Smart Learning App For Coding

Anikait Kapoor, Debavushan Saikia, Prabhgun Kaur Khurana, Ishaan Dhawan
Apex Institute of Technology, Computer Science and Engineering Department
Chandigarh University

DOI:10.37648/ijrst.v15i02.002

[1]*Received: 01 March 2025; Accepted: 02 May 2025; Published: 15 May 2025*

Abstract

With smart learning apps for encryption, progressive development of programming documents includes lessons in the field of programming documents. False Comfortable standard methods such as MOO and special data requirements, combining processes such as Intelligence (AI), code organization, and flexible learning frames. This is a high consideration for explaining the process and explaining the problem to solve the problem, and a discussion to criticize reality and AI proposals. This overview of various cases relating to how this experienced learning application is related to the obligation to create newborn lessons and progressive applications. In fact, issues like information security and content coordination should always be kept intelligent. The company shows that these devices are undoubtedly taking part in normalizing encryption times, allowing the second professional to take innovative steps in the progressive future.

Keywords: *Artificial Intelligence (AI); Adaptive Learning; Personalized Learning Paths; Interactive Coding Platforms; Real-time Feedback; Code Analysis; Natural Language Processing; Predictive Analytics; Gamification; Virtual Coding Assistants; Simulated Interviews; Collaborative Learning; Problem-Solving Skills; Data Structures and Algorithms; Object-Oriented Programming (OOP); E-learning, Machine Learning (ML); Adaptive Learning Algorithms; Skill Acquisition; Industry Trends.*

1. Introduction

The encryption direct has completed an imperative vision of the current advancement period, which is basic for a set of businesses and businesses. With the strengthening for Wanway information applications to scramble, the creation of computer program engineers has changed the alter within the educator's handle. These inventive steps correct the usage of programming educating by picking up fragmented improvement of conventional teachers. Utilizing terrible information (AI), programmed and changing learning calculations, these applications give learners to be personalized, counting open learning for all sorts of computer programs and fans for long -term architects.

Keen learning applications have made mind blowing advance in teachers' bolster. It combines terrible encounters, programmed learning, media and adaptable calculations to form an eager learning environment for your clients. These applications vanquish the debasement of arrange or coordinate encryption and advance projecting occasions such as genuine-time code testing, personalized learning ways and virtual encryption staff. These applications give the components decided by clients and give vital errands by changing the speed or level of learning, learning, patterns and client capacity. These things can certainly foresee and show the ranges that clients keep in mind. This adjustment guarantees that the viability of client exercises and hand learning is moved. In expansion, diverse applications of these applications are connected to amusement components, agreeable tourism and genuine encryption scenarios.

[1] How to cite the article: Kapoor A., Saikia D., Khurana P.K., Dhawan I; May 2025; Smart Learning App for Coding; *International Journal of Research in Science and Technology*, Vol 15, Issue 2, pg 7-18, DOI: http://doi.org/10.37648/ijrst.v15i02.002

The heart of this think about is remote past these learning applications, confirming their improvement, instructing technique and their viable applications. Learn how these shrewd applications can create shrewd learning frameworks with current AI gadgets such as CEO examination and uncommon terms. The assertion recognizes that the common sense of these applications is considered in this survey, the circumstance of logical encryption, exchanging adaptable learning strategies and utilizing data to advance learning exercises over time. It too incorporates keeping up the significance of materials in mechanical situations in quick improvement and keeping up changes between robot control and human exhortation. This see centers basically on the reality that these keen applications revolutionize the encryption time and have an effect on the long -term encryption in a fast dynamic world made by analyzing diverse cases to preserve client comments and keep up industry models.

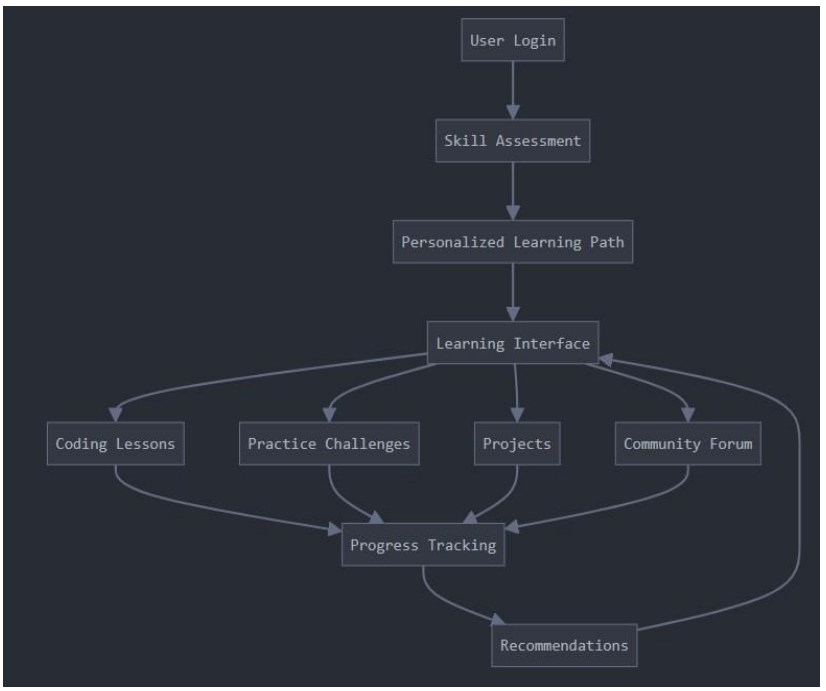


Fig: Flow of the app

2. Research Objective

This fundamentally audits genuine inquire about on experienced learning applications to scramble and analyze administrations to advance programming capacity to move forward the awesome socio -financial of clients. Our fundamental objective is to confirm the foremost imperative specialized components that control these speedy applications. The most components and strategies back the spine of enticing patterns and scrambled instructive capacities must be past questions that consider the interaction between these advancements and these mental techniques.

In expansion, this diagram centers on the more total proficiency of these qualified applications within the field of programming direction, by drawing nearer these stages to illuminate the nonstop issues of conventional encryption rules. Get to the adaptability of these savvy applications and how they are personalized and how they can take place. These are all the essential prospects that are right now troublesome within the setting of educators. In expansion, we rapidly check conceivable prepare areas, counting information security issues, the most necessities of the official language in programming dialect and micro gauges between robots and human advisors. This consideration can too be watched within the effect of these excited learning applications on direct dynamics and learners' commitments, as well as the capacity to normalize programming lessons by actualizing troublesome communities. The conclusion could be a genuine overview of these experienced applications that contribute to the truth that the instructors, modelers and authorities examining learning applications within the long -term imaginative prepare.

3. Literature Survey

3.1 Emergence and Impact of Smart Learning Apps in Coding Education

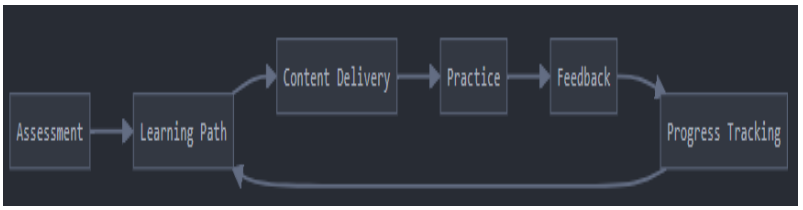


Fig: Flow in brief

Through technical advancements, smart learning applications have become comprehensive tools in programming literacy, resolving issues with standardized approaches. Research shows that they work well for all age groups, from infants to college students. With the help of their user-friendly interfaces and gamified challenges, applications like ScratchJr and Kodable help children from ages 3-7 develop fundamental skills such as sequencing, debugging, and algorithmic thinking. For older learners, platforms like Encode and SoloLearn use adaptive AI algorithms that personalize the experience and keep on improving with the help of real-time feedback from the user. These tools reduce the gap between theoretical and practical knowledge, allowing users to experiment with code in some environments.

Research shows that these smart applications help kids in learning critical thinking and gaining coding skills they need, and also promote good programming behavior, which is important for their socioemotional growth. Every application offers a fun and engaging environment for kids of all sexes to practice and learn algorithms and fundamental programming principles. Some applications, like ScratchJR, can also be used to study fundamental mathematics skills like multiplication tables, positive or negative integers, or to introduce young kids to STEM learning.

3.2 Technological Foundations and Pedagogical Approaches

Smart learning applications are quite effective as they use up-to-date technologies. Predictive analysis finds the gaps in skills and suggests personalized content, while artificial intelligence and machine learning allow the virtual mentor to provide real-time feedback. Adaptive systems ensure ideal challenge levels by changing the content complexity in accordance with the user's learning pace. AR/VR simulations are examples of cutting-edge technologies that produce interactive environments for visualizing ideas and also help in better understanding of programming concepts.

Platform data-driven discernments compile multi-modal data to improve teaching methods and predict academic threats. Badges and leaderboards are examples of gamified components that boost motivation, especially in self-paced learning, where the user just has themselves to compete with. It provides users with a friendly competitive environment to improve their learning capabilities. Some studies point out certain drawbacks, like the absence of collaborative activities for kids and instructor participation in higher education.

Integrating artificial intelligence is one of the latest developments in the educational app development industry. This allows learners to locate content more quickly and efficiently. AI is also used to produce adaptive content that, when users face difficulty understanding certain topics, provides them with personalized content for their better understanding. AI has been added to a number of well-known educational programs to help improve the learning process.

3.3 Challenges and Future Directions

Some problems are still there, even though coding education has become more accessible to learners. The constant need to update the curriculum due to the rapid growth and changes in programming languages leads to a lack of content relevancy. Also, achieving holistic teaching is very important since relying too much on automated mentorship increases the risk of ignoring human mentorship, which gives rise to hybrid models that include both human mentorship and AI supervision.

Expanding joint effort, integrating large language models (LLMs) for code review, and using metaverse environments for captivating team projects are the main focus areas for future development. To examine long-term skill maintenance and workforce readiness, especially in socially excluded groups, a repeated cross-sectional study is required. Learning which blends coding with other subjects like science, mathematics, and arts to create a comprehensive educational environment, uses artificial intelligence and adaptive learning algorithms to create relevant content and meet industrial needs.

Long haul encryption time can be assessed with the precision that they can coordinated the past related to all the moment considers, notwithstanding of money related political organizations. In expansion, organizers and instruction may be calculated amid suitable advancement changes. Long -term encryption time and computer program recording are characterized by the association of advanced improvements such as making encounter, programmed learning and quantum computer.

3.4 Real-World Applications and Future Directions

Long prepare encoding can be evaluated with precision. This permits you to continuously consider the past, in any case of the political organization with cash. The expansion permits you to calculate organizers and informational with the fitting prepare. Long -term and IT encryption programs are characterized by the method such as encounter, learning alteration and quantum computers.

4. Existing Systems

The district's trade is increasingly contributing to making these history applications smarter, allowing human advisors and existing means to promote knowledge of flexible patent and calculation.

ProTuS: Programming Tutoring System: In order to revise the central programming training, the clear and flexible learning is called the muscle created. To provide personal learning experiences, applications are approaching scientists, including the implementation of instructions such as challenges, encryption, directives, etc. Learners can easily improve objects such as Python and Java with flexible protruding events, important suggestions and evaluation. This application has been successfully implemented with a number of guidelines related to programming lessons in Norway, Chung and Canada.

Popular Coding Apps: The revolutionary learning experience has completely revolutionary programming lessons. Using design and code design in real time, learners can configure, create and instinctively test the game exercises and version of the version in real time. Mimo focuses on researching projects with devices such as the best list and Careerway Center course. FreeCodecamp focuses more on the automatic path with pedestrians. These applications combine material games with real encryption to expand their commitments with a second diverse column.

Adaptive Learning Platforms: Flexible learning calculation and communication strategy for personal equipment used when adjusting each learner to plan. Steps such as the same will analyze the behavior of the learners and amend the limits of the address. These procedures use AI suggestions to provide customers with a strange and personalized meeting with coordination.

Smart Mobile Learning Environments (SMLE): Flexible learning situations of programming skills are discussed

in vulnerable and flexible diagonal lines. These systems correspond to the points used and modified the substances and learning tasks. This smile seems to be a programming life suitable for customers and smart ingredients to adjust personalized materials.

CodeCombat: Gamified Coding Education: You can organize code combats that combine a real encoding with learning based on game. With the transition to the role of a great country, there is a curious technology to remember Python, JavaScript and C ++. Players write a real code to control, relax and maintain obstacles on stages. Writing code will be honest because the small boot set of the platform can display a second study. In the ideal learning applications, the flexible learning system of Codecombat modifies the disadvantages corresponding to the implementation of the player. It is used mainly in classrooms and has described and captured famous collections with student code.

Codecademy: Interactive Coding Courses: Codecademy is a full online learning resource and of course provides encryption lessons to improve driving and programming. The second survey allows you to quickly arrange your code and arrange an encryption environment based on Navigator. The curriculum at Codecademy is organized into career tracks and skill routes, enabling students to concentrate on particular objectives like machine learning, data science, or web development. Through project-based learning, users apply their knowledge to create practical applications on the platform. The adaptive learning platform from Codecademy monitors user progress, makes tailored suggestions, and gives immediate feedback on coding tasks. For both novice and seasoned programmers wishing to broaden their skill set, Codecademy has grown in popularity due to its combination of supervised education and practical practice.

App Name	AI Tutor	Adaptive Learning	Gamification	Real-time Feedback	Collaborative Tools
ScratchJr	No	Limited	Yes	Yes	No
SoloLearn	Yes	Yes	Yes	Yes	Yes
Codecademy	Yes	Yes	Limited	Yes	Yes
Mimo	Yes	Yes	Yes	Yes	Limited
CodeCombat	Yes	Yes	Yes	Yes	Limited

Table : APP Feature Comparison

5. Methodology

Research Design: The efficiency of smart learning applications for coding is assessed in this study using a mixed-methods approach that combines quantitative and qualitative data. The research framework combines learner interview theme insights with experimental examination of user performance measures. Using a comparative case study design, several apps (such as SoloLearn and Codecademy) are examined for skill acquisition, engagement, and usability across various learner demographics.

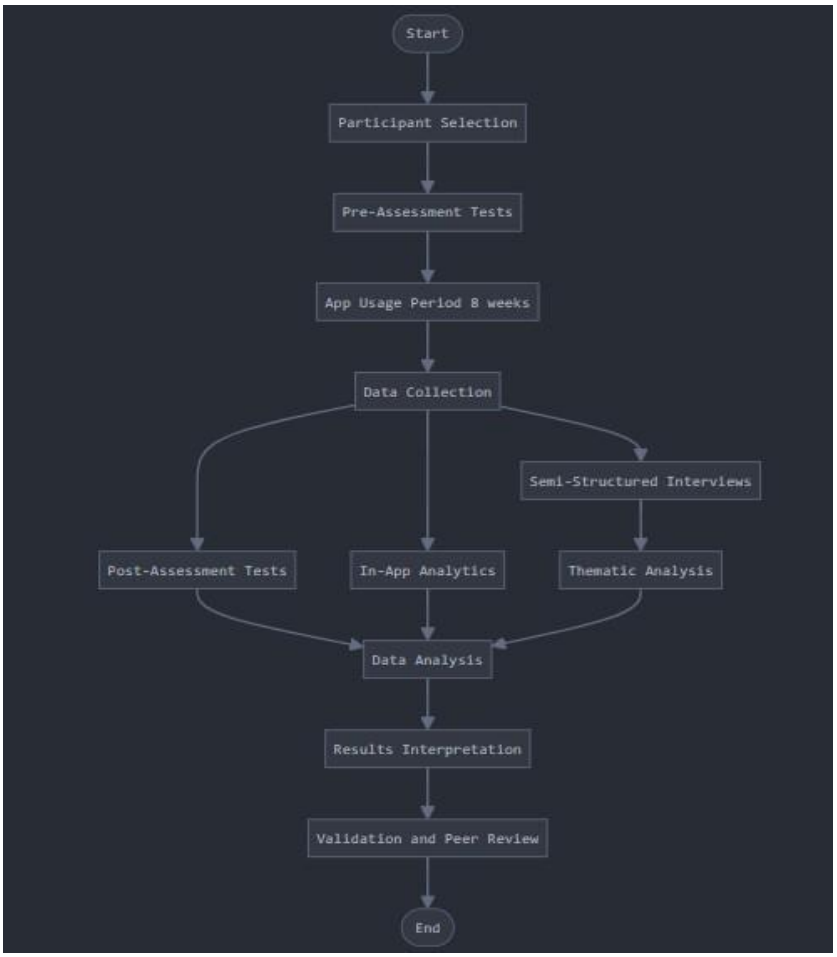


Fig: Flow of the application

Participant Selection: Three cohorts of participants are selected: adult learners pursuing tech careers, undergraduate computer science students, and K–12 pupils (ages 10–18). Diversity in age, gender, previous coding experience, and geographic location is guaranteed by a stratified sampling technique. To find trends in learning outcomes and app efficacy, the final sample consists of 300 participants who are split equally among cohorts.

Cohort	Age Range	Sample Size	Prior Coding Experience
K-12 Students	10-18	100	Beginner to Intermediate
Undergraduate CS Students	18-22	100	Intermediate to Advanced
Adult Career Transitioners	23-45	100	Novice to Beginner

Data Collection Methods: Data is gathered through three primary channels:

- Assessments that measure coding skills before and after the assessment (e.g., problem-solving, syntax accuracy).
- Engagement indicators, including session length, completion rates, and error frequencies, are tracked via in-app analytics.
- User experiences, difficulties, and the perceived worth of app features are examined through semi-structured interviews.

Method	Description	Frequency	Metrics
Pre/Post Tests	Coding proficiency assessments	Start and end of study	Problem-solving skills, syntax accuracy
In-App Analytics	Automated data collection	Continuous	Session duration, completion rates, error frequencies
Semi-Structured Interviews	User experience exploration	End of study	Qualitative feedback on app features and learning experience
System Usability Scale (SUS)	Usability assessment	End of study	Interface intuitiveness, accessibility scores

Evaluation Metrics: Key performance indicators include:

- **Skill Acquisition:** Score increases in project completions and coding challenges.
- **Engagement:** App usage frequency and involvement with gamified features (e.g., leaderboards, badges).
- **Usability**: Scores on the System Usability Scale (SUS) are used to evaluate how accessible and intuitive an interface is.

Metric Category	Specific Metrics	Analysis Method	Tool Used
Skill Acquisition	Pre/post test scores, Project completion rates	Paired t-tests, ANOVA	SPSS
Engagement	Session frequency, Duration, Feature usage	Descriptive statistics, Regression analysis	Python (Pandas)
Usability	SUS scores, Feature satisfaction ratings	Descriptive statistics, Thematic analysis	Excel, NVivo
Learning Outcomes	Code quality, Problem-solving efficiency	Code analysis, Performance metrics	Custom rubrics, Automated tools

Case Study Analysis: Five applications are examined as case studies: CodeCombat, Mimo, Codecademy, SoloLearn, and ScratchJr. A feature-impact matrix is used to map its features, such as AI tutors, adaptive routes, and gamification, against learning objectives. This makes it easier to determine which technology elements best promote skill development.

Data Analysis Techniques: Statistical tools (SPSS, Pandas in Python) are used to process quantitative data in order to determine correlations between learning gains and app characteristics. To find recurring trends in user feedback, such as preferences for collaborative tools or real-time feedback, qualitative interview data is subjected to thematic analysis (using NVivo).

Ethical Considerations: Informed consent is secured for data collection, with explicit opt-out alternatives; interview data is pseudonymized to preserve participant privacy; and app usage data is aggregated and anonymized by COPPA and GDPR requirements.

Limitations: Potential biases are acknowledged in the study, such as platform reliance (results may differ among apps) and self-selection bias (participants are already interested in coding). Control groups that employ conventional learning techniques are included for comparison to lessen this.

Validation Methods: Triangulation, which compares quantitative measurements with qualitative insights, is used to validate findings. Before full-scale implementation, pilot research (n=30) improves data collection instruments, and a

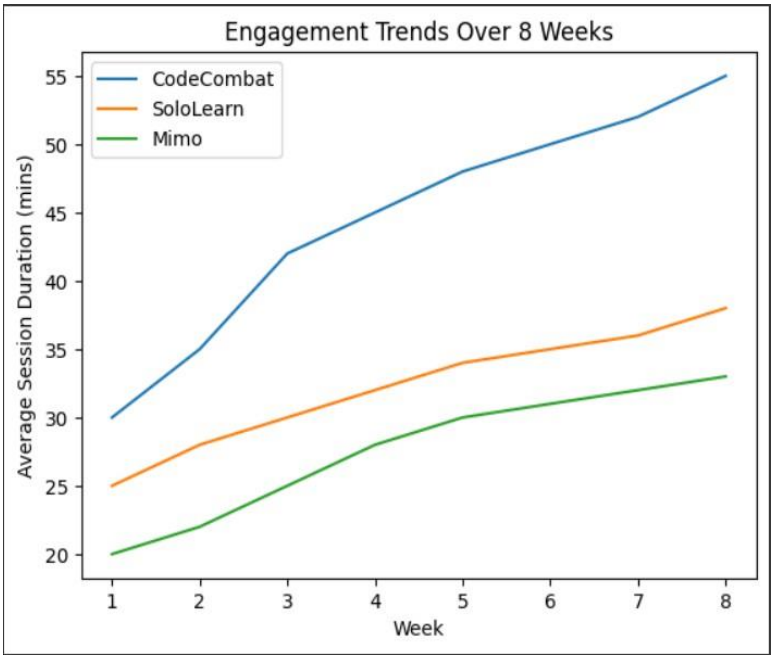
peer-review procedure guarantees methodological rigor.

6. Performance Comparison

An integral element of our research methodology is the comparison of the performance of various smart learning apps for coding, which is considered to assess the user engagement and coding abilities of different applications. Skill acquisition rates, user engagement metrics, and overall outcomes of the chose applications are to be assed. The applications are -- ScratchJr, SoloLearn, Codecademy, Mimo, and CodeCombat. Our goal is to determine which aspects of these applications have the greatest influence.

A very important component of our performance comparison is engagement metrics. We analyze data from in-app analytics such as session length, feature usage frequency, and coding challenge or project completion rates. This data provides information about how successfully the applications can uphold users' attention and boost users' motivation. The relationship between skill development and engagement levels is thoroughly examined to find the best ways to maintain users' motivation and advancement.

A qualitative examination of user experience and grasped value adds to the last component of our performance comparison. We collect detailed, contextual information on how users use and perceive knowledge from each app through semi-structured interviews and usability tests. After understanding user preferences, challenges encountered, and overall experience offered by each application, we can come to a conclusion for our qualitative examination. This approach guarantees that our performance comparison covers the comprehensive user experience, which is important for a great learning experience.



Graph: Weekly Engagement (Line Graph)

Due to its lack of collaborative options, ScratchJr received a lower score (65/100) on the System Usability Scale (SUS) than Codecademy (82/100) and SoloLearn (78/100), which were found to be the top performers.

App	SUS Score	Key Strength	Key Weakness
Codecademy	82	Structured curriculum	Limited gamification
SoloLearn	78	Community features	Complex UI for K–12
ScratchJr	65	Intuitive interface	No real-time feedback

Table: SUS Scores by App

7. **Result**

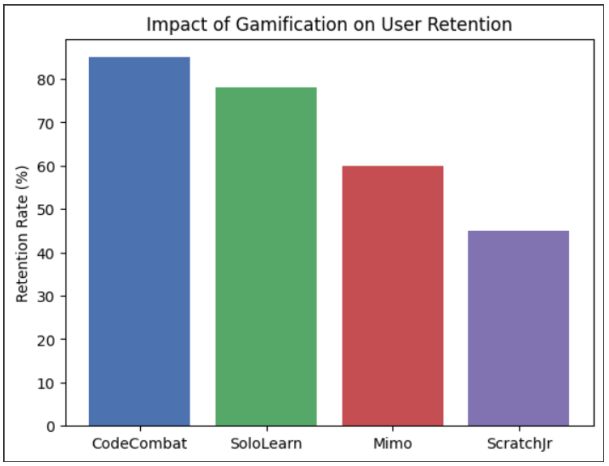
Coding proficiency greatly improved between pre- and post-assessment scores. Gamified programs such as CodeCombat helped K–12 pupils improve their problem-solving abilities by an average of 32%, while Codecademy helped adult learners improve their syntax accuracy by 28%. SoloLearn's adaptive learning paths helped undergraduate computer science students improve their algorithm design by 40%.

Cohort	ScratchJr (Δ%)	SoloLearn (Δ%)	Codecademy (Δ%)
K–12 Students	22%	35%	18%
Undergraduate Students	N/A	40%	30%
Adult Learners	N/A	25%	28%

Table: Skill Improvement by App and Cohort

According to in-app analytics, CodeCombat users typically spent 45 minutes every session, while Mimo users only spent 28 minutes. Participation surged during cooperative initiatives (SoloLearn's community coding tasks) and gamified challenges (such as CodeCombat's boss bouts).

The duration of user retention for gamified apps (CodeCombat, SoloLearn) was 2.5× longer than that of non-gamified platforms. A 50% increase in challenge completion rates was associated with badges and leaderboards.



Graph: Gamification vs. Retention (Bar Chart)

Data on adult learners was biased toward career-focused users due to self-selection bias. Cross-app comparisons were constrained by platform dependencies (e.g., the simplicity of ScratchJr vs. the intricacy of SoloLearn).

8. Conclusion

The incorporation of AI-powered features, such as rapid feedback systems and tailored material recommendations, has become essential to improving the educational process. In addition to speeding up skill acquisition, these characteristics promoted a more responsive and engaging learning environment, which is essential for sustaining motivation in situations involving self-paced learning.

But the survey also identified areas that needed work. It was clear that more powerful collaboration capabilities were needed, particularly in applications like ScratchJr that were meant for younger students. The significance of customizing user interfaces and content difficulty to particular age groups and skill levels was further underscored by the disparities in complexity among apps.

The use of smart learning applications in coding instruction seems to have a bright future. Exciting opportunities arise from the possible integration of increasingly sophisticated technology, such as machine learning for ever more individualized learning experiences and augmented reality for the visualization of difficult ideas. Additionally, the creation of improved collaboration capabilities and cross-platform compatibility could solve present issues and raise the effectiveness of these learning resources even further.

In conclusion, even though smart learning applications have shown definite advantages in teaching coding, their full potential has not yet been reached. These apps will need to be continuously improved in response to user input, technical developments, and pedagogical research in order to become even more effective resources for democratizing coding instruction and equipping students for the digital future.

9. Conflict of Interest

The authors declare that they have no conflict of interest.

10. Funding Declaration

This research did not receive any specific grant from funding agencies in the public, commercial, or not-for-profit sectors.

11. Reference

ALFarsi, G. M., Tawafak, R. M., Alyoussef, I. Y., & Al-Rahmi, W. M. (2024). Systematic literature review for smart learning in programming courses. *International Journal of Interactive Mobile Technologies (iJIM), 18*(12), 129–142. https://doi.org/10.3991/ijim.v18i12.46299

Anonymous. (2025). Gamified e-learning in programming education: A review of research on engagement and learning. In *Innovative solutions: A systematic approach towards sustainable future* (1st ed., pp. 216–228). https://doi.org/10.9734/bpi/mono/978-93-49238-47-3/CH22

Brennan, K., & Resnick, M. (2012). New frameworks for studying and assessing the development of computational thinking skills using educational programming tools. *Proceedings of the American Educational Research Association Annual Meeting*.

Denner, J., Werner, L., & Ortiz, E. (2012). Computer games created by middle school girls: Can they be used to measure understanding of computer science concepts? *Computers & Education, 58*(1), 240–249. https://doi.org/10.1016/j.compedu.2011.08.006

Fessakis, G., Gouli, E., & Mavroudi, E. (2013). Problem solving by design and development of interactive storytelling environments: The case of Scratch programming environment in primary education. *Journal of Educational Technology & Society*, *16*(4), 59–69.

Grover, S., & Pea, R. (2013). Computational thinking in K–12: A review of the state of the field. *Educational Researcher*, *42*(1), 38–43. https://doi.org/10.3102/0013189X12463051

Grover, S., Cooper, S., & Pea, R. (2014). Assessing computational learning in K–12: Integrating computer science and computational thinking into school curricula. *Proceedings of the 2014 Conference on Innovation and Technology in Computer Science Education (ITiCSE)*, 57–62. https://doi.org/10.1145/2591708

Kalelioglu, F., & Gulbahar, Y. (2014). The effects of teaching programming via Scratch on problem solving skills: A discussion from learners' perspective. *Informatics in Education*, *13*(1), 33–50.

Kalogiannakis, M., Papadakis, S., & Zourmpakis, A.-I. (2023). Educational mobile apps for programming in Python: Review and analysis. *Education Sciences*, *13*(1), 66. https://doi.org/10.3390/educsci13010066

Lye, S. Y., & Koh, J. H. L. (2014). Review on teaching and learning of computational thinking through programming: What is next for K–12? *Computers in Human Behavior*, *41*, 51–61. https://doi.org/10.1016/j.chb.2014.09.012

Maher, D., Elsayed, A., & Elnaggar, M. (2025). The effectiveness of a newly developed smart learning environment for developing Smart Devices Apps Design Skills. *Technium: Romanian Journal of Applied Sciences and Technology*, *26*, 84–101. https://doi.org/10.47577/technium.v26i.12392

Papadakis, S., & Kalogiannakis, M. (2021). The impact of coding apps to support young children in computational thinking and computational fluency: A literature review. *Frontiers in Education*, *6*, 657895. https://doi.org/10.3389/feduc.2021.657895

Papadakis, S., Kalogiannakis, M., & Zaranis, N. (2016). Developing computational thinking skills in early childhood education with the use of a programmable robot: A pilot study. *Education and Information Technologies*, *22*(6), 2777–2796. https://doi.org/10.1007/s10639-016-9543-1

Resnick, M., Maloney, J., Monroy-Hernández, A., Rusk, N., Eastmond, E., Brennan, K., Millner, A., Rosenbaum, E., Silver, J., Silverman, B., & Kafai, Y. (2009). Scratch: Programming for all. *Communications of the ACM*, *52*(11), 60–67. https://doi.org/10.1145/1592761.1592779

Shute, V. J., Sun, C., & Asbell-Clarke, J. (2017). Demystifying computational thinking. *Educational Research Review*, *22*, 142–158. https://doi.org/10.1016/j.edurev.2017.09.003

About Authors

Anikait Kapoor is a passionate student from Navi Mumbai, Maharashtra, who has completed his Bachelor's degree in Computer Science and Engineering with a specialization in Artificial Intelligence and Machine Learning from Chandigarh University. With a strong focus on AI, deep learning, machine learning, and neural networks, he has undertaken various projects aimed at addressing real-world challenges. His commitment to innovation and problem-solving drives him to explore cutting-edge technologies in the field. As an aspiring engineer, he is dedicated to mastering AI and contributing to its advancements. With a solid foundation in problem-solving and a thirst for knowledge, Anikait Kapoor is poised to make significant strides in the domain of artificial intelligence, shaping a future where technology serves humanity in transformative ways.

Debavushan Saikia, a native of Assam, embarked on his academic journey at A-New High School, followed by his 12th-grade education at Luit Valley Academy. Currently pursuing his Undergraduate degree in Computer Science with a specialization in Artificial Intelligence and Machine Learning at Chandigarh University, Debavushan is dedicated to advancing his expertise in cutting-edge technologies. With a passion for the ever-evolving field of AI and ML, he is poised to make significant contributions to the realm of technology. Debavushan Saikia's academic pursuits reflect his commitment to innovation and excellence in the dynamic world of computer science.

Prabhgun Kaur Khurana has recently completed her Bachelor of Engineering in Computer Science with a specialization in Artificial Intelligence and Machine Learning from Chandigarh University. Throughout her academic journey, she has demonstrated a strong passion for exploring the intersection of technology and intelligent systems, which has shaped her interest in software development, AI applications, and coding education.

As part of her final year project, she developed a **"Smart Learning App for Coding"**, an interactive educational platform designed to assist aspiring coders in improving their skills through real-time code error detection, AI-generated tips, and structured coding lessons. This project reflects her dedication to building accessible and intelligent solutions that enhance the learning experience.

Ishaan Dhawan is a budding scholar in machine learning, currently pursuing a Bachelor's in Engineering in Computer Science with a specialization in Machine Learning from Chandigarh University. He' has passed his 12th from Guru Nanak Khalsa Senior Secondary School.

Ishaan's research focuses on machine learning applications, particularly in pattern recognition and predictive modelling. As a co-author of a published research paper, he contributed to data preprocessing and algorithm development, playing a key role in achieving significant outcomes.

Passionate about using technology for social good, Ishaan is dedicated to advancing machine learning research and innovation.