FENS Fakultet prirodnih i tehničkih nauka
Faculty of Engineering and Natural Sciences

INTERNACIONALNI UNIVERZITET U SARAJEVU

Graduation project, SPRING 2022

Unity mobile gaming-”Quizmaster”

Play various pop-quizzes and expand your knowledge using your Android/iOS device

Student: **Mentor:**

Ajdin Fazlić *Associate Professor Dr. Kanita Karađuzović-Hadžiabdić*

14th June 2022, Sarajevo

Table of Contents

Abstract

Gaming, especially mobile gaming, has become a very popular among users of all age. Every user of the smartphone has played at least one game on that smartphone, and most of those users have returned to play the game again. There are many successful games on the Google Play and App store, and some of them are quiz games such as “Who wants to be a millionaire”, “HQ Trivia” and “SongPop 2”. “Quizmaster” is the game which offers a unique and most complete pop-quiz experience. Due to variations in quizzes and questions, this game is playable by a wide range of audiences of different ages and interests. In general, playing pop-quizzes is not as popular as some other activities, but everyone enjoys playing a couple of rounds or watching “Who wants to be a millionaire” on TV and trying to “win” the money. Because of that, all of the people who casually watch and play pop-quizzes will enjoy the “Quizmaster” since they can play a quick game of sports, music, movie or general knowledge quiz, stop at any time, and later pick up right where they left off. Due to this being a mobile game, Unity engine and C# are used for development. The game will allow users to play different types of quizzes with different types of questions in each quiz while trying to set a new high score and practice specific categories or quizzes. Overall, games such as “Quizmaster” would popularize quizzes and quiz-games, enable people to have lots of fun and learn a lot of fun facts in the process.

Chapter 1 - Introduction

1.1 Introduction

Gaming has always been popular, since 1958 when the first game was created by William Higinbotham, then through numerous gaming consoles starting from Magnavox Odyssey, Atari 2600, and Nintendo Entertainment System to contemporary consoles such as Sony PlayStation 5 and Microsoft Xbox Series X. Over the last years, popularity of gaming has increased drastically, reaching the number of 250 million players in Europe only [1]. Furthermore, gaming is becoming increasingly popular with the female population, which was not the case in the previous years. In 2018 it was estimated that 46% of Europe's gaming population were females [1]. With the increasing number of players, game development industry has grown proportionally. In 2018, game development industry was worth approximately 20 billion euros in 2018 [1], and it is safe to say that the profits have only increased since. One more testament to popularity of gaming is that the gaming revenues in 2021 were approximately $180 billion, whereas the biggest entertainment competitor, movie industry, has earned less than $20 billion from global box office revenues. With the new technologies developed, game development companies have introduced new ways of playing video games such as augmented and virtual reality which breaks the fourth wall between the player and the device. However, the player base mostly increased with the introduction of smartphones and smartphone games. Nowadays mobile gaming is the most popular and the most profitable part of game development accumulating $93 billion of the previously mentioned $180 billion, in 2021. The reason for this popularity of mobile games are ease of usage, ease of installation and option to play anywhere. Some of the very popular mobile games such as "PUBG Mobile", "Honor of Kings", "Candy Crush Saga" and "Pokémon GO" have collectively earned $8 billion in 2021. Finally, the most important feature of mobile games is the user retention. The goal of every mobile game is not to reach as many downloads as possible, but to make users return to play the game every day. The top echelon of mobile games, such as "Pokémon GO" have achieved this goal and have millions of active players.

In this project I have created mobile quiz game called "Quizmaster". There are two main reasons why I have chosen a quiz game, first reason being that quiz games can be highly beneficial studying tools, as explained in the paper written by Kate Wilkinson, George Dafoulas, Hemda Garelick and Christian Huyck [2], and second reason is that quiz games are simply interesting and everyone

wants to play at least one round when presented the opportunity. “Quizmaster” is a unique pop-quiz game, available for Android and iOS, that unifies the most popular pop-quizzes into one single game. This game will enable users to play various types of pop-quizzes such as general knowledge quiz, movie quiz, sports quiz and music quiz. However, not all quizzes will have the same format with multiple choice questions, but various types of questions such as “Have you scene it” questions for movie quizzes, “Guess the song” for music quizzes, etc. Targeted users of “Quizmaster” are people between 20 and 60 years old, and of any occupation. Regarding the system constraints, the only thing in question is whether the database will be able to support a large number of users and an even larger number of questions. The only thing that could take a lot of bandwidth from the Firebase database are images and videos which will be displayed in some questions. Current bandwidth limit is 100 GB, but it can be expanded if needed.
The engine used for the development of this mobile game is Unity (version 2020.3.32f1) by Unity Technologies. Unity is a cross-platform game engine which supports 2D and 3D graphics. It is primarily used for mobile game development, but it can also be used to develop games for consoles, PCs, and even VR and interactive simulations. Some notable and famous games developed in Unity are “Pokemon Go”, “Escape from Tarkov”, “Call of Duty: Mobile”, “Rust” etc. Game development in Unity is done using two main techniques: Unity predefined objects such as UI elements, elements for game physics and C# scripts. The biggest advantages of Unity are its simplicity and the ability to build the same project on multiple platforms without having to make any major changes in the scenes or code of the game itself. Switching from Android to iOS can be done by simply clicking the button for iOS and selecting the option “Switch platform” and making some minor tweaks so everything runs smoothly on iOS devices, thus choosing Unity as a game engine for this project was the best possible choice.

1.2. Objectives

The main objective of this project is to design and implement a fully operational mobile game application which can run on multiple Android and iOS devices using Unity Engine and programming practices learned throughout the years in numerous courses.

1.3 Thesis structure

Chapter one of this thesis presents introduction about gaming in general, progression of game development industry throughout the past few years and short description of Unity and “Quizmaster”. Chapter 2 describes in - depth about Unity and how it works. Chapter 3 discusses the market research about quiz games done before deciding on the design of “Quizmaster”. Chapter 4 breaks down system features, user and functional requirements, non – functional requirements and detailed use cases. Finally, chapter 5 describes system design in terms of class and sequence diagrams. Additionally, chapter 5 also displays the look of the user interface.

Chapter 2 – Technologies used

As previously mentioned, the main technology used in this project is Unity. Unity is a game engine and integrated development environment (IDE) created by 3 European developers David Helgason, Joachim Ante and Nicholas Francis. The initial idea behind Unity was to bring game development closer to amateur programmers who simply want to create simple games for fun. First version of Unity was released on June 6, 2005 and since then, from version 1.0.0 until version 2020.3.32 (the newest stable version, and version in which this project was built), it has provided very reliable toolset for building games for multiple platforms. The success of Unity can be seen in the individual success of its co-founders who have a combined net worth higher $2.6 billion. David Helgason and Joachim Ante have entered the world of angel investors and have created and lift up many successful companies, whereas Nicholas Francis has turned back to his roots and is currently working as a game developer.

The main feature of Unity is that it represents “technology that executes the graphics, the audio, the physics, the interactions, the networking” as pointed out by co-founder and CEO David Helgason [3]. In today’s version of Unity, one game developer can create whatever game he/she wants from scratch without great expertise in graphical design, physics, animations, etc. All that the developer needs can be found as a tool in Unity, or as a part of Unity asset store. Furthermore, Unity allows a great range of publishing targets such as Android, iOS, PC, PlayStation, Xbox, augmented and virtual reality. Additionally, Unity offers browser-based options such as Unity Web Player, Google Native Client and Flash, which is not supported in newer Unity versions. When developer wants to publish to multiple platforms, it can be done by using the same code and assets, and simply changing some settings and importing some important dependencies for the particular platform. Naturally, the responsibility for performance of the game such as game fps, model rendering, shader settings are completely on the developer. Thus, developers have to plan for which platforms they want to create the game and write code, create models and program shaders according to the selected platform. Even though it was firstly meant to be a game development engine, Unity has found its application in non-game applications such as architecture, art and data science[4]. Founders of the Unity believe that this broad application along with the ease of learning and use, and broad and thorough documentation are the main reasons for the Unity’s success.

The Unity Editor is the backbone of the Unity game engine. It is the place where the whole development takes place, from creating and setting the scenes, creating game objects, assigning functionalities to game objects, creating animations, to testing. Unity editor consists of many sub-windows, and the developer has complete freedom regarding which windows he/she will use. Even though the sub-windows differ from one project to another the most used ones are Project browser, Inspector, Hierarchy, Scene view, Game view, Animator, etc. Moreover, the developer can add additional sub-windows which are not included by default by importing different packages and assets.

The project browser is the sub-window which contains all assets such as sprites, animations, scripts, which are accessible and usable in the current project. Project browser is visually very similar to Finder in Mac OS X and File explorer in Windows. The aim of this similarity is to make developers feel more familiar with their surroundings and to navigate easily through the projects in new environment.

Inspector is the sub-window in which developer can see all details about selected game object. Inspector displays all components which are attached to selected game object such as scripts, materials, UI elements, audio elements and transforms. Developer is able to change and experiment with values of various components, for example position and scale of game object or color of the UI elements, and instantly see the change in the Scene view. Furthermore, the developer can change and tweak the values of variables from scripts which are made serializable or public.

Hierarchy sub-window contains a list of all game objects available in the scene. The list of game objects is automatically updated when new game objects are created in the scene. In order to retain simplicity, developer can assign parents or children using a simple drag and drop. The order of the list in the hierarchy is very important since the game objects which are lower in hierarchy are placed in front of objects which are higher in hierarchy.

Scene view is the place where developer constructs the whole game. In scene view developer places game objects to their respective positions and imports assets such as models and particle effects. The placement of objects and importing of assets is based on a drag and drop systems which is another testament to the simplicity of Unity. In addition, developer can set anchor and pivot points of game objects in the scene view, which is very important feature. Anchor points are used to measure the distance of game object from a specific point such as top corners of the screen.

When the anchor points are set, the game object shall always be placed on the same distance of top corners, no matter the size of screen. Since nowadays there are screens of various sizes in use, to create game that is playable for everyone, developer must make use of anchor points.

Game view is the sub-window which provides a complete preview of the game. After creating a scene and adding functionality to game objects, developer can enter the game view and test the whole game without building and deploying it. Alternative to game view is Device simulator view imported from “Device simulator” package. Device simulator is mostly used in Android and iOS development, and it offers a variety of devices of different specifications to test the game on.

Animator is a window used to place animations in the desired order. Developer can set which animations are played after certain events in the game and in what order.

Finally, Unity supports three types of programming languages: UnityScript, C# and Boo. What is common for all three languages is that they are run on open-source version of Microsoft’s .NET platform called Mono. Main advantages of Mono are that it has fast compile times, and it allows compatibility on different platforms which is perfect for engine like Unity. As previously mentioned, developer is able to change values of serializable and public variables directly from editor, without making any changes in the code. This is very useful in situations where developer needs to tune some values in-game to get the correct value. Additionally, all changes made to variables during play mode are not saved, which leaves a space for experimenting without the fear of ruining the whole project.

UnityScript is programming language similar to JavaScript and it is used by a majority of Unity developers. It is easy to learn and type, and more importantly it does a lot of “behind the scenes” work such as type casting instead of developer. Because of these features UnityScript is considered as a great option for beginners, but it is commonly used among experienced developers as well.

In contrast, C# is a general purpose, multi-paradigm programming language which is considerably harder to learn and type. C# requires developer to manually manage most of the components manually, unlike UnityScript which handles these things automatically. However, developers have complete and precise control over their scripts which C# which can be very important. This project is fully developed using C#.

Boo is programming language which represents the mix of Python and UnityScript. It uses syntax similar to Python, but it is structured similarly to UnityScript. Boo is used by minority of developers since it does not offer anything different than UnityScript.

Chapter 3 – Related work

Related work for this project includes analysing and looking into quiz games which are already available on Google Play in order to decide the main structure of "Quizmaster". First two games analysed are two biggest quiz games on the market, "HQ trivia" and "SongPop 2". These 2 games have the similar core, in terms of both being quiz games with the same core gameplay. However, the biggest distinction between these two games is that the "HQ Trivia" is based on single-round elimination tournament where the game is finished as soon as user gives the wrong answer to the given question, whereas "SongPop 2" applies round robin system in which user must answer to all questions to obtain the final score. In the article written by Tara Khairiyah Md Zali, Nor Samsiah Sani, Abdul Hadi Abd Rahman and Mohd Aliff it is stated that quiz games with single-round elimination tournament are highly dependent on the player's skill and knowledge since the difficulty increases with every question. On the other hand, games similar to "SongPop 2" are more casual and less dependent on the skill level but depend heavily on player's luck [5]. In a nutshell, quiz games such as "SongPop 2" are more attractive to casual players who simply want to relax while playing the game, whereas "HQ Trivia" and similar games are mostly played by hardcore pop-quiz and trivia fans. Since the main goal of "Quizmaster" is to increase the attractiveness of mobile quiz games, it was implemented using the round robin system.

Additionally, other popular quiz games with high ratings and large number of players were analyzed. Analysis has clearly shown that there are a lot of quiz games available and most of them are either very similar to classical quizzes such as "Who wants to be a millionaire"[6] or "Jeopardy" whose model was even applied in classrooms [7]. On the other hand there are some games which take a whole different approach to traditional quizzes. The best examples of those games are "94%" by Scimob in which the user is given a scenario and has to find 94% of the answers which fit that scenario, similar to popular TV game show "Family Feud" and "Quiz: Logo game" by Lemmings at work in which the user has to guess the name of the company based on the company logo. The available game on the market that is the most similar to the "Quizmaster" is "QuizUp" by Glu Mobile in terms of various quizzes and many different categories, but it does not provide various types and formats of questions like "Quizmaster". Because of that, "Quizmaster" has the potential to become a standard for pop quiz mobile games worldwide. The main goal of the "Quizmaster" is to provide a platform on which everyone will be able to find a quiz for their liking, learn various new things, set up impressive high scores, and most importantly have fun while doing that.

Chapter 4 – System features and use cases

This chapter represents all features the game will have. Each feature has been assigned a priority level according to the MoSCoW prioritization: MUST have, SHOULD have, COULD have and WOULD have. Additionally, this chapter includes non – functional requirements regarding hardware and software interface, performance, security, usability, portability, reusability and installability.

4.1 System Features

SF1: Manage user accounts (Must Have)

After entering the game for the first time the user shall be able to see an authentication window which will prompt him/her to log in. After logging in, the user shall be able to set high scores on the leaderboards.

UR1.1 Login

FR1.1.1 Login via Google: The user shall be able to select the option "Continue with Google", after which the user shall be prompted to select a Google account with which he/she wants to log in into the "Quizmaster".

FR1.1.2 Login with Guest account: The user shall be able to select the option "Continue as guest", after which the user shall be assigned a random guest number and will be able to proceed to the game, but with limited functionalities.

SF2: Play various quizzes (Must Have)

After logging in and clicking the play button the user shall be able to select which quiz, he/she wants to play and start it by clicking the appropriate button.

UR2.1 Display various types of questions

FR2.1.1 Display questions with images: In case a question contains an image, the user shall be able to see that image clearly below the question text.

FR2.1.2 Display questions with videos: The user shall be able to see the imported video regarding the question. The user shall also be able to replay or stop the video. If the user tries to replay a video more than once, he/she will be notified that another replay will result in losing points in the game. If the user chooses to replay the video anyway points shall be deducted and the video will be played again, and if the user chooses not to replay the video, the notification popup will simply be closed.

FR2.1.3 Display questions with sounds: The user shall be able to hear the sound regarding the question and see the visualization of that sound. The sound shall be played on the loop and will be stopped when users enter or select an answer.

UR2.2 Answer the questions

FR2.2.1 Answer the question from the input field: The user shall be able to answer the given question by entering the answer in the input field. The user can change his/her answer as much as he/she wants until the time runs out. If the timer reaches 0, the question is automatically evaluated as wrongly answered. The user shall be able to submit the entered answer by clicking the "Lock in" button.

FR2.2.2 Answer the question from the multiple choices: The user shall be able to select a correct answer from the given four choices. The user can select his answer by clicking on it. Once the user clicks on the answer it is immediately locked in and checked for correctness, and the button has changed the color to yellow. If the selected answer is correct the button shall turn green, and if it is incorrect the button shall turn red. Additionally, if the timer for that question runs out, the user shall not be able to select any answer, and the system shall give 0 points to the user for that question. Furthermore, the correct answer shall be displayed by changing the corresponding button to green.

UR2.3 Use lifelines

FR2.3.1 Use "Hint" and "Show options" lifelines: The user shall be able to use the "Hint" lifeline to see some additional information about the correct answer. The user shall be able to use this lifeline once per game. Furthermore, the user shall be able to use the "Show options" lifeline to see 4 possible options from which one is the correct answer. When the user clicks on the "Show options" button he/she will be notified that using this lifeline will result in losing half of the points on that question. If the user chooses to use the lifeline anyway, he/she will receive only half of the points if he/she guesses the correct answer, and if the user chooses not to use the lifeline the popup will be closed, and game will be resumed.

FR2.3.2 Use "50:50", "Ask a friend" and "Replace the question" lifelines: These lifelines shall only be available after the user has used the "Show options" lifeline and the user shall be able to use each abovementioned once per game. The user shall be able to use the “50:50” lifeline to remove 2 incorrect answers from the selection. Furthermore, the user shall be able to “ask a friend” and receive answer which will be correct 80% of the time. Additionally, the user shall be able to use the “Replace the question” lifeline to receive a new question from the same category.

UR2.4 Unlock new quiz level

FR2.4.1 Unlock Medium/Hard level: Each quiz shall have 3 levels: Easy, Medium and Hard. The first level, “Easy”, shall be unlocked by default and the user shall be able to play it instantly. If the user reaches a score equal or higher than the certain threshold in the first level, the user shall unlock and be able to play a higher level. User only needs to unlock level once, and he/she shall have it unlocked all the time.

UR2.5 Play various levels

FR2.5.1 Select level to play: After selecting the type of quiz, the user shall be prompted to select which level he/she wants to play. The user shall be able to select only unlocked difficulties, whereas the locked difficulties shall be greyed out and the user shall not be able to select them.

FR2.5.2 Display different questions on each level: Each level shall have different sets of questions with different difficulty level. Namely, the first level shall have the easiest questions, and the third level shall have the hardest set of questions.

UR2.6 Save/Load quiz progress

FR2.6.1 Save current quiz progress: User progress shall be saved automatically when the user quits the game or goes back to the main menu. Each quiz progress shall be saved separately for each quiz, and the system shall save current quiz difficulty, current question category, current question, lifelines used and score.

FR2.6.2 Load quiz progress: Quiz progress shall be loaded only if the user chooses to continue from where he/she left off. If the user clicks “Continue playing” button, the system shall load the difficulty, category, question, lifelines used and score for the selected quiz.

FR2.6.3 Start new game: If the user wishes to start quiz from the beginning by clicking on “Start new game” button, the progress saved for the selected quiz shall be deleted and new save file shall be created when conditions which trigger save function occur.

UR2.7 Set a high score

FR2.7.1 Set a new high score: After finishing a game, the user shall be able to see how much points has he/she scored. If that score is higher than the previously set high score, the user shall be notified that the new high score has been set. On the other hand, if the score is lower than the high score, the user shall see the achieved score and the high score in the same popup.

SF3: Enter practice mode (Should Have)

Besides being able to play a regular game mode, the user shall be able to enter a practice mode for a particular quiz by clicking an appropriate button.

UR3.1 Practice the whole quiz

FR3.1.1 Practice the whole quiz: The user shall be able to select a “Play simulation” option to simulate quiz from the regular play mode with various difficulties, if the user has those difficulties unlocked in the regular play mode. Simulation and regular play mode are very similar but have a few clear differences such as: in the simulation mode the user shall be able to use all the lifelines as many times as he/she wishes to, instead of once for some lifelines, namely “50:50”, “Ask a friend”, “Replace the question” and “Hint”, in the regular play mode. Moreover, the user shall not be limited by time in the simulation mode and will have as much time as he/she likes to answer the question.

UR3.2 Practice particular questions

FR3.1.2 Practice particular type of questions: The user shall be able to practice particular groups of questions which are specific to a certain quiz such as “Guess the song”, “Have you scene it”, “Guess the artist/actor”, “Guess the poster”, etc. After selecting this practice mode, the user shall be prompted to select which types of question he/she wants to practice and at which difficulty. The user shall be able to select only difficulties which are unlocked in a regular play mode for that quiz. In addition, the user shall be able to use all lifelines without the one per quiz limit, and will not have a question timer, same as in the simulation practice mode.

SF4: Manage game settings (Should Have)

The user shall be able to change default game settings using the “Settings” tab. Settings which shall be adjustable are sound, music, theme and avatar.

UR4.1 Toggle game sound

FR4.1.1 Turn sound off: The sound shall be turned on by default. The user shall be able to turn it off by using the sound toggle button. When the user clicks on the toggle, all sounds in the game shall automatically be muted. However, this does not refer to sounds which are produced by questions in the music quiz, such as questions from “Guess the song” category.

FR4.1.2 Turn sound on: If the user, at some point, wishes to turn sound back on, he/she shall be able to do so by using the sound toggle. When the user clicks on the toggle to turn the sound back on, all sounds in the game shall be automatically unmuted.

UR4.2 Toggle game music

FR4.2.1 Turn background music on: The music shall be turned off by default. The user shall be able to turn it on by using the music toggle button. When the user clicks on toggle, the background music shall automatically start playing. The background music shall be looped and will start from the beginning each time the user turns it on.

FR4.2.2 Turn background music off: If the user, at some point, wishes to turn the background music back off, he/she shall be able to do so by using the music toggle. When the user clicks on the toggle to turn the background music off, the track with the background music shall be automatically stopped.

UR4.3 Change game theme

FR4.3.1 Select different game theme: The user shall be able to change a game theme by selecting whichever he/she wants to from the settings menu. The basic, first theme shall be the current color scheme of the game, whereas other themes will be different color schemes. When the user selects another theme, it shall change instantly and changes shall affect main menu, practice menu and modes, regular play mode, settings tab and all popups.

UR4.4 Change in-game avatar

FR4.4.1 Select different in-game avatar: The user shall be able to change in-game avatar. By default, the in-game avatar shall be a male head silhouette, and another available avatar shall be female head silhouette. The user shall be able to select an avatar by clicking on the corresponding icon of whichever one he/she wishes to select. When the user clicks on the icon, it shall be automatically changed in all game modes which display the avatar.

UR4.5 Save/Load user settings

FR4.5.1 Save/Load user settings: Every change which user makes in the "Settings" tab shall be automatically saved using custom binary files. The saving function shall be triggered whenever the user changes something in the "Settings" tab such as toggles sound or background music, changes theme or in-game avatar. Additionally, save function shall be triggered when the user quits the game as a failsafe. User settings shall be loaded automatically each time the user enters the game or when the scene which has the "Settings" tab is reloaded.

SF5: Help tab (Could Have)

The user shall be able to access a help tab to see some additional explanations regarding the quizzes, various types of questions, lifelines and scoring system. Furthermore, the user shall be able to see frequently asked questions which can provide additional insight, if the previously written instructions were not enough.

UR5.1 Open and browse help tab

FR5.1.1 Open and browse help tab: The user shall be able to open the help tab by clicking the corresponding button which indicates either a question mark or "HELP". After clicking on the button, the user shall see the tab with buttons for "Quizzes", "Types of questions", "Lifelines", "Scoring" and "Frequently asked questions". The user shall be able to close the help tab by clicking on the "X" button in the upper right corner, or by using the Android system back button.

FR5.1.2 Open "Quizzes" help tab: The user shall be able to view helpful information regarding the quizzes by clicking on the "Quizzes" button in the help tab. When the user clicks the aforementioned button, he/she shall se the textual explanation of all quizzes available in the game. The user shall be able to navigate back to the help tab by clicking the “Back” button at the bottom of the tab.

FR5.1.2 Open “Types of questions” help tab: The user shall be able to view textual information about all types of questions which can be found in all quizzes in the game by clicking on the “Types of questions” button in the help tab. The user shall be able to navigate back to the help tab by clicking the “Back” button at the bottom of the tab.

FR5.1.3 Open “Lifelines” help tab: The user shall be able to view additional information about lifelines by clicking on the “Lifelines” button in the help tab. After clicking on the previously mentioned button, the user shall be able to see lifeline icon with textual explanation for each lifeline in the game. The user shall be able to navigate back to the help tab by clicking the “Back” button at the bottom of the tab.

FR5.1.4 Open “Scoring” help tab: The user shall be able to view explanation of the scoring system in the game by clicking on the “Scoring” button in the help tab. The user shall be able to navigate back to the help tab by clicking the “Back” button at the bottom of the tab.

FR5.1.5 Open “Frequently asked questions” tab: The user shall be able to see the most frequently asked questions by clicking on the “Frequently asked questions” button in the help tab. After clicking the button, the user shall be able to see all frequently asked questions and the answers for each question below it. The user shall be able to navigate back to the help tab by clicking the “Back” button at the bottom of the tab.

SF6: Manage leaderboards (Would have)

The user shall be able to see leaderboards for each type of quiz in the regular play mode. Each leaderboard entry shall have the score achieved and the time it took to finish the whole quiz.

UR6.1 View leaderboard

FR6.1.1 View quiz leaderboard: After finishing the quiz, the user shall be able to see the leaderboard for that type of quiz and that difficulty. The leaderboard shall consist of ten best results for that particular quiz. In addition, the user shall be able to see his/her current result in two possible ways: either in the top 10 results in the leaderboard, considering that the score is good enough, or at the bottom below the tenth result.

UR6.2 Enter the score to the leaderboard

FR6.2.1 Update the leaderboard: The leaderboard shall be updated after each game when the user has scored a better score than the ones on the leaderboard. The results shall firstly be compared by the score achieved. If the current result has a higher score than the one at the leaderboard, it shall go above it in the leaderboard. However, if the user achieves the same score as the one the leaderboard, the results shall be compared by the time it took to complete the quiz. If the current result has the same score as the one at the leaderboard, but is finished in less time, it shall take its place on the leaderboard. Additionally, there is a rare case in which the current result and the one at the leaderboard have the same score and are finished in the same time. In that case, the two results shall share the same place on the leaderboard.

SF7: Report a problem (Would have)

User shall be able to fill out a form, where he/she shall be asked to state any problems, issues and bugs that they have encountered while playing the game.

UR7.1 Problem inquiry

FR7.1.1 User dispute: The user shall be able to fill out a form in which he/she can explain in detail the bugs and issues which have occurred while playing the game.

FR7.1.2 Problem resolving: After the problem has been inspected and fixed, the game shall receive a forced update in order to ensure that all the users do not experience the same bug again.

4.2 Non-functional requirements

NFR1: Hardware Interface

Description: Since the game will be running on iOS and Android, the users will need to meet certain hardware requirements on their devices. Luckily, the game is not demanding so it is safe to say that any smartphone, which was built after 2014, will be able to run the app. However, older smartphones might experience slight performance issues. The requirements will be similar for both iOS and Android, so the following specifications are a broad estimate of the average for both operating systems. Each phone will need to have at least a 1x Core Processor clocked at 2.0GHz, 1GB of RAM, 100MB free space, Android 6 or later, iOS 7 or later.

More details:

Operating System: Android 6+ iOS 7+

Processor: 1x Core 1.7GHz

RAM: 1GB

Free Space: 100MB

Other: Network adapter/module which supports stable connections of at least 1Mbps

NFR2: Software interface

Description: Since a database is needed for all of the question and player data, database used in this project is Firestore database from Google's Firebase since it has straightforward and easy-to-use interface, it is free to use up to 100 GB bandwidth and very cheap for further bandwidth expansions, and it is easily implemented in Unity.

More details:

Database: Firestore Database

NFR3: Performance

Description: The given configuration will make sure to be able to hold up to 500 requests per second of both static and dynamic requests. Each request will take from 1ms (static) to up to 40ms (dynamic). Some partitions of the storage will be reserved to ensure maximum bandwidth usage.

NFR4: Security

Description: Only users with Admin privileges/accounts will be able to edit question, quizzes, add/delete questions, view problem reports etc. Since the authentication is done either via Google or Facebook, passwords and other sensitive information shall be protected according to Google/Facebook standards.

NFR5: Usability

Description: All targeted user groups of any age group (20-60) will be able to select and play or practice a quiz within minutes. The simple and minimalistic interface and features provide maximum efficiency and usability.

NFR6: Portability

Description: Since the Unity engine is used for the development, the game shall be easily portable to Android and iOS without significant source code changes.

NFR7: Reusability

Description: The game can be expanded with multiple other quizzes since the core gameplay, namely displaying and answering questions, uses the same source code for each quiz. Only changes needed to be done are entering new question data into database.

NFR8: Installability

Description: Novice/Amateur as well as experienced smartphone users will be able to install the game and start playing within 5 minutes (at most). The only requirement is 100MB free space and a stable internet connection.

4.3 Use cases

Below are the representations of each system feature as a use case table.

Primary actor	Use case
User	UR1.1 Login UR2.1 Display various types of questions UR2.2 Answer the questions UR2.3 Use lifelines UR2.4 Unlock new quiz level UR2.5 Play various levels UR2.6 Save/Load quiz progress UR2.7 Set a high score UR3.1 Practice the whole quiz UR3.2 Practice particular questions UR4.1 Toggle game sound UR4.2 Toggle game music UR4.3 Change game theme UR4.4 Change in-game avatar UR4.5 Save/Load user settings UR5.1 Open and browse help tab UR6.1 View leaderboard UR6.2 Enter the score to the leaderboard UR7.1 Problem inquiry

The figure below shows all uses cases for the first version of the “Quizmaster”:

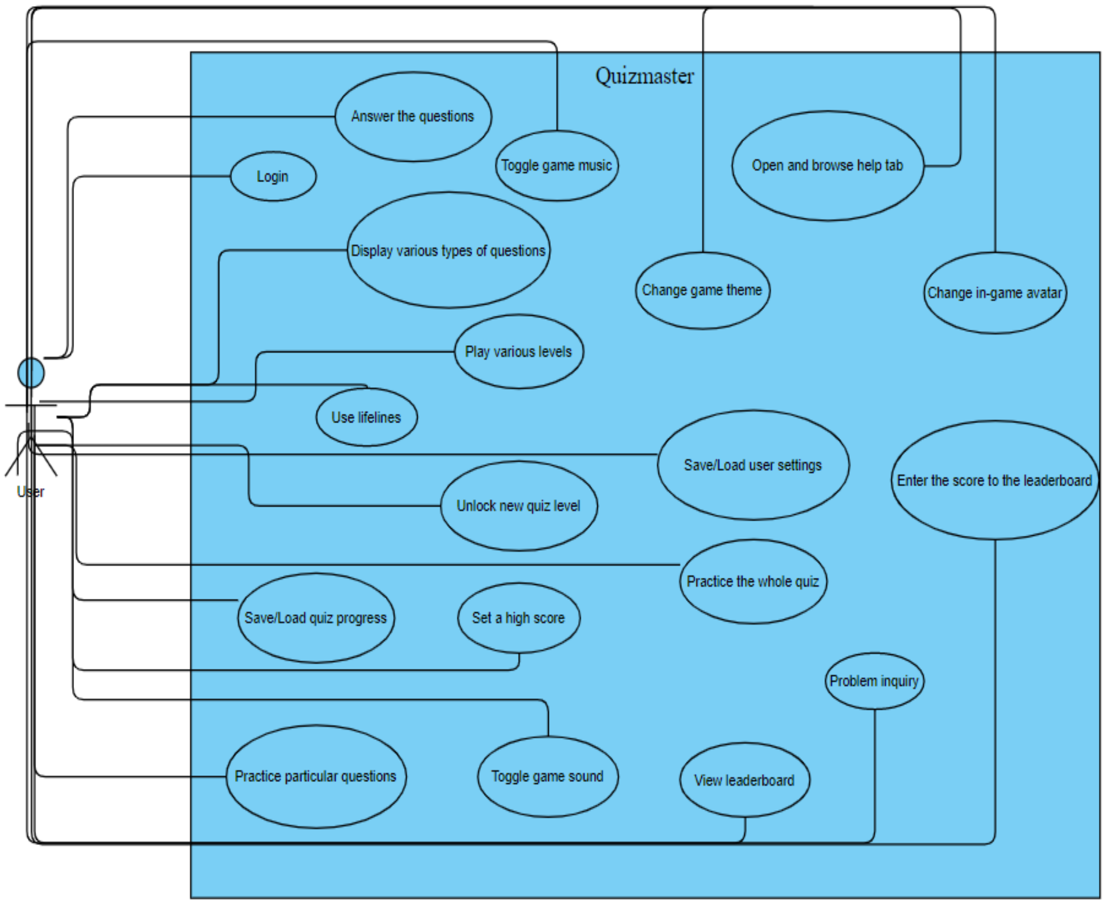


Figure 1. Use case diagram

4.3.1 Detailed use cases

Login:

User story: As a user I want to successfully log in to the game using my Google or Facebook account.

Alternative: As a user I want to successfully enter the game using a guest account.

Description: The user shall be able to successfully log in to the game when he/she opens it for the first time. The user shall be able to use Google or Facebook account. Furthermore, the user can also enter the game using a guest account.

Use case ID: UR1.1

Priority: Must have

Pre-condition: PRE-1: The user has opened the game for the first time.

Post-condition:

POST1: The user has successfully logged in using Google or Facebook account and is redirected to the “Main menu” screen.

POST2: The user has successfully obtained a guest account and is redirected to the “Main menu” screen.

POST3: The user has entered a wrong password for the Google or Facebook account and is prompted to reenter the password.

Basic flow:

1. The user enters the application for the first time.
2. The user selects which method he/she wants to use to log in (Google or Facebook).
3. The user enters his/her credentials.
4. The user is redirected to the “Main menu”.

Alternate flow:

1. The user enters the application for the first time.
2. The user selects “Continue as a guest” option
3. The user is assigned a guest number.
4. The user is redirected to the “Main menu”.

What can go wrong: The user can enter wrong credentials when trying to connect with Google or Facebook account. In that case Google/Facebook services shall notify the user that the credentials are wrong and prompt him/her to reenter them.

Display various types of questions:

User story: As a user I want to be able to see a variety of questions such as textual questions, questions with images, videos or questions with sounds.

Description: The user shall be able to see various types of questions regarding of the type of quiz the user is playing. For example, if the user is playing a music quiz, the majority of questions shall be questions involving sounds, or if the user is playing movie quiz most of the questions shall have images or videos, and if the user is playing general knowledge quiz, he/she will see textual questions.

Use case ID: UR2.1

Priority: Must have

Pre-condition:

PRE-1: The user has clicked the "Play" button and selected a certain type of quiz.

Post-condition:

POST1: The user has successfully started a quiz and can see textual question.

POST2: The user has successfully started a quiz and can see a question with the appropriate image regarding the question.

POST3: The user has successfully started a quiz and can see a question with the appropriate video regarding the question.

POST4: The user has successfully started a quiz and can see a question and hear the appropriate sound regarding the question.

Basic flow:

1. The user enters the application for the first time and clicks on the "Play" button.
2. The user selects which type of quiz he/she wants to play ("General knowledge", "Sports quiz", "Movie quiz" or "Music quiz")
3. The user starts the game and sees the question appropriate for the given quiz and category.

Answer the questions:

User story: As a user I want to be able to answer to the given question either by typing the answer or by selecting an answer from the provided options.

Description: The user shall be able to give an answer to the question by typing it into the given text input field and clicking the “Lock in” button, or by clicking on the button with the answer user wants to select.

Use case ID: UR2.2

Priority: Must have

Pre-condition:

PRE-1: The user has seen the question and has typed the answer into the input field and clicked the “Lock in” button.

PRE-2: The user has seen the question, clicked on the “Show options” button and has clicked on the button with the answer he/she wants to select.

Post-condition:

POST1: The answer user typed in or selected is checked, deemed as a correct answer and the user is notified.

POST2: The answer user typed in or selected is checked, deemed as a incorrect answer and the user is notified.

Basic flow:

1. The user starts the quiz and sees the question.
2. The user types in an answer into the input field and click the “Lock in” button.
3. The answer is checked for exactness and the user is notified of the result.

Alternate flow:

1. The user starts the quiz and sees the question.
2. The user clicks on “Show options” button and sees four given answer options.
3. The user selects one answer by clicking on it.
4. The selected answer is checked for exactness and the user is notified of the result.

What can go wrong: The user may try to click on the “Lock in” button when there is not text entered in the input field. In that case the user shall be notified that he/she has to enter some text into the input field in order to be able to lock in the answer.

Use lifelines:

User story: As a user I want to be able to answer to use given lifelines once per game in order to receive help in answering the given question.

Description: The user shall be able to use various lifelines throughout the game. The user shall be able to use every lifeline once, except the “Show options” lifeline which shall be available through the whole game, but the user shall receive points penalty after using it.

Use case ID: UR2.3

Priority: Must have

Pre-condition:

PRE-1: The user has seen the question and has clicked on one of the lifeline buttons: “Hint”, “Show options”, “50:50”, “Ask a friend” and “Replace the question”.

Post-condition:

POST1: The user sees the “Hint” popup with the provided hint.

POST2: The user can no longer see and use the input field, instead the user sees the four answer options.

POST3: The user can no longer see two randomly selected incorrect answer options.

POST4: The user sees the “Ask a friend” popup with the answer provided from the “friend”.

POST5: The user receives new question.

Basic flow:

Flow-1:

1. The user starts the quiz and sees the question.
2. The user clicks on the button for the “Hint” lifeline.
3. The user sees the “Hint” popup.
4. The user closes the “Hint” popup and the button for the “Hint” lifeline is greyed out and disabled.

Flow-2:

1. The user starts the quiz and sees the question.
2. The user clicks on the button for “Show options” lifeline.

3. The user is notified about the points penalty and prompted if he/she wants to continue.

3.1. The user declines the use of this lifeline, and the popup is closed.

3.2. The user accepts the use of this popup, the input field is removed, and the answer options are visible to user.

Flow-3:

1. The user starts the quiz and sees the question.
2. The user clicks on the button for “50:50” lifeline.
3. The two incorrect answers are removed and the button for the “50:50” lifeline is greyed out and disabled.

Flow-4:

1. The user starts the quiz and sees the question.
2. The user clicks on the button for “Ask a friend” lifeline.
3. The user sees the “Ask a friend” popup.
4. The user closes the “Ask a friend” popup and the button for the “Ask a friend” lifeline is greyed out and disabled.

Flow-5:

1. The user starts the quiz and sees the question.
2. The user clicks on the button for “Replace the question” lifeline.
3. The user receives a new question from the same category and the button for the “Replace the question” lifeline is greyed out and disabled.

Unlock new quiz level:

User story: As a user I want to be able to unlock new, harder levels for each quiz.

Description: The user shall be able to unlock two new levels per each type of quiz by scoring a certain number of points on a one level lower difficulty.

Use case ID: UR2.4

Priority: Must have

Pre-condition:

PRE-1: The user has finished the quiz and has scored the minimum number of points required for unlocking the next level or higher.

Post-condition:

POST1: The user gets notified that he/she has unlocked new level.

Basic flow:

1. The user has completed the quiz and has scored enough points to unlock the new level.
2. The user is notified that the new level is unlocked.
3. The user can start playing the new level right away.

Play various levels:

User story: As a user I want to be able to play all levels I have unlocked for a certain quiz.

Description: The user shall be able to select and play any unlocked level for every type of quiz.

Use case ID: UR2.5

Priority: Must have

Pre-condition:

PRE-1: The user has unlocked desired difficulty in some previous games.

PRE-2: The user has selected a difficulty he/she wants to play on.

Post-condition:

POST1: The game with the selected difficulty is started.

Basic flow:

1. The user has chosen a type of quiz he/she wants to play.
2. The user sees “Select difficulty” screen and selects and unlocked difficulty.
3. The game starts with the questions appropriate for the selected difficulty.

Alternate flow:

1. The user has finished a game and unlocked a new difficulty as explained in use case UR2.4.
2. The user chooses to start new game with new unlocked difficulty right away.
3. The new game is started with the questions appropriate for the new unlocked difficulty.

What can go wrong: The user may try to select and play difficulty which is not unlocked yet. In that case the user shall not be able to start the game until he/she selects available difficulty.

Save/Load quiz progress:

User story: As a user I want to be able to save my current quiz progress and continue playing from where I left of sometimes later.

Description: The user shall be able to save the quiz progress automatically by leaving the game. Furthermore, the user shall be able to load quiz progress by clicking on the "Continue playing" button.

Use case ID: UR2.6

Priority: Must have

Pre-condition:

PRE-1: The user is playing the quiz and has clicked the "Back to menu" button.

PRE-2: The user has selected the quiz and the difficulty and has clicked the "Continue Playing" button.

Post-condition:

POST1: The game progress is automatically saved after returning to the menu.

POST2: The game resumes from where the last save was made.

Basic flow:

Flow-1:

1. The user is playing the game and clicks on the "Back to menu" button.
2. The user is prompted whether he/she wants to quit the game.
3. The user clicks "Yes" button and is returned to the menu, and the game progress is automatically saved.

Flow-2:

1. The user selects the quiz and the difficulty he/she was playing previously.
2. The user is prompted whether he/she wants to continue playing from where he/she left off.
3. The user clicks on the "Continue playing" button.

4. The game is resumed from the last saving point with the score and available lifelines loaded.

Alternate flow:

1. The user selects the quiz and the difficulty he/she was playing previously.
2. The user is prompted whether he/she wants to continue playing from where he/she left off.
3. The user clicks on the "Start new game" button.
4. The save file associated with the selected quiz and difficulty is deleted and the new game is started.

Set a high score:

User story: As a user I want to be able to set a high score for each difficulty in every type of quiz.

Description: The user shall be able to set a high score for every difficulty in every type of quiz. If there is no high score set, the current score will be marked as a high score. On the other hand, if there is a high score set, the current result is compared to the high score and if it is larger, then it will be the new high score.

Use case ID: UR2.7

Priority: Must have

Pre-condition:

PRE-1: The user has successfully finished the quiz.

Post-condition:

POST1: There is no high score set and the current score is set as a high score.

POST2: There is a high score set and the current score is lower than the high score, thus nothing has changed.

POST3: There is a high score set and the current score is higher than the high score, thus the current score is set as a high score and the user is notified about the new high score.

Basic flow:

1. The user has successfully finished the quiz.
2. The current score is compared to the high score, and it is deemed as higher.

3. The user sees the endgame popup with the notification that the new high score is set.

Alternate flow:

1. The user has successfully finished the quiz.
2. The current score is compared to the high score, and it is deemed as lower.
3. The user sees the endgame popup with the achieved score and the high score.

Practice the whole quiz:

User story: As a user I want to be able to practice the whole quiz from start to end in order to have better scores.

Description: The user shall be able to practice any type of quiz on any unlocked difficulty as a simulation of real quiz. The difference from the real quiz is that in the practice mode, the user shall have unlimited lifelines and no time limit to answer the question.

Use case ID: UR3.1

Priority: Should have

Pre-condition:

PRE-1: The user has entered the practice panel and selected which quiz and what difficulty he/she wants to practice.

Post-condition:

POST1: The practice game of selected quiz with selected difficulty has started.

Basic flow:

1. The user has clicked on the “Practice” button in the main menu.
2. The user has selected “Practice whole quiz” option from the practice menu.
3. The user has selected the type of quiz and difficulty
4. The practice game starts.

Practice particular questions:

User story: As a user I want to be able to practice only the specific type of questions from specific categories.

Description: The user shall be able to practice a specific type of questions of all unlocked difficulties from specific categories. Types of questions include textual questions,

questions with sound, questions with images and questions with videos. One practice session shall consist of 5 randomly chosen questions from the selected category.

Use case ID: UR3.2

Priority: Should have

Pre-condition:

PRE-1: The user has entered the practice panel and selected which category and on which difficulty he/she wants to practice.

Post-condition:

POST1: The practice game of selected category with selected difficulty has started.

Basic flow:

1. The user has clicked on the “Practice” button in the main menu.
2. The user has selected “Practice a category” option from the practice menu.
3. The user has selected the category and difficulty
4. The practice game starts.

Toggle game sound:

User story: As a user I want to be able to turn the in-game sound on and off.

Description: The user shall be able to toggle in-game sounds on and off by clicking on the toggle button in the settings panel. The in-game sounds shall be turned on by default.

Use case ID: UR4.1

Priority: Should have

Pre-condition:

PRE-1: The user has entered the settings panel from the main menu and has clicked on the “Sound” toggle.

Post-condition:

POST1: The in-game sound is turned off.

POST2: The in-game sound is turned on.

Basic flow:

1. The user has clicked on the “Settings” button in the main menu.
2. The user has entered settings panel.
3. The user has clicked on the “Sound” toggle.

4. The in-game sounds are turned off.

Alternate flow:

1. The user has clicked on the “Settings” button in the main menu.
2. The user has entered settings panel.
3. The user has clicked on the “Sound” toggle.
4. The in-game sounds are turned on.

Toggle game music:

User story: As a user I want to be able to turn the background music on and off.

Description: The user shall be able to toggle background music on and off by clicking on the toggle button in the settings panel. The background music shall be turned off by default.

Use case ID: UR4.2

Priority: Should have

Pre-condition:

PRE-1: The user has entered the settings panel from the main menu and has clicked on the “Music” toggle.

Post-condition:

POST1: The background music is turned on.

POST2: The background music is turned off.

Basic flow:

1. The user has clicked on the “Settings” button in the main menu.
2. The user has entered settings panel.
3. The user has clicked on the “Music” toggle.
4. The background music is turned on.

Alternate flow:

1. The user has clicked on the “Settings” button in the main menu.
2. The user has entered settings panel.
3. The user has clicked on the “Music” toggle.
4. The background music is turned off.

Change game theme:

User story: As a user I want to be able to change the main theme of the game.

Description: The user shall be able to change the main theme of the game by clicking on the button corresponding to the new desired theme in settings panel.

Use case ID: UR4.3

Priority: Should have

Pre-condition:

PRE-1: The user has entered the settings panel from the main menu and has clicked on the button to select the new theme.

Post-condition:

POST1: The main theme of the game is changed.

Basic flow:

1. The user has clicked on the "Settings" button in the main menu.
2. The user has entered settings panel.
3. The user has clicked on the button for different game theme.
4. The main theme of the game is changed.

Change in-game avatar:

User story: As a user I want to be able to change my in-game avatar.

Description: The user shall be able to change the in-game avatar by clicking on the button corresponding to the desired avatar in settings panel. The default avatar shall be male silhouette.

Use case ID: UR4.4

Priority: Should have

Pre-condition:

PRE-1: The user has entered the settings panel from the main menu and has clicked on the button to select the new avatar.

Post-condition:

POST1: The in-game avatar is changed.

Basic flow:

1. The user has clicked on the "Settings" button in the main menu.
2. The user has entered settings panel.
3. The user has clicked on the button for different avatar.
4. The in-game avatar is changed.

Save/Load user settings:

User story: As a user I want the changes I have made in the settings to be saved every time I start the game.

Description: The changes made in the settings tab shall be automatically saved when the user exits the settings tab. Furthermore, the settings shall be automatically loaded each time the user enters the application.

Use case ID: UR4.5

Priority: Should have

Pre-condition:

PRE-1: The user has entered the settings panel from the main menu, made some changes and left the settings panel.

PRE-2: The user has entered the application.

Post-condition:

POST1: The user settings are automatically saved.

POST2: The user settings are automatically loaded on the application start.

Basic flow:

1. The user has clicked on the "Settings" button in the main menu.
2. The user has changed some settings.
3. The user has clicked on the "Back" button.
4. Game settings are saved.

Alternate flow:

1. The user has entered the application.
2. Game settings are loaded on the application start.

Open and browse help tab:

User story: As a user I want to open the help tab since I want to know more about the types of quizzes and questions, scoring system and other frequently asked questions.

Description: The user shall be able to open help tab by clicking on the corresponding "?" button in the main menu. In the help tab the user shall be able to choose between different sections which have the information about quizzes, categories, lifelines, scoring system and frequently asked questions.

Use case ID: UR5.1

Priority: Could have

Pre-condition:

PRE-1: The user has clicked the "?" button in the main menu.

Post-condition:

POST1: The user has entered the help tab.

Basic flow:

1. The user has entered main menu.
2. The user has clicked on the "?" button.
3. Help tab is opened.
4. The user clicks on the desired category and sees corresponding information.

View leaderboard:

User story: As a user I want to see the leaderboard containing the best results for the quiz and difficulty I have completed.

Description: The user shall be able to see the leaderboard at the end of every game. The leaderboard shall contain 10 best results for the quiz and difficulty the user has completed. Every quiz and every difficulty shall have its own separate leaderboard.

Use case ID: UR6.1

Priority: Would have

Pre-condition:

PRE-1: The user has successfully finished the game.

Post-condition:

POST1: The user sees the leaderboard with 10 best results.

Basic flow:

1. The user has successfully completed the quiz
2. The user sees the endgame popup and closes it.
3. The user sees the leaderboard with 10 best results for that particular quiz and difficulty.

Enter the score to the leaderboard:

User story: As a user I want to be able to enter the leaderboard with my score.

Description: The user shall be able to enter the leaderboard if the current score is higher than any score which is already in the leaderboard. Additionally, if the scores are equal, time required to complete the quiz is taken into consideration.

Use case ID: UR6.2

Priority: Would have

Pre-condition:

PRE-1: The user has successfully finished the game.

Post-condition:

POST1: The score user achieved is higher than some score at the leaderboard or the two scores are equal, but the current attempt is finished in less time. The achieved score enters the leaderboard.

POST2: The score user achieved is lower than some score at the leaderboard or the two scores are equal, but the current attempt is finished in more time. The achieved score does not enter the leaderboard and it remains unchanged.

Basic flow:

1. The user has successfully completed the quiz.
2. The user sees the endgame popup and closes it.
3. The user sees the leaderboard with 10 best results for that particular quiz and difficulty.
4. The user gets notified that achieved score is better than some score on the leaderboard.
5. The user sees the achieved score on the leaderboard.

Alternate flow:

1. The user has successfully completed the quiz.
2. The user sees the endgame popup and closes it.
3. The user sees the leaderboard with 10 best results for that particular quiz and difficulty.

Problem inquiry:

User story: As a user I want to be able to report problems and concerns about the game.

Description: The user shall be able to report possible issues and bugs that have occurred during his/her usage of the application.

Use case ID: UR7.1

Priority: Would have

Pre-condition:

PRE-1: The user has submitted the form from the "Report a problem" tab.

Post-condition:

POST1: The user is notified that the report is received.

Basic flow:

1. The user clicks on the "Report a problem" button and opens a "Report a problem" tab.
2. The user fills out the given form.
3. The user submits the form.
4. The user receives the confirmation that the form was sent successfully and that it is received.

What can go wrong: The user might not fill out all required fields in the form. In that case user shall not be able to submit the form until those fields are filled with the appropriate information.

Chapter 5 – System design

In this chapter design of the whole game shall be explained. This includes class diagram for the whole game, as well as sequence diagram for each use case and screenshots from the game.

5.1 Class Diagram

Figure below depicts the class diagram for “Quizmaster”. The whole game has 12 classes, namely: QuizMaster, Player, Guest, RegisteredPlayer Controller, Question, ImageQuestion, VideoQuestion, SoundQuestion, Quiz, Practice and SettingsController. QuizMaster consist of Player and Question and it has a list for both of those classes. Player class has variables for first name, last name, password and a list of high scores for every quiz and every difficulty played. Player class also has functions for Google and Facebook login, guest login, account deletion, setting a high score, and functions for managing setting save and load. Moreover, Player is inherited by Guest which has additional variable for guest number and RegisteredPlayer which is able to add score into leaderboard. On the other hand, Question has variables for text of question, correct answer, answer options and hint. Question class has a single function which is used to fill list with answer options. Question is inherited by ImageQuestion, VideoQuestion and SoundQuestion which have variables for image, video clip, and audio clip respectively. Furthermore, Quiz has variables for unanswered questions for current quiz, current question, current score and timer for current question. All functions in the Quiz class are used for controlling various functionalities during the gameplay. Quiz is aggregated from Questions. Player plays Quiz and also plays Practice which has functions which enable user to simulate a quiz or play specific category. Additionally, Player uses Controller which is used for starting the quiz which user has selected on difficulty user has selected, and for every controller there are 4 quizzes (4 types of quiz). Finally Player uses SettingsController for changing in – game settings.

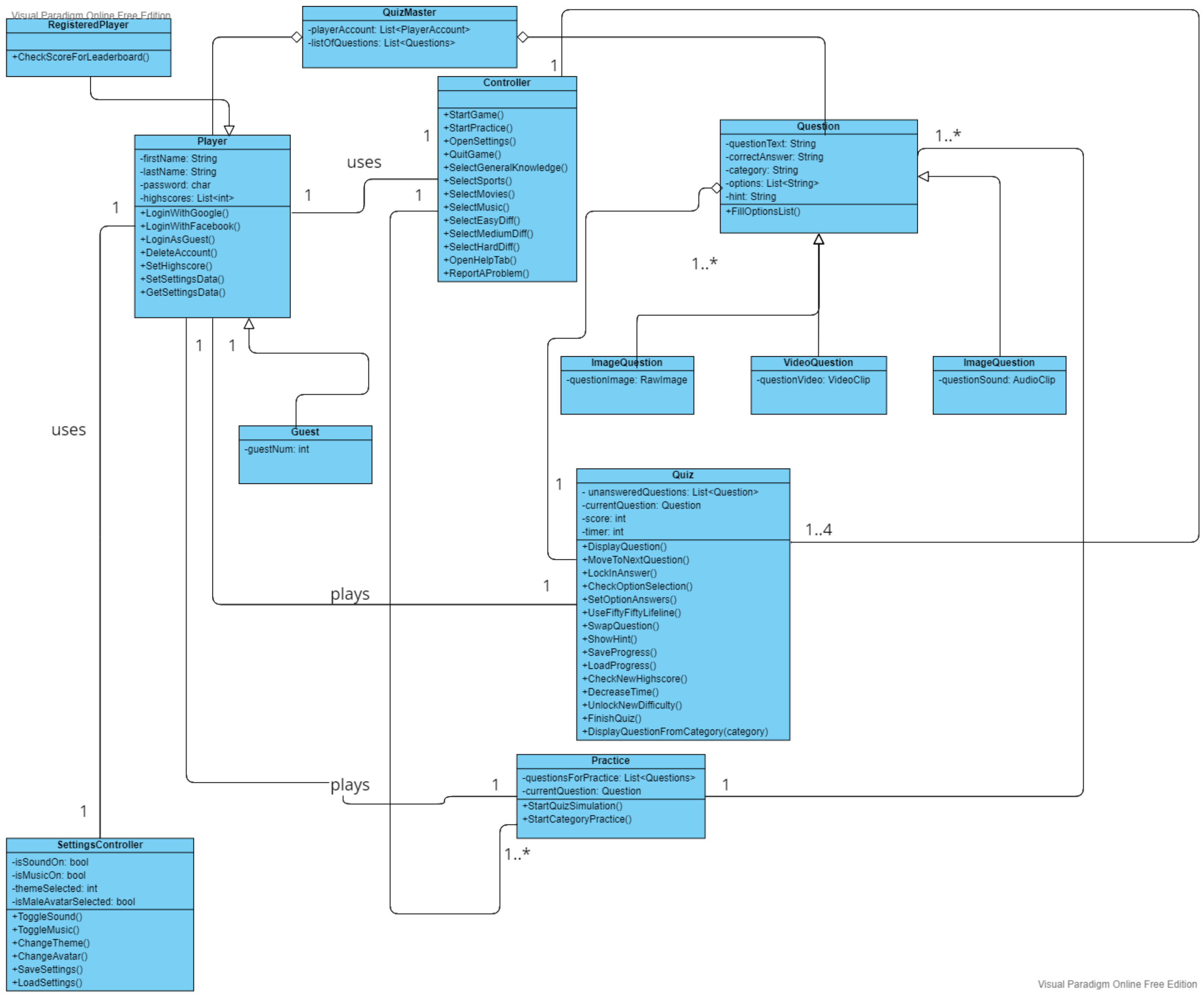


Figure 2. Class diagram

5.2 Sequence Diagrams

Sequence diagram below describes use case **Login** with use case ID: **UR1.1**

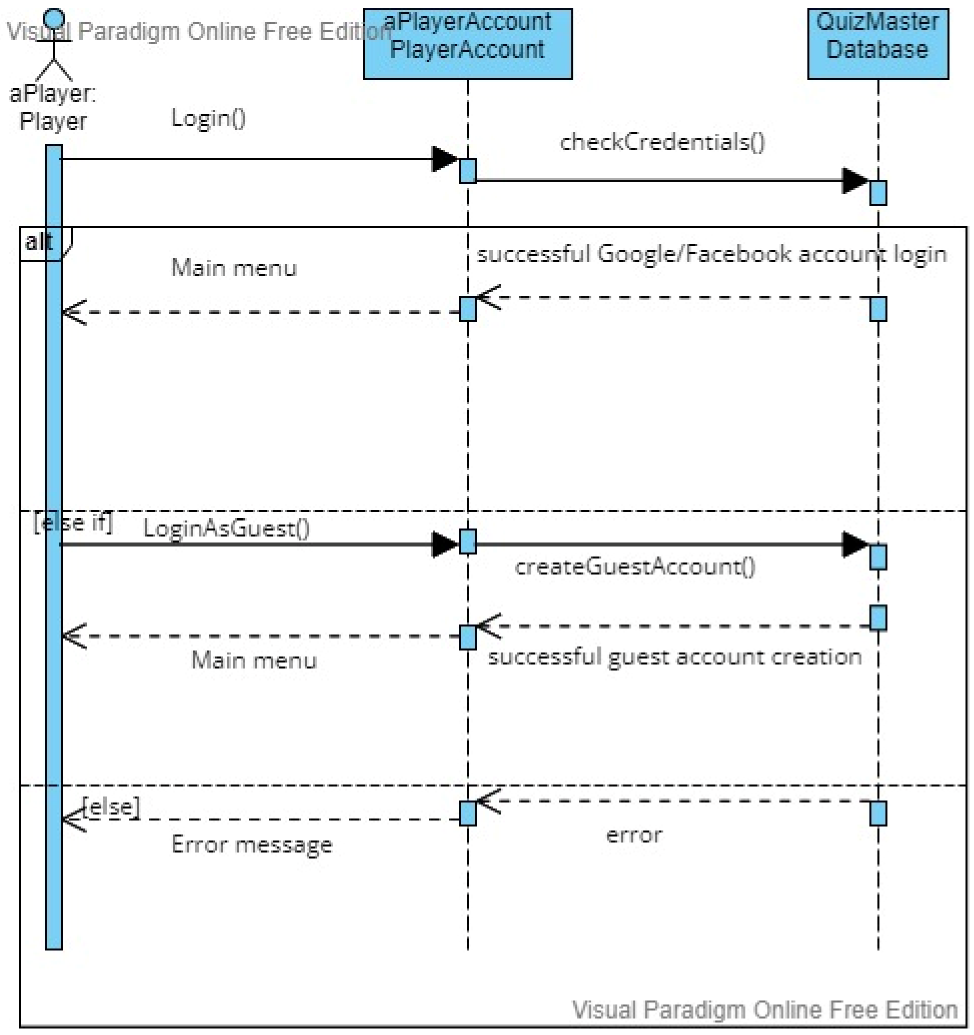
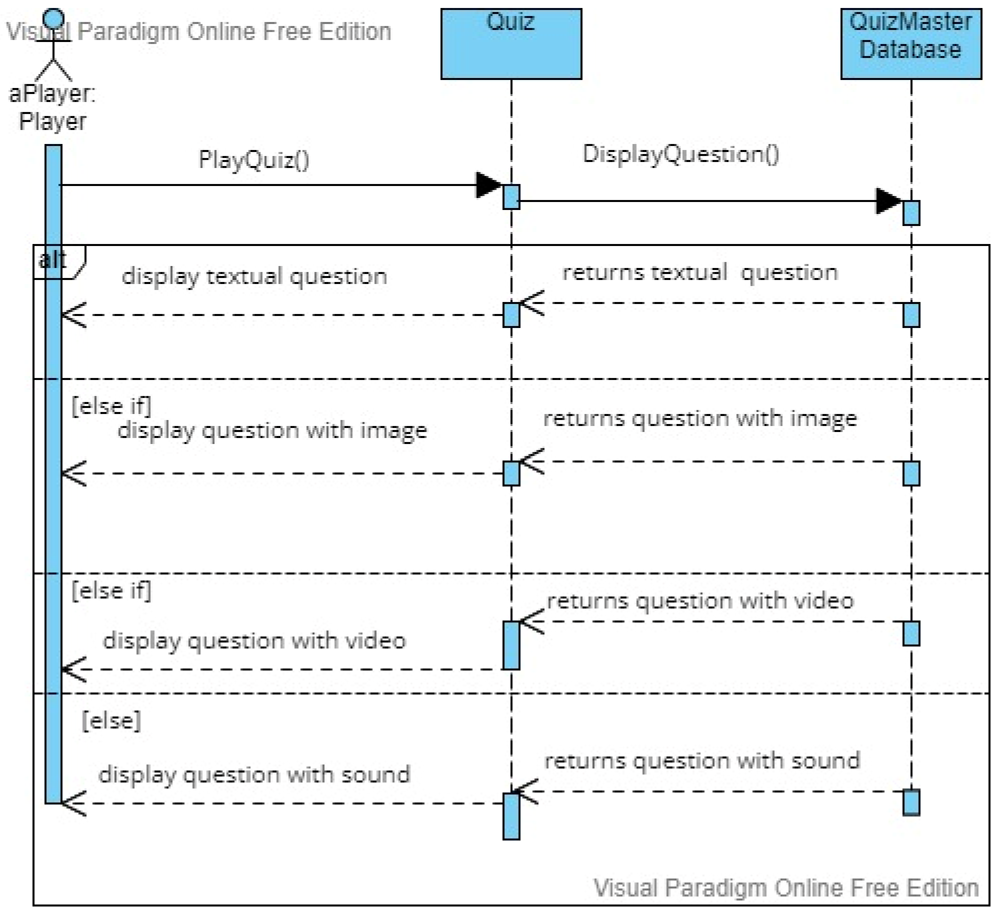

Visual Paradigm Online Free Edition
aPlayer: Player
aPlayerAccount PlayerAccount
QuizMaster Database
Login()
checkCredentials()
alt
successful Google/Facebook account login
Main menu
[else if]
LoginAsGuest()
createGuestAccount()
successful guest account creation
Main menu
[else]
error
Error message
Visual Paradigm Online Free Edition

Figure 3. Sequence Diagram 1.1

Sequence diagram below describes use case **Display various types of questions** with use case ID: **UR2.1**



Visual Paradigm Online Free Edition
aPlayer: Player
Quiz
QuizMaster Database
PlayQuiz()
DisplayQuestion()
alt
display textual question
returns textual question
[else if]
display question with image
returns question with image
[else if]
returns question with video
display question with video
[else]
returns question with sound
display question with sound
Visual Paradigm Online Free Edition

Figure 4. Sequence Diagram 2.1

Sequence diagram below describes use case **Answer the questions** with use case ID: **UR2.2**

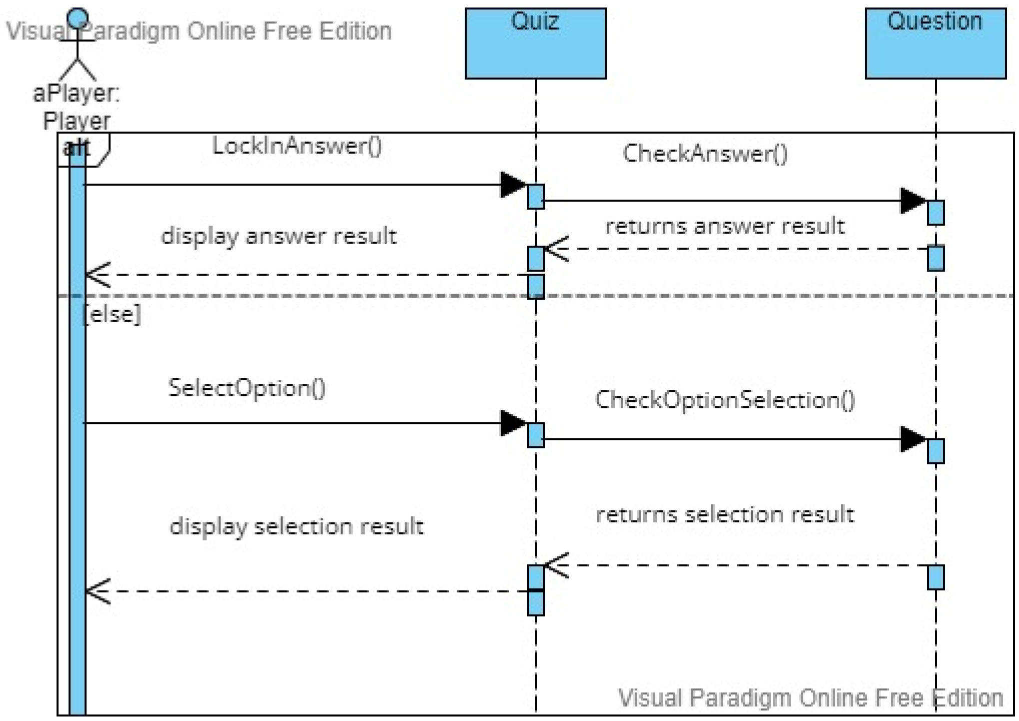


Figure 5. Sequence Diagram 2.2

Sequence diagram below describes use case **Use lifelines** with use case ID: **UR2.3**

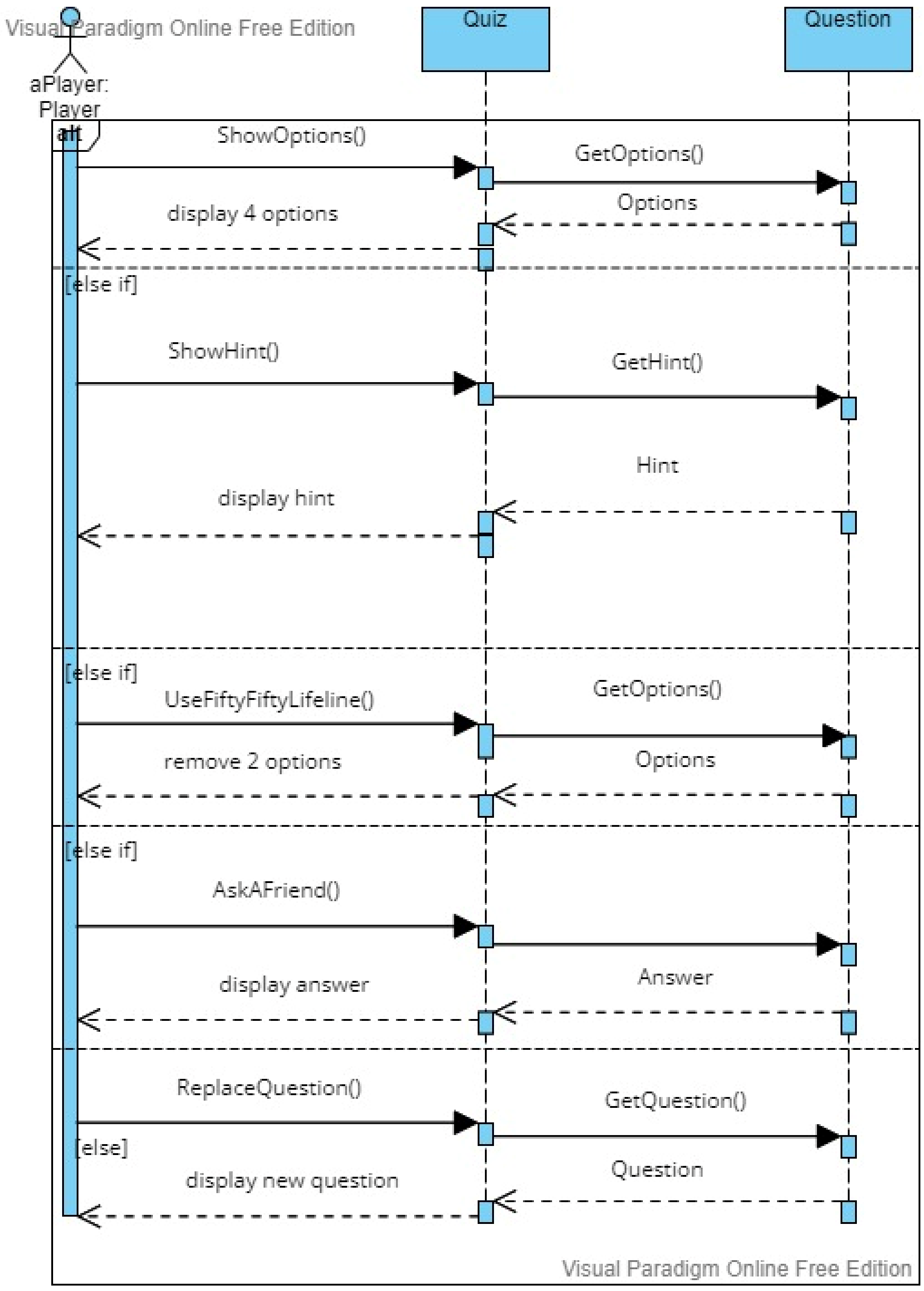


Figure 6. Sequence Diagram 2.3

Sequence diagram below describes use case **Unlock new quiz level** with use case ID: **UR2.4**

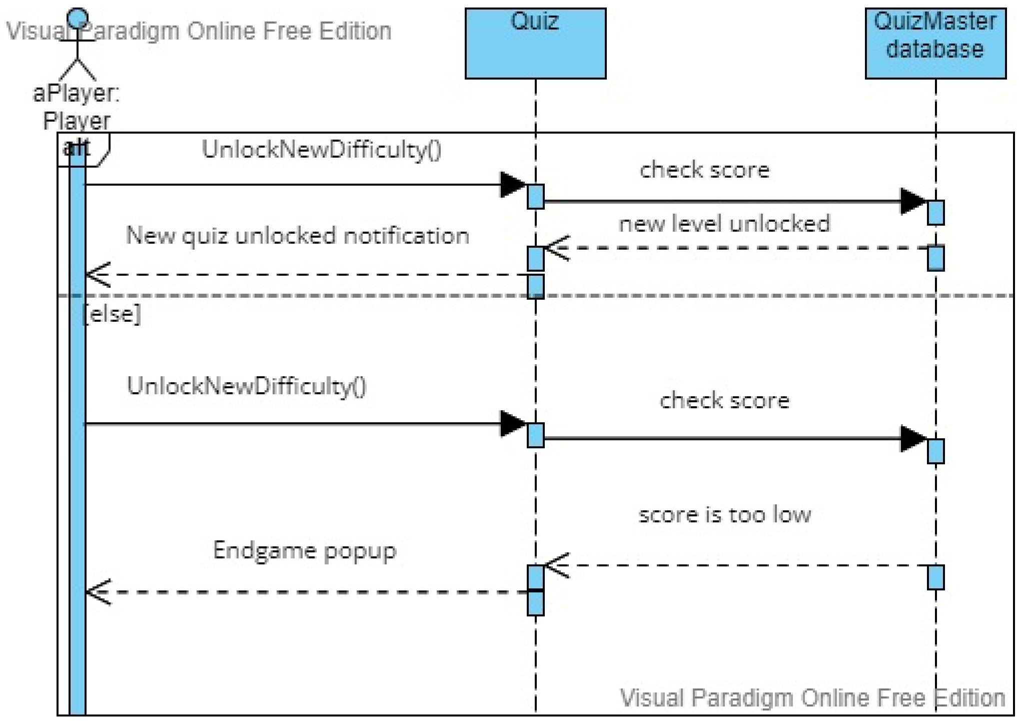


Figure 7. Sequence Diagram 2.4

Sequence diagram below describes use case **Play various levels** with use case ID: **UR2.5**

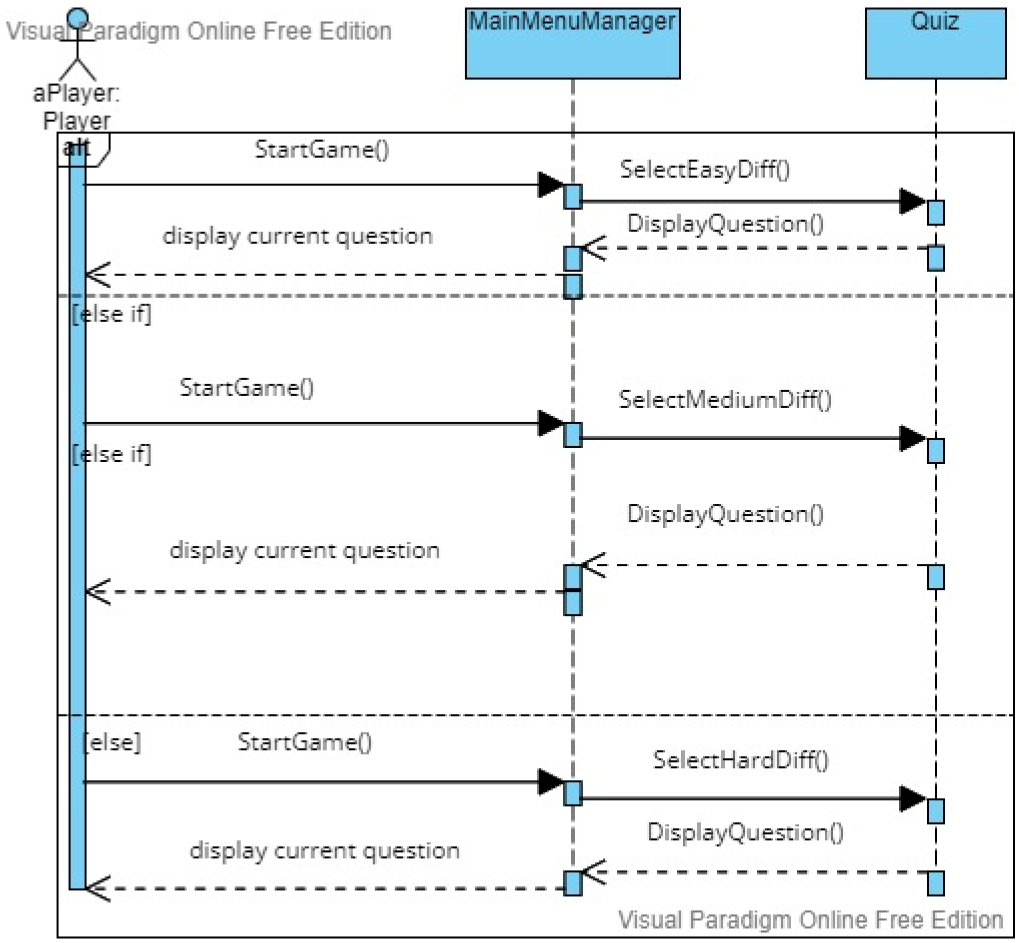


Figure 8. Sequence Diagram 2.5

Sequence diagram below describes use case **Save/Load quiz progress** with use case ID: **UR2.6**

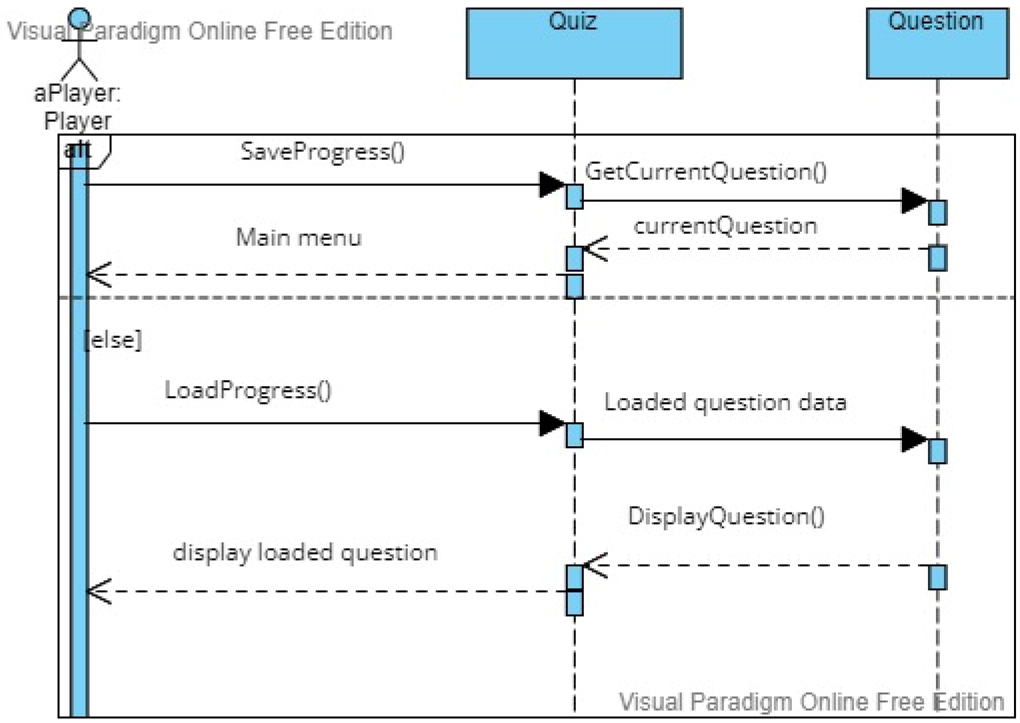


Figure 9. Sequence Diagram 2.6

Sequence diagram below describes use case **Set a high score** with use case ID: **UR2.7**

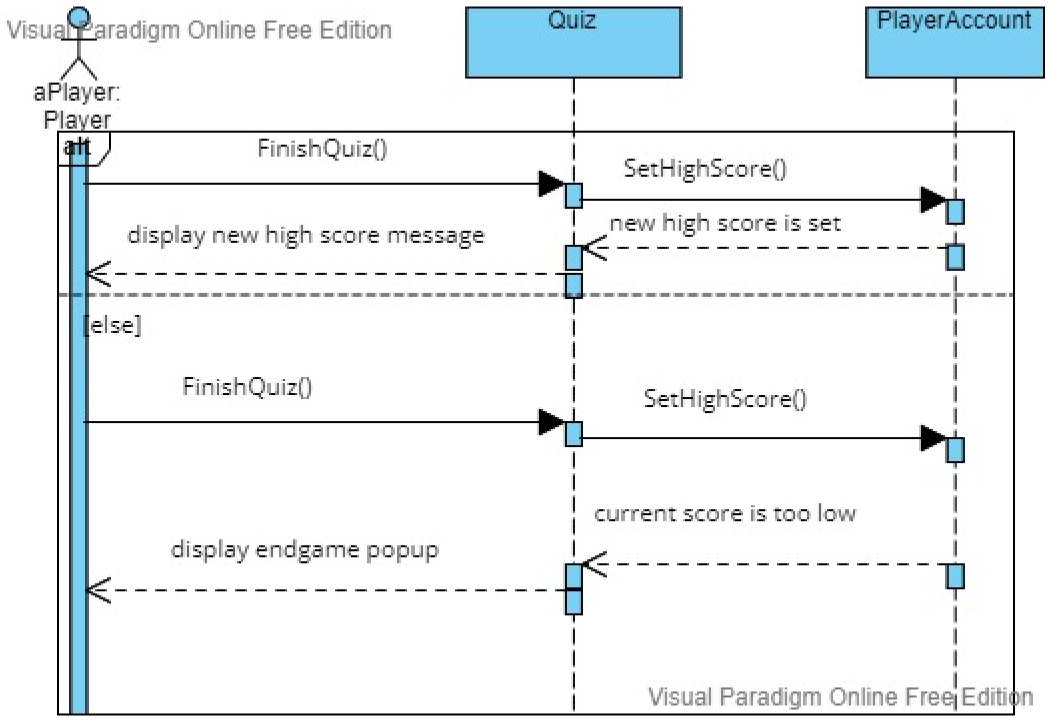


Figure 10. Sequence Diagram 2.7

Sequence diagram below describes use case **Practice the whole quiz** with use case ID: **UR3.1**

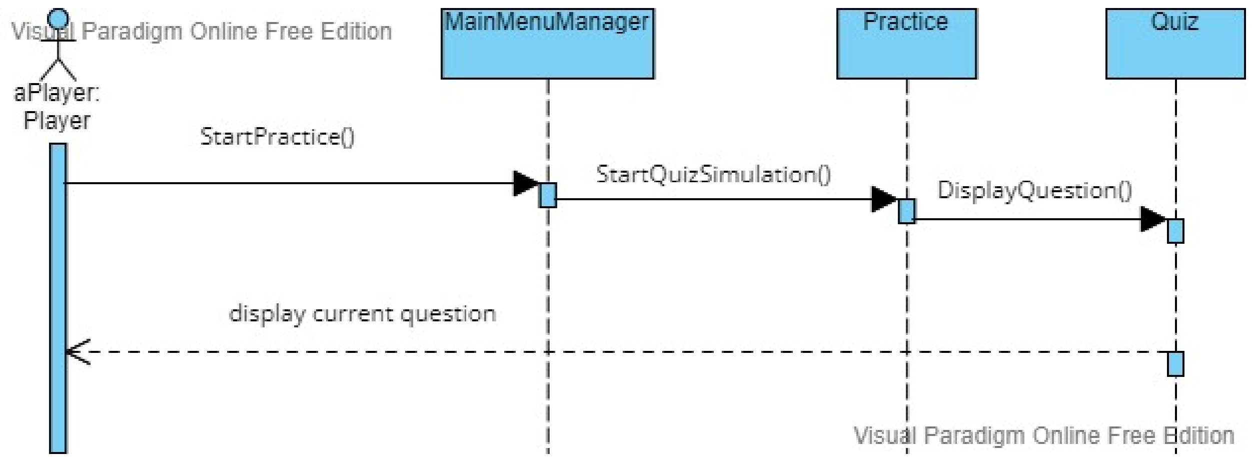
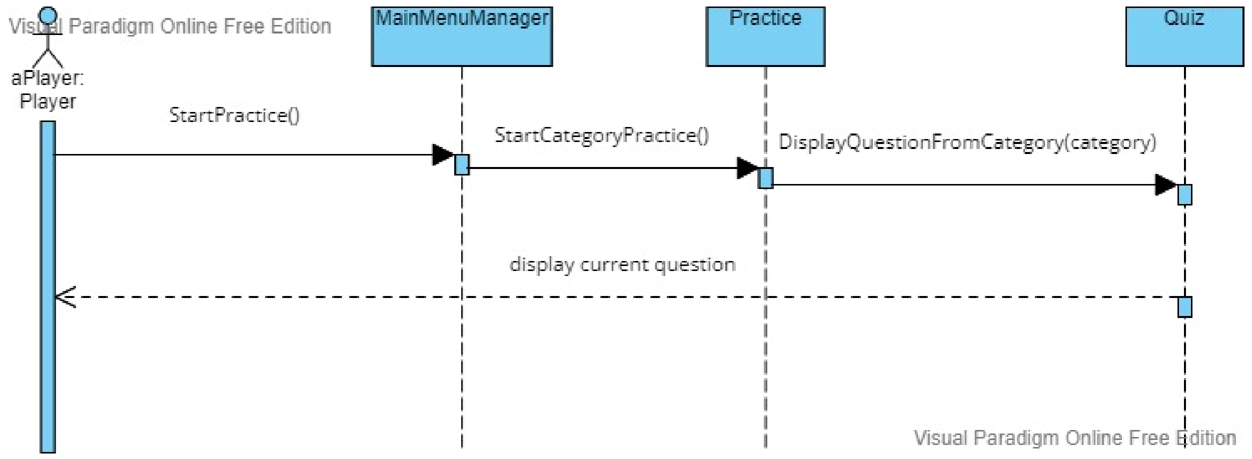


Figure 11. Sequence Diagram 3.1

Sequence diagram below describes use case **Practice particular questions** with use case ID: **UR3.1**



Visual Paradigm Online Free Edition
aPlayer: Player
MainMenuManager
Practice
Quiz
StartPractice()
StartCategoryPractice()
DisplayQuestionFromCategory(category)
display current question
Visual Paradigm Online Free Edition

Figure 12. Sequence Diagram 3.2

Sequence diagram below describes use case **Toggle game sound** with use case ID: **UR4.1**

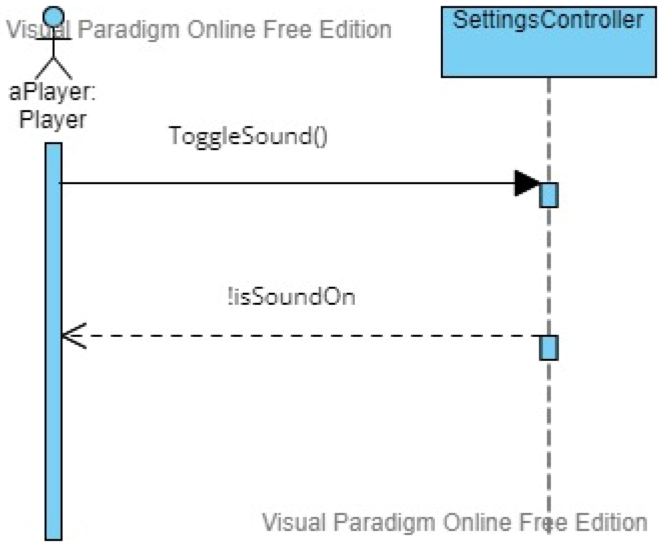

Visual Paradigm Online Free Edition
aPlayer: Player
SettingsController
ToggleSound()
!isSoundOn
Visual Paradigm Online Free Edition

Figure 13. Sequence Diagram 4.1

Sequence diagram below describes use case **Toggle game music** with use case ID: **UR4.2**

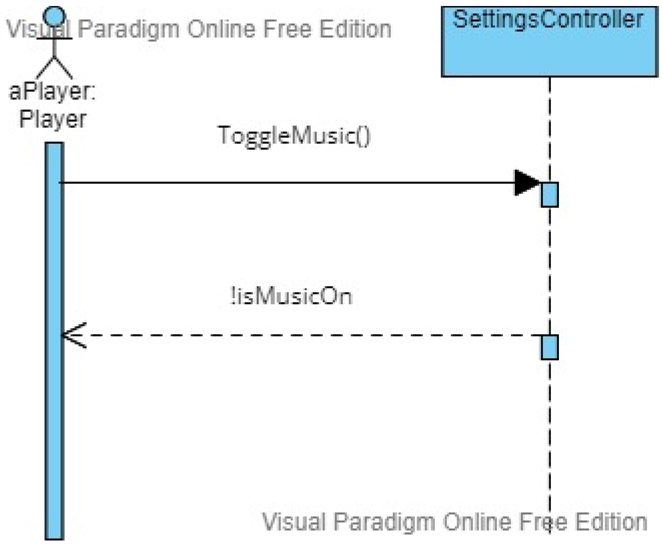

Visual Paradigm Online Free Edition
aPlayer: Player
SettingsController
ToggleMusic()
!isMusicOn
Visual Paradigm Online Free Edition

Figure 14. Sequence Diagram 4.2

Sequence diagram below describes use case **Change game theme** with use case ID: **UR4.3**

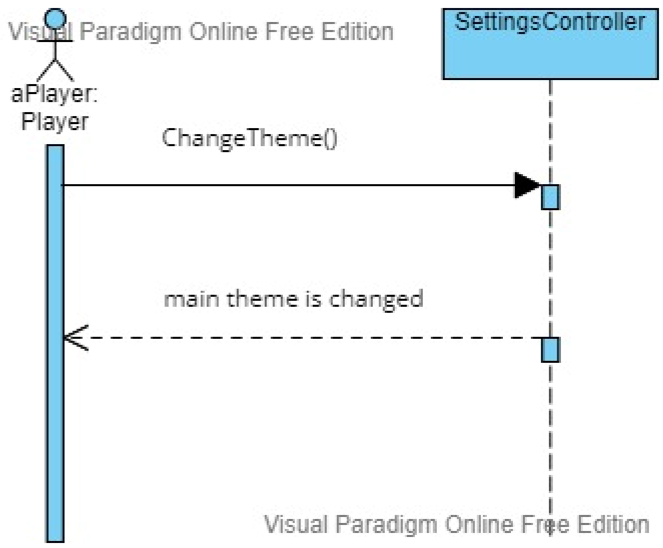

Visual Paradigm Online Free Edition
aPlayer: Player
SettingsController
ChangeTheme()
main theme is changed
Visual Paradigm Online Free Edition

Figure 15. Sequence Diagram 4.3

Sequence diagram below describes use case **Change in-game avatar** with use case ID: **UR4.4**

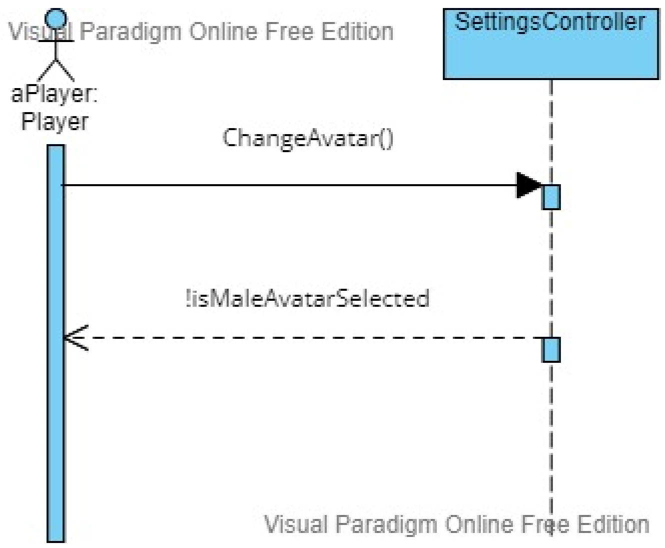
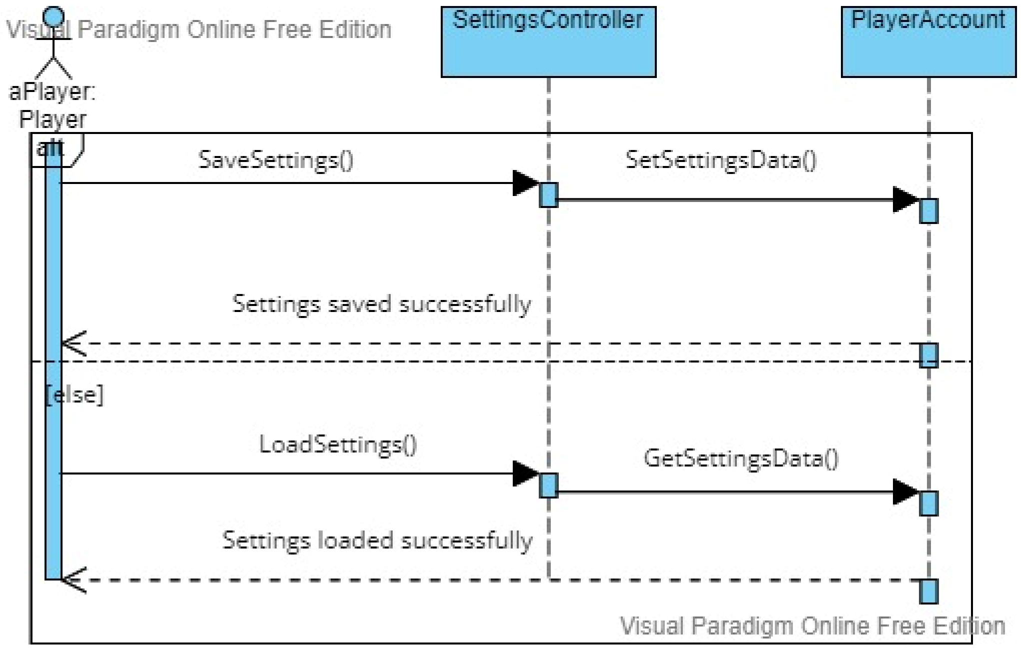

Visual Paradigm Online Free Edition
aPlayer: Player
SettingsController
ChangeAvatar()
!isMaleAvatarSelected
Visual Paradigm Online Free Edition

Figure 16. Sequence Diagram 4.4

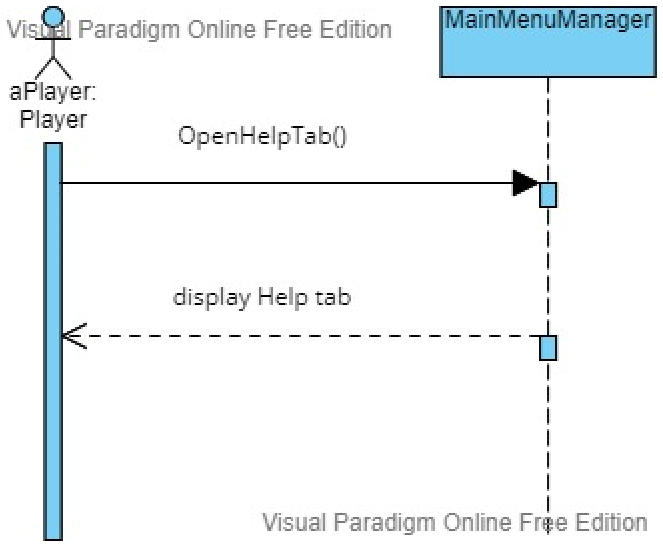
Sequence diagram below describes use case **Save/Load user settings** with use case ID: **UR4.5**



Visual Paradigm Online Free Edition
aPlayer: Player
SettingsController
PlayerAccount
alt
SaveSettings()
SetSettingsData()
Settings saved successfully
[else]
LoadSettings()
GetSettingsData()
Settings loaded successfully
Visual Paradigm Online Free Edition

Figure 17. Sequence Diagram 4.5

Sequence diagram below describes use case **Open and browse help tab** with use case ID: **UR5.1**



Visual Paradigm Online Free Edition
aPlayer: Player
MainMenuManager
OpenHelpTab()
display Help tab
Visual Paradigm Online Free Edition

Figure 18. Sequence Diagram 5.1

Sequence diagram below describes use case **View leaderboard** with use case ID: **UR6.1**

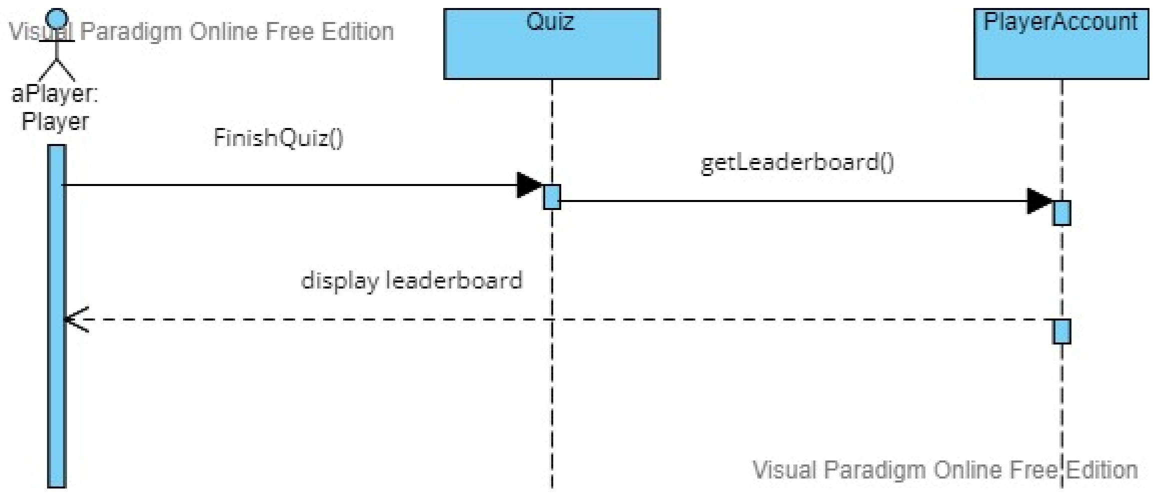


Figure 19. Sequence Diagram 6.1

Sequence diagram below describes use case **Enter the score to the leaderboard** with use case ID: **UR6.2**

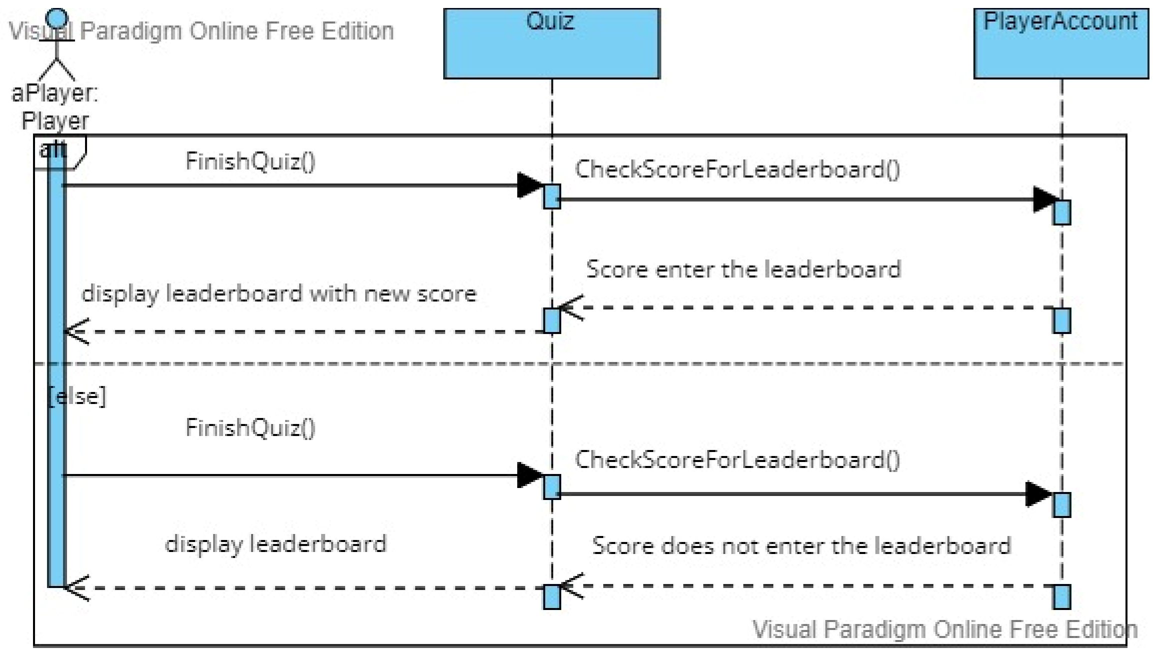

Visual Paradigm Online Free Edition
aPlayer: Player
Quiz
PlayerAccount
alt
FinishQuiz()
CheckScoreForLeaderboard()
Score enter the leaderboard
display leaderboard with new score
[else]
FinishQuiz()
CheckScoreForLeaderboard()
display leaderboard
Score does not enter the leaderboard
Visual Paradigm Online Free Edition

Figure 20. Sequence Diagram 6.2

Sequence diagram below describes use case **Problem inquiry** with use case ID: **UR7.1**

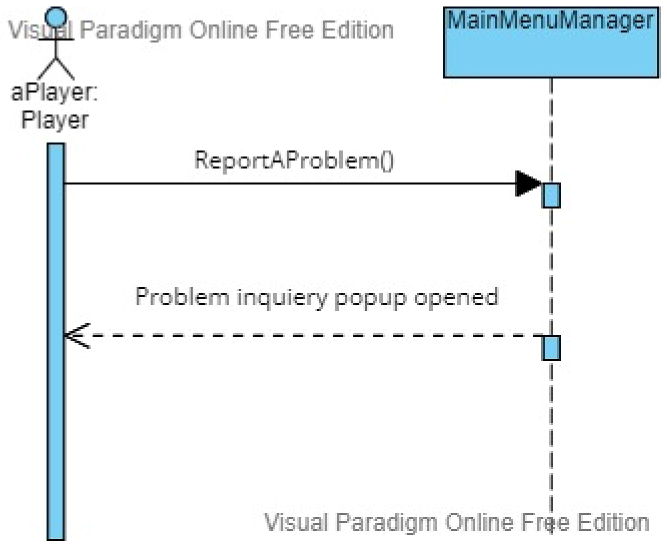

Visual Paradigm Online Free Edition
aPlayer: Player
MainMenuManager
ReportAProblem()
Problem inquiery popup opened
Visual Paradigm Online Free Edition

Figure 21. Sequence Diagram 7.1

5.3 User Interface

UI #1 Login – Screen on which user is able to login using Google account, Facebook account or login as guest

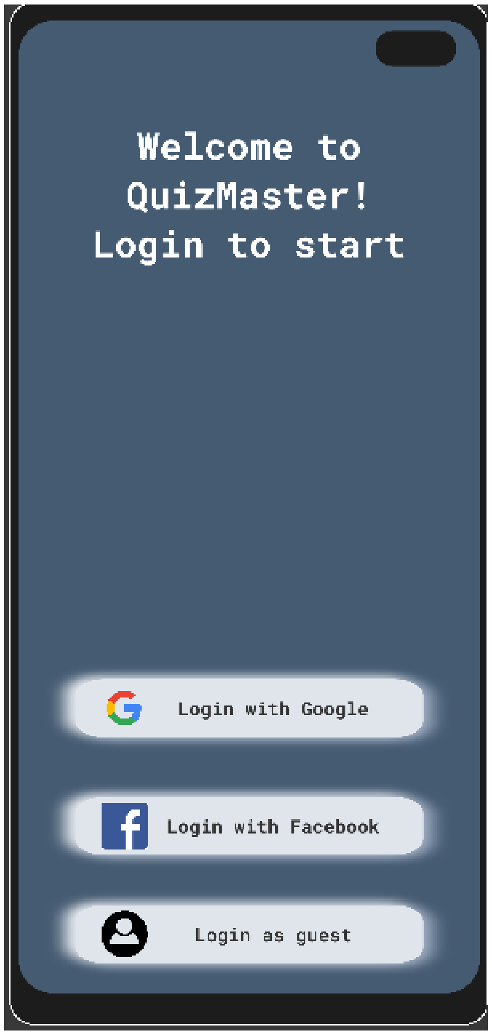


Figure 22. Login screen

UI #2 Main menu – After logging in user is redirected to the screen below which is the main menu of the game

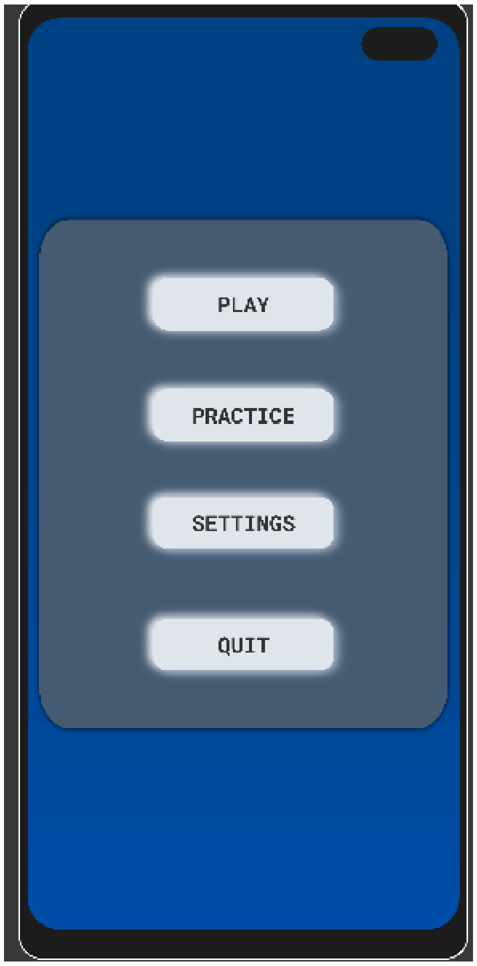


Figure 23. Main menu

UI #3 Quiz selection – After the user click on “Play” or “Practice” button, he/she is redirected to the screen below in which the user can pick which quiz to play/practice.

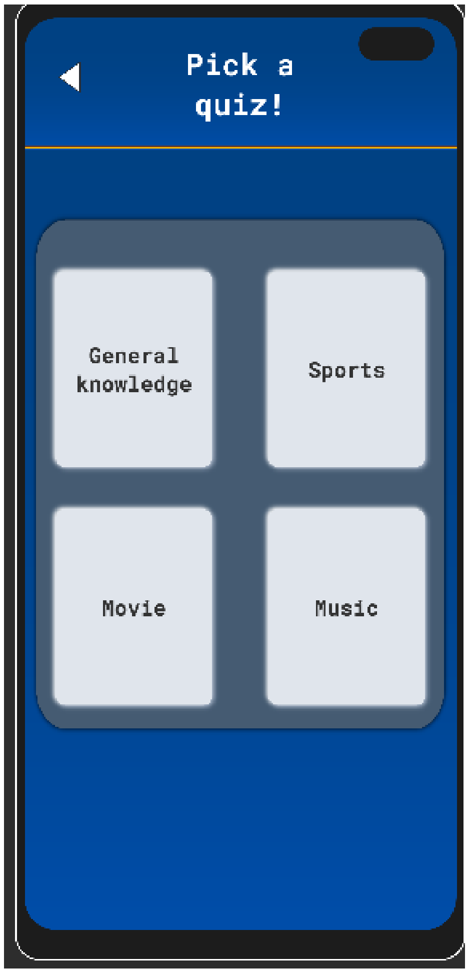


Figure 24. Quiz selection

UI #4 Difficulty selection – After picking the quiz, the user is prompted to select difficulty to play/practice on.

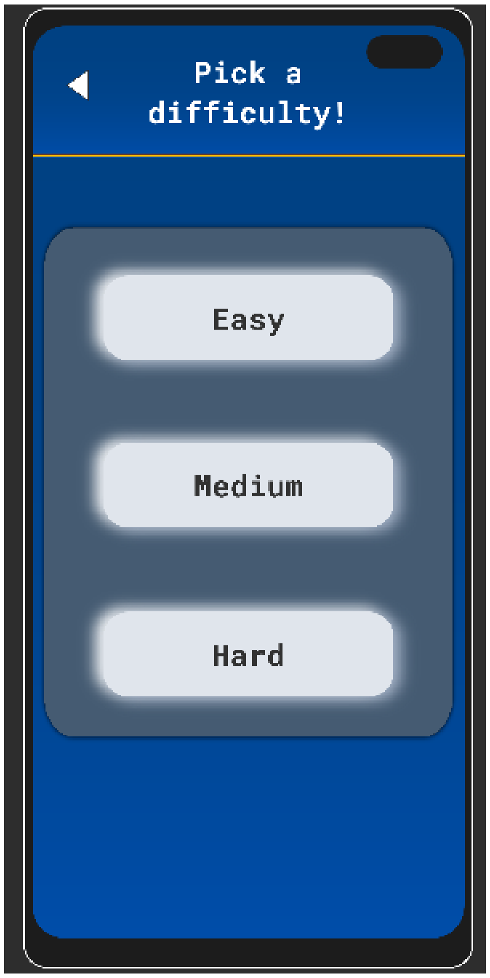
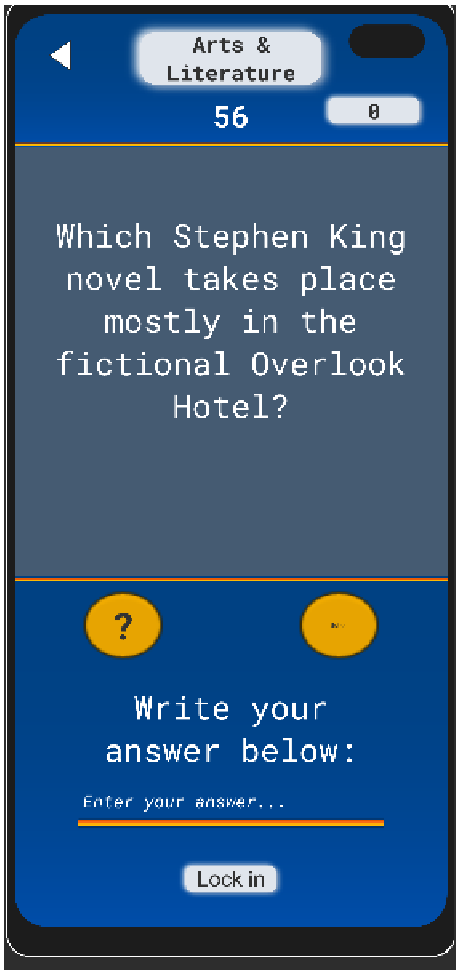


Figure 25. Difficulty selection

UI #5 Textual question – When playing the quizzes, user shall be able to see 4 types of questions. One of those types, textual question is shown in the figure below. User is also able to see the answer panel with input field, category name, score and timer.



Which Stephen King novel takes place mostly in the fictional Overlook Hotel?

Figure 26. Textual question

UI #6 Image question – Image question is shown in the figure below. User is also able to see the answer panel with input field, category name, score, and timer.

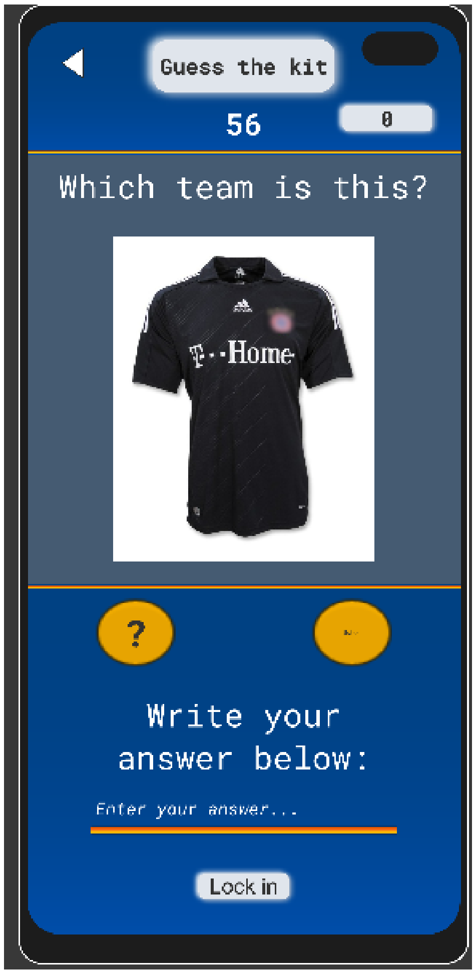


Figure 27. Image question

UI #7 Video question – Video question is shown in the figure below. User is also able to see the answer panel with input field, category name, score, and timer.

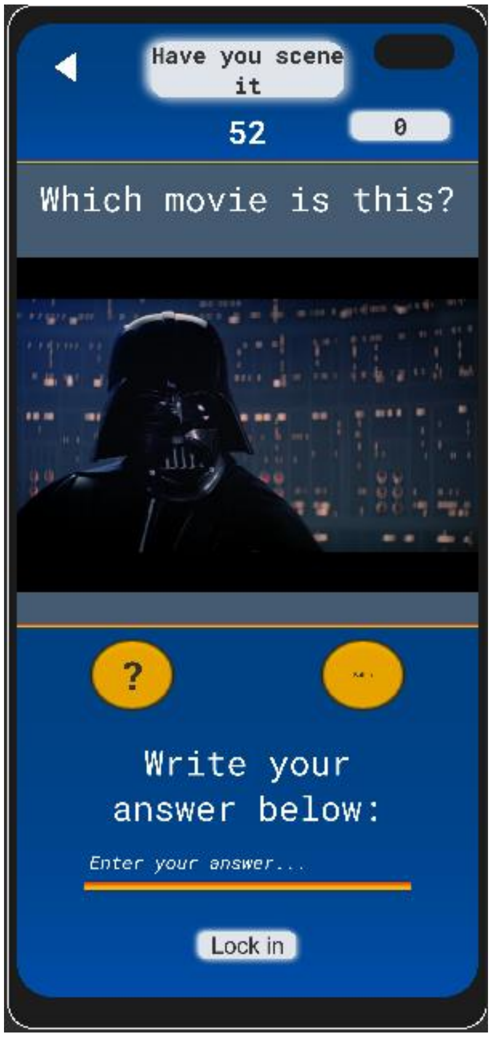


Figure 28. Video question

UI #8 Sound question – Sound question is shown in the figure below. User is also able to see the answer panel with input field, category name, score, and timer.

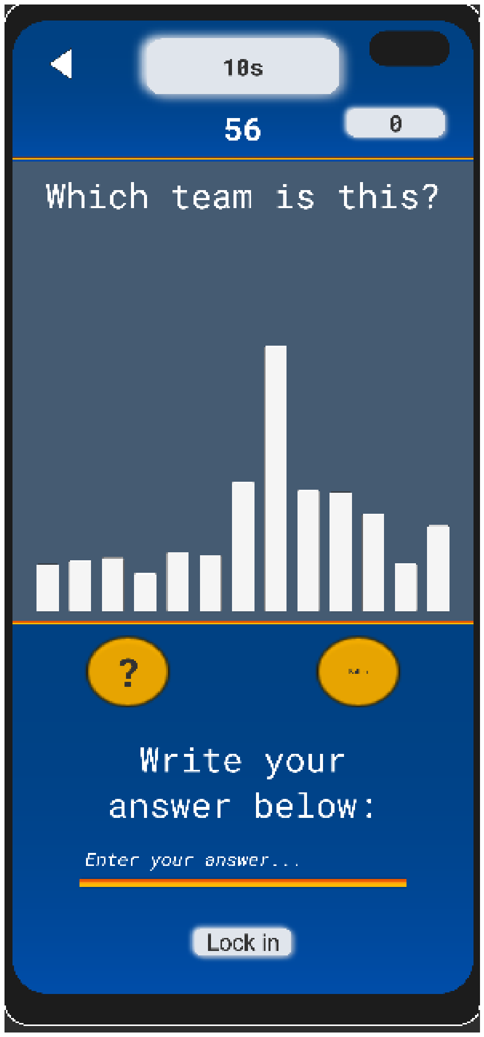


Figure 29. Sound question

UI #9 Answer options – User is able to reveal 4 answer options, as shown on the figure below, by clicking on the “Show options” button.

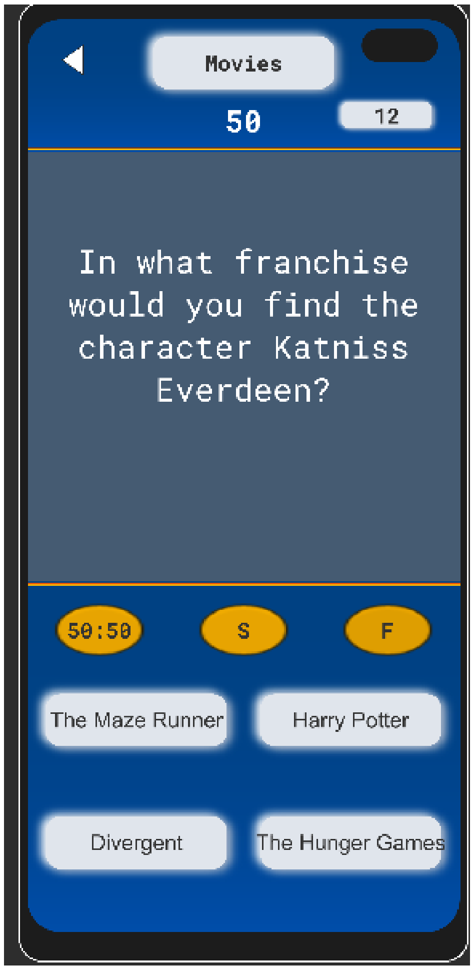
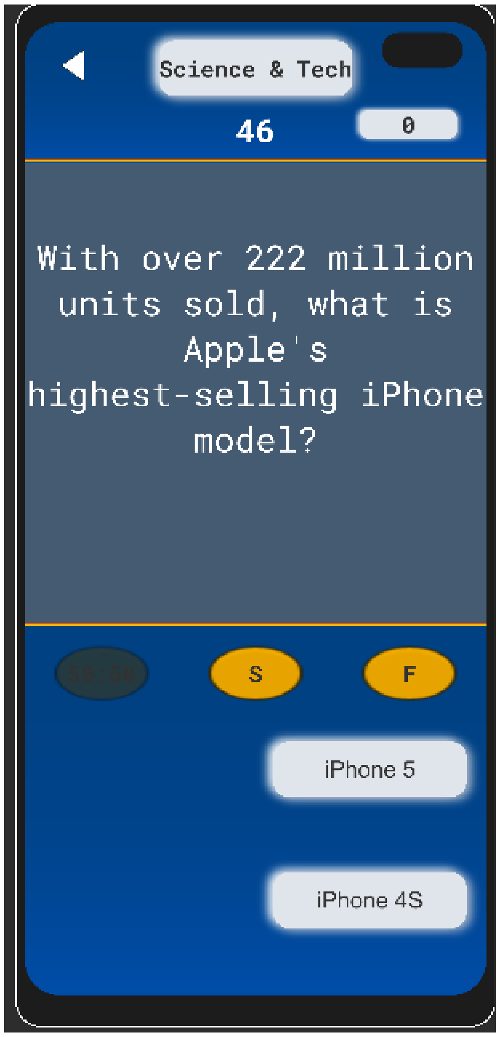


Figure 30. Answer options

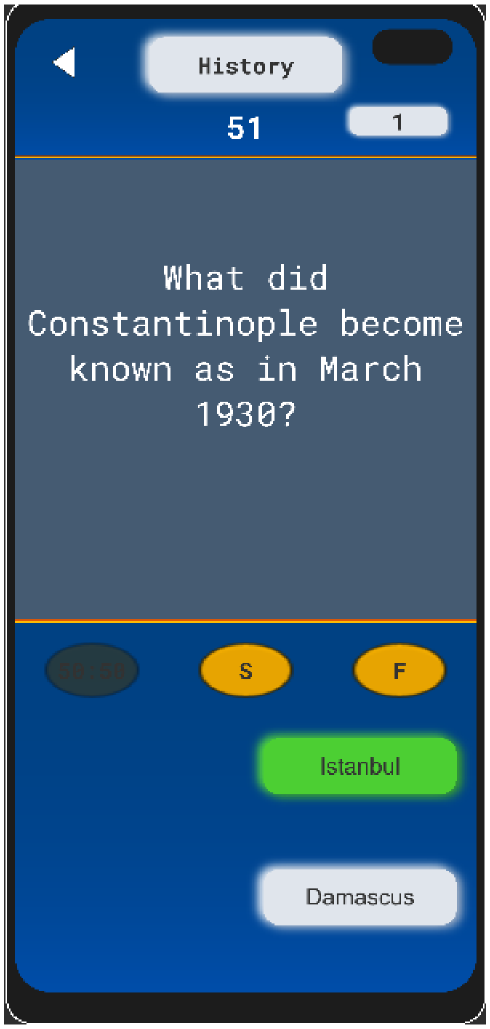
UI #10 50:50 lifeline – User can remove 2 incorrect answers, once per game, by clicking on the "50:50" button. After using this lifeline, the button is greyed out and disabled.



With over 222 million units sold, what is Apple's highest-selling iPhone model?

Figure 31. 50:50 lifeline

UI #11 Correct answer selected – If the user selects the correct answer, the button with the correct answer will change its color to green, to notify the user that the selected answer was correct



What did Constantinople become known as in March 1930?

Figure 32. Correct answer from selected options

UI #12 Ask a friend lifeline – User is able to use “Ask a friend” lifeline, once per game, by clicking on the button with letter “F”. The “friend” will then type out the answer with 90% correctness rate. After using this lifeline, the button is greyed out and disabled.

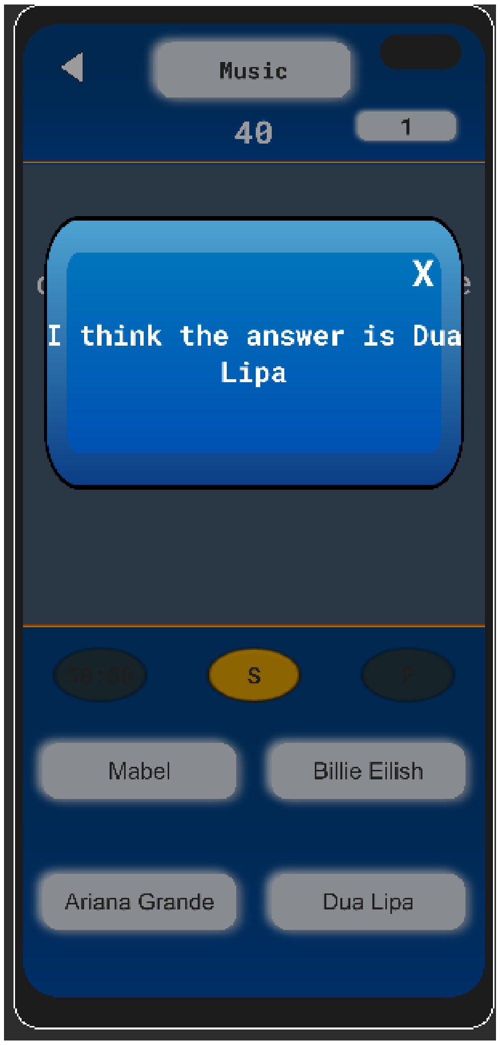
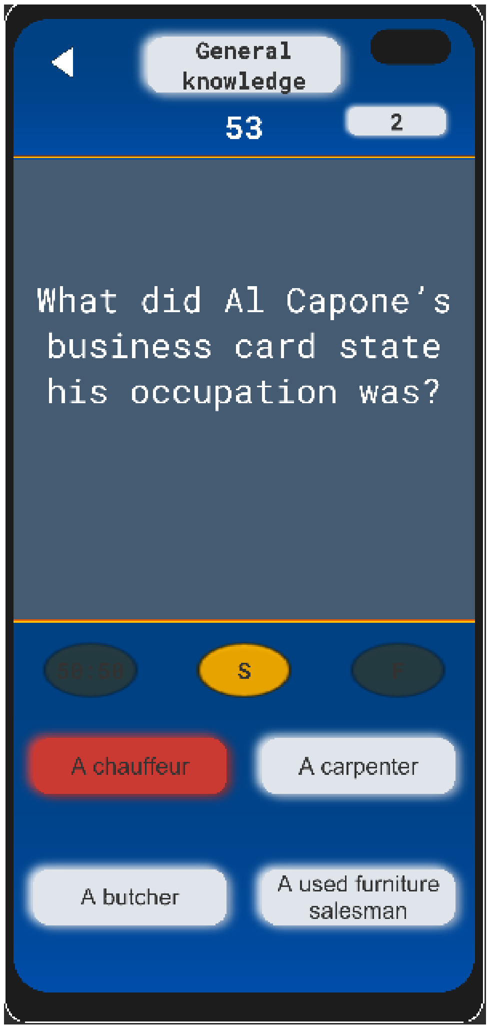


Figure 33. Ask a friend lifeline

UI #13 Incorrect answer selected – If the user selects the incorrect answer, the button with the incorrect answer will change its color to red, to notify the user that the selected answer was incorrect



What did Al Capone's business card state his occupation was?

A chauffeur

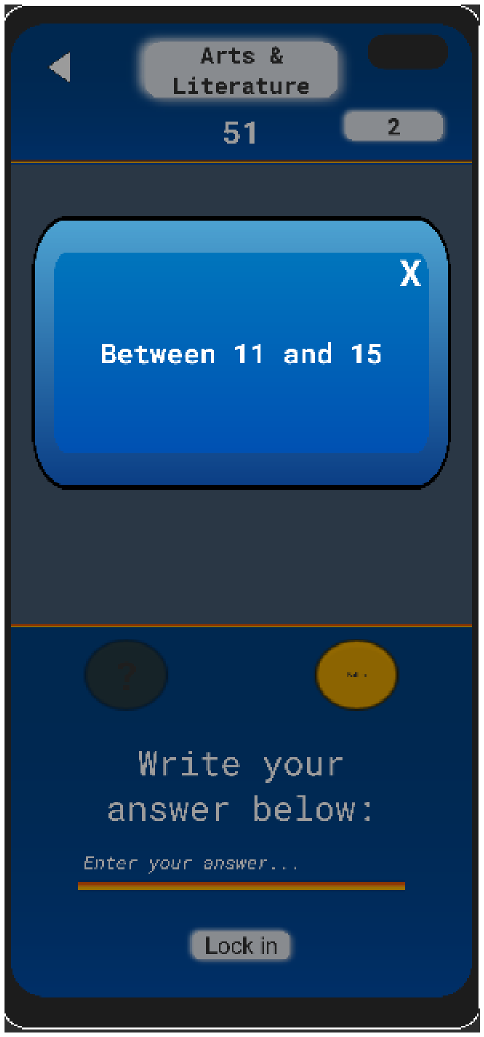
A carpenter

A butcher

A used furniture salesman

Figure 34. Incorrect answer from options

UI #14 Hint lifeline – User can use Hint lifeline, once per game, by clicking on the button with "?" character. After clicking the button, the user will see a popup with a hint which might help to remember the correct answer. After using this lifeline, the button is greyed out and disabled.



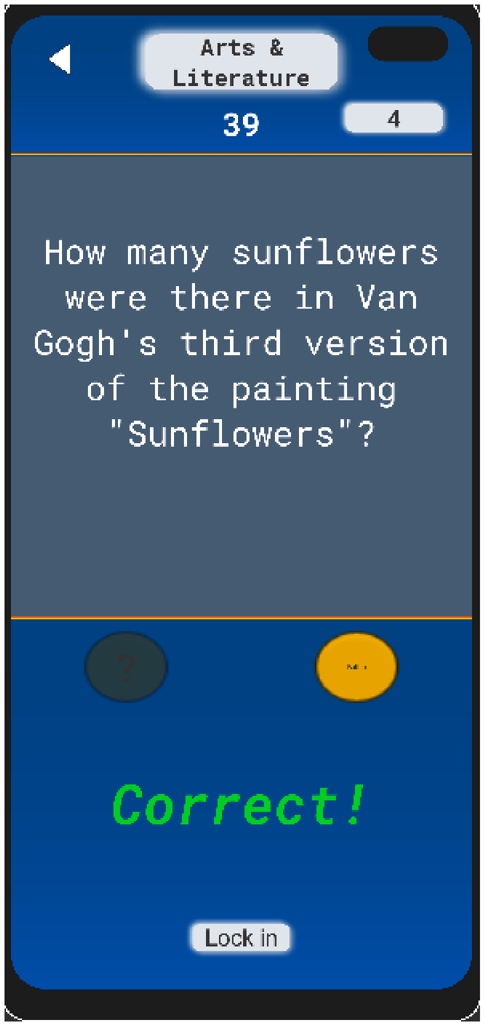
Between 11 and 15

Write your answer below:

Enter your answer...

Figure 35. Hint lifeline

UI #15 Correct answer from input field – If the user types in the correct answer and clicks “Lock in” button, then the user shall receive a message notifying the user that the entered answer was correct.



How many sunflowers were there in Van Gogh's third version of the painting "Sunflowers"?

Correct!

Figure 36. Correct typed in answer

UI #16 Incorrect answer from input field – If the user types in the incorrect answer and clicks "Lock in" button, then the user shall receive a message notifying the user that the entered answer was incorrect.

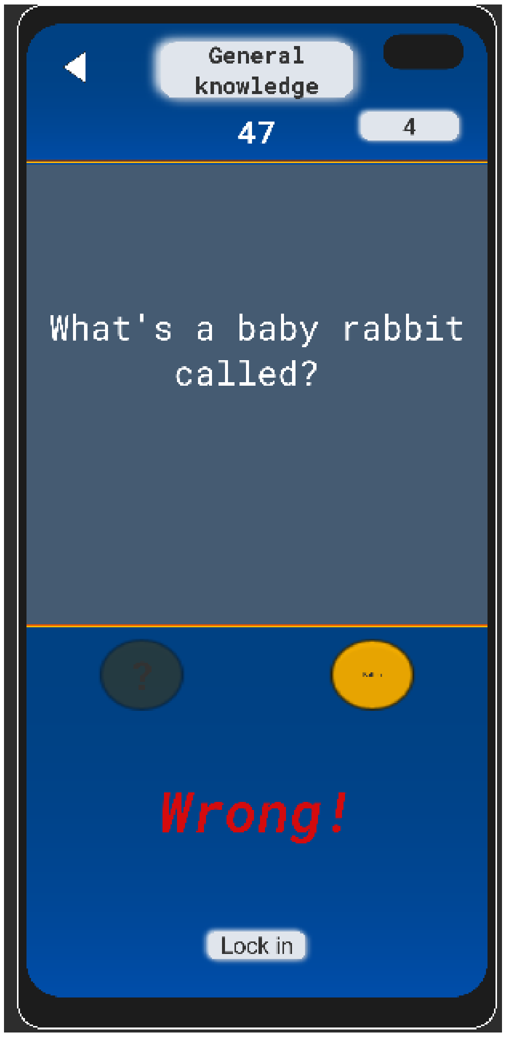


Figure 37. Incorrect typed in answer

UI #17 Endgame popup – When the user finishes the quiz, endgame popup shall appear, as shown in the image below. The popup will display achieved score and notify user if the new level is unlocked.

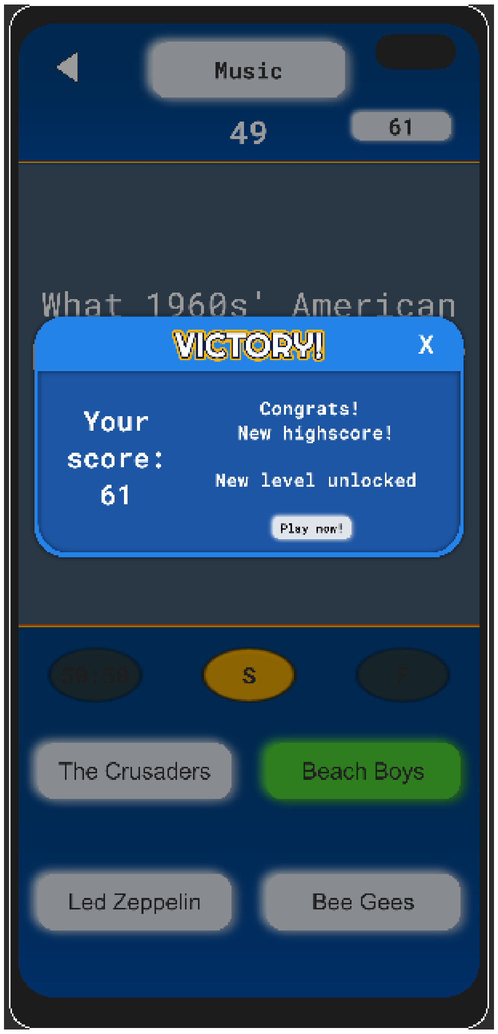


Figure 38. Endgame popup

UI #18 Practice panel – When user clicks on the “Practice button” and selects the type of quiz and difficulty, then the user is redirected to the screen on the figure below. From this screen user is able to choose either quiz simulation or category practice by clicking on their respective buttons.

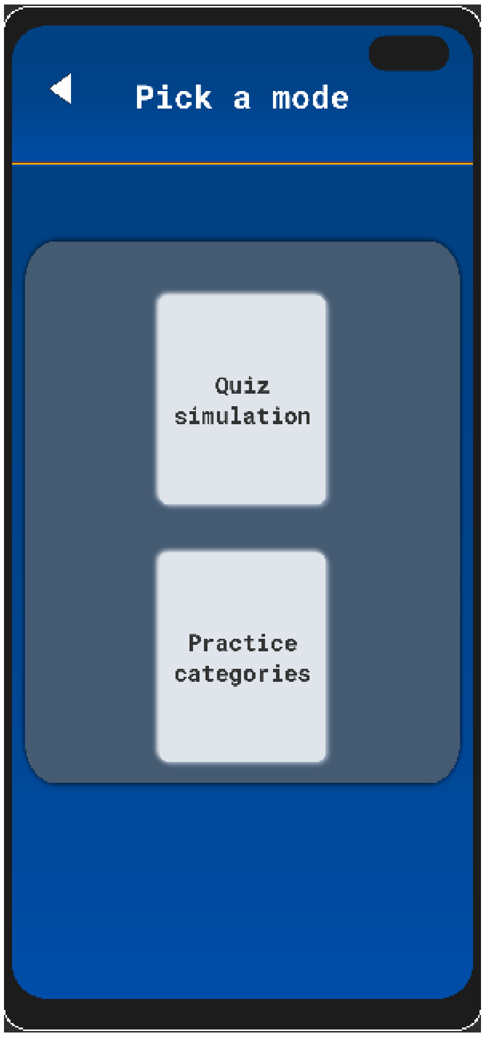


Figure 39. Pick a practice mode

Chapter 6 – Conclusion

To sum up, this paper shows the complete design and implementation of quiz game "Quizmaster" for Android and iOS devices. The game was fully developed using Unity engine and C# programming language. Unity is a game engine which offers numerous, user - oriented functionalities required for game development and other industries such as art and data science. Those functionalities include use of various types of UI such as images, videos and buttons, animations, device simulators, etc. Furthermore, related market research was done in order to decide which features are popular among users of similar games and what new features are needed in order to make "Quizmaster" competitive on the market, and more importantly attractive to users. Additionally, system is broken down into seven system features, each with one or more user and functional requirements. User requirements are explained in comprehensively using detailed use cases and sequence diagrams, whereas the bigger picture of the system design is shown in the class diagram (Figure 2).

Moreover, some examples of the future work which can be done in this mobile game include monetization, introduction of multiplayer mode and web version of "Quizmaster". Monetization of the game includes the addition of banners, interstitial ads, and reward videos, and it is a very important aspect of game development in general. It can be done using various services available such as "Google AdMob", "AppLovin Max", "Facebook ads" and "Unity ads". Introduction of multiplayer mode such as head – to – head matches between two users and web version of the game shall increase the attractiveness of the game and expand the player base. Finally, web version can also include quiz creation which would enable users to create their own quizzes which would later, if satisfiable, be imported into the game. In a nutshell, these examples of the future work in combination with the existing system features could turn "Quizmaster" into one of the most beloved quiz games in the market.

References

[1]: de Hesselle, L. C., Rozgonjuk, D., Sindermann, C., Pontes, H. M., & Montag, C. (2021). The associations between Big Five personality traits, gaming motives, and self-reported time spent gaming. Personality and Individual Differences, 171, 110483.

[2]: Wilkinson, K., Dafoulas, G., Garelick, H., & Huyck, C. (2019). Are quiz-games an effective revision tool in Anatomical Sciences for Higher Education and what do students think of them? British Journal of Educational Technology, 51(3), 761–777.

[3]: Jon Brodkin, N.p.. Web. 4 Dec 2014. http://slashdot.org/topic/cloud/how-unity3dbecome-a-game-development-beast/, Last accessed on 11th June 2022.

[4]: Haas, J. (2014) A history of the unity game engine. Worcester, UK:
Worcester Polytechnic Institute.

[5]: Zali, T. K. M., Samsiah, N., Hadi, A., & Aliff, M. (2019). Attractiveness Analysis of Quiz Games. International Journal of Advanced Computer Science and Applications, 10(8).

[6]: Aydin, B. I., Yilmaz, Y. S., & Demirbas, M. (2017). A crowdsourced "Who wants to be a millionaire?" player. Concurrency and Computation: Practice and Experience, 33(8)

[7]: Sanders, J., Arce-Trigatti, A., & Arce, P. (2020). Promoting student problem-identification skills via a Jeopardy-inspired game within the Renaissance Foundry. Education for Chemical Engineers, 30, 49–59.

[8]: Foxman, M. (2019). United We Stand: Platforms, Tools and Innovation With the Unity Game Engine. Social Media + Society, 5(4), 205630511988017.

[9]: Sommerville, I. (2018). Software Engineering, 10th Edition (10th ed.). Pearson India