algorithm.

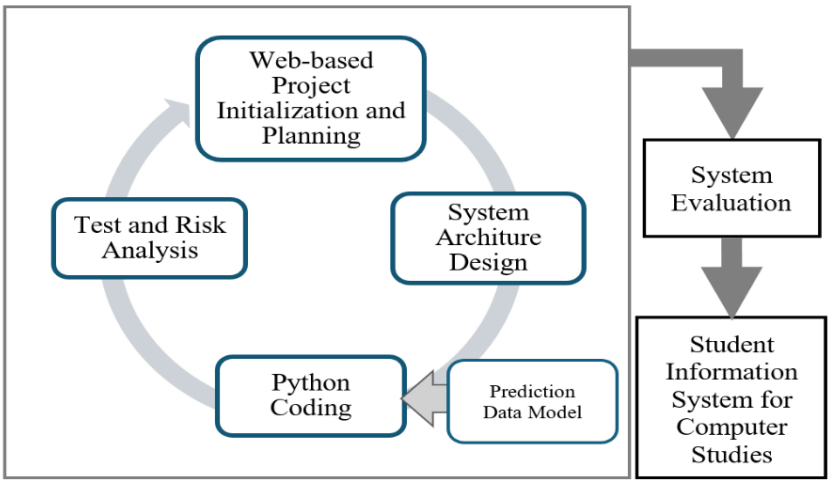


Fig. 1. Research Paradigm.

## II. Methodology

This study adopted a developmental research approach, which involved analyzing, describing, and evaluating the final software product (Ibrahim, 2016). The web-based system was developed using software development techniques, incorporating the integration of a pre-developed predictive data model.

The methodology employed by the researchers was the Spiral Model for software development, facilitating iterative refinement and continuous improvement throughout the development process.

### Spiral Model for Software Development

This model is an iterative software development methodology which provides systematic and controlled procedures in building iterative prototypes. The process strategically breaks the project into smaller, more manageable segments, enabling adaptability to growing requirements. In this model, the improvement focusses on adding functionality to the applications serves as an increasing spiral until it is ready for deployment and implementation (Yas et al., 2023). The model consists of four steps were as follows:

1) *Requirement Analysis:* It involved the collection of essential information to establish baseline data. As the system progressed through successive spirals, further requirements were explored. The main objective of this step was to identify the features and configurations of the system, along with the necessary modules specific to the context of students' programming anxiety levels.

2) *Design:* In this step, conceptual designs are drawn using online diagraming tools which include architectural, logical modules, wire frames, and the mockup design for the succeeding spirals. The researchers presented a use case analysis to show the different users and their corresponding functions, along with their access points within the system.

3) *Coding:* It focused on the development of actual software product, which involved translating the identified requirements and features into executable Python code with the help of open-source libraries and frameworks. Unit integration, local testing, and debugging were performed to enhance the software product, continuing along the spiral development model. The integration of the developed prediction data model for students' programming anxiety levels was also carried out during this process. Additionally, a custom heterogeneous algorithm was developed and implemented to generate balanced group recommendations based on predicted anxiety levels. The primary goal was to create a system that functions with an acceptable level of operations.

4) *Testing and Risk Analysis:* To determine the feature quality of prototype/next version, manual and automated unit tests were implemented through the integration of some software and testing techniques. The initial and succeeding user acceptance test was also performed and serves as the evaluation and risk assessment. The prototype was evaluated with the use of initial requirements gathered with feedback from the end-user and researcher. The identified functions and modules should operate accurately and meet the minimum requirements of the software. The same procedures were followed for additional features and functionality.

### ISO/IEC 25010 Software Quality Model

The ISO/IEC 25010 standard outlines a comprehensive framework for evaluating the quality of software through eight key characteristics, each with specific sub-characteristics (International Organization for Standardization and International Electrotechnical Commission, 2023).

- *Functional Suitability:* Evaluate the system's ability to meet user needs, focusing on correctness, completeness, and appropriateness.

ISSN 2278-2540 | DOI: 10.51583/IJLTEMAS | Volume XIV, Issue III, March 2025

- *Performance Efficiency:* Evaluates resource usage, including response time and scalability.
- *Compatibility:* Focuses on interoperability and coexistence with other systems.
- *Usability:* Measures the system's accessibility, learnability, and user satisfaction.
- *Reliability:* Ensures stability and fault tolerance under specified conditions.
- *Security:* Evaluate the system's ability to protect data through confidentiality, integrity, and authentication.
- *Maintainability:* Evaluates ease of modification, focusing on modularity and reusability.
- *Portability:* Examines the system's adaptability across different environments, including ease of installation and transferability.

The researcher designed a 31-item questionnaire with a 4-point Likert scale, with scores ranging from 1.00 to 1.49 ("Disagree") to 3.50 to 4.00 ("Strongly Agree") to evaluate the system's performance. The instrument underwent validation by experts, with content validity indices results ranging from 0.90 to 0.98. These results confirmed that the items were highly relevant and aligned with the objectives of the evaluation. The instrument was then pilot tested for reliability, and feedback was used to refine its clarity and consistency.

**Respondents of the Study**

In evaluating the performance of the system, the researchers included a group of respondents selected based on their relevant backgrounds. The respondents consisted of experts who evaluated the system using the eight quality characteristics outlined in ISO/IEC 25010, also known as the Software Product Quality Model. Over one to four weeks, these experts provided insights and recommendations through quantitative ratings and additional feedback.

Table 1 Respondents of the Study for System Performance Evaluation

| **Respondent** | **Frequency** | **Description** |
|---|---|---|
| School Administrator | 5 | Who are engaged in policy development. |
| IT Professional | 5 | Who has been involved in system development projects. |
| Computer Studies Faculty | 5 | Who has been involved in teaching computer programming subjects for 5 years and above. |
| Mental Health Professional | 5 | Who are registered psychologists. |
| **Total** | **20** | |

Table 1 presents the number of respondents involved in evaluating the system's performance. Purposive sampling techniques were employed to select participants. This non-probability sampling method leverages the researcher's expertise to identify and recruit individuals with relevant knowledge and expertise from the community, ensuring they contributed valuable insights to the academic investigation.

**III. Results and Discussion**

The findings are detailed in the following sections, covering the features of the developed web-based system, its performance evaluated against the eight characteristics of the Software Quality Model, and expert evaluation based on the criteria outlined in these standards.

**Key Features of the Developed Student Information System for Computer Studies**

The key features of the system were designed to enhance data management, facilitate student support, and improve decision-making for faculty and administrators. The following provides a detailed description of these features:

- *Student Profile Management:* Allows storage and retrieval of student details, including demographics, academic performance, learning preferences, and other relevant data.
- *Programming Anxiety Level Prediction:* Utilizes a machine learning model to predict a student's programming anxiety level based on predefined attributes.
- *Data Visualization:* Provides graphical representations of data, such as anxiety level distributions, academic performance trends, and other metrics, for easier interpretation and decision-making.
- *Automated Balanced Group Recommendation:* Automatically creates balanced student groups based on course, year level, section, and programming anxiety levels to promote collaborative learning.

ISSN 2278-2540 | DOI: 10.51583/IJLTEMAS | Volume XIV, Issue III, March 2025

The benchmarking analysis, presented in Table 2 and grounded in data extracted from Verywell Mind's medically reviewed compilation of top-rated anxiety management applications (Marschall, 2024), compared the developed system with nine leading platforms. While 5 out of 9 of these applications (approximately 55%) employ AI/ML, all are designed to manage general anxiety and lack features tailored to specific academic contexts. In contrast, the developed system uniquely targets programming anxiety through predictive analytics and includes an automated balanced grouping mechanism to foster peer support. These distinctions highlight the system's novel, student-centered approach and its potential to fill a gap in mental health support within computing education.

Table 2 Benchmarking Analysis results from Anxiety-Related platform

| Name | Platform | Key Features | AI/ML |
|---|---|---|---|
| SIS for Computer Studies | Web | Predictive analytics for programming anxiety, Automated Group Recommendations | Yes |
| Aura | Android, iOS, Web | Real-time feedback on breathing patterns and techniques | Yes |
| Breathwrk | Android, iOS, Web | Breathing exercises for relaxation and stress relief | Yes |
| Calm | Android, iOS, Web | AI voice technology assists users with relaxation exercises, breathing techniques, and sleep stories. | Yes |
| Headspace | Android, iOS, Web | Empathetic AI chatbot that encourages self-reflection. | Yes |
| InsightTimer | Android, iOS, Web | Guided meditation, mood tracking, sleep aids, live events | Yes |
| BetterHelp | Android, iOS, Web | Online therapy with licensed counselors | No |
| Medito | Android, iOS, Web | Free meditation with guided sessions for mindfulness | No |
| Sleep Reset | Web | Sleep improvement techniques, scientific-based methods | No |
| Real | iOS | Productivity scheduling with customizable routines | No |

**Integration of Prediction Data Model Based on Logistic Regression Algorithm**

The prediction data model was developed using the Cross Industry Standard for Machine Learning (CRISP-ML) framework. The model incorporated significant attributes identified through feature selection techniques, including Working Status, Course, Year Level, Previous General Weighted Average (GWA), Final Grade in Computer Programming 1 subject, Senior High Track, ICT Equipment, Learning Style, and Sleep Duration, ensuring optimal predictive performance. Among five classification algorithms evaluated, Logistic Regression achieved the highest accuracy of 98%. The model's performance was further evaluated using machine learning validation metrics with the following results: Precision of 98%, Recall of 98%, F-measure of 99%, and Cohen's Kappa of 0.96. These outcomes demonstrate that Logistic Regression is the most effective algorithm for predicting students' programming anxiety levels.

The trained model was serialized in a pickle file format for seamless integration into the web-based system, enabling real-time predictions and automated insights into programming anxiety levels.

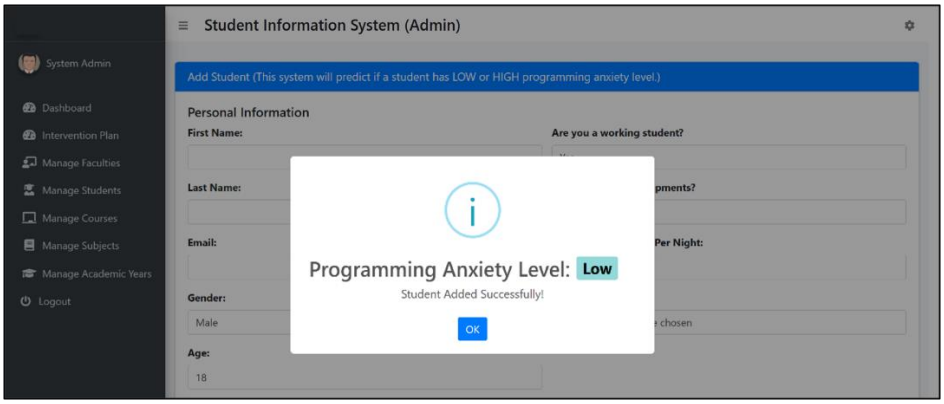


Fig. 2. System's "Low" Programming Anxiety Level Prediction

ISSN 2278-2540 | DOI: 10.51583/IJLTEMAS | Volume XIV, Issue III, March 2025

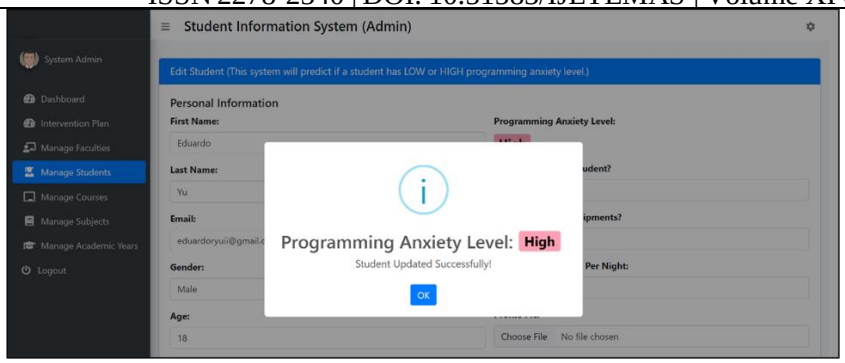


Fig. 3. System's "High" Programming Anxiety Level Prediction

Figures 2 and 3 illustrate the system's graphical user interface (GUI) following the integration of the Logistic Regression-based predictive model. The interface was designed with usability and accessibility in mind, allowing faculty and administrators to efficiently add or update student records through well-structured web forms. Upon submission, the system automatically generates a prediction of the student's programming anxiety level, categorized as High or Low. The interface features clearly labeled input fields, intuitive navigation, and a clean, responsive layout that supports user efficiency. Visual elements such as contrasting colors and readable typography enhance clarity and reduce cognitive load, particularly during data entry and review of prediction results. By including these detailed UI elements, the system offers a seamless and accessible user experience.

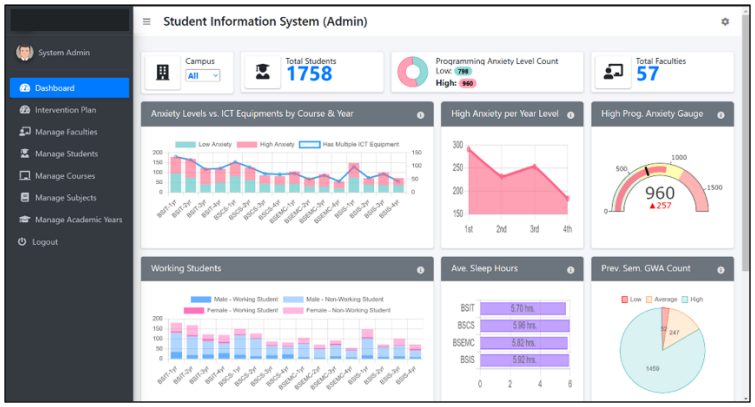


Fig. 4. System's Data Visualization

Figure 4 presents the system's interactive dashboard, developed with expert guidance to visualize student programming anxiety levels derived from the predictive model. The dashboard is structured to support the needs of faculty and administrators by offering intuitive, real-time access to aggregated data and individual-level insights. Its layout emphasizes clarity and usability, with organized sections for key metrics, color-coded indicators for anxiety levels, and interactive charts that aid in identifying trends and risk patterns across student cohorts. These visualizations facilitate timely, insight-based decision-making for implementing targeted interventions and support strategies. The dashboard's responsive design, combined with accessible visual elements, enhances user experience and encourages consistent use in academic monitoring.

**Automated Balanced Grouping Based on Predicted Programming Anxiety Levels**

One of the key features of the system is its ability to automatically generate balanced student groups based on prediction results. This function supports the existing intervention plan by ensuring that groups include a mix of programming anxiety levels, fostering improved support and collaboration among students.

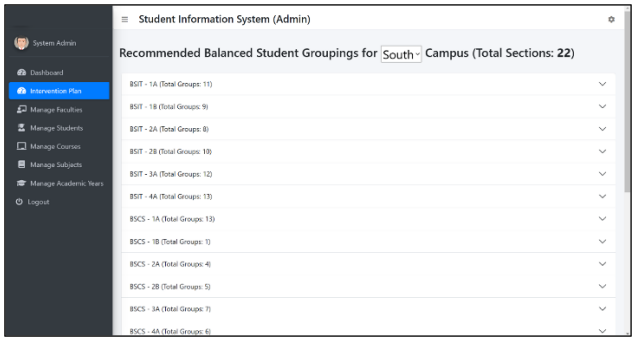


Fig. 5. System's Automated Balanced Groups

ISSN 2278-2540 | DOI: 10.51583/IJLTEMAS | Volume XIV, Issue III, March 2025

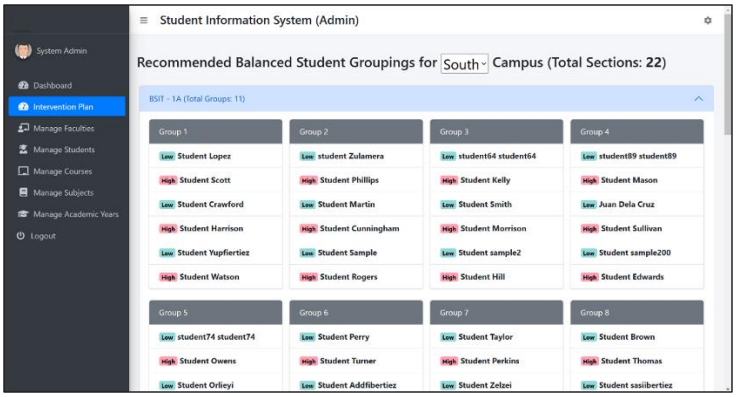


Fig. 6. System's Automated Balanced Groups per Course, Year, and Section

Figures 5 and 6 display the system's automated balanced grouping feature, which organizes students based on their predicted programming anxiety levels. The system ensures that each group contains a mix of students with both high and low anxiety levels, promoting collaboration and mutual support. Figure 5 uses a collapsible UI, allowing users to expand or collapse group details for a more streamlined viewing experience. Figure 6 provides a more detailed view of groupings organized by course, year, and section. This feature enhances the diversity and effectiveness of student groups, fostering a supportive learning environment.

The automated process for creating balanced student groups employs a heterogeneous grouping algorithm, which is based on predicted programming anxiety levels and incorporates factors such as course, year level, and section. This approach ensures that the resulting groups are not only balanced in terms of anxiety levels but also comply with established size requirements. The following steps outline how the algorithm recommends balanced groupings:

1) *Target Group Size:* Each group is designed to have 4 to 5 students, with the possibility of having up to 6 students if necessary. This ensures flexibility while keeping group sizes manageable.
2) *Desired Mix of Anxiety Levels:* This step ensures that each group contains both low-anxiety and high-anxiety students. For every high-anxiety student, there should be at least one low-anxiety student. This balance helps prevent groups from becoming too heavily weighted towards one anxiety level.
3) *Calculating the Number of Groups:* The total number of groups is calculated by taking the total number of students (both low and high anxiety) and dividing it by 5 (since 5 is the target group size). The result is rounded up, meaning there will always be enough groups to accommodate all students.
4) *Initial Distribution of Students:* Students are placed into groups by alternating between low-anxiety and high-anxiety students. This step helps maintain the balance of anxiety levels across groups.
5) *Handling Remaining Students:* If there are any groups with fewer than 4 students, these smaller groups are merged with others. If a group ends up with more than 6 students, the extra students are moved to groups that have fewer than 5 students.

**ISO/IEC 25010 or Software Product Quality Model**

Evaluating the system was a crucial phase in the software development cycle. This process provided valuable insights into how users perceived the system in relation to predefined criteria, ensuring it aligned with user expectations and requirements. The evaluation helped identify areas for improvement, enhancing the system's overall quality, usability, and effectiveness.

Table 4 System Performance Evaluation Summary

| **Indicators** | **Weighted Mean** | **Rank** | **Verbal Interpretation** |
|---|---|---|---|
| Functional Suitability | 3.82 | 2 | Strongly Agree |
| Performance Efficiency | 3.65 | 6 | Strongly Agree |
| Compatibility | 3.60 | 7 | Strongly Agree |
| Usability | 3.78 | 3 | Strongly Agree |
| Reliability | 3.73 | 4 | Strongly Agree |
| Security | 3.85 | 1 | Strongly Agree |
| Maintainability | 3.78 | 3 | Strongly Agree |
| Portability | 3.68 | 5 | Strongly Agree |
| **Overall Weighted Mean** | **3.74** | | **Strongly Agree** |

ISSN 2278-2540 | DOI: 10.51583/IJLTEMAS | Volume XIV, Issue III, March 2025

As gleaned from Table 4, The system evaluation results reveal strong approval across all criteria, with an overall weighted mean of 3.74, interpreted as "Strongly Agree". The highest rating was for security, with a mean of 3.85, emphasizing its importance in safeguarding sensitive student data and preventing unauthorized access. The consistently high ratings across the remaining characteristics reflect the system's robust performance, efficiency, usability, and reliability. These results indicate that the system meets expert expectations in terms of functionality and overall quality, positioning it as a reliable and secure tool for academic use. anxiety.

In determining the existence of significant difference between the result of evaluation using the characteristics of ISO/IEC 25010 or the Software Product Quality Model when grouped according to respondents' category, the one-way analysis of variance or ANOVA was used. This robust statistical tool allows the researchers to simultaneously examine arithmetical means across multiple group.

Table 5 Analysis of Variance Between the Evaluation of Experts

| **Source of Variance** | **Sum of Squares** | **Degree of Freedom** | **Mean Square** | **F-Stat** | **P-Value** |
|---|---|---|---|---|---|
| Between Groups | 0.014 | 3 | 0.005 | 2.680 | 0.942 |
| Within Groups | 4.271 | 120 | 0.036 | | |
| **Total** | 4.285 | 123 | | | |

Table 5 shows that the computed significant value (P-Value), which is 0.942, is higher than the alpha level of 0.05. This indicates that no significant difference was recorded in terms of the means of the group of respondents. This concludes that their perception in terms of system evaluation is the same across all the respective criteria.

## IV. Conclusion

This study concluded that the integration of a pre-developed predictive model, based on the Logistic Regression algorithm with a demonstrated accuracy of 98%, can be effectively implemented in real-world applications to predict students' programming anxiety levels. The system's automated balanced grouping feature successfully addressed the limitations of manual group formation by ensuring a heterogeneous mix of students with varying anxiety levels, thereby promoting collaboration and peer support. Benchmarking analysis against existing top-rated anxiety management applications highlighted the system's distinctiveness in targeting programming-specific anxiety, a gap unaddressed by general mental health platforms. Furthermore, system evaluation based on the ISO/IEC 25010 Software Quality Model confirmed its compliance with key software quality attributes. The results of the analysis of variance (ANOVA) indicated no statistically significant differences in expert evaluations, suggesting a strong level of agreement regarding the system's overall effectiveness. Collectively, these findings underscore the system's potential to contribute meaningfully to educational practice and mental health support by informing data-driven interventions aimed at mitigating programming anxiety and enhancing student well-being in computing education.

### Recommendation

Based on the system's evaluation using the ISO/IEC 25010 Software Quality Model, which demonstrated high levels of interoperability and functional suitability, the developed system is deemed ready for integration into educational environments. While the system exhibits strong potential for scalability as indicated by its modular architecture and robust performance, future research is recommended to validate its deployment across various academic institutions and explore integration with existing Learning Management Systems (LMS). These initiatives would support its seamless adoption by faculty and administrators, facilitate data-driven interventions to address programming anxiety, and promote its sustained use in computing education.

### References

1. Department of Health (Philippines). (2021). DOH townhall session | May 19, 2021. YouTube. Retrieved January 24, 2025, from https://www.youtube.com/watch?v=DD-OEzkjPJY.
2. Ibrahim, A. (2016). Definition, purpose, and procedure of developmental research: An analytical review. Asian Research Journal of Arts & Social Sciences, 1(6), 1–6. Retrieved January 24, 2025, from https://doi.org/10.9734/arjass/2016/30478.
3. International Organization for Standardization and International Electrotechnical Commission. (2023). Systems and software quality requirements and evaluation (SQuaRE) - Product quality model (ISO/IEC 25010:2023). Retrieved January 24, 2025, from https://www.iso.org/obp/ui/en/#iso:std:iso-iec:25010:ed-2:v1:en.
4. Khor, E. T., & Darshan, D. (2024). Prediction of students' performance in online learning using supervised machine learning. International Journal of Information and Learning Technology, 41(2), 166–179. Retrieved January 24, 2025, from https://doi.org/10.1108/ijilt-08.
5. Mahalakshmi, G., & Sujatha, G. (2023). Predictions of college students' mental stress using machine learning algorithms. 2023 7th International Conference on Intelligent Computing and Control Systems (ICICCS), Madurai, India, 288–295. Retrieved March 28, 2025, from https://doi.org/10.1109/ICICCS56967.2023.10142672.
6. Marschall, A., Hardy, N., & Owens, H. (2024, May 10). Tried, Tested, and Expert-Approved: The 9 Best Apps for Anxiety. Verywell Mind. https://www.verywellmind.com/best-apps-for-anxiety-3575736

ISSN 2278-2540 | DOI: 10.51583/IJLTEMAS | Volume XIV, Issue III, March 2025

7. Mendoza, N. B., King, R. B., & Haw, J. Y. (2023). The mental health and well-being of students and teachers during the COVID-19 pandemic: Combining classical statistics and machine learning approaches. Educational Psychology, 43(5), 430–451. Retrieved January 24, 2025, from https://doi.org/10.1080/01443410.2023.2226846.
8. Mutalib, S. (2021). Mental health prediction models using machine learning in higher education institutions. Turkish Journal of Computer and Mathematics Education, 12(5), 1782–1792. Retrieved January 24, 2025, from https://doi.org/10.17762/turcomat.v12i5.2181.
9. Tomić, B., Stojanović, T., & Stojanović, T. (2022). Examining students' test anxiety and pre-university programming education in an undergraduate introductory programming course. EDULEARN Proceedings. Retrieved January 24, 2025, from https://doi.org/10.21125/edulearn.2022.0938.
10. World Health Organization. (2024). Anxiety disorders. WHO. Retrieved March 2, 2024, from https://www.who.int/news-room/fact-sheets/detail/anxiety-disorders.
11. Yas, Q., Alazzawi, A., & Rahmatullah, B. (2023). A comprehensive review of software development life cycle methodologies: Pros, cons, and future directions. ResearchHub. Retrieved January 24, 2025, from https://www.researchhub.com/paper/4153511/a-comprehensive-review-of-software-development-life-cycle-methodologies-pros-cons-and-future-directions.
12. Yildirim, O. G., & Özdener, N. (2022). Development and validation of the Programming Anxiety Scale. International Journal of Computer Science Education in Schools, 5(3), 17–34. Retrieved January 24, 2025, from https://doi.org/10.21585/ijcses.v5i3.140.
13. Zhai, Y., Zhang, Y., Chu, Z., Geng, B., Almaawali, M., Fulmer, R., Lin, Y. D., Xu, Z., Daniels, A. D., Liu, Y., Chen, Q., & Du, X. (2024). Machine learning predictive models to guide prevention and intervention allocation for anxiety and depressive disorders among college students. Journal of Counseling & Development, 103(1), 110–125. https://doi.org/10.1002/jcad.12543.