|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |

|  |
| --- |
| Software Development Team |
| Let’s Quiz |
| Initial Requirement Model |

Contents

[1 Introduction 6](#_Toc526445071)

[1.1 Purpose 7](#_Toc526445072)

[1.2 Scope 7](#_Toc526445073)

[1.3 Deemed out of Scope 7](#_Toc526445074)

[2 System-Wide Functional Requirements 7](#_Toc526445075)

[3 System Qualities 8](#_Toc526445076)

[3.1 External interface requirements (Non-functional requirements) 8](#_Toc526445077)

[3.1.1 Usability 8](#_Toc526445078)

[3.1.2 Reliability 8](#_Toc526445079)

[3.1.3 Accessibility 8](#_Toc526445080)

[3.1.4 Performance 8](#_Toc526445081)

[3.1.5 Capacity 8](#_Toc526445082)

[4 System Constraints 9](#_Toc526445083)

[5 Assumptions and dependencies 9](#_Toc526445084)

[6 Diagrams 9](#_Toc526445085)

[6.1 Domain Diagram 9](#_Toc526445086)

[7 Functional Requirement Use Case Descriptions 10](#_Toc526445087)

[7.1 Use Case Descriptions 10](#_Toc526445088)

[7.1.1 Use Case: Launching Game 10](#_Toc526445089)

[7.1.2 Use Case: Register 10](#_Toc526445090)

[7.1.3 Use Case: Login 10](#_Toc526445091)

[7.1.4 Use Case: Login Registered Users-Sub-function End Goal: Login 10](#_Toc526445092)

[7.1.5 Use Case: Login with Facebook-Sub-function End Goal: Login 11](#_Toc526445093)

[7.1.6 Use Case: Play as Guest-Sub-function End Goal: Login 11](#_Toc526445094)

[7.1.7 Use Case: End Goal: User Submit Question 11](#_Toc526445095)

[7.1.8 Use Case: End Goal: Start a Game 11](#_Toc526445096)

[7.1.9 Use Case: Choose Game Mode-Sub-function End Goal: Start a Game 11](#_Toc526445097)

[7.1.10 Use Case: Continue an existing game-Sub-function End Goal: Start a Game 11](#_Toc526445098)

[7.1.11 Use Case: End Goal: Answer question 11](#_Toc526445099)

[7.1.12 Use Case: End Goal: Facebook share-only with Facebook approval 11](#_Toc526445100)

[7.1.13 Use Case: End Goal: Facebook Challenge/Invite 12](#_Toc526445101)

[7.1.14 Use case: End Goal: Submit Score 12](#_Toc526445102)

[7.1.15 Use Case: End Goal: Check Leader board 12](#_Toc526445103)

[7.1.16 Use Case-Submit Question 12](#_Toc526445104)

[7.1.17 Use Case-Voting 12](#_Toc526445105)

[7.1.18 Use Case-Choose a category 12](#_Toc526445106)

[7.1.19 Use Case: End Goal: Exit application 12](#_Toc526445107)

[8 Full Description for Core, Critical, Risky, Difficult (CCRD) Use Case 13](#_Toc526445108)

[8.1 Register 13](#_Toc526445109)

[8.1.1 High Level Description 13](#_Toc526445110)

[8.1.2 Event-response story 13](#_Toc526445111)

[8.1.3 Trigger 13](#_Toc526445112)

[8.1.4 Actors 13](#_Toc526445113)

[8.1.5 Stakeholders 13](#_Toc526445114)

[8.1.6 Related Use Cases 13](#_Toc526445115)

[8.1.7 Pre-conditions 13](#_Toc526445116)

[8.1.8 Post Conditions 14](#_Toc526445117)

[8.1.9 Normal Flow 14](#_Toc526445118)

[8.1.10 Alternative Flows 14](#_Toc526445119)

[8.1.11 Key Scenarios 15](#_Toc526445120)

[8.2 Login 15](#_Toc526445121)

[8.2.1 High Level Description 15](#_Toc526445122)

[8.2.2 Event-response story 15](#_Toc526445123)

[8.2.3 Trigger 15](#_Toc526445124)

[8.2.4 Actors 15](#_Toc526445125)

[8.2.5 Stakeholders 16](#_Toc526445126)

[8.2.6 Related Use Cases 16](#_Toc526445127)

[8.2.7 Pre-conditions 16](#_Toc526445128)

[8.2.8 Post Conditions 16](#_Toc526445129)

[8.2.9 Normal Flow 16](#_Toc526445130)

[8.2.10 Alternative Flows 17](#_Toc526445131)

[8.2.11 Sub Flows 18](#_Toc526445132)

[8.2.12 Key Scenarios 19](#_Toc526445133)

[8.3 Answer Question 19](#_Toc526445134)

[8.3.1 High Level Description 19](#_Toc526445135)

[8.3.2 Event-response story 19](#_Toc526445136)

[8.3.3 Trigger 19](#_Toc526445137)

[8.3.4 Actors 19](#_Toc526445138)

[8.3.5 Stakeholders 19](#_Toc526445139)

[8.3.6 Related Use Cases 20](#_Toc526445140)

[8.3.7 Pre-conditions 20](#_Toc526445141)

[8.3.8 Post-conditions 20](#_Toc526445142)

[8.3.9 Normal Flow 20](#_Toc526445143)

[8.3.10 Alternate Flows 20](#_Toc526445144)

[8.3.11 Exception Flows 21](#_Toc526445145)

[8.3.12 Key Scenarios 21](#_Toc526445146)

[8.4 Submit Score 22](#_Toc526445147)

[8.4.1 High Level Description 22](#_Toc526445148)

[8.4.2 Trigger 22](#_Toc526445149)

[8.4.3 Actors 22](#_Toc526445150)

[8.4.4 Stakeholders 22](#_Toc526445151)

[8.4.5 Related Use Cases 22](#_Toc526445152)

[8.4.6 Pre-conditions 22](#_Toc526445153)

[8.4.7 Post-Conditions 23](#_Toc526445154)

[8.4.8 Normal Flow 23](#_Toc526445155)

[8.4.9 Exception Flows 23](#_Toc526445156)

[8.4.10 Key Scenarios 23](#_Toc526445157)

[8.5 Submit User Question 24](#_Toc526445158)

[8.5.1 High Level Description 24](#_Toc526445159)

[8.5.2 Event-response story 24](#_Toc526445160)

[8.5.3 Trigger 24](#_Toc526445161)

[8.5.4 Actors 24](#_Toc526445162)

[8.5.5 Stakeholders 24](#_Toc526445163)

[8.5.6 Let’s Quiz Server 24](#_Toc526445164)

[8.5.7 Related Use Cases 24](#_Toc526445165)

[8.5.8 Pre-conditions 24](#_Toc526445166)

[8.5.9 Post Conditions 25](#_Toc526445167)

[8.5.10 Normal Flow 25](#_Toc526445168)

[8.5.11 Exception Flows 25](#_Toc526445169)

[8.5.12 Not logged in – Guest play 25](#_Toc526445170)

[8.5.13 Key Scenarios 25](#_Toc526445171)

[**8.6** **Voting** 26](#_Toc526445172)

[**8.6.1** **High Level Description** 26](#_Toc526445173)

[8.6.2 Event-response story 26](#_Toc526445174)

[**8.6.3** **Trigger** 26](#_Toc526445175)

[8.6.4 Actors 26](#_Toc526445176)

[**8.6.5** **Stakeholders** 26](#_Toc526445177)

[**8.6.6** **Let’s Quiz Server** 26](#_Toc526445178)

[**8.6.7** **Related Use Cases** 26](#_Toc526445179)

[**8.6.8** **Pre-conditions** 26](#_Toc526445180)

[**8.6.9** **Post Conditions** 26](#_Toc526445181)

[**8.6.10** **Normal Flow** 27](#_Toc526445182)

[**8.6.11** **Exception Flows** 27](#_Toc526445183)

[**8.6.12** 27](#_Toc526445184)

[**8.6.13** **Not logged in – Guest play** 27](#_Toc526445185)

[**8.6.14** **Key Scenarios** 27](#_Toc526445186)

[8.7 Choosing a Category 27](#_Toc526445187)

[**8.7.1** **High Level Description** 27](#_Toc526445188)

[8.7.2 Event-response story 28](#_Toc526445189)

[**8.7.3** **Trigger** 28](#_Toc526445190)

[8.7.4 Actors 28](#_Toc526445191)

[**8.7.5** **Stakeholders** 28](#_Toc526445192)

[**8.7.6** **Let’s Quiz Server** 28](#_Toc526445193)

[**8.7.7** **Related Use Cases** 28](#_Toc526445194)

[**8.7.8** **Pre-conditions** 28](#_Toc526445195)

[**8.7.9** **Post Conditions** 28](#_Toc526445196)

[**8.7.10** **Normal Flow** 28](#_Toc526445197)

[**8.7.11** **Exception Flows** 29](#_Toc526445198)

[**8.7.12** **Key Scenarios** 29](#_Toc526445199)

[9 Use case diagrams 29](#_Toc526445200)

[10 Activity Diagrams for Internal Use Cases 30](#_Toc526445201)

[10.1 Register 30](#_Toc526445202)

[10.2 Login 30](#_Toc526445203)

[10.3 Start a game 31](#_Toc526445204)

[10.4 Answer Question 31](#_Toc526445205)

[10.5 Play round 32](#_Toc526445206)

[10.6 Submit New Question 32](#_Toc526445207)

# Introduction

The aim of the Let’s Quiz project is to design an online, multi-player, trivia question game which asks users science fiction and fantasy genre questions. The game will to allow users to compete against each other in a timed, turn base setting. Users will also have the ability to submit their own questions, which allows the question pool to grow and improve over time. The application will allow players to register a Let’s Quiz account or login using social media accounts (see section 1.3) or to bypass the login process altogether and play as a guest. It will allow social interactions such as ‘share’. The game will allow highest score recording via global ranking and meaningful, fun game play.

We are going to create the application with the game engine Unity 3D, using C# as the primary programming language. Unity has many advantages for game development included extended support for 27 platforms. Unity has a game engine, an IDE and a user interface application all geared towards game development.

## Purpose

This Initial Requirements Model document will describe the requirements and specifications of the Let’s Quiz online trivia game. We will use this document to set the expectations for the development of this project. A requirements document is needed to guide the developers through to completion and should assist, our developers to define the intended functionality and parameters needed to develop this project.

## Scope

1. Produce an on-line single game.
2. Produce a multiplayer trivia turn based game
3. Incorporate Facebook and Google Play Services SDK’s into game (please see section1.3 for further information)
   1. Facebook login-awaiting approval
   2. Facebook sharing-awaiting approval
   3. Google Play Services Leader board-dropped
   4. Google Play Services Achievements-dropped
   5. Google Play Services Login-dropped
4. Database integration
   1. Global high score
   2. Player statistics
   3. Managing game states of open games between players
5. Finalize all implementations of game to be playable
6. Complete all tests to remove bugs

## Deemed out of Scope

Although we had originally planned for Google Play Services to be integrated, but due to compatibility reasons it has been dropped from the current project. Also Facebook integration although fully implemented and working in the test environment, needs approval from Facebook and will only be used if approval is granted prior to the 5th of October, to leave time for beta testing.

# System-Wide Functional Requirements

1. F-2-Persistence Services

i) At the end of each round the application will upload game data to an online SQL database

b) F-3-Language Services

i) Language services will be integrated as English to start off with a hope of extending in future updates to support other languages

ii) Services that allow players with a disability would be beneficial and will be implemented wherever possible

c) F-4 Networking Services

* 1. Networking services will be required for multi-player playing or to login with Facebook or Google Services, and for the majority of interactions available in the game like checking the highest score and checking leader board. Only required for Facebook if approval is granted.

# System Qualities

The non-functional requirements are known as system qualities. These are attributes of the system and describe how the software will do a required task, not what the task entails. The FURPS+ acronym sums up the requirements F-Functional, and the URPS+ meaning Usability, Reliability, Performance and Supportability are the non-functional requirements. The plus in FURPS+ is the system constraints seen below in section Four.

## External interface requirements (Non-functional requirements)

### Usability

* + - 1. The user interface should be easy to use
      2. Interface should be compatible with mobile device screens
      3. The game mechanisms should be easy to learn and navigate around.
      4. The users should be able to compete tasks in a reasonable amount of time

### Reliability

##### Availability

* + - * 1. The game should be available to players on request at least 99% of the time
        2. The application should have no more than 1 hour down time in any 2-month period

### Accessibility

* + - * 1. Once the game is installed the user should have 24/7 availability of use.

### Performance

##### Response Times

* + - * 1. The users should be able to see a response from their interactions instantly
        2. For game launch the game should be playable within 30 seconds of launching optimum 10 seconds

### Capacity

* + - * 1. The system should be capable of handling 100 users at any one time

#### Supportability

##### Compatibility

* + - * 1. The game will need to be compatible with both Android and IOS devices

##### Maintainability

* + - * 1. The Game may wish to be added to in future updates, so it will be necessary to begin with refactorable, clean code and thorough documentation

##### Documentation Requirements

* + - * 1. All required documentation will follow version control. We will supply all documentation necessary for the project.

#### Security

* + - 1. Security services are needed to authenticate users’ logins and will assist in these processes. These services will include a Let’s Quiz account which will authenticate users. Facebook and Google logins will have their logins authenticated by their systems authentication processes.

# System Constraints

Constraints are the plus in FURPS+ and include;

* 1. The game user interface will need to be designed to fit a mobile screen
  2. The application will need to be used within the limits of mobile phone power
  3. The application will be restrained by user’s phone data limits

# Assumptions and dependencies

1. That the mobile device the application is installed on will meet the minimum system requirements.
2. It is assumed the user will have the technical ability to operate a touch screen
3. That all developers have knowledge of the required IDE and other necessary aspects including Unity 3d and Facebook SKD.
4. It will be assumed that the Facebook servers are available

# Diagrams

## Domain Diagram



# Functional Requirement Use Case Descriptions

## Use Case Descriptions

### Use Case: Launching Game

When the user

Wants to start the application they click the application icon on their device

So that the application opens to show the login screen

### Use Case: Register

When the user

Wants to register a Let’s Quiz account, they must input user details and click register

So that the application creates an account for them

### Use Case: Login

When the user

Wants to login, they must choose login option

So that that the application allows login and displays pregame screen

### Use Case: Login Registered Users-Sub-function End Goal: Login

When the user

Wants to play by logging in, they then press login

So that that the application opens to the login screen

### Use Case: Login with Facebook-Sub-function End Goal: Login

When the user

Wants to play by logging in with Facebook, they then press Facebook login

So that that the application connects to the Facebook Authentication server and allows login

### Use Case: Play as Guest-Sub-function End Goal: Login

When the user

Wants to play without logging in or first registering they press play as guest

So that the application opens to the pre-game screen

### Use Case: End Goal: User Submit Question

When the user

Wants to submit a question they will press the submit question button

So that the application opens to the submit question scene

### Use Case: End Goal: Start a Game

When the user

Wants to start a new game they will press the start new game button

So that the application will either start a new game or join an existing game

### Use Case: Choose Game Mode-Sub-function End Goal: Start a Game

When the user

Wants to Choose game mode they close the game mode by pressing Corresponding mode

So that the application opens the correct game state

### Use Case: Continue an existing game-Sub-function End Goal: Start a Game

When the user

Wants to take their turn in a previously started game they press the games description

So that the application opens the correct game state

### Use Case: End Goal: Answer question

When the user

Wants to answer a question they select the correct answer

So that the game can check the answer for correctness

### Use Case: End Goal: Facebook share-only with Facebook approval

When the user

Wants to share game they click share on face book button/link

So that the application connects to the Facebook server and allows sharing

### Use Case: End Goal: Facebook Challenge/Invite

When the user

Wants to Challenge/Invite they click the challenge button

So that so that the application connects to the Facebook server and sends invitation

### Use case: End Goal: Submit Score

When the system

Wants to submit score, the system connects to Let’s Quiz Server

So the application can send score data to data base for updating

### Use Case: End Goal: Check Leader board

When the user

Wants to check the leader board scores they press the leader board button

So that the application connects to the Google Play Services server to display the leader board

### Use Case-Submit Question

When the user

Wants to add a question to the Let’s Quiz question pool

They select Submit Question from the Main Menu

So that the application will open the Submit Question Scene

### Use Case-Voting

When the user

Wants to vote on a question to the Let’s Quiz question pool

They select a like/dislike button from the question page

So that the application will add or subtract their vote.

### Use Case-Choose a category

When the user

Wants to choose a category

They select a category from the dropdown menu

So that the application will allocate a category.

### Use Case: End Goal: Exit application

When the user

Wants to exit the application they press the back button on android

So that the application closes down

# Full Description for Core, Critical, Risky, Difficult (CCRD) Use Case

## Register

### High Level Description

When the user

Wants to register a Let’s Quiz account, they must input user details and click register

So that the application creates an account for them

### Event-response story

When the user clicks register it will open the register panel and ask for username, email and password the user inputs these, then clicks register which should cause the application to connect to database and register an account otherwise the user can choose to skip registration and will be logged in as a guest

### Trigger

When a user clicks registration

### Actors

#### User

The user is the person who clicks the register button.

#### Let’s Quiz Server

The application will save account information to the server’s database for the user’s future use to login

### Stakeholders

#### User

The user expects the application to register their account details for future use by the application

### Related Use Cases

Login

Play as Guest

### Pre-conditions

Game must be installed

The user has started the application

There must be an open internet connection

### Post Conditions

#### Minimal guarantee

The user will get an error response explaining they cannot register at present and be asked if they wish to play as a guest

#### Success guarantee

The user will be registered and have an account to log into for future game play

### Normal Flow

The use case begins when the user presses register to make a Let’s quiz account

|  |  |
| --- | --- |
| Actor | System |
| 1. User inputs username, email and password |  |
| 1. The user presses register button | 1. System connect to Let’s Quiz server to create a let’s Quiz account. 2. System registers user and displays login screen |

Use case ends

### Alternative Flows

#### User already registered

If at step 1 if user is already registered

|  |  |
| --- | --- |
| Actor | System |
| 1. User navigates to registration screen. |  |
|  | 2. System will show message to user that the credentials are already used please enter other credentials or login |

#### User decides not to register- Close Game

At 1 user decides not to register- close game

|  |  |
| --- | --- |
| Actor | System |
| 1. User closes game |  |

#### User decides not to register- Skip Registration

At 1 user decides not to register but to skip registration

|  |  |
| --- | --- |
| Actor |  |
| 1. User presses skip button |  |
|  | 2. System creates guest account |
|  | 3. System opens choose game screen |

#### Exception Flows

#### No connectivity to the Let's Quiz Server

If at step 3 the normal flow the app cannot connect to the Let’s Quiz Server

|  |  |
| --- | --- |
| Actors | System |
|  | 1. System cannot connect to lets quiz server to create account 2. Application checks for server connection 3. A popup message alerts the user to the server error 4. The user can play using a guest account |

### Key Scenarios

User wants to register a Let’s Quiz account

User wants to play as guest

## Login

### High Level Description

When the user

Wants to login, they must choose a login option

So that that the application allows login and displays pregame screen

### Event-response story

When user decides on which login method to use, they press the corresponding button, which then allows login by either guest-pressing skip, Let’s Quiz account, Facebook or Google Play Services. Once the user logs in the application, it will show the pregame screen

### Trigger

The user chooses a login method

### Actors

#### User

The user chooses and presses preferred login button

#### Let’s Quiz Server

The application will ask the server for the user’s data relating to their login. The user’s data will be compared with user’s imported data.

#### Facebook Server- only with Facebook approval

The application will connect to the Facebook Servers and log the user in via their social media account

### Stakeholders

#### User

The user expects the application to log them in, via their choice of login method.

### Related Use Cases

Register

Login with Facebook

Login with Google Play Services

Play as Guest

### Pre-conditions

Game must be installed

The user has started the application

There must be an open internet connection

### Post Conditions

#### Minimal guarantee

The user will get an error response explaining they cannot log in at present and be asked if they wish to play as a guest

#### Success guarantee

The user will be logged in via the method they chose and continue onto the pregame screen for further game play.

### Normal Flow

The use case begins when the user presses and choses to login with an existing Let’s Quiz account

|  |  |
| --- | --- |
| Actor | System |
| 1. User inputs username and password |  |
| 1. The user presses chosen login button | 1. System connect to Let’s Quiz server to verify user login details. 2. User is logged in and pregame screen is displayed |

The use case ends

### Alternative Flows

#### User already logged in

If at step 1 if user is already logged in

|  |  |
| --- | --- |
| Actor | System |
| 1 User navigates to login screen. |  |
|  | 2. System will show user as logged and show menu screen |
|  |  |

#### User decides not to login- Close Game

At 1 user decides not to login

|  |  |
| --- | --- |
| Actor | System |
| 1. User closes game |  |

#### User decides not to login- Skip Login

At 1 user decides not to login but to skip login

|  |  |
| --- | --- |
| Actor | System |
| 1. User presses skip button |  |
|  | 2. System creates guest account |
|  | 3. System opens choose game screen |

#### Exception Flows

#### Cannot verify login details-Wrong Username

If at step 3 verification cannot be performed

|  |  |
| --- | --- |
| Actor | System |
|  | 1. System connects to lets quiz server but cannot verify details |
|  | 1. System displays message username provided is incorrect |
| 1. The user inputs corrects username | 1. User is logged in and pregame screen is displayed |

#### Cannot verify login details-Wrong Password

If at step 3 verification cannot be performed

|  |  |
| --- | --- |
| Actor | System |
|  | 1. System connects to lets quiz server but cannot verify details |
|  | 1. System displays message password provided is incorrect |
| 1. The user inputs correct password | 1. User is logged in and pregame screen is displayed |

#### Cannot verify login details-No details

If at step 3 verification cannot be performed

|  |  |
| --- | --- |
| Actor | System |
|  | 1. System displays a message username cannot be empty |
| 1. The user inputs username | 1. System displays a message password cannot be empty |
| 1. The user inputs password | 1. User is logged in and pregame screen is displayed |

#### No connectivity to the Let's Quiz Server – Registered User

If at step 3 the normal flow the app cannot connect to the Let’s Quiz Server

|  |  |
| --- | --- |
| Actors | System |
| 1. The user inputs login details. |  |
|  | 1. System cannot connect to lets quiz server to verify user details 2. Application checks for server connection 3. A popup message alerts the user to the server error 4. The application exits to the main menu 5. The user can play using a guest account |

### Sub Flows

#### Facebook Login

At 1 User choses Facebook Login

|  |  |
| --- | --- |
| Actor | System |
| 1. The user presses chosen Facebook login button | 1. System displays Facebook popup login |
| 1. The user inputs Facebook email/username and password | 1. System connect to Facebook server to verify user login details. 2. User is logged in and pregame screen is displayed |

Play as guest

At 1 user decides not to login

|  |  |
| --- | --- |
| Actor | System |
| 1. The user decides to play as guest |  |
| 1. The user presses skip sign in | 1. System assigns user name ‘guest’ to player 2. User is logged in as guest and pregame screen is displayed |

### Key Scenarios

User wants to login

User wants to login using Facebook

User wants to login using Google Play Services

User wants to play as guest

User gets logged in

## Answer Question

### High Level Description

When the user

Wants to answer a question they select the correct answer

So that the game can check the answer for correctness

### Event-response story

When the user is ready to answer a question, there must be a question displayed, either from a new or existing game and the round timer will start. They then select the answer they believe to be right which makes the program supply a new question to be answered. This repeats until timer ends, at which time correct answers are displayed in green and incorrect in red, as well as current score.

### Trigger

The user has started a game.

### Actors

#### User

The user answers questions to complete the round.

#### Let’s Quiz Server-

The application will ask the server for the question/answer data

### Stakeholders

#### User

The user requires the application to supply questions promptly and correctly.

#### Opponent

Another player, whether they have joined a game yet or not. An opponent is needed to compete against the user. How the user answers each question will affect the score the opponent needs to beat.

### Related Use Cases

Submit score

Check Leader board

### Pre-conditions

An Internet connection must be established so a connection with the server can pull the updated question list on game start up

Application must be installed on mobile device

A new or pre-existing game must be started

The application must be working.

### Post-conditions

#### Minimal guarantee

Application will give an error to the user explaining existing problem or in the case of an unrecoverable error the application will safely terminate, and the user can restart it.

#### Success guarantee

The application continues to supply questions till round timer ends.

### Normal Flow

The use case begins when user gets asked a question

|  |  |
| --- | --- |
| Actor | System |
| 4. The user selects desired answer | 1. The application will display a question   1. Application starts timer   5. Score is added if correct   1. Application displays next question 2. Application ends timer 3. Application loads correct and incorrect answers 4. Application calculates and shows score |

The use case ends.

### Alternate Flows

#### User decides not to answer question

At 4 user decides not to answer question

|  |  |
| --- | --- |
| Actor | System |
| 4.1. User does not answer question |  |
|  | 4.2 Timer runs out |
|  | 4.3 No score given |

#### User decides not to answer- shut game

At 1 user decides to close game

|  |  |
| --- | --- |
| Actor | System |
| 1.1. User closes game |  |
|  | 1.2. System closes game |
|  |  |

### Exception Flows

#### No connectivity to the Let's Quiz Server

If at step 1 of the normal flow no question is shown

|  |  |
| --- | --- |
| Actor | System |
|  | 1.1 Question not displayed by application |
|  | 1.2 Application checks for server connection |
|  | 1.3 A popup message alerts the user to the server error |
|  | 1.4 The application exits to the main menu |

#### The user selects an incorrect answer

If at step 6 of the normal flow the user supplies incorrect answer

|  |  |
| --- | --- |
| Actor | System |
|  | 5.1 The incorrect answer is highlighted in red. |
|  | 5.2 The correct answer is highlighted in green. |
|  | 5.3 Three points are subtracted from the users score |
|  | Normal Flow will continue from step 6 |

### Key Scenarios

User wants to answer question

Application retrieves an answer

User is given a score

Asked question is removed from applicable questions

Another question is randomly picked

Round timer ends

Scores are tallied and shown

Application ends round

Game data is uploaded to server.

## Submit Score

### High Level Description

When the system

Wants to submit score, the system connects to Let’s Quiz Server

So the application can send score data to data base for updating

#### Event-response story

When a user finishes a round it will make the application submit the users score where it will be compared with the opponents score so the application can total the scores and will then calculate a winner.

### Trigger

User finishes their round in Let’s Quiz

### Actors

#### User

The person playing the round who’s score is submitted for calculation

#### Opponents

The other person playing, to which the scores are compared to come up with a winner

### Stakeholders

#### User

The player wishing to play Let’s Quiz and get a score for each round to clarify a winner.

#### Game Opponent

The other player in the game, whom the user competes against to validate a winner by calculating scores.

#### Let’s Quiz Server

The device needs to submit a score to the Let’s Quiz server to be calculated and stored till the end of the game so that a winner can be determined

### Related Use Cases

Answer question

### Pre-conditions

An internet connection will be needed to allow the application to connect to the Let’s Quiz server to submit the score

A player must have completes at least one round

### Post-Conditions

#### Minimal Guarantee

An error message will be displayed asking user to rectify error and will save a local copy of score till this is done.

#### Success Guarantee

The application will submit the score of the user to the Let’s Quiz server at the end of each of the three rounds and will calculate and display both scores at end of the game to pronounce a winner.

### Normal Flow

The use case starts when a player finishes answering a round of questions game screen

|  |  |
| --- | --- |
| Actor | System |
| 1. The user answers questions for present round | 1. The application timer runs out and finishes each round 2. The application connects to server to store the present rounds score |
| 1. User finishes game of three rounds | 1. Total scores are then submitted to server and displayed to show a winner |

This use case ends.

### Exception Flows

#### No connectivity to the Let's Quiz Server-Submit Round Score

If at step 3 of the normal flow the app cannot connect to the Let’s Quiz Server

|  |  |
| --- | --- |
| Actor | System |
|  | 1. A popup message alerts the user to the error and saves the score locally until connection occurs |

#### No connectivity to the Let's Quiz Server-Submit Total Score

If at step 5 of the normal flow the app cannot connect to the Let’s Quiz Server

|  |  |
| --- | --- |
| Actor | System |
|  | 1. A popup message alerts the user to the error and asks them to rectify problem |

### Key Scenarios

User finishes a round

The application submits round scores

The application submits total score

The application displays winner score

The application declares a winner

## Submit User Question

### High Level Description

When the user

Wants to add a question to the Let’s Quiz question pool

They select Submit Question from the Main Menu

So that the application will open the Submit Question Scene

### Event-response story

When the user selects Submit Question

It causes the application to load the Submit Question scene

The user then adds data in the order asked

So that the serialised data can then be sent to the Let’s Quiz server to be added to the Questions SQL table

### Trigger

The user presses Submit Question from the Main Menu

### Actors

#### User

The user presses Submit Question and then follows the prompts to add their question to the server

### Stakeholders

#### User

The user is trying to add their own question to the question pool for future Let’s Quiz games

### Let’s Quiz Server

The device needs to be able to access the server to add the question to the Question table

#### All Let’s Quiz players

Due to the communal nature of all the questions every user is a stakeholder in the quality of the questions being submitted.

### Related Use Cases

Launch Game

### Pre-conditions

The user has started the application

The user has logged in but not as guest

### Post Conditions

#### Minimal guarantee

Give an error message to the user so they can rectify the problem, or upon an unrecoverable error the application will safely terminate and the user can restart it.

#### Success guarantee

The user enters a new question that is submitted to the Question Table on the Let’s Quiz server.

### Normal Flow

The use case starts when a player chooses to submit a question to the question pool

|  |  |
| --- | --- |
| Actor | System |
| 1. The user presses submit new question button | 1. Application opens the submit question |
| 3. User presses submit question | 4.The application connects to server to store the users question |

This use case ends.

### Exception Flows

#### No connectivity to the Let's Quiz Server- Submit New Question

If at step 4 of the normal flow the app cannot connect to the Let’s Quiz Server

|  |  |
| --- | --- |
| Actor | System |
|  | 4.1. A popup message alerts the user to the error asking them to fix it |

### Not logged in – Guest play

If at step 1 of the normal flow user is logged in under guest account

|  |  |
| --- | --- |
| Actor | System |
|  | 1.1. System displays error guests cannot submit new questions |

### Key Scenarios

User wants to submit new question

User adds new question and answers

New questions saved to Let’s Quiz server

* 1. **Voting**
     1. **High Level Description**

When the user

Wants to vote on a question to the Let’s Quiz question pool

They select a like/dislike button from the question page

So that the application will add or subtract their vote.

### Event-response story

When the user selects like/dislike

It causes the application to add or subtract a point from the questions vote total

So that the serialised data can then be sent to the Let’s Quiz server to be added to the Questions SQL table

* + 1. **Trigger**

The user presses the like/dislike buttons below the question/answer panel

### Actors

* + - 1. ***User***

The user presses like/dislike and then follows the prompts to add their question vote to the server

* + 1. **Stakeholders** 
       1. ***User***

The user is trying to add their vote to the question for future Let’s Quiz games

* + 1. **Let’s Quiz Server**

The device needs to be able to access the server to add the vote to the Question table

* + - 1. ***All Let’s Quiz players***

Due to the communal nature of all the questions every user is a stakeholder in the quality of the questions being voted on.

* + 1. **Related Use Cases**

Play Game

* + 1. **Pre-conditions**

The user has started the application

The user has logged in but not as guest

The question must be open on the screen

* + 1. **Post Conditions**
       1. ***Minimal guarantee***

Give an error message to the user so they can retry, or upon an unrecoverable error the application will safely terminate and the user can restart it.

* + - 1. ***Success guarantee***

The user enters a vote of like or dislike for a question that is submitted to the Question Table on the Let’s Quiz server.

* + 1. **Normal Flow**

The use case starts when a player chooses to vote on a question

|  |  |
| --- | --- |
| Actor | System |
| 1. The user presses like or dislike button | 2. The application connects to server to store the users vote |

This use case ends.

* + 1. **Exception Flows**
       1. ***No connectivity to the Let's Quiz Server- Submit New Question***

If at step 4 of the normal flow the app cannot connect to the Let’s Quiz Server

|  |  |
| --- | --- |
| Actor | System |
|  | 4.1. A popup message alerts the user to the error asking them to fix it |

* + 2. **Not logged in – Guest play**

If at step 1 of the normal flow user is logged in under guest account

|  |  |
| --- | --- |
| Actor | System |
|  | 1.1. System displays error guests cannot vote on a question |

* + 1. **Key Scenarios**

User wants to vote on a question

User’s vote is saved to Let’s Quiz server

## Choosing a Category

* + 1. **High Level Description**

When the user

Wants to choose a category

They select a category from the dropdown menu

So that the application will allocate a category.

### Event-response story

When the user selects a category

It causes the application to load questions from that category

So that the user is asked only questions from their selected topic

* + 1. **Trigger**

The user selects a category

### Actors

* + - 1. ***User***

The user chooses then selects a category of their liking

* + 1. **Stakeholders** 
       1. ***User***

The user is trying to choose a category for their next Let’s Quiz game

* + 1. **Let’s Quiz Server**

The device needs to be able to access the server to store the chosen category

* + - 1. ***All Let’s Quiz players***

Due to the communal nature of all the categories every user is a stakeholder in that the categories are across all games

* + 1. **Related Use Cases**

Launch Game

* + 1. **Pre-conditions**

The user has started the application

The new game screen must be open

* + 1. **Post Conditions**
       1. ***Minimal guarantee***

Give an error message to the user so they can retry, or upon an unrecoverable error the application will safely terminate and the user can restart it.

* + - 1. ***Success guarantee***

The user should be able to choose a category for that game and that category is saved to the Let’s Quiz server.

* + 1. **Normal Flow**

The use case starts when a player starts a new game

|  |  |
| --- | --- |
| Actor | System |
| 1. The user presses category chooses | 2. The application opens a drop down menu with category chooses |
| 3. The user chooses their category | 4. The system stores the new category |

This use case ends.

* + 1. **Exception Flows**
       1. ***No connectivity to the Let's Quiz Server- Submit New Question***

If at step 4 of the normal flow the app cannot connect to the Let’s Quiz Server

|  |  |
| --- | --- |
| Actor | System |
|  | 4.1. A popup message alerts the user to the error asking them to fix it |

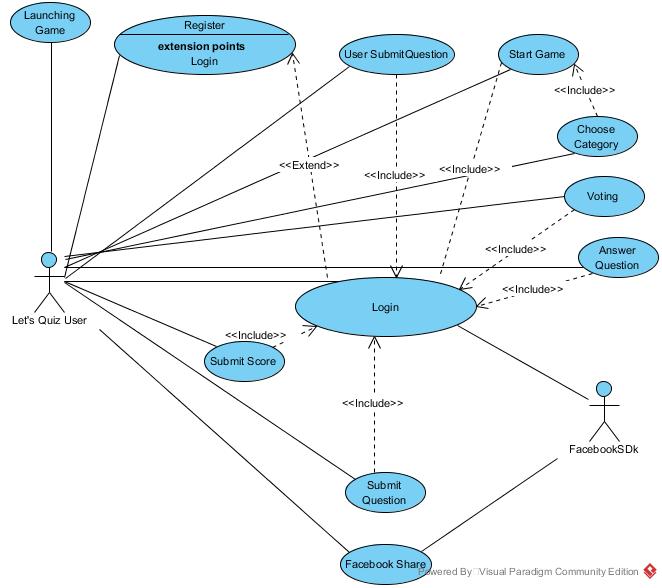
* + 1. **Key Scenarios**

User can choose a category

User’s category is saved to Let’s Quiz server

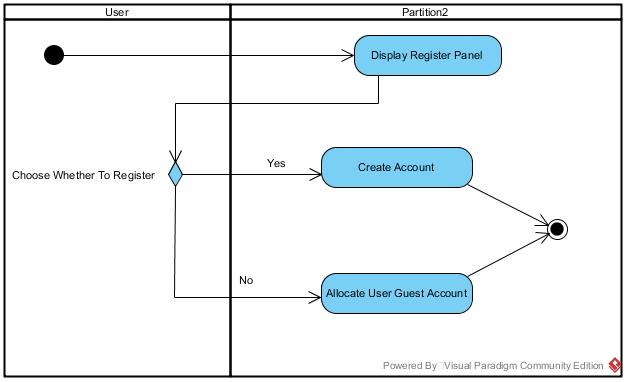
# Use case diagrams

Full Use Case

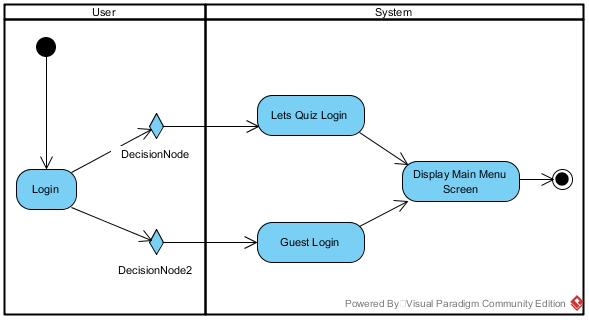


# Activity Diagrams for Internal Use Cases

## Register



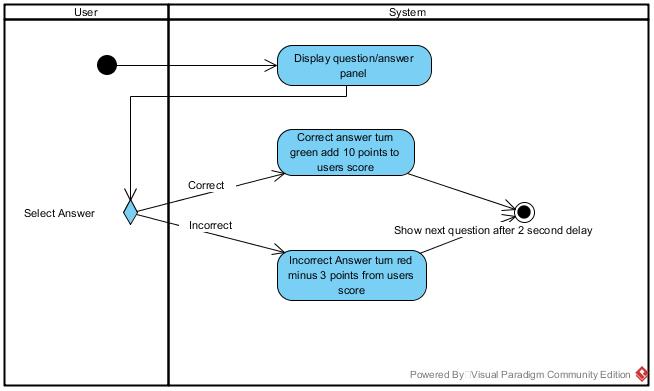
## Login



## Start a game



## Answer Question



## Play round



## Submit New Question

