Managing Gamified Programming Courses with the FGPE Platform

Article *in* Information (Switzerland) · January 2022

DOI: 10.3390/info13020045

5 authors, including:

José Carlos Paiva
Institute for Systems and Computer Engineering, Technology and Science (INESC …
23 PUBLICATIONS **161** CITATIONS
SEE PROFILE

Ricardo Queiros
Polytechnic Institute of Porto
113 PUBLICATIONS **633** CITATIONS
SEE PROFILE

José Paulo Leal
University of Porto
133 PUBLICATIONS **1,115** CITATIONS
SEE PROFILE

Jakub Swacha
University of Szczecin
85 PUBLICATIONS **517** CITATIONS
SEE PROFILE

information

Article

Managing Gamified Programming Courses with the FGPE Platform

José Carlos Paiva [1,2,*], Ricardo Queirós [1,3], José Paulo Leal [1,2], Jakub Swacha [4,*] and Filip Miernik [4]

[1] CRACS—INESC Porto LA, 4169-007 Porto, Portugal; ricardoqueiros@esmad.ipp.pt (R.Q.); zp@dcc.fc.up.pt (J.P.L.)
[2] Department of Computer Science, Faculty of Sciences, University of Porto, 4169-007 Porto, Portugal
[3] uniMAD—ESMAD, Polytechnic of Porto, 4480-876 Vila do Conde, Portugal
[4] Department of Information Technology in Management, University of Szczecin, 70-453 Szczecin, Poland; filip@flexile.io
* Correspondence: jose.c.paiva@inesctec.pt (J.C.P.); jakub.swacha@usz.edu.pl (J.S.)

Abstract: E-learning tools are gaining increasing relevance as facilitators in the task of learning how to program. This is mainly a result of the pandemic situation and consequent lockdown in several countries, which forced distance learning. Instant and relevant feedback to students, particularly if coupled with gamification, plays a pivotal role in this process and has already been demonstrated as an effective solution in this regard. However, teachers still struggle with the lack of tools that can adequately support the creation and management of online gamified programming courses. Until now, there was no software platform that would be simultaneously open-source and general-purpose (i.e., not integrated with a specific course on a specific programming language) while featuring a meaningful selection of gamification components. Such a solution has been developed as a part of the Framework for Gamified Programming Education (FGPE) project. In this paper, we present its two front-end components: FGPE AuthorKit and FGPE PLE, explain how they can be used by teachers to prepare and manage gamified programming courses, and report the results of the usability evaluation by the teachers using the platform in their classes.

Keywords: learning environment; programming exercises; gamification; programming learning; automatic assessment; usability evaluation

check for updates

Citation: Paiva, J.C.; Queirós, R.; Leal, J.P.; Swacha, J.; Miernik, F. Managing Gamified Programming Courses with the FGPE Platform. *Information* **2022**, *13*, 45. https://doi.org/10.3390/info13020045

Academic Editor: Christo Dichev

Received: 30 November 2021
Accepted: 17 January 2022
Published: 19 January 2022

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.

1. Introduction

There is a major concern among educational researchers with the disengagement of the students from the learning activities, which frequently leads to academic failure and, lastly, dropout. This issue is even more noticeable in distance learning [1], which became the default way of learning in many countries in times of the current pandemic. One way to counteract this problem is through gamification, which has been proven as a capable tool to reduce the dropout rate [2].

Feedback is considered as one of the three high-level concepts (together with competition and cooperation) essential for gamification [3]; however, even apart from gamification, it has been considered for a long time as an important element of successful knowledge and skill acquisition [4,5]. In programming education, fortunately, instant feedback can be generated automatically based on the submission code, providing the student with relevant information about general concepts, task constraints, mistakes made, hints on how to proceed, or even meta-cognition [6].

The combined use of automated assessment, which provides instant feedback to the students experimenting with their code, and gamification, which provides additional motivation for the students to intensify their learning effort, can help students to overcome the barrier of difficulty in acquiring computer programming skills. Its practical application, however, requires a number of interconnected tools allowing the programming teachers

to prepare and manage gamified courses. OneUp [7], a customizable platform aimed at facilitating the process of gamifying academic courses, has been developed as a response to these challenges in education in general (i.e., not only programming). Unfortunately, to the best of the authors' knowledge, (1) it remains a closed prototype (source code not available and access to a live deployment is only granted by request), (2) formats for sharing and reusing exercise packages and gamification layers are not public (if they exist), and (3) supported gamification elements could be enriched (i.e., more than points, badges, leaderboards, levels, and virtual currency).

Until now, there was no platform of this kind (i.e., for providing gamified programming courses with automatic assessment) that would be simultaneously: open source (rather than commercial), general purpose (rather than integrated with a specific course on a specific programming language), and featuring a meaningful selection of gamification components (rather than merely awarding points and badges). Developing such a fully customizable platform was the aim of the Framework for Gamified Programming Education (FGPE) international project realized by partners from Poland, Portugal, Denmark, and Italy, and financially supported from the Erasmus+ program [8]. In this paper, we describe two key software components of the platform—FGPE AuthorKit and FGPE PLE—and explain how these can be used by teachers to prepare and manage gamified programming courses. Furthermore, we report the conducted study for evaluating the usability of the platform from the teachers' perspective and present its results.

The rest of this paper is organized as follows. Section 2 reviews the current state of gamification in education, focusing on empirical studies describing experiments that applied game elements into a learning environment. Section 3 presents the architecture of the FGPE Platform, and explains the role of its key components. Section 4 describes the purpose and design of the FGPE AuthorKit tool, and explains how it can be used to prepare programming exercises with an attached gamification layer. Section 5 presents the FGPE Programming Learning Environment, and describes its functionality in student and teacher modes. Section 6 describes the usability evaluation of the platform from the teachers' perspective. Finally, Section 7 summarizes the contributions of this work and indicates some future directions.

2. Related Work

Gamification of education has been a top trend in the last years [9], which can be explained by the positive boost in motivation and user experience it is accredited with. There are several case studies in the literature, particularly performed in high school and universities, covering different learning subjects, ranging from Computer Science/Information Technology (CS/IT) [10–16] to Mathematics (Maths) [17–20], Foreign Languages (FL) [21,22], Communication and Multimedia (C&M) [23–26], and Medicine/Biology (M/B) [27–29]. To better understand the effects of gamification in learning, a search for experimental studies, between 2014 and 2018 (start date of FGPE project), that apply gamification methods in educational e-learning environments was conducted. To this end, we have queried Google Scholar and ScienceDirect databases using combinations of the following keywords: gamification, serious games, education, and learning. From the returned results, only the first 100 studies by relevance from each database were considered. These 200 results were further narrowed down by applying the following inclusion criteria:

1. Have reported findings in high school and university subjects of the previously mentioned areas,
2. Research methods are explained.

The result was a set of about 50 empirical research studies, of which 20 were selected according to their relevance. Table 1 presents a comparative study of the collected research papers, regarding the applied game elements.

Table 1. Game elements reported in the selected research papers.

	CS/IT								Maths			FL		C&M				M/B	
Game Elements	[10]	[11]	[12]	[13]	[14]	[15]	[16]	[17]	[18]	[19]	[20]	[21]	[22]	[23]	[24]	[25]	[26]	[27]	[28]	[29]
Badges	✓	✓	✓	✓	✓	✓	✓		✓			✓	✓	✓	✓	✓			✓	
Collaboration	✓			✓			✓		✓				✓	✓	✓	✓		✓	✓	
Direct Competition [1]							✓													
Content Disclosure		✓		✓		✓	✓							✓						
Exhaustible Resources								✓			✓	✓		✓	✓		✓			
Leaderboards	✓	✓		✓		✓			✓	✓		✓		✓	✓	✓		✓	✓	
Levels							✓		✓				✓	✓		✓	✓		✓	
Points [2]	✓	✓	✓	✓	✓	✓	✓	✓	✓		✓	✓	✓	✓		✓	✓	✓	✓	
Progress Indicator	✓	✓		✓								✓		✓		✓	✓			✓
Quests							✓						✓	✓		✓				✓
Real-world Rewards																		✓		
Serious Games/Immersion							✓	✓		✓			✓	✓						✓
SLP [3]			✓			✓						✓		✓	✓					
Status	✓						✓		✓									✓		

[1] Only core competition, i.e., one player (or team) plays against other, if one wins, the other loses. [2] Includes experience points (XP), skill points (score), influence points (rating, reputation), and automatic grades. [3] Similar Learning Path (SLP) consists of using data from other students to predict and display the current progress of a student (not leaderboards).

From this study, we can observe that points, badges, and leaderboards (PBL) are the most applied game elements. This was expected as those are the easiest elements to reuse in distinct educational scenarios, having multiple general-purpose game backend-as-a-service providers offering them as a plug-and-play API. In contrast, only a few works use, for instance, quests (5), content disclosure (5), status (4), and direct competition (1) as these are harder to implement and less reusable. Therefore, teachers aiming to provide a meaningful gamification layer on top of the course contents they teach, either opt for the "all-round" PBL-based gamification or have to implement a new environment from scratch.

Surprisingly, even though there is much more work on gamification of CS/IT education compared to other areas of education, no significant difference in game elements could be found other than the use of direct competition. Despite the great interest and, typically, the existence of a more adequate environment in this field to run complex scenarios, the cost of recreating the necessary tools for slightly different contexts and gamification elements hinders their application. This emphasizes the importance of tools/services supporting customizable gamification in education with more impactful game elements.

Regarding the experiments, the vast majority of the studies compare a control group against the experimental group. Both groups have the same background and face the same (or very similar) educational activities, but the former has no contact with gamification whereas the latter interacts with a gamified environment. The reported effects on students are mostly slightly positive or neutral, with a few exceptions [19,24]. Nevertheless, in the area of CS/IT, the results are overall positive, i.e., they either promote engagement, participation, dedicated time, or learning outcome. As expected, more advanced game elements tend to achieve better results than those relying solely on PBL-based gamification [16,23].

3. FGPE Platform Architecture

The FGPE Platform is a web-based environment that aims to deliver a single interface to a complete ecosystem for gamified programming education.

As can be observed in Figure 1, the FGPE ecosystem includes the following key components:

- FGPE AuthorKit [30], a web tool for preparing and managing the programming exercises and gamification rules,
- An open repository (hosted at GitHub), where the gamified programming exercises are stored,

- Gamified programming exercises—the educational contents represented using two formats, one for describing programming exercises—YAPExIL [31]—and another for gamification layers—GEdIL [32],
- FGPE Gamification Service [33], which processes gamification rules and keeps the game state,
- Mooshak 2.0 [34], the sandbox for execution of the programs submitted by students and automatically assessing their correctness,
- FGPE Programming Learning Environment (PLE) [35], providing the user interface for both students (to interact with gamified programming exercises, edit, and test their solutions) and teachers (to arrange exercise sets for their students and follow the students' progress with the exercises).

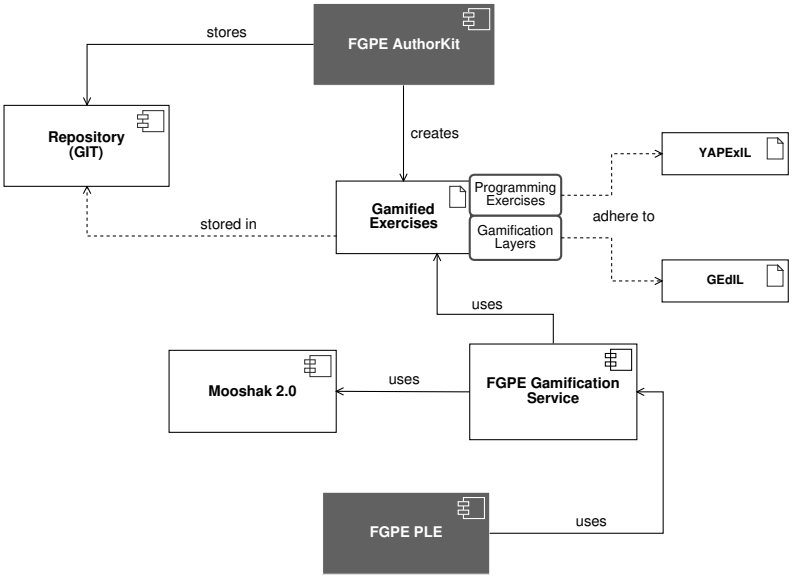

Figure 1. Architecture of the FGPE ecosystem, highlighting the FGPE AuthorKit and FGPE PLE.

In the following two sections, we shall present the two components that are essential for managing the gamified programming courses with the FGPE platform, namely AuthorKit and PLE. This does not mean that the remaining software components are not important, but that they are not used directly by the user doing the management job.

The automated evaluation engine (Mooshak 2.0 [34]) guarantees accurate and timely feedback to students. Such an engine is responsible for marking and grading exercises. In this ecosystem, an evaluation engine will: (1) receive the identification of the student, a reference to the exercise, and the student's attempt to solve it; (2) load the exercise by reference from the repository (if necessary); (3) compile the solution and run the tests against the student's program, comparing the obtained output to the expected; and (4) build report of the evaluation with passed/failed test cases, the grade, and, possibly, some feedback. An important feature of Mooshak 2 is that it supports custom static and dynamic analysis scripts, which enables assessment for the non-traditional programming exercises.

The FGPE Gamification Service [33] is a GraphQL service designed to consume a gamification layer defined using GEdIL, and transform it into a game, played by students. Hence, the service has complete support for various rewarding mechanisms such as points, badges, coupons, virtual items, leaderboards, locked and secret content, and different activity modes (e.g., time-bomb and speedup). Furthermore, the FGPE Gamification Service is the single point of access of the FGPE PLE to this ecosystem. Consequently, it should not only apply gamification rules, but also deliver the challenge and activity statements as well as handle the automated assessment of activities. To this end, the service uses plugins, i.e., consumers of the evaluation engine APIs, that leverage the connected evaluation engine, which is adequate to assess given type of activity.

4. AuthorKit

4.1. Purpose and Design

From the teachers' perspective, the most daunting task in conducting an online programming course is its creation, including either authoring new programming exercises or selecting and adapting existing exercises available from past years. Before FGPE, the latter suffered from (1) the lack of a standard format to describe programming exercises and (2) the absence of an open repository of programming exercises from which teachers could choose and import. Introducing gamification makes the task even more challenging, as a strategy to combine exercises and gamification elements in an engaging way needs to be designed and implemented. The first gap has been addressed by the introduction of GEdIL [32], and the second with FGPE AuthorKit.

FGPE AuthorKit is a web-based application with a dedicated API that provides authors with a wizard-based editor to develop programming exercises featuring gamification elements that boost students' motivation to increase their learning effort. The core functionalities of the tool include (1) a wizard for creating programming exercises and their associated metadata, (2) a wizard for selecting and tuning adequate gamification techniques for a specific exercise or collection, (3) a tree-based component to design the content structure and add sequencing rules, and (4) mechanisms to import and export the content in the formats of choice. Moreover, authored educational content is automatically synchronized with GitHub and can be shared internally with other peers.

4.2. Preparing Programming Exercises

Effective gamification needs to be justifiable, adequate, and easy to set up. Implementing gamification in a specific course is, thus, a thorough process. Separating the gamification layer from the educational content can significantly alleviate this task, allowing to reuse and adapt either the solicitously designed gamification rules, educational activities, or both. Let us remind that before the introduction of the FGPE platform, one could either use existing gamified courses and platforms or develop a new platform, as there was no open repository of programming exercises with connected gamification rules nor open platforms to handle them.

In the FGPE platform, the process of authoring gamified programming exercises is divided for simplicity and reusability, allowing the user to import content from popular non-gamified exercise formats (e.g., past courses run on Mooshak [34]) and wire up gamification later. To this end, two dedicated formats are employed, Yet Another Programming Exercises Interoperability Language (YAPExIL) [31] and Gamified Education Interoperability Language (GEdIL) [32]. The former describes programming exercises without gamification, comprising several types of activities, which will be described further on. The latter represents the set of gamification concepts and rules applied to activities, separately from their actual educational content.

AuthorKit supports all types of programming exercises that can be specified using YAPExIL, including `BLANK_SHEET`, which asks the students to develop their solution from scratch; `EXTENSION`, which presents a partial solution for the student to complete; `IMPROVEMENT`, which provides a correct initial source code that needs some optimization; `BUG_FIX`, that gives a buggy solution to the student who has to correct it; `FILL_IN_GAPS`, which provides code with missing parts; `SPOT_BUG`, that asks students to merely indicate the location of the bugs in a provided solution; and `SORT_BLOCKS`, which asks students to sort several blocks of code to form a working solution. To this end, it allows the author to develop multiple solutions, skeletons, test cases, source code and behavior checkers, and statements per exercise.

The user interface for developing programming exercises, presented in Figure 2, is a multi-step wizard that maps each facet of YAPExIL to a step of the wizard. Hence, it contains four steps:

1. Metadata, which allows the author to enter information about the exercise such as the name, keywords, programming languages, among others;

2. Presentation, which includes files that are presented to either the student or the teacher (e.g., problem statements, instructions, and skeletons);
3. Evaluation, which encompasses files that enter in the evaluation phase (e.g., tests, solutions, and output/behavior checkers);
4. Tools, which contains executable scripts that complement the exercise (e.g., test generators).

Each resource field of the form (e.g., test) has an associated sidebar where the author can enter its information (e.g., weight, arguments, and feedback messages) and upload related files (e.g., input and output text files).

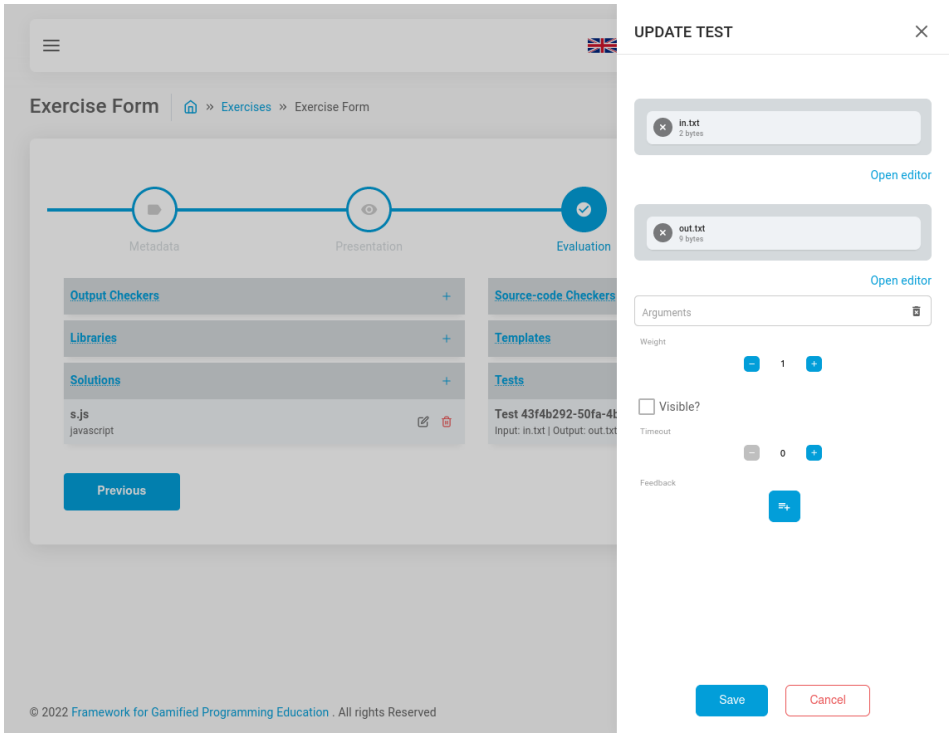

Figure 2. Screenshot of FGPE AuthorKit while editing a test on the exercise wizard.

4.3. Adding Gamification

After completing the definition of activities planned for students, authors can devise and attach gamification elements, both to single activities and groups of them, defining gamification layers adhering to GEdIL [32]. Even though GEdIL does not target a specific subject, it was designed to cover a vast collection of rewarding mechanisms, such as points, badges, virtual items, and leaderboards to provide extrinsic motivation, as well as unlockable and secret content, different activity modes (e.g., speedup and duels), among others.

The user interface to author the gamification layers, depicted in Figure 3, has five steps according to GEdIL constituents, one for the metadata and one for each main gamification element of GEdIL, i.e., rewards, rules, leaderboards, and challenges. The metadata step allows defining metadata of the gamification layer, including the name, description, keywords, and status. The rewards step is where the author can specify rewards to be delivered based on rules. Here, the type of reward (i.e., badge, point, virtual item, coupon, revealable, unlockable, hint, and message) can be chosen and the respective information provided (e.g., upload an image for badges, indicate the number of points, etc.). Rules are defined in the third step, requiring action or time-based triggers, criteria for application, and the actions to execute, if applicable. The fourth step allows the author to define global leaderboards, only by specifying their name and metrics. Finally, the challenge step contains an editable tree component, in which intermediate nodes are the challenges and leaf nodes correspond to local-scoped rules, rewards, and leaderboards. Challenges wrap one or more activities

with gamification attributes for locking, hiding, and tuning its mode (e.g., limiting activity to a certain length of time).

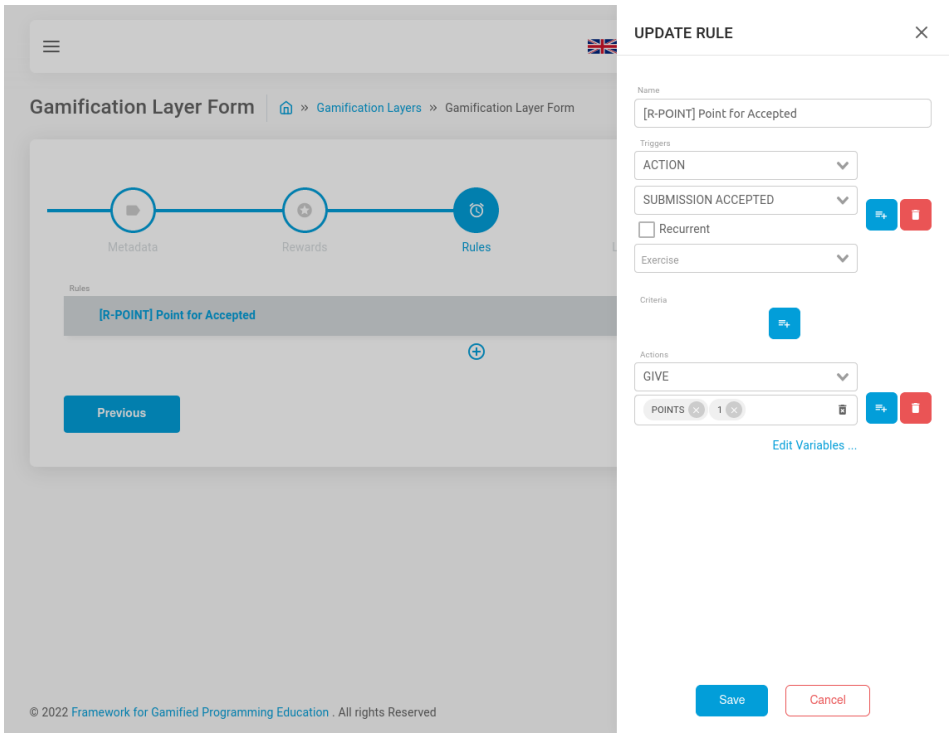

Figure 3. Screenshot of FGPE AuthorKit while editing a rule on the gamification layer wizard.

4.4. Managing and Sharing Projects

Projects are the shareable units of AuthorKit and enclose related educational content (e.g., exercises about machine learning), confining the scope of relationships between gamification layers and activities. Users can share projects with other peers either internally, by inviting them through their accounts' email to participate in the project, or exposing the project publicly for them to export. In addition to that, each project has a corresponding remote git repository, to which all its contents are synchronized.

The user interface organizes programming exercises and gamification layers inside projects. Projects can be created and modified directly from the dashboard, which contains a listing of the projects, through a toggleable sidebar as shown in Figure 4. The sidebar form lets the user enter meta-information about the project, such as the name, description, status, and availability. Within a project, the user can see the listing of both the collection of programming exercises and gamification layers.

AuthorKit does not target a specific evaluation engine nor gamification service, aiming to be used independently as an authoring tool for gamified and non-gamified programming exercises; therefore, all efforts are headed into facilitating the creation, management, and conversion from/to other programming exercise formats. For instance, programming exercises can be imported/exported directly from their in-app listing in the user's preferred format. Currently, the formats supported are YAPExIL [31], Mooshak Exchange Format [36], and SIPE [37]. Consequently, authors can start with a collection of pre-existing programming exercises (i.e., importing them), optionally wire them up with gamification rules, and export to their tool of choice in an adequate format.

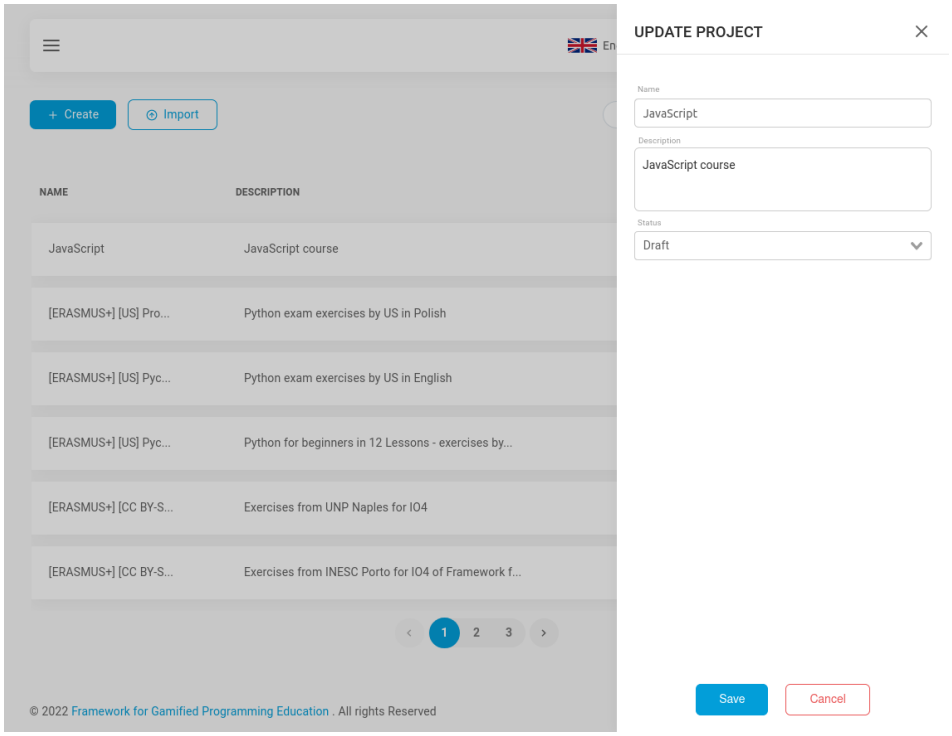

Figure 4. Screenshot of FGPE AuthorKit while editing a project.

5. Programming Learning Environment

5.1. Purpose and Design

The goal of the FGPE Programming Learning Environment (PLE) is to engage learners while learning to program. To this end, it combines automated assessment with a meaningful composition of gamification elements, both provided by the FGPE ecosystem. The PLE is effortlessly reusable across distinct courses and programming languages, being independent of the gamification layer linking the activities.

The PLE has been implemented as an open-source progressive web application. It was developed in ReactJS—"a JavaScript library for building user interfaces" [38]. It is based on the Apollo Client, a community-driven effort to build an easy-to-understand, flexible, and powerful GraphQL client, to manage and consume remote data from the FGPE Gamification Service [33] without worrying about infrastructure code for networking and caching. It also makes use of Keycloak [39], an open-source identity and access management solution providing features such as centralized user management, various authentication mechanisms, single sign-on and identity brokering, and social login.

As a web application, PLE is intended to work on any platform that runs a modern web browser, which includes both desktop and mobile devices. Moreover, it also features a multilingual user interface; at the moment of writing these words, eight languages were provided: German, English, Italian, Polish, Portuguese, Russian, Spanish, and Ukrainian.

PLE handles two types of users: students—who register at the platform and complete the programming courses in which they are enrolled, by solving the proposed challenges—and teachers—who manage the courses, enroll students, assign them into groups, and supervise their progress. Note that PLE is not used for authoring the exercises, which is the role of AuthorKit (see Section 4).

5.2. Student's User Interface

The student's user interface (StUI), whose exemplary view is presented in Figure 5, follows a component-driven development approach, which consists in dissecting and splitting the interface into small repeatable patterns, developing these patterns, and joining them together to form larger components and pages.

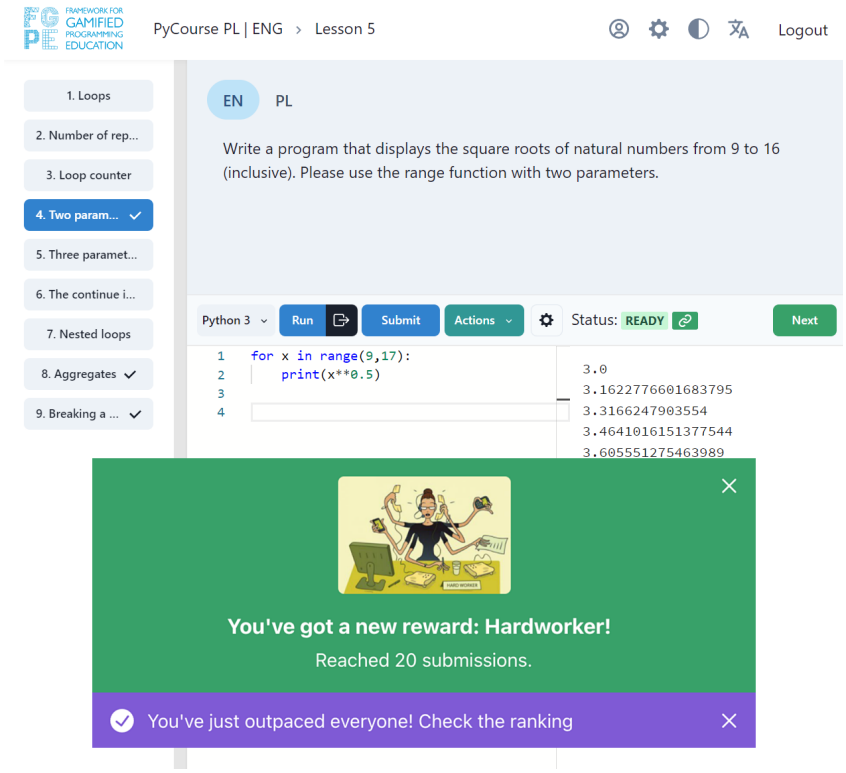

Figure 5. Exemplary screenshot of the FGPE PLE Student UI.

The main components of the StUI are the following:

Code Editor is based on Monaco Editor [40], the very same editor that powers Visual Studio Code. This editor allows students to code starting from a skeleton provided by the exercise author, taking advantage of a vast set of features such as syntax highlighting, parameter hints, smart code navigation, and code completion. For challenges that can be solved in more than one programming language, a language switch is attached at the top of the editor allowing the student to choose the programming language of the editor.

Console is where the results of the code execution and feedback based on the submission evaluation are presented. The code execution consists in taking the inputs provided by the student through a popup and running the program against these inputs. In distinction, on submission, the code is run against the complete set of test cases provided by the exercise author. While the execution is meant to help the students test their code, only after the submission, the code can be acknowledged as correct, triggering gamification rules, which in turn may result in a reward for the student or unlock a new section of the course.

Statement Viewer is where the activity statement is displayed. This component can display HTML, MarkDown, PDF and raw text files.

Leaderboard is the component responsible for displaying the usernames and scores, sorted according to certain metrics. Leaderboards can be challenge-scoped or course-scoped, use any of the available metrics to sort users by, and optionally have group visibility.

Push Notifications are small rectangle boxes displayed based on events received from registered GraphQL subscriptions. For instance, received rewards, results of both processed submissions and validations, and other updates.

Profile is the space to show off student's achievements, including badges, virtual items, experience points, and course progress.

While StUI is primarily designed for students practicing with the gamified exercises, it is also thought to be used by the exercise authors and teachers to check how the exercises

will be presented to students, verify the reactions of the automatic evaluation engine to various solution attempts, and ensure relevant gamification rules are triggered when they should be.

5.3. Teacher's User Interface

The teacher's user interface (TeaUI) addresses two basic needs of the teachers using PLE: game management and student management.

As for the game management, the following key functional areas are covered:

Setting up a game which lets the teacher provide the students a new game, consisting of a chosen set of programming exercises and the relevant gamification layer. The teacher can also limit the availability of the game to a specific time period as well as make it private, i.e., accessible only for the invited students (the public games are visible and can be played by anyone who registers at the platform).

Supervising a running game which includes viewing the students' progress in a particular game (i.e., how many exercises they have completed)—see Figure 6 for an exemplary screenshot, viewing how well the students dealt with a specific exercise (i.e., how many students have completed it) and letting the teacher modify game settings.

Deleting a game, which lets the teacher make a finished game no longer available for students.

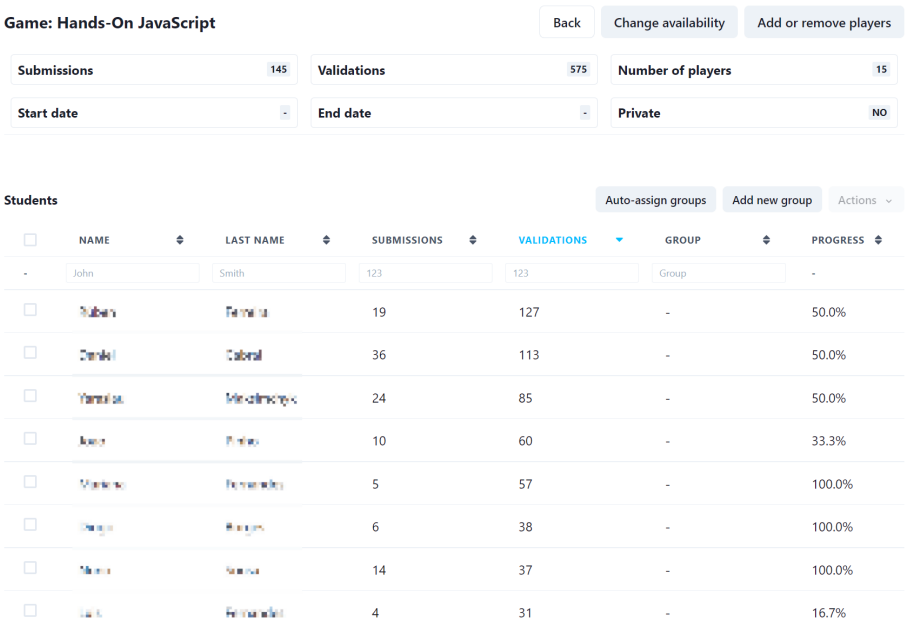

Figure 6. Exemplary screenshot of the FGPE PLE Teacher UI: game management.

With regard to the student management, the following key functional areas are covered:

Game participants management, which lets the teacher assign students to a particular game. This is especially important for private games, which cannot be joined by students on their own, unless the teacher provides them with a special link or adds them manually. This also allows the teacher to remove students from a game (because, e.g., they joined a wrong one).

Student group management, which lets the teacher assign students to groups. This makes it easier to, e.g., analyze progress of students from one class, as the other players may be easily filtered out. The assignment pertains to a specific game—so one student may belong to different groups in different games. The students may also be automatically assigned to randomly chosen groups—this feature is dedicated to support gamification scenarios based on group competition.

User profile, which gives teachers the access to any student's basic profile data (name, username, and email address) and their status in each game he or she plays (the assigned group, the number of gathered points and rewards, the number of submissions and validations made, and the progress).
Submission history, which lets the teacher see the code of any of the submissions made by a particular student (see Figure 7 for an exemplary screenshot). This serves a number of purposes—primarily, it lets the teacher know the reason for which some student struggles with a given exercise, and possibly be able to give him or her a hint; but it also allows the teacher to verify that a solution accepted by the automatic evaluation engine is actually correct—depending on the complexity of the exercise and the coverage of attached tests, it is more or less easy to make the engine accept a solution that is wrong yet produces the expected results.

Figure 7. Exemplary screenshot of the FGPE PLE Teacher UI: submission view.

6. Evaluation

Teachers typically have a short amount of time to prepare their classes and, thus, are not open to spending much time handling platform particularities; therefore, user experience is a crucial aspect for teachers' acceptance of the platform. As user experience is a very broad concept, for this particular study, we measure it in terms of usability metrics, such as effectiveness, efficiency, and satisfaction, and only from the teachers' perspective. The aim of this study is to evaluate the usability of the FGPE platform from the teachers' perspective during the preparation of the educational content for their classes. We, therefore, focused on AuthorKit, which is the FGPE component used for this purpose.

6.1. Methodology

The usability of AuthorKit has been measured through an online survey following the Usability Metric for User Experience (UMUX) Questionnaire model developed by Finstad [41]. UMUX is a short four-item Likert scale questionnaire used for the subjective

assessment of the perceived usability of an application after a short period of interaction. It provides results similar to those obtained with the longer ten-item System Usability Scale (SUS) (UMUX correlates with the SUS at a rate higher than 0.80 [42]) while also following the definition of usability established in ISO 9241-11 [41].

One of the key characteristics of UMUX is that it alternates positive and negative statements that respondents rate their agreement with. In particular, it includes the following statements:

UMUX.Q1 [The system's] capabilities meet my requirements. *(positive)*
UMUX.Q2 Using [the system] is a frustrating experience. *(negative)*
UMUX.Q3 [The system] is easy to use. *(positive)*
UMUX.Q4 I have to spend too much time correcting things with [the system]. *(negative)*

In this study, all items were measured in a 7-point Likert scale, including from the most negative to the most positive: "strongly disagree" (1), "disagree" (2), "somewhat disagree" (3), "neither agree or disagree" (4), "somewhat agree" (5), "agree" (6), and "strongly agree" (7). Rates assigned to negative statements are measured inversely to standardize results (i.e., odd items are scored as [user score] and even items are scored as (7 − user score + 1)); therefore, we will consider UMUX.Q2* and UMUX.Q4* as the positive version of UMUX.Q2 and UMUX.Q4, respectively, while presenting results.

The questionnaire has been administered to 8 programming teachers and 15 graduate students from six universities of four different countries, including the University of Szczecin (Poland), University of Porto and Media Arts and Design School of the Polytechnic of Porto (Portugal), Aalborg University (Denmark), and Parthenope University of Naples (Italy). Responses were collected at least a few weeks after the participants had their first contact with FGPE AuthorKit. Only participants who created no less than two programming exercises and one gamification layer using the platform took the questionnaire. All respondents participated in the survey voluntarily and anonymously at their preferred time and location, and were informed about the purpose of gathering the data.

6.2. Results

Figure 8 depicts the results in a percentage heatmap, with values, average, and standard deviation per question. From a visual perspective, it is clear that the direction of response for all variables is towards the degree of somewhat agree (5), agree (6), and strongly agree (7), which means that the measurement of the variable is acceptable [43]. For instance, UMUX.Q2* has a mean close to agree (6), whereas the others have means greater or close to somewhat agree (5). The lowest mean is 4.7 for question UMUX.Q4*.

To confirm this observation, Table 2 extracts the minimum, first quantile, median, mean, third quantile, and maximum for each question. According to Nielsen [44], the result of the usability measurement can be described by the mean value of each variable. In this case, we considered both the mean and median. For UMUX.Q1, UMUX.Q2*, and UMUX.Q3, the first quantile score is already categorized as accepted (6, 5, and 5), so the median is obviously accepted too. As for UMUX.Q4*, the median (5) is also classified as acceptable. Finally, all the means (5.7, 5.91, 5.35, and 4.7, respectively) are categorized as accepted. The overall average (5.42) indicates clearly positive respondents' general opinion about AuthorKit's usability.

Table 2. Statistical summary of the results.

Question	Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
UMUX.Q1	2	6	6	5.70	6	7
UMUX.Q2*	3	5	6	5.91	7	7
UMUX.Q3	2	5	6	5.35	6	7
UMUX.Q4*	3	4	5	4.70	6	6

Figure 8. Results of the UMUX questionnaire. Rates assigned to negative questions have already been inverted.

6.3. Discussion

Even though the results of the UMUX questionnaire do not allow us to conclude as precisely as a quantitative usability measurement could, they can still help to identify possible problems. For instance, UMUX.Q4*, which asks about the feeling of participants regarding the time spent to solve some issue, has received the lowest scores in this study. This may mean that a part of respondents feel that they are wasting some time due to the platform, so there are things that could be redesigned to let the users achieve their intended aims in less time.

An obvious limitation of the evaluation results presented here is the small sample size. Nonetheless, its heterogeneous composition, especially the fact the respondents hailed from four countries, gives the results some legitimacy.

Comparing the obtained results to those reported in the literature, edCrumble, a visual authoring tool for blended learning, was evaluated using UMUX by a group of 56 users on average at 5.208 (UMUX.Q1 at 5.428, UMUX.Q2* at 5.534, UMUX.Q3 at 5.527, and UMUX.Q4* at 4.341) [45] (calculated based on Table V therein), i.e., below AuthorKit on average and in all considered aspects but the ease of use. Taking a wider perspective, Lewis reported usability evaluation results for four widely used software products: Excel (UMUX average of 5.176, n = 390), Word (5.53, n = 453), Amazon (6.088, n = 338), Gmail (5.68, n = 256) [46] (the percentages reported in Table 8 therein were remapped to the 1–7 scale). In this context, AuthorKit's usability (on average, evaluated at 5.42) could be ranked between Excel's and Word's.

7. Conclusions

This work describes how the software ecosystem developed within the FGPE project [8] can be used to provide and manage gamified computer programming courses. While our previous work presented the formats needed to convey programming exercise [31] and gamification layer [32] definitions, and the gamification service [33], handling students' submissions, triggering gamification rules, and updating the game state, in this paper, we focus on the two remaining key elements of the ecosystem: FGPE AuthorKit and FGPE PLE. The first can be used to prepare programming exercise sets and relevant gamification rules, and the other to provide the students with the ability to play with gamified exercises and allow the teachers to set up challenges for the students and follow their progress. The presented results of usability evaluation show that teachers positively assessed the tool in all considered dimensions.

The ecosystem developed within the FGPE project [8] is the first customizable open-source environment to support teachers in applying gamification to computer programming education in all phases of the process, from designing the exercises to analyzing the progress students make in their courses. While the ecosystem is already fully operational, its development continues under the FGPE Plus project [47], whose main aims are to make the PLE embeddable in any LTI-compliant Learning Management System and to enhance the PLE mobile user experience; therefore, further usability evaluation involving students is envisaged to help indicate the proper direction of PLE development in this regard. Further, more thorough experiments with quantitative measurements to evaluate the learning and behavioral effects of different gamified scenarios are planned.

Author Contributions: Conceptualization, J.C.P., R.Q., and J.S.; Data curation, J.C.P., and J.S.; Formal analysis, J.C.P. and J.S.; Funding acquisition, J.S.; Investigation, J.C.P., R.Q., and J.S.; Methodology, J.C.P., R.Q., and J.S.; Project administration, R.Q., J.P.L., and J.S.; Resources, J.C.P. and F.M.; Software, J.C.P. and F.M.; Supervision, R.Q., J.P.L., and J.S.; Validation, J.C.P., R.Q., J.P.L., J.S., and F.M.; Visualization, J.C.P., J.S., and F.M.; Writing—original draft, J.C.P., R.Q., J.P.L., J.S., and F.M.; Writing—review and editing, J.C.P., R.Q., J.P.L., J.S., and F.M. All authors read and agreed to the published version of the manuscript.

Funding: The work described in this paper was achieved within two projects supported by the European Union's Erasmus Plus programme: the Framework for Gamified Programming Education (2018-1-PL01-KA203-050803) and FGPE Plus: Learning tools interoperability for gamified programming education (2020-1-PL01-KA226-HE-095786).

Institutional Review Board Statement: Not applicable. No sensitive or personal data were collected.

Informed Consent Statement: Informed consent was obtained from all subjects involved in the study.

Data Availability Statement: Data are available from the corresponding author on request.

Acknowledgments: The authors would like to thank all the people who participated in testing and evaluation of AuthorKit.

Conflicts of Interest: The authors declare no conflict of interest. The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript; nor in the decision to publish the results.

References

1. Prenkaj, B.; Stilo, G.; Madeddu, L. Challenges and Solutions to the Student Dropout Prediction Problem in Online Courses. In Proceedings of the 29th ACM International Conference on Information & Knowledge Management, CIKM '20, Online, 19–23 October 2020; ACM: New York, NY, USA, 2020; pp. 3513–3514. [CrossRef]
2. Ghaban, W.; Hendley, R. How Different Personalities Benefit From Gamification. *Interact. Comput.* **2019**, *31*, 138–153. [CrossRef]
3. Deterding, S. The lens of intrinsic skill atoms: A method for gameful design. *Hum.-Comput. Interact.* **2015**, *30*, 294–335. [CrossRef]
4. Bellon, J.J.; Bellon, E.C.; Blank, M.A. Teaching from a Research Knowledge Base. *NASSP Bull.* **1992**, *76*, 121–122. [CrossRef]
5. Shute, V.J. Focus on Formative Feedback. *Rev. Educ. Res.* **2008**, *78*, 153–189. [CrossRef]
6. Keuning, H.; Jeuring, J.; Heeren, B. Towards a Systematic Review of Automated Feedback Generation for Programming Exercises. In Proceedings of the 2016 ACM Conference on Innovation and Technology in Computer Science Education, Arequipa, Peru, 11–13 July 2016; ACM: New York, NY, USA, 2016; pp. 41–46. [CrossRef]
7. Dicheva, D.; Irwin, K.; Dichev, C. OneUp: Supporting Practical and Experimental Gamification of Learning. *Int. J. Ser. Games* **2018**, *5*, 5–21. [CrossRef]
8. FGPE Consortium. Framework for Gamified Programming Education. 2018. Available online: https://fgpe.usz.edu.pl (accessed on 19 November 2021).
9. Swacha, J. State of Research on Gamification in Education: A Bibliometric Survey. *Educ. Sci.* **2021**, *11*, 69. [CrossRef]
10. Utomo, A.Y.; Amriani, A.; Aji, A.F.; Wahidah, F.R.N.; Junus, K.M. Gamified E-learning model based on community of inquiry. In Proceedings of the 2014 International Conference on Advanced Computer Science and Information System, Jakarta, Indonesia, 18–19 October 2014; pp. 474–480. [CrossRef]
11. Anderson, P.E.; Nash, T.; McCauley, R. Facilitating Programming Success in Data Science Courses Through Gamified Scaffolding and Learn2Mine. In Proceedings of the 2015 ACM Conference on Innovation and Technology in Computer Science Education, ITiCSE '15, Vilnius, Lithuania, 4–8 July 2015; ACM: New York, NY, USA, 2015; pp. 99–104. [CrossRef]
12. Auvinen, T.; Hakulinen, L.; Malmi, L. Increasing students' awareness of their behavior in online learning environments with visualizations and achievement badges. *IEEE Trans. Learn. Technol.* **2015**, *8*, 261–273. [CrossRef]

13. Bernik, A.; Bubas, G.; Radosevic, D. A Pilot Study of the Influence of Gamification on the Effectiveness of an e-Learning Course. In *Central European Conference on Information and Intelligent Systems*; Faculty of Organization and Informatics, University of Zagreb: Varaždin, Croatia, 2015; pp. 73–79.
14. Hakulinen, L.; Auvinen, T.; Korhonen, A. The Effect of Achievement Badges on Students' Behavior: An Empirical Study in a University-Level Computer Science Course. *Int. J. Emerg. Technol. Learn.* **2015**, *10*, 18–29. [CrossRef]
15. Paiva, J.C.; Leal, J.P.; Queirós, R.A. Enki: A Pedagogical Services Aggregator for Learning Programming Languages. In Proceedings of the 2016 ACM Conference on Innovation and Technology in Computer Science Education, Arequipa, Peru, 11–13 July 2016; ACM: New York, NY, USA, 2016; pp. 332–337.
16. Topîrceanu, A. Gamified learning: A role-playing approach to increase student in-class motivation. *Procedia Comput. Sci.* **2017**, *112*, 41–50, https://doi.org/10.1016/j.procs.2017.08.017.
17. Faghihi, U.; Brautigam, A.; Jorgenson, K.; Martin, D.; Brown, A.; Measures, E.; Maldonado-Bouchard, S. How Gamification Applies for Educational Purpose Specially with College Algebra. *Procedia Comput. Sci.* **2014**, *41*, 182–187, https://doi.org/10.1016/j.procs.2014.11.102.
18. Yildirim, I. The effects of gamification-based teaching practices on student achievement and students' attitudes toward lessons. *Internet High. Educ.* **2017**, *33*, 86–92. [CrossRef]
19. Christy, K.R.; Fox, J. Leaderboards in a virtual classroom: A test of stereotype threat and social comparison explanations for women's math performance. *Comput. Educ.* **2014**, *78*, 66–77. [CrossRef]
20. Pedersen, M.K.; Svenningsen, A.; Dohn, N.B.; Lieberoth, A.; Sherson, J. DiffGame: Game-based Mathematics Learning for Physics. *Procedia-Soc. Behav. Sci.* **2016**, *228*, 316–322, https://doi.org/10.1016/j.sbspro.2016.07.047.
21. Hasegawa, T.; Koshino, M.; Ban, H. An English vocabulary learning support system for the learner's sustainable motivation. *SpringerPlus* **2015**, *4*, 99. [CrossRef] [PubMed]
22. Perry, B. Gamifying French Language Learning: A Case Study Examining a Quest-based, Augmented Reality Mobile Learning-tool. *Procedia-Soc. Behav. Sci.* **2015**, *174*, 2308–2315, https://doi.org/10.1016/j.sbspro.2015.01.892.
23. Barata, G.; Gama, S.; Jorge, J.; Gonçalves, D. Studying student differentiation in gamified education: A long-term study. *Comput. Hum. Behav.* **2017**, *71*, 550–585. [CrossRef]
24. Hanus, M.D.; Fox, J. Assessing the effects of gamification in the classroom: A longitudinal study on intrinsic motivation, social comparison, satisfaction, effort, and academic performance. *Comput. Educ.* **2015**, *80*, 152–161. [CrossRef]
25. Holman, C.; Aguilar, S.J.; Levick, A.; Stern, J.; Plummer, B.; Fishman, B. Planning for Success: How Students Use a Grade Prediction Tool to Win Their Classes. In Proceedings of the Fifth International Conference on Learning Analytics And Knowledge, Poughkeepsie, NY, USA, 16–20 March 2015; ACM: New York, NY, USA, 2015; pp. 260–264. [CrossRef]
26. Jang, J.; Park, J.J.Y.; Yi, M.Y. Gamification of Online Learning. In *Artificial Intelligence in Education*; Conati, C., Heffernan, N., Mitrovic, A., Verdejo, M.F., Eds.; Springer International Publishing: Cham, Switzerland, 2015; pp. 646–649.
27. Pettit, R.K.; McCoy, L.; Kinney, M.; Schwartz, F.N. Student perceptions of gamified audience response system interactions in large group lectures and via lecture capture technology. *BMC Med. Educ.* **2015**, *15*, 92. [CrossRef]
28. Nevin, C.R.; Westfall, A.O.; Rodriguez, J.M.; Dempsey, D.M.; Cherrington, A.; Roy, B.; Patel, M.; Willig, J.H. Gamification as a tool for enhancing graduate medical education. *Postgrad. Med. J.* **2014**, *90*, 685–693. [CrossRef]
29. Bonde, M.T.; Makransky, G.; Wandall, J.; Larsen, M.V.; Morsing, M.; Jarmer, H.; Sommer, M.O.A. Improving biotech education through gamified laboratory simulations. *Nat. Biotechnol.* **2014**, *32*, 694–697. [CrossRef]
30. Paiva, J.C.; Queirós, R.; Leal, J.P.; Swacha, J. FGPE AuthorKit—A Tool for Authoring Gamified Programming Educational Content. In Proceedings of the 2020 ACM Conference on Innovation and Technology in Computer Science Education, ITiCSE '20, Trondheim, Norway, 15–19 June 2020; ACM: New York, NY, USA, 2020; p. 564. [CrossRef]
31. Paiva, J.C.; Queirós, R.; Leal, J.P.; Swacha, J. Yet Another Programming Exercises Interoperability Language (Short Paper). In Proceedings of the 9th Symposium on Languages, Applications and Technologies (SLATE 2020), Online, 13–14 July 2020; Simões, A., Henriques, P.R., Queirós, R., Eds.; OpenAccess Series in Informatics (OASIcs); Schloss Dagstuhl–Leibniz-Zentrum für Informatik: Dagstuhl, Germany, 2020; Volume 83, pp. 14:1–14:8. [CrossRef]
32. Swacha, J.; Paiva, J.C.; Leal, J.P.; Queirós, R.; Montella, R.; Kosta, S. GEdIL–Gamified Education Interoperability Language. *Information* **2020**, *11*, 287. [CrossRef]
33. Paiva, J.C.; Haraszczuk, A.; Queirós, R.; Leal, J.P.; Swacha, J.; Kosta, S. FGPE Gamification Service: A GraphQL Service to Gamify Online Education. In Proceedings of the 9th World Conference on Information Systems and Technologies, Azores, Portugal, 30 March–2 April 2021; Springer: Azores, Portugal, 2021; Volume 1368, pp. 480–489. [CrossRef]
34. Leal, J.P.; Silva, F. Mooshak: A Web-based multi-site programming contest system. *Softw. Pract. Exp.* **2003**, *33*, 567–581. [CrossRef]
35. Paiva, J.C.; Queirós, R.; Leal, J.P.; Swacha, J.; Miernik, F. An Open-Source Gamified Programming Learning Environment. In Proceedings of the Second International Computer Programming Education Conference (ICPEC 2021), Braga, Portugal, 27–28 May 2021; Henriques, P.R., Portela, F., Queirós, R., Simões, A., Eds.; Open Access Series in Informatics (OASIcs); Schloss Dagstuhl—Leibniz-Zentrum für Informatik: Dagstuhl, Germany, 2021; Volume 91, pp. 5:1–5:8. [CrossRef]
36. Paiva, J.C.; Queirós, R.; Leal, J.P. Mooshak's Diet Update: Introducing YAPExIL Format to Mooshak. In Proceedings of the 10th Symposium on Languages, Applications and Technologies (SLATE 2021), Póvoa de Varzim, Portugal, 1–2 July 2021; Queirós, R., Pinto, M., Simões, A., Portela, F., Pereira, M.J.A., Eds.; Open Access Series in Informatics (OASIcs); Schloss Dagstuhl—Leibniz-Zentrum für Informatik: Dagstuhl, Germany, 2021; Volume 94, pp. 9:1–9:7. [CrossRef]

37. Swacha, J. SIPE: A Domain-Specific Language for Specifying Interactive Programming Exercises. In *Towards a Synergistic Combination of Research and Practice in Software Engineering*; Kosiuczenko, P., Madeyski, L., Eds.; Springer International Publishing: Cham, Switzerland, 2018; pp. 15–29. [CrossRef]
38. Facebook. React: A JavaScript Library for Building User Interfaces. 2021. Available online: https://reactjs.org (accessed on 16 January 2021).
39. Keycloak. Keycloak: Open Source Identity and Access Management Solution. 2014. Available online: https://www.keycloak.org (accessed on 9 January 2021).
40. Microsoft. Monaco Editor. 2021. Available online: https://microsoft.github.io/monaco-editor/ (accessed on 16 January 2021).
41. Finstad, K. The Usability Metric for User Experience. *Interact. Comput.* **2010**, *22*, 323–327. [CrossRef]
42. Brooke, J. SUS-A quick and dirty usability scale. *Usability Eval. Ind.* **1996**, *189*, 4–7.
43. Babbitt, B.; Nystrom, C. Questionnaire construction manual (Research Product 89-20). In *Fort Hood, TX: US Army Research Institute for the Behavioral and Social Sciences*; Defense Technical Information Center: Fort Belvoir, VA, USA, 1989.
44. Nielsen, J. *Usability Engineering*; Morgan Kaufmann: Burlington, MA, USA, 1994.
45. Albó, L.; Hernández-Leo, D. edCrumble, a Data-Enriched Visual Authoring Design Tool for Blended Learning. *IEEE Trans. Learn. Technol.* **2021**, *14*, 55–68. [CrossRef]
46. Lewis, J.R. Measuring Perceived Usability: SUS, UMUX, and CSUQ Ratings for Four Everyday Products. *Int. J. Hum.-Comput. Interact.* **2019**, *35*, 1404–1419. [CrossRef]
47. FGPE Consortium. FGPE Plus. 2021. Available online: https://fgpeplus.usz.edu.pl/ (accessed on 19 November 2021).