SYSTEMS REQUIRMENT DOCUMENT

The Unstoppable Banana-Mobile

MCKENZIE STATON, JAYSON WILLIAMSON, SENGTHIDA LORVAN, AND PATRICK KSOR

Author: McKenzie Staton

1. INTRODUCTION

1.1 TITLE

1.1.1 PROJECT NAME

The Bananas Trivia Game

1.1.2 TEAM NAME

The Unstoppable Banana Mobile

1.1.3 DATE LAST UPDATED

04/27/21

1.1.4 TEAM MEMBERS

Jayson Williamson, McKenzie Staton, Patrick Ksor, and Sengthida Lorvan

1.1.5 HONOR CODE

WE HAVE ABIDED BY THE UNCG ACADEMIC INTEGRITY POLICY ON THIS ASSIGNMENT.

1.2 TABLE OF CONTENTS

1.3 PURPOSE

To create a functional and fun Trivia game as a desktop application using Java, JavaFX to create a GUI, and using the OpenTriviaDatabase API to generate questions and answers. To save player data (like score) we are using a MySQL database as our persistent data storage.

1.4 DOCUMENT CONVENTIONS

API stands for Application Programming Interface and allows you to access offsite data to use in projects.

We are using the required style guide for this CSC340 class, CSC340 Style Guide.

1.5 INTENDED AUDIENCE

The intended audience of our project is anyone wanting to test their knowledge on a multitude of subjects in a fun way. They do have to have previous knowledge related to some of the categories like the Anime & Manga category. The General Knowledge category will have questions anyone can answer, and with the adjustable game difficulty players will never run out of challenges.

1.6 DEFINITIONS/JARGON

API: Application Programming Interface

GUI: Graphic User Interface

SOLID: a set of software design principles that are utilized to make the software understandable, flexible, and maintainable.

MVC Architecture: separating the logical components of the program into models (actual code of a program), views (UI logic/code for what the user will see), and controllers (the middlemen between the two).

MySQL database: a relational database that stores information

JRE: Java Runtime Environment

JDK: Java Development Kit

1.7 PROJECT SCOPE

Create a complete desktop application using Java that connects to a third-party API, and has a component of persistent data storage. Developers will not have to worry about marketing, deployment, or maintenance after development. Project must conform to SOLID principles, MVC architecture, and other Software Engineering techniques.

1.8 TECHNICAL CHALLENGES

Most of the technical challenges that this group has experienced lie within the GUI. This project is the first time during our computer science schooling where we were tasked with creating a user interface, so this means we have never worked with JavaFX before. Another aspect we were unfamiliar with was the persistent data storage though some in the group have written in SQL before.

1.9 REFERENCES

CSC340 lectures, previous class experiences, online tutorials for JavaFX.

2. OVERALL DESCRIPTION

2.1 PRODUCT FEATURES

Our project allows the selection of many different settings to customize the player's experience. These settings being the length of the game, the difficulty of the game, player name, and the category of questions being asked. The categories include General Knowledge, Video Games, Music, Science & Nature, Computers, Film, and Anime & Manga.

The leaderboard will take player data (length, difficulty, name, and category played) that is sent to a MySQL database and present for any user to see.

2.2 USER CHARACTERISTICS

People who are using this application want to test their knowledge in a fun way no matter their knowledge of the subject matter. They also could be using the application to check out the scores of previous players as well.

2.3 OPERATION ENVIRONMENT

This project is expected to work on Windows 10, but because all that is needed is the ability to run Java application it should run on most computers and operating systems.

2.4 DESIGN AND IMPLEMENTATION CONSTRAINTS

We decided to limit the types of questions available to the user to be multiple choice, at first because of the many variables when dealing with string-based user input. Second, even though the API gives us the potential to ask true false questions we wanted to keep a consistent look of the GUI.

2.5 ASSUMPTIONS AND DEPENDENCIES

Assumes that the user knows about the category they have chosen. We do not control the questions given to the user, except by the specified settings determined by the user at the setup stage.

3. FUNCTIONAL REQUIREMENTS

3.1 PRIMARY

When setting up the Trivia game the user needs to make sure to enter all the settings necessary. If any of the game settings are not selected the API call cannot be made to get the contents of the game. Connection to the API is also necessary to receive the information needed. The game settings along with the player score and name are sent to our MySQL database at the end of the game

3.2 SECONDARY

Our leaderboard allows us to view previous players' scores to compare against their own. The leaderboard also can be sorted based on many different attributes like category, difficulty, score, and name. We do this by querying our MySQL database and displaying the result through the leader board.

4. TECHNICAL REQUIREMENTS

4.1 OPERATING SYSTEMS/COMPATIBILITY

The project can be run on all modern operating systems and there should not be any difference in gameplay on these different operating systems. The only requirements are the connection to the API, connection to the database, and the ability to run Java executable files.

4.2 INTERFACE REQUIREMENTS

4.2.1 USER INTERFACE

The start screen has three options: Set-up questions (Game Settings), View Leaderboard, and quit.

- Game Settings has four parts that must have selections before you can continue: Category, Game Length, Game Difficulty, Name.
    - The category has a drop-down menu with options: General Knowledge, Music, Video Games, Science & Nature, Computers, Film, and Anime & Manga.
    - Game Length is another drop-down menu with options: 10, 20, and 30.
    - Game Difficulty is another drop-down menu with only three options: easy, medium, and hard. Note: These difficulty levels are determined by the Admins that look over questions being submitted to the API, not members of our team.
    - Name is a text box that requires input from the keyboard

Then after all the settings are selected there is a button to begin the quiz. Or there is an option to return to the previous screen (Start Screen).

- The Quiz screen displays the Question at the top. Then the answer choices are displayed above their associated buttons. Selecting the answer choice is done by clicking the button associated with the answer.
- Once the game is played the player's score will be displayed on the screen.
- The Leaderboard displays the High Scores as a table with the Player's name, the category they chose, their game length, game difficulty played, and their score.
    - There is also a button to return the previous screen in this one, it returns you to the Start Screen.

4.2.2 HARDWARE INTERFACE

The hardware requirements for this application are a keyboard and mouse to enter and click user selections. Also needed is an internet connection to connect to both the API and database.

4.2.3 SOFTWARE INTERFACE

The software needed for this project is Java 16 as that is the JDK used for this application.

4.2.4 COMMUNICATIONS INTERFACE

GET requests are used to connect the API and retrieve the information specified by the user, this information with then be formatted for our use. The database is connected via the JDBC, which allows us to query the MySQL database to form the leaderboard.

5. NONFUNCTIONAL REQUIREMENTS

5.1 PERFORMANCE REQUIREMENTS

Requirements for this application are:

OS: Any modern OS that can run Java executable files

JRE: JDK 16

5.2 SAFETY/RECOVERY REQUIREMENTS

Any unprecedented breakdown or unexpected crash shouldn't affect the subsequent runs of the application. But player data may not have been sent to the database if the crash happened before the completion of the quiz.

5.3 SECURITY REQUIREMENTS

No login but player data (name, game length, game difficulty, category chose, and score) sent to the MySQL database.

5.4 POLICY REQUIREMENTS

Must use GitHub for versioning control. Must incorporate a 3rd party API in the creation of the project. The project must follow the following software requirements Dependency inversion (SOLID principles), Adherence to the Style Guide, Proper use of Enumeration, MVC Architecture, Separation of Concerns, and Loose Coupling.

5.5 SOFTWARE QUALITY ATTRIBUTES

5.5.1 AVAILABILITY

This application should always be accessible once downloaded. Exceptions to that are if the Open Trivia Database goes down to no fault of the project team members. If the necessary internet connection does not exist, the application will work.

5.5.3 CORRECTNESS

Answers to the quiz questions may not be correct as we do not control the accuracy of the information given. That accuracy is determined by the Open Trivia Database as they review the user-submitted questions before allowing them to be called.

5.5.4 REUSABILITY

At this point, there are no plans to reuse this application, but the option for that is available.

5.5.5 PORTABILITY

The application should be able to be installed and ran on any computer that meets the Performance Requirements.

5.6 PROCESS REQUIREMENTS

5.6.1 DEVELOPMENT PROCESS USED

Development began with the creation of the many documents necessary for the project this was done with input from all parties. Then we began the prototyping phase for our API connection and Persistent Data Storage, and after this, we began to code. We have been meeting two times a week (Mondays & Wednesdays) to go over what we all need to do and the help we may need. We also were available to each other outside these times to work on problems that may have arisen.

5.6.2 TIME CONSTRAINTS

Development began in January 2021 and will end in late April 2021. Final Project due on April 30 at 7 pm, with 2 code reviews due earlier.

5.6.3 COST AND DELIVERY DATE

The application comes at no cost to the user and our team. The final development is on April 30, 2021