**DigiPen Academy Studio Project Proposal**

**Math Game (Name TBD)**

9/27/2021

**STUDENT**

Gryphon McLaughlin

gryphon.mclaughlin@digipen.edu

**INSTRUCTOR**

Jeremy Kings

**PROJECT DESCRIPTION**

I plan to use the Xamarin.Forms framework to create a math game that I’ve prototyped in C before. This project is interesting to me because I think that this game would be something cool to share with the public and possibly make an income through.

**PROJECT LEARNING OBJECTIVES**

The most important learning objective from this project is working with Xamarin.Forms. I believe that understanding how to create mobile apps and games could help me branch out from only working on computer apps and games.

Once the game is finished and working properly I might consider finding a way to display advertisements within the app, but that’s not a certainty.

**PROJECT DELIVERABLES**

The final deliverable for this Studio Project, is a mobile build of the math game that has a home screen, a game setup screen, a game screen, a results screen, and a settings/help screen. This includes a full randomizer that dynamically chooses the numbers based on the previous numbers, a robust way to read the equation given by the user, and a way to play the game with multiple different settings. It may not be completely ready to be shipped at this point, but the base game should be prepared for testing.

**PROJECT RESOURCES**

Materials and Equipment: Computer, Phone to test app on (or an emulator), Maybe a way to test iPhones.

Software: Visual Studio, Xamarin.Forms, Possible Phone Emulator

Instructional Resources: Xamarin.Forms tutorial (currently using Microsoft Learn), Xamarin.Forms documentation, Xamarin.Forms resources. If a faculty member is familiar with Xamarin, they may also be helpful.

**SCHEDULE**

The schedule is a daily breakdown of specific topics, tasks, and deliverables that you will use to measure progress with the project.

|  |  |
| --- | --- |
| **Day** | **Tasks and Topics** |
| 1 | Define project, identify learning objectives, identify resources Completion and approval of individualized project proposal |
| 2 | Finish the Microsoft Learn Course Review the course to ensure understanding of topic |
| 3 | Pseudocode for how the UI will interact with the internal functionality of the game. |
| 4 | Finish the pseudocode and start work on internal functionality Should have a basic randomizer set up as well as the ability to play a game. Essentially have the C version working in C#. |
| 5 | “Alpha” (Adjustable date based on project) - Effectively the proof of concept and pipeline. Whatever base knowledge and skills that a student will need to execute the project must be evident. |
| 6 | Strengthen the randomizer, possibly add randomizer “difficulty” where the randomizer helps, is random, or doesn’t help. All difficulties will decrease the chance of 0’s drastically. |
| 7 | Create UI and work on the Equation reader to optimally understand the equation and execute order of operations correctly. |
| 8 | “Beta” (Adjustable date based on project) - “Content” is complete and implemented |
| 9 | Polishing and bug fixing. Clean up UI and possibly create art for it. “Beautify” the app. |
| 10 | Final project submission and presentation with post-mortem |

**PROJECT RISKS AND MITIGATION**

* Project could quickly become cluttered and difficult to understand. The C version of this game did so and refactoring became a monumental task.
  + I plan to not only heavily use pseudocode to prevent clutter from happening, but also plan to use C# features such as function overloading and default parameters to clean up things that did not have simple solutions in C.
* UI may be difficult to get right or link properly to the internal code, especially since I am not used to working in XAML, the markup language that Xamarin uses.
  + I can make sure I reinforce my understanding of XAML and will research ways to get certain effects to work properly.
  + I can also find examples similar to what I’m trying to create so I can use templates to get a head start on parts of the UI.
* Changes to the randomizer could make it less random and only provide a few identical sets of numbers after much use.
  + I can repeatedly test the randomizer and save the output of using it multiple times to see if there is a pattern in the first place.
  + If there is, dial back the changes and instead make the numbers random, with stops to prevent duplicates and 0’s, but not change based on how the numbers interact.

**EVALUATION:**

The end goal of this project is to have a clean and functional Xamarin project that runs the math game. Not only do I want the code to be clear and easy to understand, but I would also like it to be able to comfortably run on a mobile device. If possible, I would like it to look like a program that could be sent out to beta testers if there were any interested in testing it.

Rubric:

|  |  |
| --- | --- |
| Topic | Expectations to Meet Requirements |
| Documentation/Readability | Clearly commented where necessary Self-documenting code No magic numbers Function and File Headers |
| Reasonably run on a mobile device | Optimized Code that doesn’t have extra variables or loops that could be reasonably done simpler. No strange workarounds that don’t make sense |
| Could be sent to Beta Testers | Screens don’t take too long to load XAML is used efficiently to create proper screens No obvious graphical bugs. |