Game-based Learning: ProgGames

Makayla Menges

School of Computing Science

Sir Alwyn Williams Building

University of Glasgow

G12 8RZ

A dissertation presented in part fulfillment of the requirements of the Degree of Master of Science at the University of Glasgow

September 6th, 2019

Abstract

Games have been used frequently to teach all age ranges new knowledge and video games are no different. They teach many different aspects of problem-solving, stress management, etc. but they also lend their abilities to education. This project studies the impacts of using education-based games within a cooperative environment to understand if playing through these education-based video games with a classmate can help reinforce or teach a student new course. To start this process a system called 'ProgGames' was created with the following process; understanding the background and current research of education-based video games, utilizing that research to generate requirements for the students and teachers who may use the 'ProgGames' system, implementing the requirements into a design for the project, and evaluating the system with participants to gather data required to understand our initial question.

Education Use Consent

I hereby give my permission for this project to be shown to other University of Glasgow students and to be distributed in an electronic form.

Name: Signature:

Acknowledgements

I would like to thank my supervisor for this project for not only keeping me on track but also for their many suggestions and time set aside to meet with me individually. To all the individuals around me and across the ocean who have given endless amounts of encouragement and support through this process and to all the participants who volunteered their time to this project, thank you.

Contents

Chapter 1 Introduction

1.1 What Will be Developed

Through this report, a project will be developed to understand the potential of utilizing multiplayer versus single-player education-based video games to enhance student's learning outcomes by understanding the challenges and flow of both. The project itself is a multiplayer and single-player game aimed at allowing student peers to challenge each other's or their own programming skills and knowledge. This game will allow the students the chance to test the users programming fundamentals while also allowing them to support and challenge each other.

The project was created through a Lean Game Development (LGD) strategy discussed by Rosenfield to combine the aspects of software development and game development into a functional model (Rosenfield Boeira, 2017). The LGD strategy, as shown in Figure 1, has the following main structure of Inception or background research and problem definition, Design or creation of design goals/hypothesis, and Build or code, test, and evaluate with sub-structures included in each (Rosenfield Boeira, 2017).

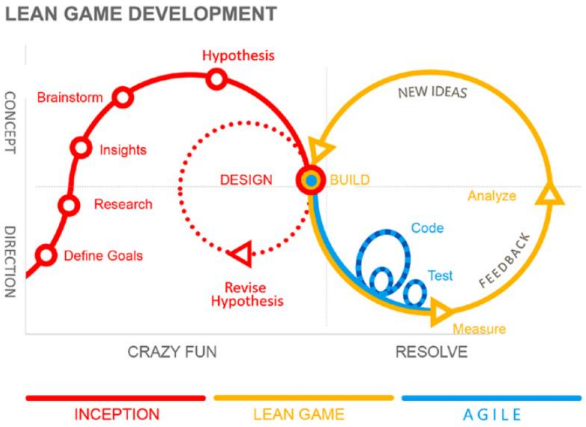


Figure 1: Lean Game Development Structure **(Rosenfield Boeira, 2017)**

1.2 Report Structure

This report has the following structure:

1. A background analysis of education-based video games which includes their origins, essentials to create and understand what impacts education-based video games, and past research that has already been done in this field.
2. A set of requirements and objectives outlined for this project.
3. An explanation of original and current designs for the project and how they were implemented.
4. The strategy and outcomes of testing the game features.
5. How evaluations were executed and the results of the evaluations themselves.
6. A discussion of project achievements and future work regarding this project.

Chapter 2 Analysis & Requirements

2.1 History and Analysis of Games as Learning Tools

The idea of learning a new subject or practicing current skills of a topic can be an uninviting thought as students may feel overwhelmed or tired of the topic itself. This literature review will discuss game-based learning and the implications it already had on the future of educational skills. It will also consider how this past research was executed before writing this paper.

2.1.1 Educational Game Origins

The origins of game-based learning and the different aspect developers should consider while creating an educational video game differ depending on the type of content intended for the student to learn (Plass, Homer, & Kinzer, 2015). These various aspects are: cognitive, affective, behavioral, and sociocultural, and all contribute to the effectiveness of the learning outcomes themselves (Plass, Homer, & Kinzer, 2015). According to Plass, Homer, & Kinzer's research, the engagements are as follows (Plass, Homer, & Kinzer, 2015):

- Behavioral engagement: learning an action or using movement and gestures.
- Sociocultural engagement: engaging in social contexts within the game, such as inviting other players to play or sharing essential items.
- Affective engagement: Emotional engagement through characters or situations within the game.
- Cognitive engagement: The cognitive processing the player goes through to understand the game and its situations.

However, it does state that cognitive engagement, over the others, is needed in all game-based learning as it is the mental processing of information to learn (Plass, Homer, & Kinzer, 2015). These engagement processes are based on the psychological actions of using video games as educational tools.

2.2 Game-based Learning Essentials

2.2.1 Flow & Engagement

When thinking about video games or games, in general, the question 'what makes them enjoyable and want to be played' must be presented. Based on this question, one argument is brought forward, the idea of flow, or the balance of the challenges within the game (Plass, Homer, & Kinzer, 2015). Flow is the ability for the players/learners to be able to go through the steps of the game without problems but also while being challenged (Hamari, et al., 2016). What this means is the player must be challenged enough to keep their interests within the context but not challenged to the point where they feel like they are lost or will fail. This leads to the meaning of challenges. By making sure the player has trials, it makes sure the player is engaged in the learning and playing (Hamari, et al., 2016). If the player feels like they will not get any knowledge or benefit from what they are

doing, then they will not have the motivation to continue the tasks or to actively try to understand the curriculum placed in front of them (Hamari, et al., 2016).

(Hamari, et al., 2016) describes engagement as having an elevation in concentration, enjoyment, and interest. These three aspects are defined as absorption, the positive feelings associated with the game, and the amount of attention being directed towards the actions, respectively. Both Hamari, Admirall, and their respective authors suggest that engagement and flow are inherently connected through tasks and challenges to create an environment students are intrigued by and therefore have the desire to learn the concepts being presented to them (Admiraal, Huizenga, Akkerman, & ten Dam, 2011) (Hamari, et al., 2016). No matter how these researches defined flow or engagement, it proposes that the ability to create an environment that is challenging but not too challenging, is engaging without distractions, and allows for feedback without breaking the flow of the games themselves could improve the ability for the student to learn or improve their current knowledge of a subject.

2.2.2 Challenges

Current education is based on a belief that standardized tests and curriculum that is universal is appropriate for all individuals. However, the fact that the students are individuals shows the fact that they will have different levels of engagement, knowledge, and ambitions. (Hamari, et al., 2016) states, these games allow for more individualization regarding challenges as the students can go through the game at their own pace and face these challenges as they progress through a system that is being designed to take them through a course they are interested in learning about.

2.3 Past Research

2.3.1 Collaborative Learning

(Admiraal, Huizenga, Akkerman, & ten Dam, 2011) investigated the impacts of game-based learning through flow, engagement, and collaborative learning within a high school class setting. These students were asked to play a game within teams to learn about Amsterdam. This was utilized to show the connection between engagement, flow, and the ability to retain knowledge acquired in the game. The authors found that when the students were not distracted by the technical issues, they gained the engagement and flow required to better preserver the information they had learned regarding Amsterdam (Admiraal, Huizenga, Akkerman, & ten Dam, 2011). High school students were asked to spend one day playing a game in teams that went head to head. At the end of the game evaluations, the teams presented their data and answers, and the team with the rightest answers won the most points (Admiraal, Huizenga, Akkerman, & ten Dam, 2011). Based on these authors investigations, students were more engaged with the program throughout the learning process, the more they learned about Amsterdam. This shows that a state of flow and lack of distractions, such as technical difficulties and other students, is beneficial to the learning process.

(Sung & Hwang, 2013) integrated Mindtools to better assists players to organize the information they learn within an education game. This integration was then tested using three groups (one of which did not use Mindtools). These three groups

were evaluated to understand how Mindtools integration could benefit educational video games and impactful learning and to understand better the impacts of collaborative learning on educational memory as groups were playing with each other throughout the process. The students were asked to participate in a pre-test to test their knowledge levels before playing the game. They were then asked to play through the game with their peers. The results regarding collaborative learning are highly crucial as these results suggest that having the ability to communicate and discuss between peers improved the learning outcomes of the students themselves. This is important as it proposes that students who have more opportunity to interact with their peers regarding their studies will have a better knowledge skillset after playing the game (Sung & Hwang, 2013).

2.3.2 Impactful Learning

(Hamari, et al., 2016) is trying to understand what will benefit learners when going through a course they may have skills in or not. As well as understanding how the flow and immersion of video games may impact engagement and learning. They used the following method to understand better if flow, engagement, and immersion have any impact on the potential implications. Two games were installed and distributed to be played throughout eleven different classrooms, one of the game's subjects was physics, and the other game subject was in engineering dynamics (Hamari, et al., 2016). These students were then given surveys at the end of their courses and took their standard school exams needed to exit the course. Overall, they concluded that video games could enhance the learning capabilities of students through engagement. This may have also been impacted by skill and challenges the students had during their time playing the game. They also showed that immersion within the game did not have a significant impact on learning.

2.4 Requirements

A generic project description was provided by Dr. Mireilla Bikanga Ada at the beginning of the project selection. This description gave a basic overview of the project to be designed and the problem to be researched within the project. Dr. Ada then aided in answering any basic questions regarding the clarity of the overall goals of the system itself.

These requirements were further expanded by researching past projects and research done within the field of game-based learning. This research was used to understand the fundamentals required to create a functioning and impactful education video game design based on the requirements found. Beyond this research and the guidance of Dr. Ada, any further specifications were based on my own personal experience in creating videos games.

The application itself must allow for users to be able to start the application within their own computer operating system and could have the option to load their progress if returning to the game from previous playthroughs. The user must also be able to choose the programming language that they would like to learn or improve their knowledge of. Once this is determined, the user must be able to select the level of difficulty within the programming language they would like to learn more about. Once this language and difficulty are chosen, the player must be able to collaborate with their peers to better understand the programming language and improve their long-term knowledge of the subject. These

requirements were used to make a Must have, Should have, Could have, Would like to have (MoSCoW) statement located in Appendix A.

2.5 Objectives

A goal of this virtual education project is to understand whether multiplayer functionality in a game-based learning environment impacts the challenges and focus needed to increase student's learning outcomes. This will be done and tested by making sure there is a flow to the game and by evaluating the levels of challenges that users feel the game presents.

Being able to progress through the game while also not feeling like the user is reading a textbook is essential to the levels of engagement that the users have within the project. To attempt to avoid this, the game will try to integrate the learning practices within the gameplay. This will be done by creating a storyline with the information needed to understand incorporated within the line of objectives the users will have to go through to play the game.

As (Plass, Homer, & Kinzer, 2015) states, cognitive engagement is vital to all types and topics of game-based learning as the overall goal of an educational video game is for students to come away from the game having learned something long-term. This requirement is an important one for this project as the goal of participants to be able to either gain new knowledge or improve on the experience they already have of a programming language. Therefore, our focus will be cognitive engagement, which will only be enhanced by the utilization of social, action, and emotional engagement through the collaborative aspect of the game. This leads to the next objective, collaborative learning.

There must be a multiplayer or collaborative portion of the game. Users must be able to challenge themselves and their peers within the programming language. This will be used to enhance their engagement better and help them through the challenges they face while learning the programming languages. The game should also have a one-player option for the players that want to play by themselves. This would entail the same requirements but would be designed for a single player.

The options for programming languages must have at least two options but can have more if time allows. There should also be the option of advanced questions if the user feels they are not being challenged or already feels that their knowledge is beyond the fundamentals.

Chapter 3 Design & Implementation

3.1 Design

This chapter will discuss the design strategies utilized to create 'ProgGames.' This is crucial to the project developed as it will explain the rationale behind design decisions and the implementations to develop them. The following sections will describe the main sections of the project, their specific goals or reasons, and how they were implemented. A full view of all initial wireframes and current game design documentation is located in Appendix B.

3.1.1 Overall Implementation

The overall project was implemented using the Unity Game engine for the game and SQLite for the database. Unity was the chosen engine to develop this project in for the following reasons; The programming language was C#, and as I have had at least a little experience using C#, it was a smaller learning curve to C++ in the Unreal Engine option. Unreal Engine has a phenomenal 3D game option, however, as the focus of this project was to experience the potential benefits of multiplayer game-based learning, a 3D game was not needed therefore a 2D option was best suited for the moment. Unity also allows for multiplayer hosts, clients, and servers and provides many opportunities for the deployment of which we are focusing on a windows-based implementation for now.

3.1.2 Single Player and Multiplayer

The single-player and multiplayer options of the game were crucial as the goal of this project was to understand the impacts of multiplayer and single-player game-based learning. This created a scenario of which aspects of the single-player mode were needed in the multiplayer mode as well. By understanding which elements could cross over, it would create an opportunity for classes and functions that could be utilized across both sections. To do this, the following list of essential game functions were generated:

Get curriculum – Teacher and Student
Login for Game – Student and Teacher
Choose Difficulty - Student
Choose Single or Multiplayer - Student
Choose Programming Language - Student
See Instructions - Student
See Game Credits - Student
See Results - Student

These functionalities are utilized by everyone who uses the system and therefore, could be created once, reducing the code produced.

The student's access the game options screen after logging into the game from the main screen which then goes to the game options screen, Figure 2, which shows the options students can choose from including playing a game, checking instructions, and game credits. In order to play a game, the student must select their choice of programming language, difficulty, and solo or multiplayer. If the student selects 'Two-Player' then the options that the local player chooses are the

options they see in the game. For instance, Player A chooses 'Python', 'Beginner', and 'Two-Player'. Player B selects 'Java', 'Intermediate', 'Two-Player'. Player A's game will display Beginner Python questions, and Player B's game will show Intermediate Java questions. This was done so that each player could choose their own challenge of subjects while still being able to play with whomever they choose to.

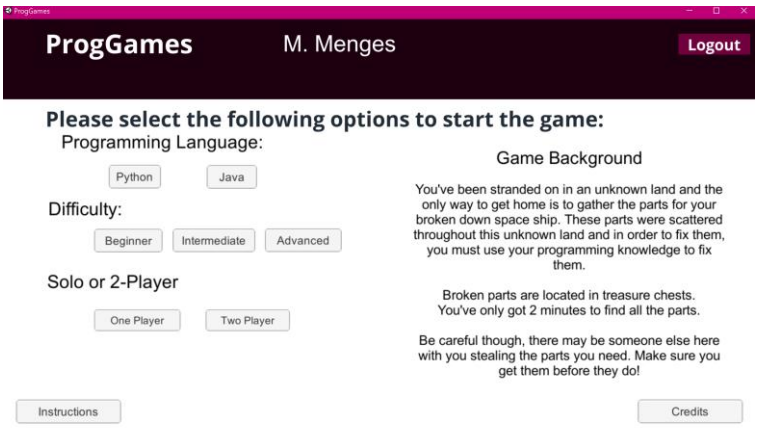


Figure 2: ProgGames Game Options Screen

Now once the student chooses between one and two-player modes, the screen will split into two different scenes. These Unity scenes differ as the single one player option takes them straight to the game while the two-player option takes them to a Network option (Figure 3) created utilizing the Unity game engine's network assets and multiplayer capabilities (Unity Technologies) (Unity - Manual: Network Lobby Manager, 2019). This allowed each local game to play with their own game settings and only shared information that needed to be shared.

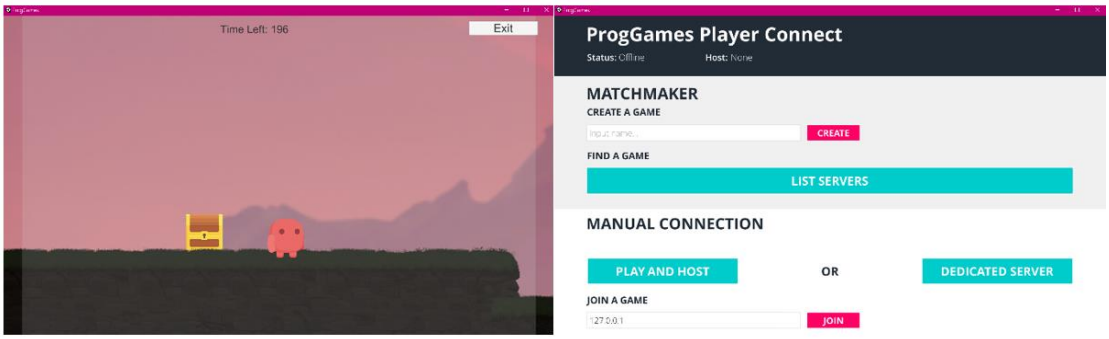


Figure 3: ProgGames One-Player Game Screen (Left) **(Bayat Games)**. Multiplayer Network Options Screen (Right) **(Unity Technologies)**.

Once the player gets to the network options screen, they then need to decide if they want to join a current game or create a new game. They must then wait until at least two players have joined the game before it will start. In both single and multiplayer games, the overall goal is to get all the treasures by answering the

questions. The single-player version handles this by decrementing a treasure count when a question is responded to and stores if it was answered correctly or incorrectly. The multiplayer version handles this slightly differently as both versions of the game has their own treasures within their individual client. The way Unity's networking options work is if a variable is required to update when it is changed it must use an option called [SyncVar] this option only updates the server's variable and not the client's variables. Therefore, an [Command] option must be used to update all the clients with this variable, and this is what handles the check if the players are still playing (Figure 4).

```
# ProgGames GameManager.cs
[SyncVar] public int treasureCount = 8;
 …
[Command]
void CmdChangeTreasure(int treas)
  {
     treasureCount = treas;
     RpcchangeTreasure(treas);
  }
```

Figure 4: This code is syncing variables across the clients and server, which is vital to the overall multiplayer gameplay.

3.1.3 Teacher

The teacher view was discussed early on as in an educational setting, there must be curriculum or questions being displayed and developed for the students. It was decided that they would need access to see their own student list and the progress the student had made in the teacher's curriculum itself. They would also need access to the curriculum to update, change, and add new options based on the student's needs. Based on these needs, a database of information was developed to gather all this information in one place.

The SQLite database was utilized to implement and create a login, student vs. teacher view, and a teacher curriculum system. This database was chosen as it is relationally based, it functions well within Unity, and the project did not require large amounts of database storage. These systems were created in one database of which is attached to the game and accessible through the game's assets and scripts.

These scripts access the database to; login the teacher or students' users based on their username, get curriculum based on individual teachers, get students based on their teachers, get curriculum specific students have completed based on their teacher, and get explanations of the questions answered wrong for the results page in the game view. Students are logged in if their username is all numbers, such as an ID number, and teachers are logged in if their username contains numbers and letters, such as their last name and ID. An Entity-Relationship diagram for this database is in Appendix A.

Within the game, once a teacher logs in a script gathers all the curriculum for the individual teacher based on their unique id located in the 'users' table. This

function also works for the students once they log in as the teacher's curriculum for that student is needed within the game. The teacher is then given two options; check on student progress or edit curriculum. Depending on the choice they make the scripts then get the information needed within the database. The student progress screen displays a list of students as well as the percentage of curriculum completed for each programming language. The teacher can then select a student and see their curriculum and what that student has completed or not. This allows them to know which questions students may be having troubles with.

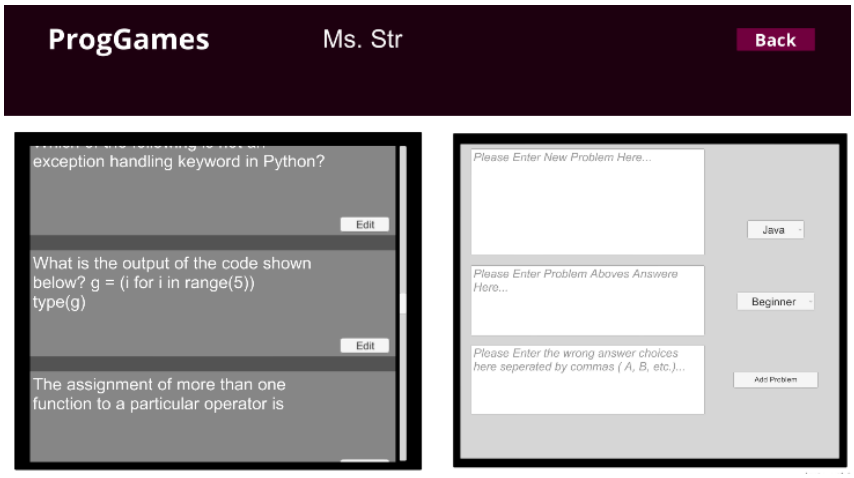


Figure 5: List of curricula that is editable (Left) and Add new curriculum (Right). Java and Python Curriculum questions, answers, and wrong answers that were utilized were retrieved from the Sanfoundry website **(1000 Java MCQs for Freshers & Experienced | Sanfoundry) (1000 Python MCQs for Freshers & Experienced | Sanfoundry)**.

If the teacher selects the edit curriculum option, they are brought to a screen with two options; Edit curriculum and Add new curriculum (Figure 5). The edit curriculum option allows the teacher to scroll through their curriculum and to edit any question they choose. That question will be updated to the database created and saved for future viewing. The add new curriculum option allows the teacher to add a new question with answers and wrong answer options. It also allows them to choose the language they would like and which difficulty the question falls under. This new question will be added to the database as well once done.

Chapter 4 Testing & Evaluation

4.1 Game Software Testing

In order to test the game features included in this project a suggestion on how to handle testing game development features by J. Rosenfield from their book, *Lean Game Development : Apply Lean Frameworks to the Process of Game Development,* was utilized and reformatted to handle this projects specification. An example of one of the tests completed is in Figure 6, and all the tests completed are located in Appendix C.

Edit Curriculum Feature

Feature	**User should edit a question and save the edits**		
Input	Mouse inputs	P	F
Tests	Check if the correct question was selected	X	
	Check if the question is editable	X	
	Check if the answer if editable	X	
	Check if the question and answer are saved in the database	X	
Output	If the user's edits were saved in the database, the test passed; otherwise, the test failed.		

Figure 6: Sample test of ProgGames **(Rosenfield Boeira, 2017)**.

The general idea of this test is to take the features drawn out in the requirements and create a set of tasks that specific feature should include in order to either pass or fail during implementation (Rosenfield Boeira, 2017). The test in Figure 6 will be used as an example to explain how these tests were completed.

- The test has a title which entails the overall feature, i.e., Edit Curriculum Feature.
- The feature section depicts what the overall feature should accomplish within the game.
- The inputs are any inputs the user must utilize to complete the test.
- The tests are any steps required within that feature that must be performed to complete a test. If even one of these tests fails, then the overall feature test fails.
- The output describes how to tell if the overall test has passed or failed.

Utilizing the tests within each feature gave me the opportunity to see what precisely was succeeding and what needed improvements throughout the development process. Once a feature was created, these tests were performed by completing each individual test and marking if they passed or failed. For instance, in Figure 6, all the tests have passed the development process, therefore, the test passed. Once all these tests passed, I then moved on to the next feature that was part of the requirements discussed in the requirements section in chapter 2. If any part of the test failed, I then went back to the code or the Unity editor and refactored them in order to achieve a passing test overall.

4.2 Evaluation

4.2.1 Procedure

The evaluations for this project were done two different ways: virtually and in person. This allowed for different kinds of people to complete this evaluation in order to gain the perspective of participants who are university students currently and those of whom are not now university students. Although evaluations were done virtually, they were planned so that those participants could call or skype with me while doing the evaluation in order to be available for any questions or problems that may occur. There was a total of 12 participants included in this evaluation, and they were set up in the following ways. The in-person ones included getting two users in the room at one time and going over the introduction script prepared beforehand. The virtual version required two people in one place but not with me physically there.

For both versions they were then given the opportunity to ask any questions and asked if they would like to participate in the experiment. They were then asked to read and sign a participation form before moving on to the demographic questionnaire asking general questions of each participant. They were then given their first task of the evaluation. Once they finished the tasks, the participants were given a questionnaire regarding the specific task they had just completed and a list of questions where the participants were asked to check if they had seen or answered the questions for the language and difficulty the chose. Within the questionnaire there were short answer, long answer, and rating questions for them to answer, and the ratings were based on a 1-7 scale rating.

In a real classroom there would be actual consequences to not learning or understanding the material, but with this evaluation, it would have been beneficial to have a test of some kind to know what questions they engaged with. Before starting this evaluation, a list of each question was entered into the evaluation questionnaire of all the questions they would see depending on the options they chose and the questions they answered were printed to a local file. Each participant was asked to enter a unique game ID in the form and in the game so that the information printed to a file would be traceable to their answers within the questionnaire. This will be used to see which questions they engaged with and which they did not by seeing which questions they remember from playing the game.

Once all the tasks and questionnaires were completed, they then completed an overall system evaluation and were read the debrief script and offered contact details again if they wanted them. A full list of questions asked and the introduction and debrief scripts are in Appendix D. It is vital to note there were technical difficulties when playing the multiplayer version of the game. If the network connection was not secure enough, then the game would disconnect, and the players would need to reconnect and start the game over. This caused some distractions for the participants that it happened to.

4.2.2 Participant Overview Results

Overall the 12 participants who participated in this evaluation had a range of prior knowledge of programming material as well as a wide range of video game and education-based video game experience or enjoyment. More than half of the

participants had played an education-based video game before participating with about 58%. While the others were either unsure if they had or had never played an education-based video game before this. This being noted the participants had a wide range of answers regarding how often they play video games. 16% either never play video games or they play them every day with the rest of the participants somewhere in between. Only 50% of the participants stated they really enjoy playing video games while the other 50% ranged from not enjoying them to moderately enjoying them. For the participants who had played an education-based video game they stated that they could make learning more fun or 'entertaining' but they can also have the tendency to be 'dry' or interruptive of the gameplay.

Among the participants only one of them prefers to study in a group while 25% have no preference for how they study and about 66% prefer to study by themselves. Now when specifically asked, if they were given the option of playing an education-based video game with classmates, about 91% of them chose by themselves overworking with classmates. Now when asked what their preferred study method is the majority of them stated they preferred 'Online Tutorials' while only one person chose 'Audio' and the other two chose 'Books/Paper.'

Java and C++ were among the most common languages between the participants with Ruby being the least prior used programming language of them all, and with 50% of the participants having no prior knowledge of programming it made it a little difficult to fully grasp the system's benefit to their educational outcomes. However, since this evaluation is not regarding the educational benefits but rather the effectiveness of engagement and flow between multiplayer and single-player education-based video games, this should not affect our understanding of this since their knowledge of the topics will not be evaluated.

4.2.3 Gameplay Results

Since there were technical difficulties within the multiplayer mode for one of the pairs, their gameplay data has been left out of the following calculations, therefore, for the following results there are only 10 total participants. During the single-player mode 60% and 70% from the multiplayer mode of the participants found the gameplay easy or very easy and the rest felt it was somewhere in the middle of easy and difficult. Based on this we can assume that the number of players did not affect the difficulty of gameplay. Now in the single-player mode the participants were asked if they were given enough time to complete the objectives within the game since there is a timer. This question was meant to correlate to the multiplayer aspect of seeing who can answer the questions first in the multiplayer mode as they can only be answered by one player.

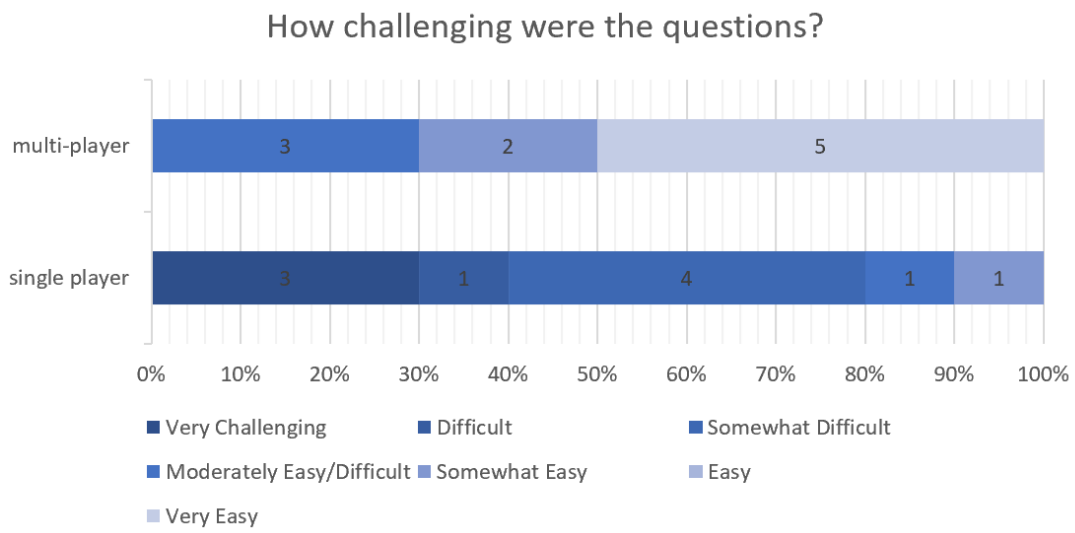


Figure 7: Graph showing the difficulty of questions perceived by participants between multiplayer mode and single-player modes.

In the single-player mode, most of them felt they had enough time where 40% 'strongly agreed' and 30% just below 'strongly agreed.' However, the multiplayer mode seemed to bring more issues to the overall education-based gameplay than expected. More than half the participants agreed that they 'did not care if they got the questions right, as long as I beat my opponent' and only one participant below the halfway point and two participants right at the halfway point of the 1-7 scale. However, when asked if there was anything that surprised them both during the questionnaire and after evaluation comments in person, most of them stated how fun it was to play against an opponent. They enjoyed having that intractability aspect within the education-based game but based on their answer regarding their engagement of the questions it makes it difficult to gauge their real understanding of the material.

This is where the participants' memory of the questions in the game will be of use to test their engagement with the questions between single and multiplayer modes. In order to calculate the percentage of questions remembered from the single and multiplayer views the total number of questions between all participants was calculated. Then the total number of remembered questions was added for both single and multiplayer mode. The remembered questions were then divided by the total questions and multiplied by 100 for the percentages located in Figure 8. These calculations and results show that while the participants do enjoy playing with other people in and education-based video game, they do not necessarily focus or engage with the questions as fully as they do when playing by themselves. It should also be noted that, as shown in Figure 7, the participants felt that the questions were also more difficult in the multiplayer mode versus the single-player mode. Considering most of the participants chose the same settings for both modes, therefore the questions themselves had the same difficulty levels, this could be from less engagement with the material with the added distractions of the opponent they are playing against. Which means that having that extra challenge of multiplayer mode negatively impacts the focus needed to engage the material.

As mentioned earlier half of the participants have no prior knowledge of programming and therefore the challenge of the questions is understood to likely be more difficult than if they had previous experience with programming. With that in mind, it is interesting that during the single-player mode the participants felt the questions were more challenging than in the multiplayer mode. This could be that in the multiplayer mode it is likely that they did not see all of the questions as they disappear of their opponent answers it first, but it could also be that because they were more focused on winning versus answering the questions correctly, as previously discussed, they felt there was less challenge to the questions themselves.

10 participants x 8 questions each = 80 total questions
67 remembered questions for single-player mode
46 remembered questions for multiplayer mode
Single-mode: 64 / 80 * 100= 80%
Multiplayer mode: 46 / 80 * 100 = 57.5%

Figure 8: Calculation of questions remembered from single and multiplayer modes.

4.2.4 Teacher and Overall System Results

When evaluating the teacher's views within the system most of the participants, 58%, felt that editing the curriculum was easy and that adding new curriculum was mostly easy with 58% selecting slightly under agreeing it was easy and 16 percent completely agreeing it was easy. They did suggest some sort of feedback as to whether the edits were saved successfully as well as the fact that the dropdown options were very small and hard to see. During the task of finding a student the most common question received was 'do we click on them?'. This was a sure-fire way of knowing that this view was confusing for the participants as the student's icons do not look like they are clickable and are hard to see since the color scheme if dark grey on black. The student progress screen's scrolling was backward compared to the other scrolls structures in the system that added more confusion. Since this view had these issues the participants felt that finding a specific student was mostly easy but still difficult and that the view was hard to understand at times.

Overall the participants responded that the ease of the system was moderate to very easy to use and that they moderately to strongly agreed the information provided was useful to them in some way. The majority of the participants felt the information provided was moderate to very easy to find with one participant disagreeing. They also felt the system was easy to moderately challenging to navigate, with one participant feeling it was difficult and four stating it was very easy. The same situation is seen when asking participants if they would use a system like this to learn a new topic.

When asking the participants which parts of the system they liked and disliked, they mostly had good things to say while also adding suggestions on how to improve the system overall. The main things they felt were that this could be a fun way of learning and that the gamification of the questions was encouraging and enjoyable. They also suggested that changing some of the color schemes and font consistencies could really help the system and the game become more comfortable to read through and understand.

Chapter 5 Conclusion

Within this project there are three main things to see and do; Play a single-player education-based video game, play a multiplayer education-based video game with peers or friends, and handle curriculum for the student's in a teacher's class. At this stage of the project each of them has their strengths and weaknesses, but they all could grow and become part of a system that could help students learn the material in a classroom setting, but in a non-traditional way. I was able to create both a single-player and a multiplayer version of the game where users could challenge their knowledge of programming material they already knew or had never seen before. I also created a teacher side of the system where they could check on their student's progress or change their curriculum to be less confusing or better based on their student's work.

Throughout the project's views there are many issues regarding the user interface, however, there were also some strong points within the interface itself. When logged in as a teacher the view to select a specific student does not look like to can click on the students and therefore, during evaluations, I received many questions as 'do I click on one?' or 'can I click on one?'. When adding a new question, it is challenging to see which options you are choosing from the two drop-down lists provided which caused some confusion as well. Given the previous examples one could tell that most of the interface issues that came to be were when viewing the teacher views which is something I would have liked to do better as there are inconsistent font sizes and the icons are inconsistent as well.

However, even with these shortcomings, the ability to look through student's progress for that specific teacher's curriculum and the ability to edit and add curriculum is something I am proud of as I have not had the opportunity in the past to utilize a database within a game or even at all. Having the opportunity to include this within my project was a highlight for me primarily since it doesn't just utilize it in the teacher's views, but the database is also utilized within the student's views as well.

The previous sections discussed some of the drawbacks of the system and the strong points within the system, but, overall, I learned a lot about how to incorporate a database into a video game, creating a local multiplayer game, client versus host versus server and even learned more about the C# programming language and Unity's game engine. These are all things I am very proud of while also knowing that there are a lot of improvements that could be made to the game and I look forward to making those improvements and finding more improvements needed after that.

5.1 Future Work

5.1.1 Feedback for Users

Throughout the system created there are many opportunities for feedback that the user could potentially utilize. Currently when a player answers a question there is no immediate feedback of whether they got the answer right or wrong. I created it this way since in personal experience seeing immediate feedback of answering

questions wrong can be a little discouraging, and for this project, I did not want to players to be discouraged in any way regarding the questions themselves. However, they do get to see their results after playing through the game in order to give them a chance to look through that information for as long as they would like.

Once the students have finished playing the game a results page appears. I would have liked this page to incorporate more information regarding the game itself and more about the questions as well. Regarding what information is already there, I am happy that it includes not only the question and correct answer, but it also includes a short explanation as to why that answer is correct and the answer the player chose. This was important to me as the player may see the correct answer but, if the player does not remember the answer, they chose then they may not fully understand how the explanation connects to their choice. I would have liked to include more information as to actual gameplay. Such as what questions their opponent got wrong or source links to more information to the question topics. All of these features could be updated or looked at more deeply in future work as well, but overall, I would have liked to plan more time for development and testing of the project and to possibly have a test group in order to test everything before going into evaluations.

Another area where feedback would be useful would be when editing or adding a new question to a teacher's curriculum. Currently, there is nothing notifying the user that their changes were saved, therefore, they may be unsure if it was successful or not.

5.1.2 Single-player Story Mode and Multiplayer Challenge Mode

In the original version of the project that I wanted to do, it would have included a complete storyline within a single-player depiction of the game and a multiplayer version where users could challenge each other on the knowledge they had learned in the single-player mode. This version turned into something a little different as the goal of the current version is to understand if a single-player or multiplayer version is best to enhance the challenges and flow needed to positively impact education-based learning. This drawback was mostly due to time as implementing a database and scripts dependent on client and hosts took up a large portion of the development time. The storyline implementation could potentially be utilized to learn new material as the player would have more time and a reason to learn through the story, while the implementation created in this project turned out to be more of a reinforcement of previously known knowledge. This is seen based on the results from the evaluations as most of the participants who had no prior knowledge of programming had the most difficulty with the questions and those who have had experience with programming did not have as many challenges as the others.

5.1.3 Incorporating Database Game Updates

Incorporating the database in the student's games was a challenge as the only questions that needed to appear for each student should be questions that are from their teacher and the programming language and difficulty that they chose. There are some shortcomings as the current set up does not change the database to update the questions as being complete or incomplete when the student answers the questions. This was implemented that way currently as for evaluations the

questions needed to be the same for everyone who chose the specific settings, but there is space in the code currently that would allow for the opportunity to create that capability. This has been noted in the code where a user either gets the questions right or wrong.

5.1.4 Teacher Capabilities

There are some capabilities for the teacher view that I knew I would most likely not have time for but think would be beneficial to a system like this overall. Those capabilities include the following:

For a teacher to put in the new curriculum, one at a time could take them a very long time. Therefore, I believe having the capabilities to choose a file of some sort to parse through and create the curriculum for a large number of questions would be less time consuming and potentially create a more positive use of the system for them.

Another opportunity for a positive response would be to add students to their class rosters. All at once or one at a time. The ability to add or remove student's one at a time would be valuable as student's move and change schools all the time so for a teacher to be able to have this feature would be beneficial to their student's no matter when they join the class.

Chapter 6 References

1000 Java MCQs for Freshers & Experienced | Sanfoundry. (n.d.). Retrieved August 1, 2019, from https://www.sanfoundry.com/java-questions-answers-freshers-experienced/

1000 Python MCQs for Freshers & Experienced | Sanfoundry. (n.d.). Retrieved August 1, 2019, from https://www.sanfoundry.com/1000-python-questions-answers/

Admiraal, W., Huizenga, J., Akkerman, S., & ten Dam, G. (2011). The concept of flow in collaborative game-based learning. *Computers in Human Behavior, 27*(3), 1185 - 1194.

Albert, B., Tullis, T., & Tedesco, D. (2010). Beyond the Usability Lab | ScienceDirect. *Sciencedirect.com*.

Barnum, C. (2011). Usability Testing Essentials | ScienceDirect. *Sciencedirect.com*.

Bayat Games. (n.d.). Free Platform Game Assets. *Unity Asset Store*. Retrieved August 4, 2019, from https://assetstore.unity.com/packages/2d/environments/free-platform-game-assets-85838

Hamari, J., Shernoff, D., Rowe, E., Coller, B., Asbell-Clarke, J., & Edwards, T. (2016). Challenging games help students learn: An empirical study on engagement, flow and immersion in game-based learning. *Computers in Human Behavior, 54*, 170 - 179.

Plass, J., Homer, B., & Kinzer, C. (2015). Foundations of Game-Based Learning. *Educational Psychologist, 50*(4), 258-283.

Rosenfield Boeira, J. (2017). An Inception in Practice. In J. Rosenfield Boeira, *Lean Game Development : Apply Lean Frameworks to the Process of Game Development* (pp. 23-32). Berkeley, CA: Apress.

Sung, H.-Y., & Hwang, G.-J. (2013). A collaborative game-based learning approach to improving students' learning performance in science courses. *Computers & Education, 63*, 43 - 51.

Unity - Manual: Network Lobby Manager. (2019). *Docs.unity3d.com*. Retrieved August 1, 2019, from https://docs.unity3d.com/Manual/class-NetworkLobbyManager.html

Unity Technologies. (n.d.). Network Lobby. *Unity Asset Store*. Retrieved August 4, 2019, from https://assetstore.unity.com/packages/essentials/network-lobby-41836

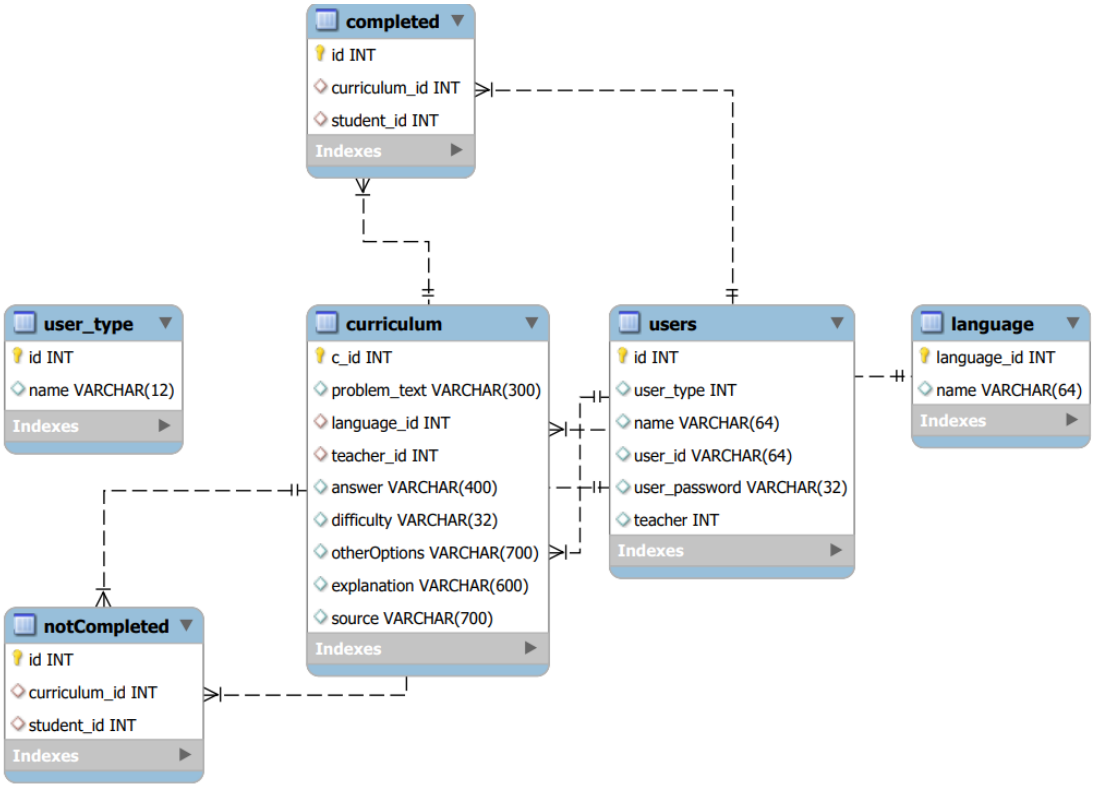
Appendix A <Requirements>

MoSCow Requirements

Must have	Should have
Students must have the option of at least two different programming languages	Students should be able to log in
Students must have at least a beginner option	Teachers should be able to log in
Students must have the option to play against an opponent/classmate	Students should be able to play by themselves
Teachers must be able to view curriculum	The database should store teacher and student login information
Teachers must be able to view their student list	The database should store student's progress
The database must store the teacher's curriculum	Student's should be able to view their results from the game
The game must have objectives for the students to complete	Teachers should be able to add new curriculum
Teachers must be able to edit their curriculum	

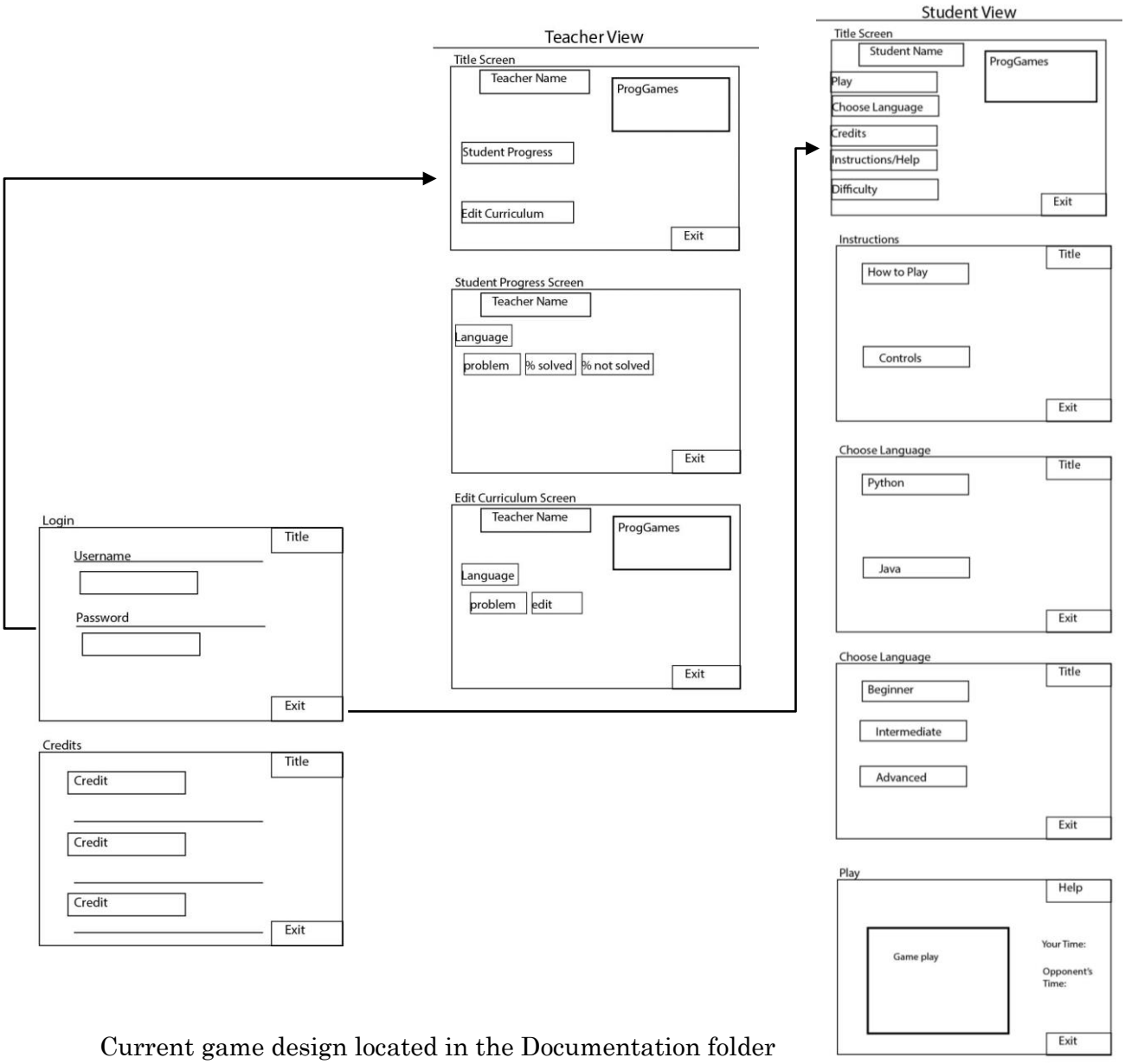
Could have	Would like to have
Teachers should be able to view individual student's progress	A Storyline for the single-player game
Students could have the option of more than beginner difficulty	Enemies to create a more challenging game experience
More than two programming languages	More and different objectives for students to play through
Students could have the ability to request help from a teacher on specific questions/topic.	A leveling system for the students to see their progress through multiple playthroughs

E-R Diagram of SQLite Database



Appendix B < Design >

Wireframes/Initial Design



Current game design located in the Documentation folder included with project.

Appendix C <Testing>

Teacher Login Feature

Feature	**User should be able to log in and enter the Teacher's view**		
Input	Keyboard and Mouse inputs	P	F
Tests	Check username input is in database	X	
	Check password input correlates to username and is in database	X	
	Check if the username is a Char string	X	
	Check if scene changes to Teacher's view	X	
Output	If the user's scene is what's expected, test passed; otherwise, the test failed.		

Student Login Feature

Feature	**User should be able to log in and enter the Student's view**		
Input	Keyboard inputs	P	F
Tests	Keyboard and Mouse inputs	X	
	Check username input is in database	X	
	Check password input correlates to username and is in database	X	
	Check if the username is an Integer	X	
Output	If the user's scene is what's expected, test passed; otherwise, the test failed.		

Answer Question Feature

Feature	**The player should answer questions on the screen**		
Input	Mouse inputs	P	F
	Check if the answer is selected	X	
	Check if question disappears from player's view	X	
Output	If the player's answer is what's expected, test passed; otherwise, the test failed.		

Add Curriculum Feature

Feature	**The player should add new questions and answers to the database**		
Input	Mouse and Keyboard inputs	P	F
Tests	Check if Question is entered	X	
	Check if Answer is entered	X	
	Check if the Programming language is selected	X	
	Check if Difficulty is selected	X	
	Check if Add Question Button is selected	X	
Output	If the player's expected question and answer are in the database, test passed; otherwise, the test failed.		

Edit Curriculum Feature

Feature	**The player should edit a question and save the edits**		
Input	Mouse and Keyboard inputs		
		P	F
Tests	Check if the correct question was selected	X	
	Check if the question is editable	X	
	Check if the answer if editable	X	
	Check if the question and answer are saved in the database	X	
Output	If the player's edits were saved in the database, test passed; otherwise, the test failed.		

Student Progress Feature

Feature	**The player should view a student's progress**		
Input	Mouse inputs	P	F
Tests	Check if student's Java progress is scrollable	X	
	Check if student's Python progress is scrollable	X	
Output	If the student's progress is what's expected, test passed; otherwise, the test failed.		

Select Student Feature

Feature	**The player should select a student to view progress**		
Input	Mouse inputs	P	F
Tests	Check if student list is viewable	X	
	Check if a student is selected from the list	X	
	Check if student progress scene is reached	X	
Output	If the player's selection is what's expected, test passed; otherwise, the test failed.		

Walking Feature

Feature	**Character Should Walk on Horizontal-Axis**		
Input	Mouse inputs	P	F
Tests	Check which key is pressed	X	
	Check when A or Left direction key is pressed, and the character moves to the left.	X	
	Check when S or Down direction key is pressed, and the character moves down.	X	
	Check when D or Right direction key is pressed, and the character moves to the Right.	X	
	Check when W or Up direction key is pressed, and the character moves to the left.	X	
	Check the other keys do not move the character.	X	
	Check if the character collides with the screen boundaries.	X	
Output	If the player's positions are what's expected, test passed; otherwise, the test failed.		

Question Feature

Feature	**The player should answer questions on the screen**		
Input	Mouse inputs	P	F
Tests	Check if the player collides with the 'treasures'	X	
	Check if the question is viewable	X	
	Check if answers are viewable	X	
Output	If the question is what's expected, test passed; otherwise, the test failed.		

Results Feature

Feature	**The player should see their results from the game**		
Input	GameManager.cs inputs	P	F
Tests	Check player's selected answer is stored in GameManager	X	
	Check player's questions are stored in GameManager	X	
	Check question's explanation is stored in GameManager	X	
	Check questions the correct answer is stored in GameManager	X	
	Check results are scrollable	X	
Output	If the player's results are what's expected, test passed; otherwise, the test failed.		

***Testing Format (Rosenfield Boeira, 2017)**.

Appendix D <Evaluations>

Full evaluation documentation included in the submitted project folder. The following questions were created in part by utilizing the following sources to formalize questions needed for this project: (Albert, Tullis, & Tedesco, 2010) (Barnum, 2011).

Demographic Questions

1. How often do you play video games? (Scale 1 – 7)
2. Have you ever played an educational video game before?
    a. If yes, what did you like/dislike about them?
3. What is your level of prior experience with programming?
    a. Less than a year, 1-2 years, 2-4 years, 4 or more years, None
4. Which languages have you worked with and how long have you worked with them?
5. Do you prefer studying alone or with a group?
6. What is your preferred study method?
7. You enjoy playing video games. (Scale 1 – 7)

Single Player

1. Given the option to play an educational video game by yourself or with classmates, which would you choose? (Scale 1 – 7)
2. How challenging did you find the gameplay? (Scale 1 – 7)
3. How challenging did you find the questions asked within the game? (Scale 1 – 7)
4. How interesting did you find the questions asked? (Scale 1 – 7)
5. Did you encounter/experience any distractions while playing?
    a. If yes, what did you find distracting?
6. Which settings did you choose?
7. I had enough time to complete the objectives given to me. (Scale 1 – 7)
8. Was there anything about the experience that surprised you?
9. I understood the questions in the game.

Multiplayer

    a. The game was: (very easy to very difficult: 1-7)
2. The questions were: (very easy to very difficult: 1-7)
3. How interesting did you find the questions asked?
4. Did you encounter/experience any distractions while playing?
    a. If yes, what did you find distracting?
5. Which settings did you choose?
6. Based on your experience in Tasks 1 and 2, would you prefer to play an educational video game with classmates or by yourself?
7. Playing against an opponent motivated me to complete the game faster. (1-7)
8. I did not care if I got the questions right, as long as I beat my opponent (1-7)
9. I was satisfied with how easy the game was to play. (1-7)
10. Was there anything about the experience that surprised you?

Teacher

1. Editing the curriculum was: 1-7 (Easy – Difficult)
2. The curriculum view was easy to understand. 1-7 (Disagree – Agree)
3. Creating a new curriculum was easy. 1-7 (Disagree – Agree)
4. Finding a specific student was easy to do. 1-7 (Disagree – Agree)
5. The student's progress view was easy to understand 1-7 (Disagree – Agree)
6. Logging into the system was not difficult. 1-7 (Disagree – Agree)
7. Is there anything you would improve in the Teacher's view?

Overall System Questions

1. I was satisfied with how easy the system was to use. 1-7 (Disagree – Agree)
2. The information provided was useful. 1-7 (Disagree – Agree)
3. The information provided was easy to find. 1-7 (Disagree – Agree)
4. The system was easy to navigate. 1-7 (Disagree – Agree)
5. I would use a System like this to learn a new topic. 1-7 (Disagree – Agree)
6. Where there any parts of the system you liked/disliked?