第 48 卷 第 5 期

七月 2023

Diqiu Kexue - Zhongguo Dizhi Daxue Xuebao/Earth Science Journal of China University of Geosciences

https://dqkx-periodicals.com

Vol. 48 No. 5

July 2023

https://doi.org/10.3799/dqkx.2023.782

FGPE+: An Empirical Assessment of the Mobile FGPE Environment and the Pareto-Optimized Model for Gamified Programming Exercise Selection.

Ricardo Queirós, Nasir Jabar, Faheem Aouse

Department of Computer Science, Faculty of Sciences, University of Porto, 4169-007 Porto, Portugal

Received: 23 June 2023

Revised: 7 July 2023

Accepted: 15 July 2023

Published: 23 July 2023

Abstract: This paper is poised to inform educators, policy makers and software developers about the untapped potential of PWAs in creating engaging, effective, and personalized learning experiences in the field of programming education. We aim to address a significant gap in the current understanding of the potential advantages and underutilisation of Progressive Web Applications (PWAs) within the education sector, specifically for programming education. Despite the evident lack of recognition of PWAs in this arena, we present an innovative approach through the Framework for Gamification in Programming Education (FGPE). This framework takes advantage of the ubiquity and ease of use of PWAs, integrating it with a Pareto optimized gamified programming exercise selection model ensuring personalized adaptive learning experiences by dynamically adjusting the complexity, content, and feedback of gamified exercises in response to the learners' ongoing progress and performance. This study examines the mobile user experience of the FGPE PLE in different countries, namely Poland and Lithuania, providing novel insights into its applicability and efficiency. Our results demonstrate that combining advanced adaptive algorithms with the convenience of mobile technology has the potential to revolutionize programming education. The FGPE+ course group outperformed the Moodle group in terms of the average perceived knowledge (M = 4.11, SD = 0.51).

Keywords: FGPE; pareto optimization; gamified programming; personalized learning; adaptive learning; progressive Web Applications (PWAs); mobile learning

1. **Introduction**

Traditional programming education techniques sometimes struggle to keep students engaged and motivated. Gamification models provide a solution by including interac- tive and game-like features that pique students' attention, improve their engagement, and create a better understanding of programming ideas. The events of the COVID-19 epidemic prompted higher education institutions around the world to quickly adjust to remote learning approaches [1,2]. The incorporation of e-learning platforms has been a popular technique, albeit the transition has not always been effective, frequently resulting in inferior performance when compared to traditional classroom-based training [3]. To fully realize the potential of gamified exercises and improve the e-learning experience, it became clear that these activities should be smoothly integrated into the existing course structure rather than given via separate platforms [4]. The Learning Tools Interoperability (LTI) standard developed as a feasible approach to support this integration. Students would be routed to a gamified learning environment within the e-learning platform itself if LTI
was implemented, where they may engage in programming exercises. Furthermore, the outcomes of these activities might be quickly conveyed back to the e-learning platform, allowing for real-time feedback [5]. Gamified programming exercise selection models have received a lot of attention in recent years as a way to improve learning and engagement in programming education. These platforms use gamification aspects in conjunction with programming exercises to incentivize and encourage learners to actively engage in the development of their programming abilities. A significant body of research now suggests that gamification techniques and serious games play a crucial role in enhancing learning outcomes in programming education [6,7]. The underlying premise is that these approaches make learning more enjoyable and engaging, which in turn leads to improved knowledge retention and application [8,9].

Gamification has a positive impact on students' motivation [10,11]. Techniques such as points, badges, leaderboards, and challenges can stimulate students' competitive nature and foster collaboration [12,13]. Serious games, especially those addressing real-world issues, such as sustainable development [14] and healthcare [15,16], can foster deeper understanding and enhance programming skills. The game 'Eco JSity' is a successful example of integrating sustainability topics into programming education [14]. Gamifica- tion can improve self-efficacy in programming [11]. The perceived ability to perform a task often translates into better performance. Game-based learning approaches enhance student experience, knowledge gain, and usability in higher education programming courses [17]. Researchers also highlight different levels of gamification, from partial to full integration, into the course. For example, the SHOOT2LEARN project [18] and the game 'SQL Island' [19] mix gameplay with programming, enhancing student engagement. Meanwhile, 'SpAI War' is used for learning artificial intelligence algorithms [20]. These studies collectively suggest that gamification and serious games provide new, innovative ways to teach programming [21]. They transform the traditionally complex and abstract programming concepts into interactive, relatable scenarios for students. This helps to reduce the intimidation factor often associated with learning to program, which in turn can lead to improved motivation and self-efficacy [22]. However, it is essential to consider the varying degrees of success in implementing gamification, which is not always granted [23]. The effective integration of these techniques into a course requires thoughtful design and ongoing evaluation to ensure they are meeting the intended learning objectives [24]. Gam- ified programming systems rely heavily on engaging learners. The studies look at the motivating elements, pleasure, and immersion that learners have when using the mobile platform [25]. Analyzing the effectiveness of gamification features in creating intrinsic motivation and sustained engagement is part of this. Moreover, while game-based learning and gamification demonstrate promise, they are not a one-size-fits-all solution. They should be used alongside traditional teaching methods to cater to diverse learning styles and needs. More empirical studies are also needed to understand their long-term impacts and potential side effects (e.g., whether they lead to an over-reliance on reward structures) [26].

Technological advancements, such as the growth of mobile devices, online learning platforms, and instructional software, have opened the opportunity for gamification meth- ods to be implemented in programming education. The availability of technologies and platforms to facilitate gamified learning experiences has fuelled the demand for these approaches. Although gamified exercises through LTI offer potential [27], there was always a need to improve the programming learning environment for mobile device users [28]. To solve this, a mobile gamified Programming Learning Environment (PLE) can be created utilizing Progressive Web Application (PWA) technology [29]. PWAs leverage new APIs and capabilities to deliver functionality comparable to native apps, such as push notifica- tions, sensor integration, and the ability to have an access icon on the device's application launcher [30]. It is critical to assess the usability of mobile platforms to ensure that learners can easily explore, participate, and finish programming tasks. Examining characteristics such as ease of use, learnability, efficiency, and mistake prevention are all part of usability evaluations. Despite the potential benefits of PWAs in education, their use is restricted and their recognition in the area is minimal. The reasons for this limited acceptance are yet unknown [31]. PWAs tend to be underused, with the method going mostly unnoticed in the programming education sector [32]. This knowledge gap emphasizes the need for more PWA research and study in the context of educational applications. The Framework for Gamification in Programming Education (FGPE) was created to fill these shortcomings. This method sought to provide a comprehensive framework for the use of gamification techniques in programming education, spanning requirement design, the collection of gam- ified exercises, and the creation of supporting software [33]. The PWA design was chosen as part of its implementation to create mobile-like web apps with increased functionality and user experience.

This paper provides an in-depth examination of the integration of the Pareto-optimized gamified exercise selection model into e-learning courses using the LTI standard. It also emphasizes the potential of mobile gamified PLEs based on PWA technology to improve the programming learning environment for mobile device users. The FGPE method offers a framework for gamification in programming instruction, as well as the development of accompanying applications. Significant efforts were made during the installation of the FGPE strategy to optimize the performance of the mobile gamified PLE with a Pareto optimal exercise selection approach. This improved the efficiency of the app in providing programming training and the overall learning experience, as was confirmed by students utilizing these methodologies. Cross-country comparisons were also conducted to help understand the differences in user experience and educational results between areas and technical aspects that can impact user experience [34] in different nations.

We have aimed to assess the general attitude of mobile device users towards the PWA version of the FGPE PLE platform, as well as the suitability of the mobile version for different learning contexts. The evaluation used self-report measures, such as rating scales, to gather feedback from the users. Additionally, the study employed the User Experience Questionnaire (UEQ) to obtain a more detailed understanding of users' views on the mobile version of the platform. The responses from students in Poland and Lithuania were compared to identify any cross-country differences. Furthermore, a knowledge evaluation survey compared the effectiveness of the FGPE+ model with a classic Moodle course in teaching programming. The survey assessed the perceived knowledge of learners in each group. Lastly, the study evaluated the effectiveness of using the Sharable Content Object Reference Model (SCORM) in the FGPE+ model.

This paper is organized as follows. Section 2 reviews existing approaches to gamifica- tion in education. Section 3 describes the conducted experiment, including information on participants, the design of the gamified programming task selection model, and the implementation of such a model in FGPE. Section 4 presents the results of the evaluation of this implementation in various aspects. Section 5 discusses the outcomes of our study and its potential impact in STEM (Science, Technology, Engineering, and Mathematics) education. Finally, Section 6 summarizes the contributions of this work.

2. Overview of Approaches to Gamification in Educational Apps

According to research, well-designed gamification models can improve learning out- comes in programming education. They can boost student motivation, encourage long-term engagement, and improve information retention and skill development. As a result, ed- ucational institutions and companies are looking for efficient ways to improve learning outcomes in programming education. By bringing game design ideas and mechanics to programming learning environments, general gamification approaches for programming education attempt to improve student engagement, motivation, and learning results [35]. We frequently encounter recognizable elements such as:

- The Points, Badges, and Leaderboard (PBL) model entails allocating points to accom- plished tasks or achievements, awarding badges for particular achievements, and keeping leaderboards to display student standings [36]. It uses competition and prizes to inspire students and promote a sense of accomplishment and growth.

- Quests and Levels model, in which learning activities are arranged into quests or missions, and learners move through a series of obstacles or levels [37]. Each level often provides new concepts or programming tasks, resulting in a logical learning path. Completing quests or levels opens additional content or features, giving the player a sense of advancement and success.
- Storytelling and narrative model that includes storytelling and narrative components into the programming learning process [38]. It develops immersive settings, characters, and plotlines to emotionally engage learners and relate learning information to real-world circumstances. Learners take on parts in the plot and embark on quests or missions, making learning more interesting and meaningful [39].
- The Achievements and unlockables model, as in video games, provides a system of achievements and unlockables [40]. Learners receive rewards when they complete particular programming assignments, grasp concepts, or reach milestones. Unlock-ables are extra challenges, added content, or special features that become available as learners progress and meet certain goals.
- The Companion or virtual pets model involves the incorporation of companion char- acters or virtual pets into the programming learning environment [41]. By completing programming assignments or demonstrating expertise, students can care for their virtual pets, earn rewards, and unlock additional capabilities. Throughout the learning process, the companions provide comments, direction, and encouragement [42].
- By integrating collaborative challenges and tournaments [43], the Collaborative chal- lenge and tournament model stimulates collaboration and competitiveness among learners. Students create groups, collaborate to solve programming issues, and com- pete against other groups. It promotes camaraderie and pleasant rivalry while encour- aging cooperation, communication, and problem-solving abilities [44].
- The Simulations and Virtual Environments Model [45] use simulations or virtual environments to provide realistic programming scenarios. Within the simulated environment, learners participate in hands-on coding activities such as constructing virtual apps or solving virtual programming tasks. Simulations provide a secure environment for experimentation and practice, allowing students to apply principles in a real-world setting [46].
- Gamification models almost always some include feedback and progress monitor- ing [47]. Learners immediately receive feedback on their coding performance, flagging problems and offering improvements. Progress tracking techniques such as progress bars, skill trees, or visual representations assist learners in seeing their progress and providing a sense of success [35].

- Gamified Learning Analytics combines learning analytics [48] with gamification [49]. It collects data from learners' interactions with gamified programming environments to gather insights into their learning habits, progress, and areas for growth. To improve learning outcomes, these insights enable tailored feedback, adaptive interventions, and instructional decision-making. Empirical studies and scientific research are used to develop evidence-based gami- fication models for programming education in order to improve learning outcomes and engagement in programming education [50]. Below, we offer a rundown of some of the more popular research-based gamification approaches used in programming education:
- Goal-Structure Theory emphasizes the importance of defining specific objectives and creating a structured learning environment [51]. It underlines the significance of defining distinct, difficult, and attainable goals in programming assignments. Clear goals provide learners a feeling of direction and purpose, which promotes motivation and attention throughout the learning process.
- Learner autonomy, competence, and relatedness are all emphasized in Self-Determination Theory [52]. It demonstrates that learners are motivated when they feel in charge of their learning, recognize their competency in programming tasks, and feel connected to others. Gamification components that encourage autonomy, skill development, and social interaction can boost student motivation and engagement.
- The goal of Flow Theory [53] is to generate a state of "flow" in which learners are completely absorbed and interested in programming tasks. Flow occurs when learners confront tasks that are appropriate for their ability level, receive fast feedback, and feel a sense of control and concentration. Flow experiences in programming education can be facilitated by gamification features that increase challenge, feedback, and focused attention.
- In programming education, Social Cognitive Theory stresses the importance of ob- servational learning, social interaction, and feedback [54]. Learners may watch and learn from other people's programming techniques, participate in collaborative coding exercises, and receive constructive comments from peers or instructors. Based on this principle, gamification models stimulate social learning, give chances for knowledge exchange, and promote positive reinforcement [55].
- The Cognitive burden Theory [56] is concerned with controlling cognitive burden during programming activities. It implies that instructional design should aim to reduce external cognitive strain while increasing internal cognitive demand. Based on this principle, gamification models may include interactive components, step-by-step assistance, and scaffolded learning to minimize cognitive load and improve learning efficiency [57].
- Mastery Learning [58] advocates for a mastery-based approach to programming in- struction. It pushes students to grasp basic programming concepts and abilities before moving on to more difficult topics. Gamification methods based on mastery learning give adaptive feedback, tailored learning routes, and chances for purposeful practice to assist learners in achieving mastery and establishing a solid programming foundation.
- Personalized and adaptive gamification models adjust the learning experience to the qualities, interests, and needs of the individual learner [59]. They use student data, such as performance history and learning styles, to modify the difficulty level, material sequencing, and feedback in programming exercises dynamically. Personalization and adaptability boost student engagement, improve learning efficiency, and accommodate a wide range of learning profiles [60].

To prevent possible downsides, it is critical to find a balance between gamification components and primary learning objectives [61]. Careful implementation, constant as- sessment, and adaptive design based on learner input may help maximize the benefits of research-based gamification models, while limiting the obstacles that come with them. Gamification models can enhance programming education by catching learners' attention and inspiring them to actively participate in learning sessions. The incorporation of game features can boost engagement and interest in programming by creating a sense of excite- ment and challenge [62]. By combining aspects such as incentives, accomplishments, and competition, gamification taps into both inner and extrinsic motivating factors. Learners are driven to complete goals, gain badges, and climb leaderboards, which might inspire them to continue studying and achieve higher outcomes. Gamification models may be tailored to meet the requirements and preferences of individual learners. Adaptive components and individualized feedback can create tailored learning experiences, allowing students to advance at their own speed while receiving focused assistance [63]. Personalization encourages a more effective and efficient learning process. Gamification models frequently include interactive coding exercises, simulations, and challenges, giving learners hands-on practice. Through active experimentation, learners may apply programming ideas in a practical setting, developing their problem-solving and coding abilities [64]. Gamified programming environments may provide learners with rapid feedback on their coding per- formance. Instant feedback enables learners to quickly detect and rectify errors, reinforcing their comprehension and enabling deeper learning.

However, there are various drawbacks to employing research-based gamification techniques [65]. There is a danger that learners will become more focused on obtaining points, badges, or rankings than on true learning. If the game aspects take precedence over the learning objectives, learners may participate in superficial engagement, prioritizing the gamification features above real comprehension and skill development. While competition may be inspiring, it can also create a climate that inhibits learners from collaborating and cooperating. Some students may become demotivated or disengaged if they believe they are slipping behind their peers, resulting in unpleasant learning experiences [66]. Gamification models should be carefully built to be compatible with the programming education setting and learning objectives. Not every game element may be appropriate or helpful for every programming topic or learning scenario. Gamification features should be relevant, meaningful, and connected with the targeted learning goals. Learners' learning methods, interests, and motivations vary [67]. Gamification methods may not appeal to all learners in the same way, and what drives one student may not inspire another. Individual variations must be considered, and gamification tactics must be tailored to different learner profiles. Maintaining and implementing gamified programming environments necessitates constant work and resources. Updating material, tracking learner progress, and fixing technological difficulties can be time-consuming and may need specialized staff or systems to support the gamification model's long-term viability [68].

In order to overcome all these drawbacks, several gamification design frameworks appeared in the last decades to provide a systematic and structured approach to avoid the ad hoc implementation of gamification components and learning theories, enabling organizations to effectively scaffold the development of a well-defined gamification strategy. Existing gamification design frameworks can be divided into process-based frameworks, component-based frameworks, and goal-based frameworks. Process-based frameworks provide a structured approach to gamification design, guiding designers through a series of steps. Following these steps, designers ensure that game design elements are incorporated effectively. Some examples of these frameworks are Octalysis, 6D, and Game Thinking. The Octalysis framework defines and builds upon the eight core drives of human motivation [69]: Epic Meaning and Calling, Development and Accomplishment, Empowerment of Creativity and Feedback, Ownership and Possession, Social Influence and Relatedness, Scarcity and Impatience, Unpredictability and Curiosity, and Loss and Avoidance. The framework is based on a human-focused design, i.e., de- signed to motivate users, as opposed to function-focused design, i.e., designed assuming users complete their tasks in a timely manner, as they are required to do so. The 6D Frame-work [70] is a design process composed of six steps, namely "Define business objectives", "Delineate target behavior", "Describe yours players", "Devise activity loops", "Don't forget the fun" (i.e., ensure that there is fun while using with the system), and "Deploy appropriate tools". The Game Thinking Framework [71] combines game design, systems thinking, design thinking, and agile/lean practices into a recipe for gamification designers. Component-based frameworks focus on specific game design elements, providing guidelines to apply these elements into non-game contexts. One of the most popular gami- fication design frameworks, the Mechanics–Dynamics–Aesthetics (MDA) framework [72], belongs to this group. The MDA framework breaks games down into three components: mechanics, dynamics, and aesthetics. Mechanics are the base game components such as rules and algorithms. Dynamics include the run-time behavior of the mechanics acting on player input, which involve other mechanics. Aesthetics are the player's emotional responses to the system, such as sensation, fantasy, narrative, challenge, fellowship, dis-covery, expression, and submission. The MDA framework provides precise definitions for these terms and seeks to explain how they relate to each other and influence the player's experience. Other component-based frameworks include the SAPS (Status, Access, Power, and Stuff) framework [73], which focuses on the four key components of gamification design: status, access, power, and stuff, and provides guidelines for incorporating these
components into non-game contexts.

Goal-based frameworks capitalize on the desired outcomes of the gamification design process (e.g., increased engagement, motivation, and learning), providing guidelines to
achieve these outcomes by using game design elements. Examples of frameworks in this category are RAMP (Relatedness, Autonomy, Mastery, and Purpose) [74] and ARCS (Attention, Relevance, Confidence, and Satisfaction) [75]. The RAMP framework focuses on four key outcomes of gamification design: retention, achievement, mastery, and progress. The framework provides guidelines for achieving these outcomes through game design elements such as points, levels/stages, badges, leaderboards, prizes and rewards, progress bars, storyline, and feedback. The ARCS model focuses on the four key components of motivation: attention, relevance, confidence, and satisfaction.

3. Materials and Methods

3.1. Materials

The 23 Lithuanian students from the Faculty of Informatics ranged in age from 18 to 25 years. Variations were noted within this range, with a few older persons falling slightly outside of it. The gender distribution among the student cohort favored male students, who had a larger representation than females. Notably, the sample includes a handful of transgenders, contributing to the group's diversity of gender identities. All 23 students in the sample were pursuing programming-related specialties. This emphasis demonstrates their unique interest in learning about creating, building, and maintaining software systems. The majority of the participants were from Lithuania, which corresponded to the location of Kaunas University of Technology.

The 49 Polish respondents were 1st-year students of IT in Business and IT & Econo- metrics attending the Introduction to Programming course at the Faculty of Economics, Finance, and Management of University of Szczecin. Almost all of them were 19 years old; 1/4 of them were female, and 3/4 male. Alike the students at Kaunas University of Technology, they have a passion for developing software systems, yet in contrast to them, they are more focused on enterprise information systems rather than software in general.

3.2. Pareto-Optimized Gamified Programming Task Selection Model

Designing a Pareto-optimized gamified programming task selection model (Figure 1) for adaptive personalized learning involves the creation of a multi-objective optimization model that seeks to balance multiple competing factors such as the learner's interests, abilities, task difficulty, novelty, and relevance to the curriculum. In this context, Pareto optimization refers to a state of allocation where it is impossible to make any one individual better off without making at least one individual worse off.

The model has the following key components:

1. Programming task bank: A repository of programming tasks, each classified according to their difficulty level, related topic, required skills, estimated completion time, etc.
2. Learner profile: A dynamic profile for each learner capturing their programming skill level, areas of interest, learning pace, historical performance on tasks, preferred learning style, etc.

3. Gamification elements: Incorporation of game design elements such as points, badges, leaderboards, achievement tracking, feedback, progress bars, storyline, etc.

Formally, the model can be described as follows:
Let L = {L1, L2, . . . , Ln} be the set of learners.
Each learner Li is represented as a tuple (idi, sli, ini, pi, hi, sti), where:
idi is the id,
sli is the skill level,
ini is the set of interests,
pi is the learning pace,
hi is the history of completed tasks,
sti is the learning style.
Let T = {T1, T2, . . . , Tm} be the set of tasks.
Each task Tj is represented as a tuple (idj, dj, tj, sj, timej), where:
idj is the id,
dj is the difficulty level,
tj is the topic,
sj is the set of required skills,
timej is the estimated completion time.
Let A : L × T → 2T be the assignment function, where 2T is the power set of T. This function assigns to each learner a set of tasks, i.e., A(Li) = {Ti1, Ti2, . . .} ⊆ T. Let O : L × T → 2T be the Pareto optimization function, defined as:
O(Li, T) = {Tj ∈ T | there does not exist Tk ∈ T such that Tk is better than Tj for Li}.
Let U : L × T × R → L be the update function, defined as:
U(Li, Tj, per f ormance) = Li′, where Li′ is the updated learner profile based on the performance on task Tj.
Here, L is the set of learners, each represented as a tuple of parameters. T is the set of tasks, each represented as a tuple of parameters. A is the assignment function that maps each learner to a set of tasks. O is the Pareto optimization function that assigns a learner a subset of tasks for which there are no better alternatives. U is the update function that updates a learner's profile based on their performance on a task.
The model starts by initializing the learners and tasks. Each learner is then assigned a set of tasks that are Pareto optimized for them. The Pareto optimization process involves finding the balance between different objectives (learner's interests, skill level, pace, etc., vs. the task's difficulty, skills required, topic relevance, etc.) to maximize the learning outcome. The learner's profile is updated after they complete a task based on their performance, and the task assignment process can be repeated as necessary. Gamification elements can be added to the model to increase learner engagement.

3.3. Implementation

Our goal was to fill a substantial knowledge gap on the potential benefits and inade- quate use of Progressive Web Applications (PWAs) in the education sector, particularly for educational programming. The FGPE+ model provides programmers with a unique and engaging mobile user experience (see Figure 2). To create an effective and pleasant learning environment, it blends the ideas of Pareto optimization, gamification, and programming exercises. The PWA-based mobile-compatible solution has a clean and straightforward user experience that facilitates navigation and interaction. The interface is designed in a modern, minimalist style, with an emphasis on usability and clarity. When new users start the app, they are met with a full onboarding experience that introduces them to the features and capabilities of the FGPE+ model. Within the app, users are encouraged to create individualized profiles. They can choose their preferred programming language (Javascript, Python, Java, C#, Cpp), skill level, and areas of interest. This data allows the FGPE+ model to personalize exercise recommendations to the user's specific requirements and goals.

1. The exercise abstract class represents a programming exercise. It has attributes such as id (exercise identifier), difficulty (difficulty level of the exercise), and learningOut- comes (a list of learning outcomes associated with the exercise). It provides meth- ods to access these attributes and defines three virtual methods: evaluateObjec- tiveWeights(to evaluate the objective weights of the exercise), calculateObjectiveVal- ues() (to calculate the objective values of the exercise), and compareTo() (to compare two exercises based on their objective values).
2. ParetoExercise class represents an exercise that includes objective values. It inherits from the Exercise class and has an additional attribute called objectiveValues, which is a map that stores objective values for the exercise. It provides methods

to obtain and set objective values for specific objectives and defines the dominates() method to check whether it dominates another ParetoExercise based on their objective values.

3. ExerciseSelector abstract class serves as the base class for the exercise selection al- gorithm. It has attributes exercises (a list of exercises to select from) and objec- tiveWeights (a map that holds the weights of different objectives). It provides meth- ods to select exercises, sets the exercises and objective weights, and defines six virtual methods that outline different steps of the algorithm: evaluateExercises() (to evaluate the exercises based on objectives), paretoOptimization() (to perform Pareto opti- mization on the exercises), diversityEnhancement() (to enhance the diversity of the exercise set), gamificationIntegration() (to integrate gamification elements into the ex- ercises), personalization() (to personalize the exercise selection), and evaluationAnd- FeedbackLoop() (to evaluate and refine the exercise selection based on feedback).

4. MyExerciseSelector. This class represents a specific implementation of ExerciseS- elector. It adds an additional attribute called the threshold (a threshold value for evaluation) and overrides the paretoOptimization() and evaluationAndFeedback- Loop() methods to provide custom implementation based on the defined threshold.

5. Objective class represents an objective to optimize in the exercise selection. It has an attribute name (the name of the objective) and provides a method to access the name.

The FGPE+ concept adds gamification aspects throughout the app to make the learning experience more interesting. For completing workouts, attaining milestones, and obtaining high scores, users gain points, badges, and virtual gifts. This gamified method encour- ages competitiveness, incentive, and ongoing progress. The app monitors and shows the progress and performance data of users. Users may check their completion rates, accuracy, and time required to complete each activity. The model offers customized reports and insights to assist users in identifying their own strengths, shortcomings, and opportunities for progress. The FGPE+ paradigm encourages user social engagement. Users may join groups, participate on discussion boards, and work together to solve puzzles. Users may also compare their performance to that of others, encouraging healthy rivalry and infor- mation exchange. Users obtain fast feedback on their workout answers, which helps them understand and improve from their mistakes. The software offers thorough explanations, code critiques, and advice to help users improve their programming skills. Users may also seek assistance from mentors or experienced programmers within the app.

Here is the pseudo code of the FGPE+ approach:

Inputs: List of exercises (ExerciseList); exercise attributes (e.g., learning outcomes, difficulty levels); objective weights (e.g., learning outcomes, engagement).

Outputs: Pareto optimal exercise set (ParetoSet).

Procedure: 1. DefineExerciseSelectionCriteria(); 2. CollectExerciseData(); 3. ParetoOp-
timization(); 4. DiversityEnhancement(); 5. GamificationIntegration(); 6. Personalization();
7. EvaluationAndFeedbackLoop().

Procedure DefineExerciseSelectionCriteria(): Specify criteria for selecting exercises based on objectives and gamification elements; define attributes to consider, such as learn- ing outcomes, difficulty levels, or programming concepts.

Procedure CollectExerciseData(): Gather information on exercises, including attributes and gamification elements; store exercise data in a suitable data structure.

Procedure ParetoOptimization(): Apply multi-objective optimization algorithm (e.g., NSGA-II, SPEA2) on the exercise data; generate a set of Pareto optimal solutions considering the defined objectives and weights.

Procedure DiversityEnhancement(): Enhance diversity in the selected exercise set; apply niching or crowding techniques to avoid redundancy and promote variety.

Procedure GamificationIntegration(): Integrate gamification elements into the selected exercises; consider elements such as points, badges, levels, leaderboards, or social interac- tion features.

Procedure Personalization(): Allow learners to personalize exercise selection based on preferences, prior knowledge, or skill levels; implement adaptive algorithms for dynami- cally adjusting exercise difficulty or sequence.

Procedure EvaluationAndFeedbackLoop(): Continuously evaluate the effectiveness of the exercise selection model; collect user feedback, learning outcomes, and engagement metrics; refine and improve the algorithm based on evaluation results.

Output ParetoSet: Return the set of exercises representing the Pareto optimal solutions.

4. Results

4.1. Evaluation of the PWA Version of the FGPE PLE Platform by Mobile Device Users

As the main purpose of redeveloping the FGPE PLE platform as a PWA was to make it more suitable for mobile device users, the first part of its evaluation was aimed at assessing to what extent we have succeeded. Although this could be evaluated with both self-report and behavioral measures [76], following the findings of [77] showing that the interpretation of the latter is not always straightforward and could be misleading, we decided to go with the former.

Apart from assessing the general students' attitude to using the FGPE PLE platform on a mobile device (Q1), we also strived to assess whether the learning place makes a difference in the mobile use of the platform (Q2 and Q3), as well as whether the mobile users still feel the need to use the PC version at all (Q4):
(Q1): How do you generally rate the mobile version of the FGPE PLE platform?
Answer range: 1 (bad)–5 (excellent).
(Q2): Do you think the mobile version of the FGPE PLE platform makes sense for students learning at home?
Answer range: 1 (bad)–5 (excellent).
(Q3): Do you think the mobile version of the FGPE PLE platform makes sense for students who study on the go to school/work?
Answer range: 1 (bad)–5 (excellent).
(Q4): Do you think it is possible to learn to write code only using the mobile version of the FGPE PLE platform—and without using the PC version at all?
Answer range: 1 (bad)–5 (excellent).
The results are summarized in Figure 4. The answers to Q1 demonstrate the over- whelmingly positive response to the mobile version of FGPE PLE in our study. The answers to Q2 and Q3 indicate the students see it as a convenient learning tool at home and during commuting to school/work (more so for the latter than the former). The answers to Q4 show that the students do not believe that programming can be learned only by using FGPE PLE on their mobile devices.

4.2. User Experience Analysis

In order to obtain a more detailed picture of how the users view the mobile version of FGPE PLE, the User Experience Questionnaire (UEQ) has been employed [78]. It is a well-established instrument used to evaluate the user experience of a product or service, designed to provide developers with a quick and straightforward way to assess the user experience of their product, be it a website, a software application, or any other kind of interactive system. It has been previously used for The Evaluation of User Experience on Learning Management Systems [79]. The UEQ measures six different scales:

- Attractiveness: covers the overall impression of the product, including whether it is pleasant or enjoyable to use.
- Perspicuity: measures how easy it is for users to understand how to use the product.
- Efficiency: evaluates the perception of how efficiently users can complete tasks using the product.
- Dependability: measures how reliable and predictable users find the product.
- Stimulation: evaluates how exciting and motivating the product is to use.
- Novelty: assesses whether the design of the product is creative and innovative and whether it meets users' expectations.

The questionnaire itself consists of 26 pairs of opposing adjectives (such as "compli- cated" vs. "easy"), and respondents rate their experience with the product on a 7-point Likert scale between these extremes (see Figure 5). The scores from these pairs of ad- jectives are then used to calculate scores on the six scales listed above. This provides a comprehensive and nuanced understanding of how users perceive the product or service.

Students from Poland and Lithuania provided answers to the User Experience Ques- tionnaire (UEQ), reflecting how they used the FGPE+ model. Each input indicates a score for a particular aspect of the user experience, such as appeal, perspicacity, effectiveness, reliability, stimulation, and innovation (Figure 6). These replies come from many student groups; therefore, they represent the distinctive perspectives and experiences of these various cohorts.
The availability of data from two countries created an opportunity for a cross-country analysis. This analysis aimed at identifying discrepancies and/or parallels in the assessment that could be linked to differences in pedagogy and cultural backgrounds of the students taught in various countries. Here is a brief analysis of the results:

- Attractiveness: The Lithuanian group had a higher average score (mean = 1.4141, std = 0.6081) compared to the Polish group (mean = 0.6607, std = 0.5785). This indicates that the Lithuanian students found the learning environment more attractive and appealing than the Polish students.
- Perspicuity: The Lithuanian group also scored higher (mean = 1.4914, std = 0.7604) than the Polish group (mean = 0.5580, std = 0.7732), suggesting that the Lithuanian students found the learning environment more clear and understandable.
- Efficiency: Again, the Lithuanian group's score was higher (mean = 1.5193, std = 0.6714) than the Polish group (mean = 0.2727, std = 0.6650), indicating that the Lithuanian students found the learning environment more efficient for achieving their tasks.
- Dependability: The Lithuanian group had a higher mean score (mean = 1.2462, std = 0.8367) than the Polish group (mean = 0.7047, std = 0.7909), indicating they found the learning environment more reliable and dependable.
- Stimulation: The Lithuanian group scored slightly higher in this dimension (mean = 1.4972, std = 0.6408) than the Polish group (mean = 1.2548, std = 0.7053). This means that the Lithuanian students found the learning environment slightly more exciting and motivating.
- Novelty: Lastly, the Lithuanian group scored higher in terms of novelty (mean = 1.3701, std = 0.6361) compared to the

Polish group (mean = 0.5738, std = 0.6177). This suggests that the Lithuanian students found the learning environment more innovative and creative.
By comparing the data, it is evident that Lithuanian students often score higher on the UEQ than Polish students. Such results indicates that, globally, Lithuanian students may have found the FGPE PLE mobile version easier to use. However, it is important to note that a thorough analysis is difficult without a complete understanding of the issues that correlate to the presented data. It may simply reflect differing cultural views, educational backgrounds, or degrees of knowledge with comparable systems rather than necessarily implying that the approach featuring mobile learning supported with FGPE is more frequently accepted in Lithuania. The lower results in the Polish student group may indicate an opportunity for the further development of the FGPE PLE in some areas to make it more flexible and advantageous for a wider range of users, but it could also be a reflection of various standards or expectations.

4.3. Knowledge Evaluation Survey: FGPE Approach vs. Classic Moodle Course

Following the guidelines proposed in [80] regarding the evaluation of area-specific effects of gamification which suggest knowledge improvement as an indicator relevant for gamification applications aimed at supporting learning, we have used the opportunity that the Lithuanian group had a parallel group learning programming without the use of the FGPE toolset, to compare the two educational approaches.
A knowledge evaluation survey was conducted comparing two groups of learners: the first group used the PWA-based FGPE exercise selection model of the Python pro- gramming course, while the second one used the typical Moodle course format of Python programming (non-gamified, lecturer assigned, and ordered programming tasks). Table 1 demonstrates the perceived knowledge evaluation of groups using the FGPE+ model and the Moodle course, demonstrating the efficacy of the model. The 'N' column refers to the sample size for each group. 'M (SD)' represents the mean and standard deviation of the perceived knowledge scores. 't' and 'p' are the t-statistic and p-value, respectively, from a t-test comparing the game and Moodle course group scores. The FGPE+ course group had a higher average perceived knowledge score (M = 4.11, SD = 0.51) than the Moodle group (M = 3.67, SD = 0.56). The t-value indicates that the Moodle course group's score was higher, and the small p-value ($p < 0.05$) suggests this difference was statistically significant. This could imply that the FGPE+ model was effective in increasing the perceived knowledge.

4.4. Effectiveness of Using Sharable Content Object Reference Model

The Sharable Content Object Reference Model (SCORM) is a collection of standards and specifications for web-based educational content. It provides a standardized approach to creating and delivering online learning content, ensuring interoperability, accessibility, and reusability. SCORM has been widely adopted by e-learning providers as it ensures that learning content and Learning Management Systems (LMS) can work seamlessly together, regardless of the developer or platform. The results of evaluation according to SCORM criteria are presented in Figure 7). Data were collected using a descriptive survey following the practice established by [81], and assessment was conducted using their suggested quasi-experimental approach. ANCOVA results (FT = 3.76; FC = 8.11; p = 0.04) revealed a substantial difference in the impacts of SCORM-conformant e-content and conventional material on academic achievement.
Seventeen people (normalized) responded from the original groups of FPGE+ and Moodle users (see Table 1).

5. Discussion

We believe that PWAs have not yet gained full momentum in the education sector, and their use remains limited [82]. There is still somewhat of a scarcity of research and different uses of PWAs in education, demonstrating that the approach is not commonly recognized or used in the programming education arena [83]. Our Pareto-optimized approach enabled a personalized and adapted exercise selection to meet the specific needs and skill levels of individual learners, demonstrating that adaptive learning algorithms and techniques can dynamically adjust the difficulty, content, and feedback of gamified programming exercises based on learners' progress and performance. To address these issues, the Framework for Gamified Programming Education (FGPE) was developed. The FGPE framework includes requirement formulation, a collection of gamified activities, and the creation of supporting software.
We would like to bring up a few critical topics for reader consideration. The first step is to examine the long-term consequences. Longitudinal studies are needed to assess the long-term influence of gamified exercises on students' programming abilities and infor- mation retention. We investigate whether incorporating gamification into programming
instruction results in better long-term learning outcomes than standard techniques [84]. Fu- ture study should look into how various gamification tactics and mechanics affect students' motivation, perseverance, and pleasure in programming tasks. Longer research would also aid in assessing social interaction and cooperation, since including social components such as competition, collaboration, and peer evaluation might improve students' learning experiences and outcomes in the long run.

5.1. Importance of FGPE+ for STEM Education

The FGPE+ model's potential extends far beyond the realm of programming education and has significant implications for the broader STEM (Science, Technology, Engineering, and Mathematics) education landscape. In an era characterized by rapid advancements in technology and the increasing relevance of digital literacy, effective STEM education is crucial in order to allow for the break-out of current rigid education schemes [85]. The FGPE+ model, with its innovative blend of personalization, gamification, and mobile learning, is a robust tool for enhancing STEM learning experiences.

One of the central tenets of the FGPE+ model is its ability to tailor learning experiences to individual learners. This approach aligns well with the diverse nature of STEM education, where learners often come with varying levels of background knowledge and abilities. The FGPE+ model can accommodate these differences, making learning more effective and enjoyable. The adaptive algorithm can be extended beyond programming tasks to include other STEM-related exercises, such as solving mathematical problems or designing engineering solutions.

The incorporation of gamification techniques in the FGPE+ model is a significant asset for STEM education [86]. Gamification elements such as badges, points, leaderboards, and levels can make complex STEM concepts more engaging and accessible, thereby fostering a positive attitude towards these subjects. These elements can also promote healthy competition and motivation, encouraging students to continually improve their understanding and mastery of STEM subjects.

With the prevalence of mobile devices, the mobile-compatible nature of the FGPE+ model offers vast potential for remote and flexible learning. This aspect can break down barriers to STEM education, allowing learners to access resources and engage with STEM concepts anytime, anywhere. Mobile learning also aligns with the digital habits of the current generation, thereby increasing its effectiveness and appeal.

The FGPE+ model's use of real-world tasks mirrors the application-based learning that is critical in STEM fields. By engaging with tasks that reflect real-world challenges, students can gain a deeper understanding of the practical applications of their learning, making the learning process more meaningful and relevant.

In conclusion, the FGPE+ model is a significant tool that can transform STEM educa- tion. By making learning more personalized, engaging, flexible, and application-focused, it aligns with the goals of modern STEM education, promoting increased participation and achievement in these crucial fields. Future research and development efforts should consider how the principles of FGPE+ can be effectively integrated and implemented across various STEM disciplines.

5.2. Limitations

Despite the potential contributions, this study has certain limitations that should be considered:

- This study used a subset of students from Poland and Lithuania, which may not be typical of the whole population. We believe that extending the study's size and altering the demographics of the participants would potentially increase generalizability.
- This research relied heavily on self-report measures, which are subjective and prone to bias. Incorporating objective measures, such as performance-based assessments or tracking system data, could provide a more comprehensive evaluation of the platform's effectiveness.

- The study aimed to assess the FGPE PLE platform in various educational settings and learning scenarios. The findings may not fully represent the intricacies and complexity of various educational settings. Future study might look at the platform's efficacy in other educational institutions, student backgrounds, and instructional environments.
- The PWA version of the FGPE PLE platform was assessed particularly for program- ming instruction in the research. The findings may not be applicable in other domains or topic areas. Replicating the study in additional educational fields would be advan- tageous in determining the platform's generalizability.
- The investigation focused on the initial user experience and perceived knowledge. Understanding the long-term impact of the FGPE PLE platform on learners' pro- gramming skill development and information retention would necessitate additional research outside the scope of this study.

5.3. Potential Lines of Research

The study on the evaluation of the PWA version of the FGPE PLE platform opened up several potential lines of research:

- We concentrated on evaluating the platform's initial user experience and perceived knowledge. More study might be conducted to investigate the long-term consequences of utilizing the FGPE PLE platform on mobile devices. Longitudinal studies might look into the long-term influence on learning outcomes, programming skill development, and knowledge retention.
- In terms of perceived knowledge, we compared the FGPE+ model to a traditional Moodle course. Future study might compare the efficacy of other educational method- ologies, such as gamified platforms such as FGPE+ vs. traditional teaching methods. Comparative research might assist to uncover the advantages and disadvantages of each strategy and provide ideas into how to improve learning experiences.
- The platform was evaluated mostly using self-report measures in this study. Tracking user interactions, completion rates, and performance statistics, for example, might give a more objective evaluation of learners' progress and engagement with the platform if learning analytics approaches are used. Analyzing such data might aid in the discovery of trends, the identification of areas for development, and the implementation of individualized learning interventions.

- Further study might look into the efficacy of certain instructional tactics used inside the FGPE PLE platform. Investigating how various gamification aspects, adaptive learning algorithms, or social interaction features influence motivation, engagement, and learning results might provide useful insights for building and enhancing educa- tional systems.
- Our investigation discovered some cross-national disparities in user experience per- ceptions. More thorough cross-cultural research might provide insight on how cultural backgrounds and educational environments impact FGPE PLE platform acceptability and efficacy. Understanding these cultural differences may help with the customiza- tion and localization of educational systems for varied learner groups.

6. Conclusions

This study explored the use of the FGPE+ model, a Pareto-optimized gamified pro- gramming exercise selection system, in a mobile learning context. The FGPE+ system, by integrating principles of Pareto optimization and gamification in a mobile-compatible platform, offered a unique, personalized, and engaging learning environment for pro- gramming students. The FGPE+ model's clean and user-friendly interface was another aspect that stood out, enabling learners to navigate and interact with the platform easily. The PWA-based system allowed learners to carry their learning environment with them, enhancing accessibility and convenience. The students appreciated the tailored exercise rec- ommendations, adaptive difficulty level adjustments, and gamification elements that kept them motivated to learn continually. The study also shed light on the effectiveness of the model in catering to diverse learners, including those who preferred different programming languages and had varying levels of skills and interests.

The overwhelmingly positive response to the FGPE+ model in our study is a promis- ing step towards transforming programming education, paving the way for an array of exciting future research opportunities. One immediate avenue for future research is to expand the scope of this investigation beyond the initial Polish and Lithuanian samples. Conducting comparative studies across different countries and cultures will provide a more comprehensive understanding of FGPE+'s cross-cultural efficacy and adaptability. This global approach can also reveal unique regional requirements or preferences, which can be integrated into the FGPE+ model to create a more universally effective learning platform. The current study focused on the learner's perspective. However, insights from educators who utilize FGPE+ could offer a different perspective, providing additional ways to improve and optimize the system. They could share valuable input on what works well in a classroom setting, potential areas of difficulty, or suggestions to improve learner engagement.

Additionally, the development of more sophisticated adaptive algorithms that lever- age artificial intelligence (AI) and machine learning (ML) techniques could augment the FGPE+'s capabilities. These techniques could further personalize the learning experience, making it more responsive to individual learner's needs. For instance, the system could predict what a student might struggle with based on historical data and preemptively provide resources to mitigate these challenges [87]. Incorporating Virtual Reality (VR) or Augmented Reality (AR) could also enhance the FGPE+ model's immersive learning experiences. VR/AR could be used to simulate real-world programming scenarios, making abstract programming concepts more tangible and engaging for learners.

The FGPE+ system, although robust, currently supports only a limited set of pro- gramming languages. Extending its support to encompass a broader range of languages, including emerging ones, will make it more versatile and valuable to a wider audience. Finally, we observe potential in exploring the impact of different gamification elements on learning outcomes. While FGPE+ currently uses a set of gamification techniques, under- standing which elements are most effective can help refine the system to maximize learner engagement and achievement.

In conclusion, the FGPE+ system is a promising approach to modernize programming education. By employing principles of gamification and Pareto optimization in a PWA platform, it provides an engaging, adaptive, and personalized learning experience. We believe that the positive results obtained highlight the system's potential for broader application. We anticipate that FGPE+ can be adapted to diverse educational contexts, playing a significant role in programming education across various age groups, skill levels, and cultural settings. However, more extensive and diverse studies are needed to validate these findings and refine the model to cater to a broader range of learners. By continuing to innovate and push boundaries, we believe that FGPE+ has the potential to revolutionize programming education. Through future research, the model could be further refined to make learning programming more accessible, engaging, and effective for all.

References

1. Mishra, L.; Gupta, T.; Shree, A. Online teaching-learning in higher education during lockdown period of COVID-19 pandemic.
Int. J. Educ. Res. Open 2020, 1, 100012. [CrossRef]
2. Breiki, M.A.; Yahaya, W.A.J.W. Using Gamification to Promote Students' Engagement While Teaching Online During COVID-19. In Teaching in the Post COVID-19 Era; Springer International Publishing: Cham, Switzerland, 2021; pp. 443–453. [CrossRef]
3. Pedro, L.; Santos, C. Has Covid-19 emergency instruction killed the PLE? In Proceedings of the Ninth International Conference on Technological Ecosystems for Enhancing Multiculturality (TEEM'21), Barcelona, Spain, 26–29 October 2021; ACM: New York, NY, USA, 2021. [CrossRef]
4. Redondo, R.P.D.; Rodríguez, M.C.; Escobar, J.J.L.; Vilas, A.F. Integrating micro-learning content in traditional e-learning platforms.
Multimed. Tools Appl. 2020, 80, 3121–3151. [CrossRef]
5. Jayalath, J.; Esichaikul, V. Gamification to Enhance Motivation and Engagement in Blended eLearning for Technical and Vocational Education and Training. Technol. Knowl. Learn. 2020, 27, 91–118. [CrossRef]
6. da Silva, J.P.; Silveira, I.F. A systematic review on open educational games for programming learning and teaching. Int. J. Emerg. Technol. Learn. 2020, 15, 156–172. [CrossRef]
7. Paiva, J.C.; Leal, J.P.; Queirós, R. Fostering programming practice through games. Information 2020, 11, 498. [CrossRef]
8. Maryono, D.; Budiyono, S.; Akhyar, M. Implementation of Gamification in Programming Learning: Literature Review. Int. J. Inf. Educ. Technol. 2022, 12, 1448–1457. [CrossRef]
9. Maskeliu¯ nas, R.; Kulikajevas, A.; Blažauskas, T.; Damaševicˇius, R.; Swacha, J. An interactive serious mobile game for supporting the learning of programming in javascript in the context of eco-friendly city management. Computers 2020, 9, 102. [CrossRef]
10. Cuervo-Cely, K.D.; Restrepo-Calle, F.; Ramírez-Echeverry, J.J. Effect Of Gamification On The Motivation Of Computer Program- ming Students. J. Inf. Technol. Educ. Res. 2022, 21, 001–023. [CrossRef]
11. Chinchua, S.; Kantathanawat, T.; Tuntiwongwanich, S. Increasing Programming Self-Efficacy (PSE) Through a Problem-Based Gamification Digital Learning Ecosystem (DLE) Model. J. High. Educ. Theory Pract. 2022, 22, 131–143.
12. Ašeriškis, D.; Damaševicˇius, R. Gamification Patterns for Gamification Applications. Procedia Comput. Sci. 2014, 39, 83–90. [CrossRef]
13. Panskyi, T.; Rowin´ska, Z. A Holistic Digital Game-Based Learning Approach to Out-of-School Primary Programming Education.
Inform. Educ. 2021, 20, 1–22. [CrossRef]
14. Swacha, J.; Maskeliu¯ nas, R.; Damaševicˇius, R.; Kulikajevas, A.; Blažauskas, T.; Muszyn´ska, K.; Miluniec, A.; Kowalska, M. Introducing sustainable development topics into computer science education: Design and evaluation of the eco jsity game. Sustainability 2021, 13, 4244. [CrossRef]
15. Damaševicˇius, R.; Maskeliu¯ nas, R.; Blažauskas, T. Serious Games and Gamification in Healthcare: A Meta-Review. Information
2023, 14, 105.
16. Francillette, Y.; Boucher, E.; Bouchard, B.; Bouchard, K.; Gaboury, S. Serious games for people with mental disorders: State of the art of practices to maintain engagement and accessibility. Entertain. Comput. 2021, 37, 100396. [CrossRef]
17. Zhao, D.; Muntean, C.H.; Chis, A.E.; Rozinaj, G.; Muntean, G. Game-Based Learning: Enhancing Student Experience, Knowledge Gain, and Usability in Higher Education Programming Courses. IEEE Trans. Educ. 2022, 65, 502–513. [CrossRef]
18. Mohanarajah, S.; Sritharan, T. Shoot2learn: Fix-And-Play Educational Game For Learning Programming; Enhancing Student Engagement By Mixing Game Playing And Game Programming. J. Inf. Technol. Educ. Res. 2022, 21, 639–661. [CrossRef] [PubMed]
19. Xinogalos, S.; Satratzemi, M. The Use of Educational Games in Programming Assignments: SQL Island as a Case Study. Appl. Sci.
2022, 12, 6563. [CrossRef]
20. Barmpakas, A.; Xinogalos, S. Designing and Evaluating a Serious Game for Learning Artificial Intelligence Algorithms: SpAI War as a Case Study. Appl. Sci. 2023, 13, 5828. [CrossRef]
21. Costa, J.M. Using game concepts to improve programming learning: A multi-level meta-analysis. Comput. Appl. Eng. Educ. 2023,
31, 1098–1110. [CrossRef]
22. Soboleva, E.V.; Suvorova, T.N.; Grinshkun, A.V.; Bocharov, M.I. Applying Gamification in Learning the Basics of Algorithmization and Programming to Improve the Quality of Students' Educational Results. Eur. J. Contemp. Educ. 2021, 10,

987–1002.
23. Toda, A.M.; Valle, P.H.D.; Isotani, S. The Dark Side of Gamification: An Overview of Negative Effects of Gamification in Education. In Communications in Computer and Information Science; Springer International Publishing: Cham, Switzerland, 2018; pp. 143–156. [CrossRef]
24. Imran, H. An Empirical Investigation of the Different Levels of Gamification in an Introductory Programming Course. J. Educ. Comput. Res. 2022, 61, 847–874. [CrossRef]

25. Chatterjee, S.; Majumdar, D.; Misra, S.; Damaševicˇius, R. Adoption of mobile applications for teaching-learning process in rural girls' schools in India: An empirical study. Educ. Inf. Technol. 2020, 25, 4057–4076. [CrossRef]
26. Tuparov, G.; Keremedchiev, D.; Tuparova, D.; Stoyanova, M. Gamification and educational computer games in open source learning management systems as a part of assessment. In Proceedings of the 2018 17th International Conference on Information Technology Based Higher Education and Training (ITHET), Olhao, Portugal, 26–28 April 2018; pp. 1–5. [CrossRef]
27. Pérez-Berenguer, D.; García-Molina, J. A standard-based architecture to support learning interoperability: A practical experience in gamification. Software Pract. Exp. 2018, 48, 1238–1268. [CrossRef]
28. Calle-Archila, C.R.; Drews, O.M. Student-Based Gamification Framework for Online Courses. In Communications in Computer and Information Science; Springer International Publishing: Cham, Switzerland, 2017; pp. 401–414. [CrossRef]
29. Sheppard, D. Introduction to Progressive Web Apps. In Beginning Progressive Web App Development; Apress: New York, NY, USA, 2017; pp. 3–10. [CrossRef]
30. Hajian, M. PWA with Angular and Workbox. In Progressive Web Apps with Angular; Apress: New York, NY, USA, 2019; pp. 331–345. [CrossRef]
31. Devine, J.; Finney, J.; de Halleux, P.; Moskal, M.; Ball, T.; Hodges, S. MakeCode and CODAL: Intuitive and efficient embedded systems programming for education. J. Syst. Archit. 2019, 98, 468–483. [CrossRef]
32. Lee, J.; Kim, H.; Park, J.; Shin, I.; Son, S. Pride and Prejudice in Progressive Web Apps. In Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security, Toronto, ON, Canada, 15–19 October 2018; ACM: New York, NY, USA, 2018. [CrossRef]
33. FGPE PLE Environment. Available online: https://github.com/FGPE-Erasmus/fgpe-ple-v2 (accessed on 10 June 2023).
34. Sutadji, E.; Hidayat, W.N.; Patmanthara, S.; Sulton, S.; Jabari, N.A.M.; Irsyad, M. Measuring user experience on SIPEJAR as e-learning of Universitas Negeri Malang. IOP Conf. Ser. Mater. Sci. Eng. 2020, 732, 012116. [CrossRef]
35. Nah, F.F.H.; Zeng, Q.; Telaprolu, V.R.; Ayyappa, A.P.; Eschenbrenner, B. Gamification of Education: A Review of Literature. In
Lecture Notes in Computer Science; Springer International Publishing: Cham, Switzerland, 2014; pp. 401–409. [CrossRef]
36. Barik, T.; Murphy-Hill, E.; Zimmermann, T. A perspective on blending programming environments and games: Beyond points, badges, and leaderboards. In Proceedings of the 2016 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC), Cambridge, UK, 4–8 September 2016; pp. 134–142. [CrossRef]
37. Prokhorov, A.V.; Lisovichenko, V.O.; Mazorchuk, M.S.; Kuzminska, O.H. Developing a 3D quest game for career guidance to estimate students' digital competences. CEUR Workshop Proc. 2020, 2731, 312–327. . [CrossRef]
38. Padilla-Zea, N.; Gutiérrez, F.L.; López-Arcos, J.R.; Abad-Arranz, A.; Paderewski, P. Modeling storytelling to be used in educational video games. Comput. Hum. Behav. 2014, 31, 461–474. [CrossRef]
39. Hadzigeorgiou, Y. Narrative Thinking and Storytelling in Science Education. In Imaginative Science Education; Springer Interna- tional Publishing: Cham, Switzerland, 2016; pp. 83–119. [CrossRef]
40. Kusuma, G.P.; Wigati, E.K.; Utomo, Y.; Suryapranata, L.K.P. Analysis of Gamification Models in Education Using MDA Framework. Procedia Comput. Sci. 2018, 135, 385–392. [CrossRef]
41. Wu, M.; Liao, C.C.; Chen, Z.H.; Chan, T.W. Designing a Competitive Game for Promoting Students' Effort-Making Behavior by Virtual Pets. In Proceedings of the 2010 Third IEEE International Conference on Digital Game and Intelligent Toy Enhanced Learning, Kaohsiung, Taiwan, 12–16 April 2010; pp. 234–236. [CrossRef]
42. Chen, Z.H.; Liao, C.; Chien, T.C.; Chan, T.W. Animal companions: Fostering children's effort-making by nurturing virtual pets.
Br. J. Educ. Technol. 2009, 42, 166–180. [CrossRef]
43. Slavin, R.E. Cooperative Learning: Applying Contact Theory in Desegregated Schools. J. Soc. Issues 1985, 41, 45–62. [CrossRef]
44. Zakaria, E.; Iksan, Z. Promoting Cooperative Learning in Science and Mathematics Education: A Malaysian Perspective.
EURASIA J. Math. Sci. Technol. Educ. 2007, 3, 35–39. [CrossRef]
45. Correia, A.; Fonseca, B.; Paredes, H.; Martins, P.; Morgado, L. Computer-Simulated 3D Virtual Environments in Collaborative Learning and Training: Meta-Review, Refinement, and Roadmap. In Progress in IS; Springer International Publishing: Cham, Switzerland, 2016; pp. 403–440. [CrossRef]