Enhancing Student Engagement and Success in Operating Systems through Simulation based Learning and Formative Assessments

[1]Parkavi R, [2]Sujitha S, [3]Sheik Abdullah A
[1,2] Department of Information Technology, Thiagarajar College of Engineering, Madurai, Tamil Nadu, India
[3]School of Computer Science and Engineering, Vellore Institute of Technology, Chennai
[1] rpit@tce.edu [2] ssiit@tce.edu [3] sheikabdullah.a@vit.ac.in

***Abstract*— Operating Systems is often perceived as a challenging course for many undergraduate IT students because the concepts are abstract, algorithm-heavy, and difficult to visualize. This makes it harder for learners to build strong conceptual connections using traditional lecture-based teaching alone. The gap becomes more visible when students struggle to apply theoretical ideas to real-world scenarios, especially in topics such as CPU scheduling, memory management, and disk operations. To address these limitations, this study implemented a blended instructional approach combining simulation-based learning, module-wise formative quizzes, and problem-driven tutorials for 60 second-semester B.Tech IT students during the 2024–25 academic year. Students designed and experimented with simulators for scheduling, paging, and disk algorithms, while continuous assessments encouraged steady learning throughout the semester. The intervention led to notable improvements in academic performance, including a complete 100% pass rate, higher average marks compared with the previous cohort, and a marked reduction in the number of low-performing students. Student feedback further indicated that simulations made complex algorithms easier to understand, and the regular quizzes helped them stay engaged with the course material. Overall, these findings show that the blended approach not only enhanced understanding but also fostered deeper participation and confidence in learning OS concepts. This study highlights the potential of integrating simulations and formative assessments to strengthen student learning outcomes in algorithm-intensive courses and offers a practical model that can be scaled to similar engineering subjects.**

***Keywords*—Simulation-Based Learning; Formative Assessment; Operating Systems Education; Active Learning Strategies; Student Engagement; Conceptual Understanding;**

***ICTIEE* Track—Assessment, Feedback, and Learning Outcomes**

ICTIEE Sub—Track: Enhancing Student Performance through Formative Feedback

I. INTRODUCTION

Operating Systems (OS) is a foundational course in the B.Tech Information Technology curriculum. It covers essential concepts such as process scheduling, memory management, deadlocks, and file systems, which form the backbone of system-level computing. Despite its importance, many students perceive the subject as abstract and difficult due to its algorithmic complexity and lack of tangible visualization. Traditional lecture-based instruction often fails to bridge the gap between theoretical knowledge and practical application.

Challenges in Teaching Operating Systems

Operating Systems is taught across several computing disciplines, including Computer Science, Software Engineering, and Big Data Analytics. However, students often find the subject challenging and disengaging. A closer examination of existing instructional practices reveals several barriers to effective learning:

Gap Between Theory and Practice

Although OS is inherently practical, instruction frequently emphasizes theory with minimal integration of hands-on activities. Fragmented examples rarely translate into real-world understanding, limiting knowledge retention and problem-solving skills.

Missed Opportunities for Value-Based Learning

The course naturally allows for discussions on societal and ethical dimensions, such as software licensing, fair resource allocation, and responsible multi-user system design. However, such opportunities are often overlooked, restricting students' exposure to broader perspectives.

Outdated Teaching Approaches

Traditional lecture-based methods dominate OS teaching, offering limited opportunities for curiosity-driven exploration or innovation. This restricts the development of critical and

R.Parkavi
Assistant Professor, Department of Information Technology, Thiagarajar College of Engineering, Madurai.rpit@tce.edu

creative thinking, which are vital for modern computing careers.

Limited and Ineffective Assessment Methods

Assessment in OS courses typically relies on assignments and a final exam. While useful for evaluating basic knowledge, this approach may not adequately capture students' applied problem-solving abilities, sometimes resulting in strong grades but limited competence.

To address these challenges, a pedagogical intervention was designed that combined simulation-based learning, problem-driven tutorials, and formative assessments delivered through online quizzes. Students were tasked with developing simulators for CPU scheduling, disk scheduling, and page replacement policies—three of the most algorithm-intensive areas of the course. This hands-on approach, supported by structured quizzes and problem-solving sessions, aimed to foster conceptual clarity, sustained engagement, and improved learning outcomes.

Over the past few academic years, internal course reviews and student feedback consistently showed that conventional lecture-driven teaching left a substantial portion of learners struggling with core OS topics. For instance, in earlier batches, more than 40% of students reported difficulty in visualizing how scheduling algorithms behave under different workloads, even after repeated classroom explanations. Students also tended to memorize steps for algorithms like LRU or Optimal Page Replacement without understanding why certain decisions were taken by the OS. These gaps were reflected in assessments as well, where lower-performing students made similar conceptual errors such as misinterpreting Gantt charts or confusing memory management terminologies. These recurring patterns highlighted the need for a more interactive and experience-based learning approach, providing the motivation for the present study.

This paper presents a case study of the instructional model applied to 60 second-semester B.Tech IT students during the 2024–2025 academic year. The results indicate that integrating simulation-based learning with formative assessments can significantly enhance student comprehension and success in complex engineering subjects such as Operating Systems.

II. Related Works

Teaching Operating Systems has long been recognized as a demanding task in engineering education because it requires learners to understand abstract concepts, analyze algorithms, and apply system-level thinking. Earlier work has emphasized the importance of aligning OS education with engineering accreditation standards, especially in strengthening analytical and problem-solving skills. For instance, Zhang & Zhang (2021) examined OS curriculum requirements and proposed mechanisms to better align course outcomes with accreditation expectations, highlighting the need for continuous improvement in pedagogy.

Several researchers have focused on redesigning the OS learning environment to address challenges such as steep learning curves and limited opportunities for hands-on experimentation. A notable contribution is the scalable OS lab platform developed by Wang et al. (2020), which supports large classes and enables automated behavior tracking. Their findings from 2015 to 2018 demonstrate that timely feedback and structured laboratory practice significantly improve student learning.

The need for scalable, well-designed infrastructure is even more apparent in large university settings. Zemanek & Muzikar (2023) describe the complexities of managing OS courses with nearly 1,000 students and emphasize robust automation, tailored exercises, and structured assessment approaches. Their work shows that without the right instructional design and infrastructure, maintaining quality becomes difficult at scale.

In addition to infrastructure-based solutions, blended and hybrid pedagogies have been explored to improve conceptual understanding. Chen (2022) introduced a hybrid online–offline teaching approach that helped students deepen engagement through staged learning activities and continuous reflection. Similarly, Hakiki et al. (2023) combined simulation-based tasks, targeted tutorials, and formative assessments to bridge the gap between theory and practice, demonstrating that a blended approach can make algorithm-heavy topics more approachable.

Mobile and web-based learning tools continue to gain traction in OS pedagogy. Using the 4D design framework, Ebling (2024) developed a mobile learning platform and reported high practicality scores from both teachers and students, reinforcing the value of accessible, self-paced learning materials.

Simulation tools specifically designed for OS topics are also emerging. Wadmare et al. (2024) presented an online tool that enables learners to visualize scheduling algorithms, memory processes, and resource allocation strategies. Although effective, the authors note areas for enhancement, such as integrating more realistic evaluation metrics and refining complex simulations like the Banker's Algorithm.

Recent advancements in generative AI have further influenced OS education research. Zhou et al. (2025) demonstrated that integrating generative AI within the BOPPPS teaching model supports active learning, improves conceptual clarity, and encourages student participation at deeper levels. Their study suggests that AI-augmented teaching strategies can be successfully adapted to engineering domains.

Virtual models of operating systems are another evolving theme. Gaziz et al. (2025) compared OS instruction delivered through virtual models versus traditional physical systems and found that virtual environments significantly improve accessibility, student motivation, and independent exploration. Their findings highlight the relevance of virtual simulations for teacher-training programs and universities working toward scalable solutions.

Learners often struggle with programming courses, but this research (Parkavi et al., 2024) aims to enhance their understanding through methods like flipped classrooms, online quizzes, and virtual labs. It integrates fuzzy logic, factor analysis, and machine learning to identify key factors affecting active learning strategies in IT courses. Using real-time data, the study compares the performance of various machine learning models, with Naive Bayes and K-Nearest Neighbors achieving the highest accuracy.

Personalized learning in Educational Technology is crucial but challenging, particularly when predicting students' abilities based on their learning styles and ICT usage. This study (Parkavi et al., 2024) proposes a decision support system combining Machine Learning, swarm intelligence, and explainable AI to assess performance, using Chaotic Particle Swarm Optimization (C-PSO), which outperforms other algorithms. The results demonstrate C-PSO's effectiveness in educational data analysis, emphasizing ICT's role in self-progress and suggesting future studies on real-world applications and comparative optimizations.

Evaluating student performance manually is time-consuming for educators, but using a Learning Management System (LMS) helps monitor and analyze students' behavior efficiently. This study (Parkavi et al., 2025) applies Exploratory Data Analysis (EDA) and Machine Learning techniques like KNN and Multiple Regression to predict and visualize students' cognitive and psychomotor skills. The model achieves an impressive 99% accuracy with Multiple Regression, offering valuable insights for educational institutions.

Predicting student performance is vital for higher education institutions, as academic records often determine admission to quality institutions. This paper (Parkavi & Karthikeyan, 2025) aims to predict students' future academic outcomes by analyzing their personal and academic data using classification techniques. The study applies methods like Multiple Linear Regression, Naive Bayes, and Decision Tree, achieving 88.44% accuracy, helping identify key factors for improvement and enabling better support for students to enhance their success rates.

Collaborative Learning, where knowledge is constructed through peer interaction, is essential in modern education, especially with the rise of smart devices. This study (Parkavi et al., 2022) examines the impact of Collaborative Learning in an elective Information Technology course, involving 176 undergraduate students. Through individual, group, formative, and summative assessments, the study finds that incorporating Collaborative Learning significantly improves course outcomes, surpassing the performance of previous student cohorts.

Despite these valuable contributions, existing research largely examines individual components such as simulation tools, hybrid teaching, mobile learning, or scalable labs in isolation. There remains limited work that integrates simulations, problem-based tutorials, and structured formative assessments into a single instructional model for Operating Systems. Moreover, few studies systematically evaluate both performance outcomes and student perceptions within large classroom contexts.

This paper addresses that gap by presenting a unified, student-centered instructional approach and examining its impact on engagement, conceptual understanding, and academic success.

III. Research Methodology

A. Motivation and Contribution

OS remains one of the most challenging subjects in computer science education due to its abstract concepts, algorithmic complexity, and lack of immediate visual feedback. Despite being a foundational course, many students struggle to connect theoretical knowledge with practical application, leading to disengagement and poor academic outcomes. Traditional lecture-based methods often fall short in fostering critical thinking, problem-solving abilities, and the ability to apply concepts in real-world contexts. This gap in pedagogy becomes even more pronounced when teaching large cohorts, where individual attention and personalized learning opportunities are limited.

Motivated by these challenges, this study explores an integrated instructional approach that combines simulation-based learning, problem-based tutorials, and module-wise formative assessments. Unlike previous studies that examine these strategies in isolation, this work evaluates their combined impact on student engagement, understanding, and performance in an OS course. The central contribution of this research lies in demonstrating how hands-on simulation projects help demystify complex algorithms, how formative assessments promote continuous learning, and how problem-based sessions reinforce critical thinking. By aligning with outcome-based education (OBE) principles and engineering accreditation goals, this model not only improves student success rates but also provides a scalable, adaptable framework for teaching OS in both traditional and blended learning environments.

While several studies have explored simulation tools, blended teaching, or formative assessments individually, very few have examined how these strategies can work together as a continuous learning ecosystem. The uniqueness of this work lies in integrating simulator development, problem-based tutorials, and module-wise formative assessments into a single structured pedagogy, and evaluating their combined influence on engagement and academic success. The study also contributes by showing how this model can be scaled for large cohorts without compromising the depth of learning an area where existing literature offers limited empirical evidence. Proposed methodology framework is given in Figure 1.

This work was undertaken to address persistent learning gaps observed in OS courses particularly difficulty in visualizing algorithms, inconsistent engagement, and performance gaps among students. To guide the investigation, the following research questions were framed:

B. Research Questions

RQ1: To what extent does simulation-based learning improve students' ability to accurately interpret and apply core OS algorithms such as CPU scheduling and page replacement?

Measure: Improvement in quiz scores, assignment accuracy, and reduction in common conceptual errors.

RQ2: How do module-wise formative assessments influence student engagement and continuous learning in an OS course?

Measure: Quiz participation rate, score progression across modules, and consistency in CAT performance.

RQ3: Does the blended instructional model (simulation + tutorials + formative assessments) lead to measurable improvements in overall academic performance compared to the previous year's traditional approach?

Measure: Differences in pass percentage, average marks, lowest marks, and distribution of score ranges.

C. Approach

This study follows a mixed-methods case study design, incorporating both qualitative and quantitative data to evaluate the effectiveness of a blended instructional model in teaching Operating Systems.

1) Participants

The participants were 60 second-semester B.Tech IT students, most of whom had prior exposure to basic programming. However, only a small proportion had hands-on experience with low-level system concepts such as process scheduling or memory management. Their performance in earlier core subjects reflected a wide range of academic backgrounds, making them a representative group for assessing the generalizability of the proposed approach.

2) Teaching Interventions

Simulation Design and Evaluation Criteria: Each simulator was designed to mimic the behaviour of real OS algorithms by requiring students to accept user inputs, generate visual timelines, and compute performance metrics such as waiting time and turnaround time. Students were evaluated based on correctness of logic, clarity of documentation, interface usability, and their ability to explain algorithmic behaviour during review sessions.

Assessment Criteria: Quizzes were designed with a mix of conceptual MCQs and short analytical problems to check immediate understanding. Tutorials were graded for completeness and accuracy, with emphasis on students' step-by-step reasoning. This structured approach ensured consistency across the cohort and supported reproducibility of the intervention.

3) Data Collection

Data was collected through multiple channels:

1. Simulator project submissions (evaluated on functionality and documentation)
2. Quiz scores for each module
3. Tutorial participation and completion records
4. Final examination results
5. Optional student feedback via reflections or anonymous surveys

4) Analysis

Descriptive statistics were used because the primary aim was to compare performance trends between two academic years and observe shifts in score distributions across assessments. Given the case-study nature of the research and the moderate sample size, descriptive analysis offered a practical way to capture both improvements and variations. Qualitative feedback was intentionally integrated to complement numerical results, as student reflections provided context to explain why certain changes occurred. A limitation of this approach is that it does not include inferential statistical tests; hence, future work could employ methods such as paired t-tests or regression analysis to provide deeper validation of the observed improvements.

IV. Results And Discussion

Beyond measurable improvements, this study highlights how simulation-based learning can strengthen the broader teaching community by offering a practical, scalable model for difficult engineering subjects. Students reported that building simulators helped them 'see' abstract algorithms in action, which reduced cognitive load and increased confidence. The integration of formative assessments fostered consistent learning habits instead of last-minute preparation. These insights demonstrate that algorithm-heavy courses benefit from visual, hands-on pedagogies, underscoring the value of redesigning traditional classrooms to promote deeper, experiential learning.

The redesigned instructional model combining simulation projects, problem-based tutorials, and formative assessments was implemented with 60 second-semester B.Tech IT students. The results across exams, assignments, and feedback provide compelling evidence of its impact. A comparison of outcomes between the 2023–24 (traditional approach) and 2024–25 (simulation + formative assessment) cohorts shows clear improvement (Table 1). The pass percentage increased from 97% to 100%, while the average marks rose by 11 points. Importantly, the lowest score also improved from 30 to 50, showing that weaker students benefited significantly from the interventions. These results indicate not only stronger performance at the top but also a narrowing of the performance gap, suggesting more equitable learning outcomes.

A. Continuous Assessment Tests

The analysis of CAT 1 and CAT 2 results across both years highlights striking differences. It is shown in Figure 2 and 3.

CAT 1: In 2023–24, most students scored in the 46–55 range, with very few above 55. In contrast, in 2024–25, a substantial shift occurred: many students moved into the >55 category, and the proportion of students in the lower ranges (25–35 and <25) almost disappeared.

CAT 2: A similar trend was observed. The 2024–25 cohort showed a sharp rise in the >55 group compared to the previous year, while the lowest-scoring categories nearly vanished.

TABLE I
STUDENT PERFORMANCE COMPARISON

Academic Year	Pass Percentage	Average Marks	Highest Marks	Lowest Marks
2023–2024 (Traditional)	97%	61	87	30
2024–2025 (Simulation + Formative Assessment)	100%	72	91	50

This demonstrates that students not only achieved higher scores but also showed more consistency after the introduction of simulations and formative assessments. The interventions seem to have encouraged continuous engagement rather than last-minute preparation, resulting in deeper understanding and sustained performance.

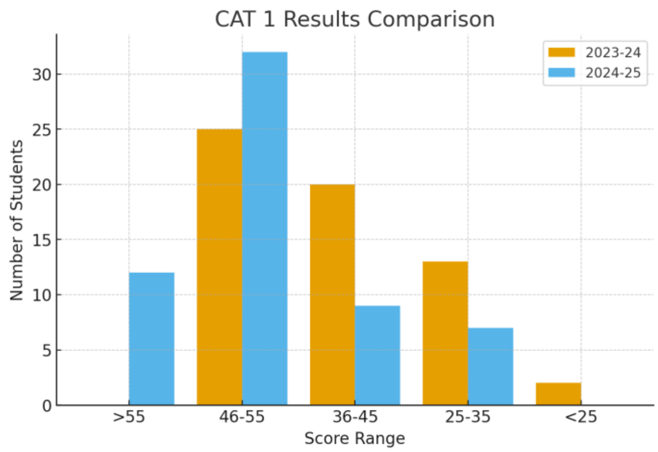

Fig.2. Comparative Analysis of CAT1

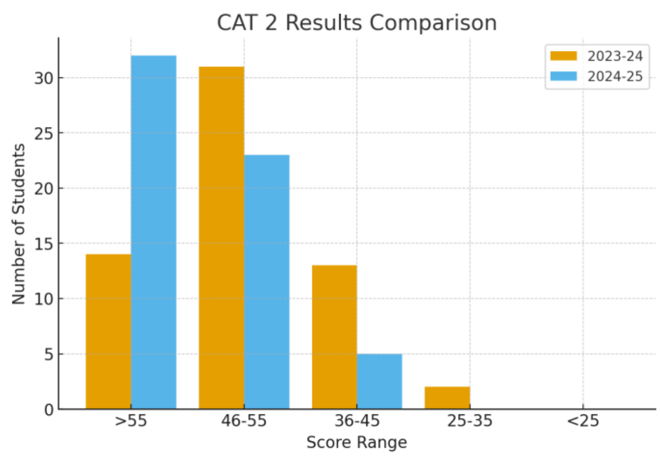

Fig.3. Comparative Analysis of CAT 2

B. Assignment Performance

Assignments were another area where performance trends were examined.

Assignment 1 In 2023–24, 53 students scored in the higher range (31–40) and 7 fell into the lower range (20–30). In 2024–25, only 2 students remained in the lower range, while 58 scored in the top bracket. This shows steady improvement and fewer struggling learners. Assignment 1 performance is given in Figure 4.

Assignment 2 In 2023–24, all 60 students scored in the higher range. However, in 2024–25, a small dip occurred, with 4 students slipping into the 20–30 range. Despite this, the majority (56 students) still performed strongly in the top range. Assignment 2 performance is given in Figure 5.

This indicates that while overall assignment performance was consistently high, the second assignment in 2024–25 might have posed greater difficulty or coincided with increased student workload. Nonetheless, the overall trajectory remains positive.

C. Student Feedback and Engagement

Students consistently reported that simulation projects helped "bring algorithms to life," while quizzes encouraged steady revision and tutorials provided confidence in traditionally

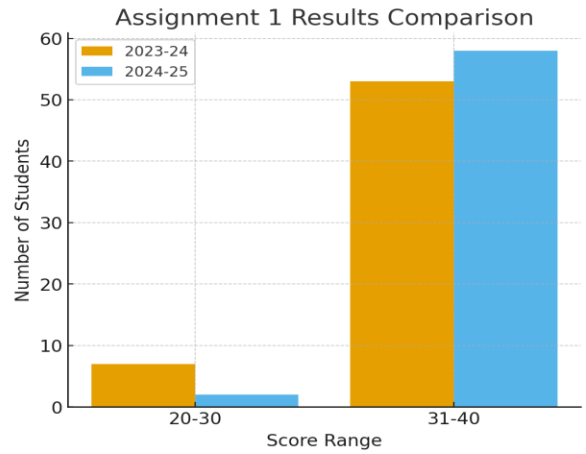

Fig.4. Comparative Analysis of Assignment 1

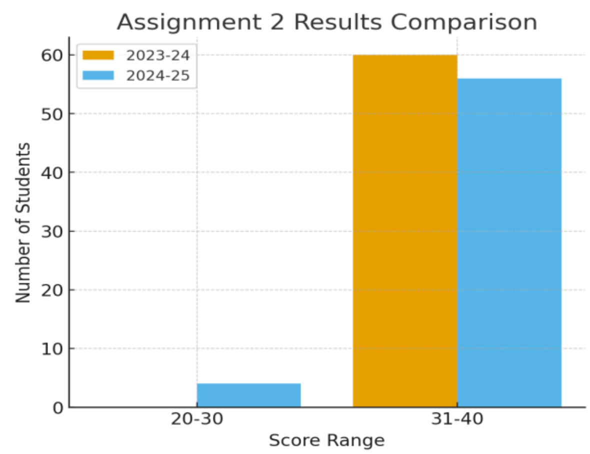

Fig.5. Comparative Analysis of Assignment 2

TABLE II
STUDENT FEEDBACK

Feedback Category	% of Students Mentioning	Sample Student Comment
Simulator enhanced understanding	85%	"I could see how Round Robin scheduling actually works."
Quizzes motivated consistent study	75%	"The quizzes made me revise after every module."
Tutorials improved problem-solving	88%	"Memory management was difficult before tutorials."

It is inferred from the table 2

85% of students mentioned that the simulator helped them understand how concepts like Round Robin scheduling work, making things clearer.

75% of students felt that quizzes encouraged them to study regularly, as they prompted them to review the material after each module.
88% of students found that the tutorials improved their problem-solving skills, especially in challenging topics like memory management.

These results underscore the effectiveness of combining simulation-based learning, problem-solving tutorials, and formative assessments in teaching a conceptually heavy subject like Operating Systems. The approach not only enhances comprehension but also promotes equity, engagement, and long-term learning.

Although the performance increase is clear, the current analysis is descriptive in nature. A more detailed statistical comparison such as using a paired t-test, effect-size calculation, or regression modeling would provide stronger evidence of the intervention's impact. The sample size is limited to a single program and academic year, which may affect generalizability. Despite these limitations, the consistent upward shift across quizzes, assignments, and exams indicates that the instructional approach had a meaningful positive influence. Future studies involving multiple cohorts could validate the findings more rigorously.

Conclusion And Future Work

This study emphasizes the effectiveness of continuous assessment as a tool to enhance student learning outcomes. The comparative analysis of CAT results across two academic years shows a noticeable improvement in performance, particularly with a larger number of students moving into higher score brackets in 2024–25. This positive shift indicates that structured assessments, timely feedback, and guided interventions contribute to strengthening conceptual understanding and improving academic confidence.

The results also demonstrate that assessments should not be viewed as mere grading exercises but as opportunities for learning and reflection. The proposed research methodology, which focuses on problem identification, targeted interventions, and reflective evaluation, offers a systematic framework for bridging performance gaps. By combining continuous evaluation with active learning strategies, educators can ensure that students are not only assessed fairly but also supported in their academic progression.

The insights from this study open avenues for further research and application. Future studies could examine the long-term impact of this blended pedagogy by tracking students across advanced courses such as Distributed Systems or Cloud Computing. Another area for exploration is how simulator-based learning influences higher-order problem-solving skills, especially in topics like synchronization and deadlock handling. Scaling the model to interdisciplinary courses or integrating analytics tools to personalize feedback are also promising directions for expanding this work.

References

Chen, Q. (2022). Exploration of Method to Improve the Teaching Effect of Course "Operating System." 2022 IEEE 2nd International Conference on Educational Technology (ICET), 71–75. https://doi.org/10.1109/icet55642.2022.9944467

Ebling, M. R. (2024). Resources for Teaching Operating Systems: A Survey of Instructors and a Literature Review. *ACM Transactions on Computing Education,*24(4),1–28. https://doi.org/10.1145/3688853

Gaziz, G., Bidaibekov, E., Grinshkun, V., & Koneva, S. (2025). Virtual models' effectiveness in teaching operating systems to future computer science teachers. *E-Learning and Digital Media.* https://doi.org/10.1177/20427530251343206

Hakiki, M., Surjono, H. D., Wagiran, Fadli, R., Budiman, R. D. A., Ramadhani, W., … Hidayah, Y. (2023). Enhancing Practicality of Web-Based Mobile Learning in Operating System Course: A Developmental Study. *International Journal of Interactive Mobile Technologies (iJIM)*, 17(19), pp. 4–19. https://doi.org/10.3991/ijim.v17i19.42389

Parkavi, R., Karthikeyan, P., Sujitha, S., & Abdullah, A. S. (2025). Enhancing Educational Assessment: Predicting and Visualizing Student Performance using EDA and Machine Learning Techniques. *Journal of Engineering Education Transformations,* 37, 240–245. Retrieved from https://journaleet.in/index.php/jeet/article/view/2336

Parkavi, R., & Karthikeyan, P. (2025). Enhancing Student Performance Prediction in Higher Education: A Data Driven Approach. *Journal of Engineering Education Transformations,* 37, 271–279. Retrieved from https://journaleet.in/index.php/jeet/article/view/2343

Parkavi, R., Karthikeyan, P., & Abdullah, A. S. (2022). Collaborative Learning : A Case Study on Information Security and Auditing Management Course. *2022 International Conference for Advancement in Technology (ICONAT)*, 1–7. https://doi.org/10.1109/iconat53423.2022.9726091

Parkavi, R., Karthikeyan, P., & Sheik Abdullah, A. (2024). Enhancing personalized learning with explainable AI: A chaotic particle swarm optimization based decision support system. *Applied Soft Computing*, 156, 111451. https://doi.org/10.1016/j.asoc.2024.111451

R., P., P., K., & A., S. A. (2024). Predicting academic performance of learners with the three domains of learning data using neuro-fuzzy model and machine learning algorithms. *Journal of Engineering Research*, 12(3), 397–411. https://doi.org/10.1016/j.jer.2023.09.006

Wadmare, J., Kolte, D., Bhatia, K., Desai, P., & Wadmare, G. (2024). Virtual Simulations Tool for Operating Systems: Advancing E-Learning in Computing Education. *Journal of Information Technology Education: Innovations in Practice*, 23, 018. https://doi.org/10.28945/5404

Wang, L., Zhen, Z., Wo, T., Jiang, B., Sun, H., & Long, X. (2020). A Scalable Operating System Experiment Platform Supporting Learning Behavior Analysis. *IEEE Transactions on Education*, 63(3), 232–239. https://doi.org/10.1109/te.2020.2975556

Zemanek, P., & Muzikar, Z. (2023). Methodology and Infrastructure for Teaching Operating Systems Courses for Large Number of University Students. *2023 18th Iberian Conference on Information Systems and Technologies (CISTI)*, 1–4. https://doi.org/10.23919/cisti58278.2023.10212062

Zhang, F., & Zhang, X. (2021). Achievement Degree Evaluation Method of Operating System Course Targets Under the Background of Engineering Education Accreditation. *2021 IEEE 3rd International Conference on Computer Science and Educational Informatization (CSEI)*, 5–9. https://doi.org/10.1109/csei51395.2021.9477752

Zhou, R., Liu, Y., Sun, L., & Zhu, S. (2025). Enhancing Operating System Education With a Generative AI-Supported Boppps Model: An Empirical Study. *Computer Applications in Engineering Education,* 33(6). Portico. https://doi.org/10.1002/cae.70114