Quiz Game

James Blankenship and Vrushank Mali

Abstract

We are writing about the group project for CSCI 352. The project will be about a short quiz game. It will also give information about a topic.You will be able to make quizzes and add them to a database for storage. You will be able to later edit the quiz.

1. Introduction

The project will be a simple quiz. The group will be trying to make an application that will work as a quiz that will shuffle through questions. The project will have a decent amount of questions and when you choose an answer it gives you information about the topic of the question.The questions will come from random topics and . The target for the project is people wanting to take quiz or find out information about a topic. The target audience will get information about topics when using the program.The target audience will be younger people so it will have likely simpler questions.The ability to add quizzes will like broaden the audience age diversity.

1.1. Background

There are not currently any terms that need to be explained. The reason for this idea is we wanted something information based and after running through some ideas decided on this on.The reason for deciding on this game was that it is a very broad program concept.For example the quizzes could have a lot of diversity in the question.You could have a quiz on history and another on on music. The game will be a general knowledge game. The game will have a timer as you go through the questions. It will have different difficulty levels.It will have a difficulty pattern similar to multiple choice.

1.2. Impacts

There are not a lot of impacts for the project besides the grade for the project. If there was a impact it would be the learning that would come out of it.The impacts could be that people learn new information.

1.3. Challenges

Classes would be them main problem getting them all connected and dealing with the database.Certain functions such as a timer might be a difficult thing to figure out. The easier part of the project will likely be getting the overall design of what we want down.If we get to the stretch goals use links to give information might be a challenge.

2. Scope

The scope of this project will be having a program that has multiple hard coded questions that will tell you if you got the question right. It will also tell you information about the topic in question.The stretch goals would likely be adding links, but we do not fully know how that works.The other stretch goal will be adding more questions.

2.1. Requirements

As part of fleshing out the scope of your requirements, you'll also need to keep in mind both your functional and non-functional requirements. These should be listed, and explained in detail as necessary. Use this area to explain how you gathered these requirements.

2.1.1. Functional.

- User needs to choose difficulty.
- User needs to have the ability to answer questions, and get information on topic.
- User needs to be able to make their own quiz.
- User needs to be able to store their quiz in a data base.
- User needs to be able to keep track of how many questions they got right.
- User needs to be able to edit the quiz they have added, or remove it.

2.1.2. Non-Functional.

- Users quiz must be stored in the data base.
- Users should be able to see the interface and interact with it.
- The quiz should stay in the database until removed.
- The quiz should function as a quiz where you get a question and you answer it.

2.2. Use Cases

Here are some brief examples of how the game should work.

Use Case Number: 1
Use Case Name: Player starts a new game
Description: A player starts the game in easy difficulty. Player will click on "Easy" button. This will load the easy level game page with first question and options for answer on the screen.

1) Player loads the game which will load the start menu on the screen.
2) Player will left-click on the "Easy" button.
3) The easy difficulty game will load on the screen with first question and the options for answer on the screen with a timer.
4) Player will choose one of the options for the answer by left-clicking on the button.
5) If the selected answer is correct, it will turn the answer button "green" and will show a dialogue box with more information about the topic which was in the question.

Termination Outcome: If the timer ends for first question, it will skip the first question and will show second question.

Use Case Number: 2
Use Case Name: Pause
Description: If the player wants to pause the game, player will click on the pause menu and it will pause the timer and will show the pause menu.

1) Player will left-click on the "Pause" button, which will open the pause page on screen.
2) Clicking pause button will pause the timer and will show the pause menu.
3) The pause menu shows "Resume", "Restart", "Volume", and "Exit" options for the player to choose.

Use Case Number: 3
Use Case Name: Restart
Description: If the player wants to restart the game, player will click on the restart button and it will restart the whole game with same difficulty and reset score and time.

1) After game getting paused by the player, the player left-clicks on the "Restart" button.
2) This will restart the whole game with the same difficulty level, reset score and time.
3) The player continues to play the game again.

You will then need to continue to flesh out all use cases you have identified for your project.

2.3. Interface Mockups

These are screen shots we made as interface mockups.

3. Project Timeline

Go back to your notes and look up a typical project development life cycle for the Waterfall approach. How will you follow this life cycle over the remainder of this semester? This will usually involve a chart showing your proposed timeline, with specific milestones plotted out. Make sure you have deliverable dates from the course schedule listed, with a plan to meet them (NOTE: these are generally optimistic deadlines).

4. Project Structure

At first, this will be a little empty (it will need to be filled in by the time you turn in your final report). This is your chance to discuss all of your design decisions (consider this the README's big brother).

4.1. UML Outline

Show the full structure of your program. Make sure to keep on updating this section as your project evolves (you often start out with one plan, but end up modifying things as you move along). As a note, while Dia fails miserably at generating pdfs (probably my fault), I have had much success with png files. Make sure to wrap your images in a .

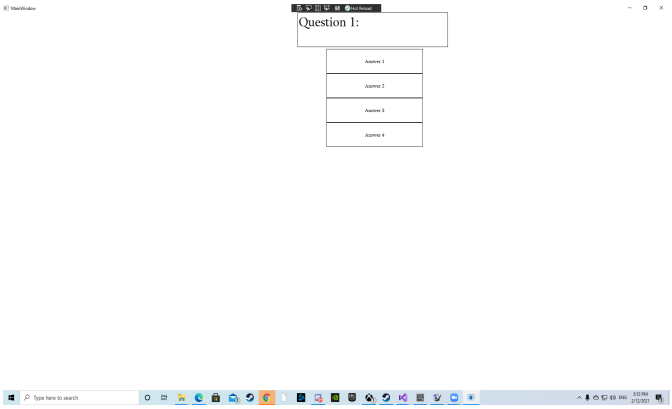

Figure 1. Interface 1

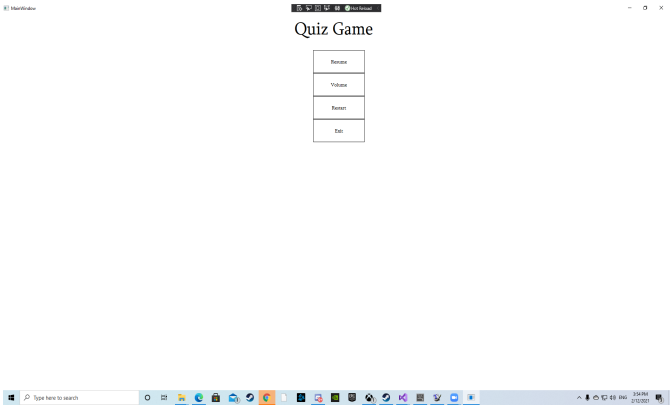

Figure 2.

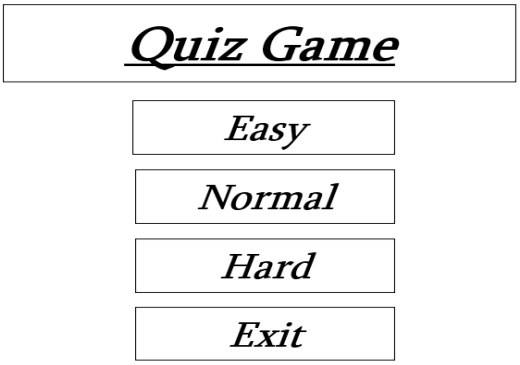

Figure 3. start menu

4.2. Design Patterns Used

Make sure to actually use at least 2 design patterns from this class. This is not normally part of such documentation, but largely just specific to this class – I want to see you use the patterns!

5. Results

This section will start out a little vague, but it should grow as your project evolves. With each deliverable you hand in, give me a final summary of where your project stands. By the end, this should be a reflective section discussing how many

of your original goals you managed to attain/how many desired use cases you implemented/how many extra features you added.

5.1. Future Work

Where are you going next with your project? For early deliverables, what are your next steps? (HINT: you will typically want to look back at your timeline and evaluate: did you meet your expected goals? Are you ahead of schedule? Did you decide to shift gears and implement a new feature?) By the end, what do you plan on doing with this project? Will you try to sell it? Set it on fire? Link to it on your resume and forget it exists?

References

[1] H. Kopka and P. W. Daly, *A Guide to LaTeX*, 3rd ed. Harlow, England: Addison-Wesley, 1999.