



Bilkent University

Department of Computer Engineering

CS319 - OBJECT ORIENTED SOFTWARE ENGINEERING

Q-bitz

Iteration 2 - Analysis Report

Sait Aktürk, Zafer Tan Çankırı, Berkin İnan, Halil Şahiner, Abdullah Talayhan

Supervisor: Eray Tüzün

Progress / Final Report
November 27, 2018

Table of Contents

1. Introduction	3
2. Overview	4
2.1 Game Modes	4
2.2 In-Game Objects.....	6
2.3 Information and Control.....	6
3. Functional Requirements.....	7
3.1 Additional Requirements	7
3.2 User related requirements.....	7
3.3 Navigation.....	7
3.4 Game Functionalities	8
4. Nonfunctional Requirements	9
4.1 Additional Requirements	9
4.2 Usability	9
4.3 Reliability	9
4.4 Performance	9
4.5 Extendibility.....	9
5. System Models.....	10
5.1 Use case model	10
5.2 Dynamics models.....	14
5.3 Object and class model.....	21
5.4 User interface	25
6. Improvement Summary	31

Analysis Report

Q-bitz

1. Introduction

In Q-bitz players use their special cubes to recreate patterns on the cards, gain points and win the game. Although the original game has three rounds which will be played in order to win a game, our game will not be round based but mode-based style which will consists of those rounds. We chose Q-bitz because it has the potential to be extended with new game modes and features that we will add.

In our game we have:

- Online Q-bitz
 - Multiplayer Game Modes
- Offline Q-bitz
 - Practice
 - Single Player Game
- Options
 - Adjustable color set
 - Sound options
 - Controls
- How to play

With this structure, the game will use the benefits of being digital, meaning that a user can choose multiplayer mode to enjoy the game with friends while in single player mode users can have a taste of an arcade game without the need of any other user.

This report will give an overview for the game by describing functional and non-functional requirements for developing the game successfully. The features of the game and how these features work. It will explain how the user will interact with the game using use case diagrams. Then It will explain the flow of the game by providing sequence, activity and state diagrams and will show the object structure using the class diagram. Finally, it will contain the user interface mockups of the screens that the game has for each scenario.

2. Overview

Q-bitz has both online and offline game mode. In online mode, we have multiple sub modes for online play. In the multiplayer modes, if a player successfully recreates the pattern, the player earns 10 points and a countdown starts for other players. Players who finish by the time countdown ends, earn points but fewer than the previously finished players.

Point spread system will be like this:

- 1st : 10
- 2nd : 08
- 3rd : 06
- 4th : 05
- 5th : 04
- 6th : 03
- 7th : 02
- 8th : 01

Each game mode is played 3 times and the player with the highest points in the end is the winner.

2.1 Game Modes

2.1.1 Single Player Mode

In single player mode, there will be three different modes such as, time attack, roll to win and memory.

2.1.1.1 Time Attack:

In Time Attack mode, the player will try to recreate a given pattern before the countdown ends. Time on the countdown will change depending on the difficulty of the pattern.

2.1.1.2 Roll to Win:

In Roll to Win mode, all cubes are rolled and players try to recreate the pattern on the card without rotating the cubes. When the player decides the cubes cannot be used for recreating the pattern more, the cubes can be rolled again without a limitation of number of rolling.

2.1.1.3 Memory:

In Memory mode, the pattern card is only revealed for a certain time and then players try to recreate the pattern from their memory. Each player can submit their board only 3 times to check if they are finished. After 3 failed attempts, they lose the round and earn 0 points.

2.1.2 Multiplayer Mode

In multiplayer mode, there will be five different modes such as, race, roll to win, memory, elimination.

2.1.2.1 Race:

In Race mode, up to 8 players will race against each other to recreate the same pattern revealed on the pattern card.

2.1.2.2 Roll to Win:

In Roll to Win mode, all cubes are rolled and players try to recreate the pattern on the card without rotating the cubes. When the player decides the cubes cannot be used for recreating the pattern more, the cubes can be rolled again without a limitation of number of rolling.

2.1.2.3 Memory:

In Memory mode, the pattern card is only revealed for a certain time and then players try to recreate the pattern from their memory. Each player can submit their board only 3 times to check if they are finished. After 3 failed attempts, they lose the round and earn 0 points.

2.1.2.4 Elimination:

Since the game is digital and we can have many boards instead of four in the original one. Up to 8 players can join to the same game room and start a tournament. In each game, the player that finishes the pattern last will be eliminated.

2.2 In-Game Objects

2.2.1 Board:

Players place their cubes onto the game board. Board size can change depending on the difficulty, such as 3x3 for easy, 4x4 for normal and 5x5 for hard mode.

2.2.2 Cubes:

Cubes have 6 faces with different designs on them. Players try to recreate the pattern on the card by rotating the cubes and placing them on the board.

2.2.3 Pattern:

Cards have patterns on them for players to recreate. We have two pattern generators with different difficulty settings. First pattern generator will generate a 2, 4 or 6 cell patterns depending on the grid size, and repeat or transpose that 2, 4 or 6 cell patterns to create a 3x3, 4x4 or 5x5 pattern. This pattern generator will be used on memory mode to create a pattern that is easier to memorize. The other pattern generator will generate a truly random pattern by assigning each grid a random pattern. These patterns will be more difficult than the patterns generated by the first pattern generator.

2.2.3 Game information:

Remaining time, pattern card and other players' progresses will be displayed on the game screen.

2.3 Information and Control

2.3.1 Options:

Players will be able to customize their controls, key bindings, color set and adjust volume in the options.

2.3.2 Controls:

Cubes have 6 faces with different designs on them. Players try to recreate the pattern on the card by rotating the cubes and placing them on the board.

2.3.3 Progress and User Level:

After completing a multiplayer or single player game, user will gain experience and after gaining enough experience, user's level will increase by one.

3. Functional Requirements

3.1 Additional Requirements

- Accounts of users will be verified by their e-mail address using an e-mail verification code.
- Configuration files for game options will be stored locally so that they cannot be changed or altered remotely.
- Room configurations for a multiplayer game will be stored in the server.

3.2 User related requirements

After launching the game, user will be directed to the login screen. On this screen, user can login to their accounts with username and password. They can also create an account if they do not have one yet or request to reset their password if they have forgotten it.

3.3 Navigation

The navigation through the game will be done by a main menu which consists of the following.

3.3.1 User Menu:

This menu is a drop-down menu which will appear after selecting user symbol to navigate through the user related functionality such as user settings, go online/offline and log out from the account.

3.3.2 Play Multiplayer Button:

The players can play the Q-bitz online through that screen. If the user is offline, the multiplayer menu will be greyed out.

3.3.3 Play Single Player Button:

The player can play the Q-bitz offline through that screen. There will be many different levels with different modes and difficulty settings player can choose from.

3.3.4 How to Play Button:

This button will open How to Play Menu. Tutorial will be in the form of an illustration. Illustrations will explain the race, roll to win, memory and elimination mode.

3.3.5 Options Button:

This button will open the menu for game controls, sound and music options and also color option related to cube color.

3.4 Game Functionalities

3.4.1 Play Game (Multiplayer):

Q-bitz is a multiplayer game and to play with other people, the player can create a room or join an existing room from the list on the screen.

3.4.1.1 Create Room:

Users can create a room for others to join and play with them. Users can create rooms with different settings, such as different game modes, difficulty, number of players and visibility of the game room. Rooms can be set to public or private, changing its visibility on the room list. Rooms can have up to 8 players.

3.4.1.2 Join Room:

Users can join an existing room from the room list on the screen. Additionally, the user can join rooms with using “room code” that helps the users to share same room sharing this code with its codes.

3.4.2 Play Game (Single player-Arcade):

Users can play the game offline by themselves in the single player mode. In single player mode there will many stages user can play any time. Each level will have rewards depending how quickly the player finishes it. The difficulties of levels will increase through to each level with adjusting the pattern card size, the pattern difficulty, memory mode and using time limitation.

3.4.3 Options:

User will be able to change several options for the game:

- **Music:** Users can change the level of the game music playing in background.
- **Effects:** Users can change the level of the effect sounds inside the game.
- **Cube Color:** Users can change the cube colors that will form the patterns.
- **Background Color:** Users can change the background color inside the game.
- **Rotation Controls:** Users can change the bindings of the keys dedicated for rotating the cube.
- **Navigation Controls:** Users can dedicate different mouse buttons for navigating a cube along the board and selecting a cube.

3.4.4 How to Play:

Users will be able to learn how to play the game using a brief and simple tutorial provided inside the game.

4. Nonfunctional Requirements

4.1 Additional Requirements

- After playing the game physically we decided that the time between selecting a cube and placing on the board should be fast enough so that the users do not lose time while placing a cube.
- Passwords of users for multiplayer games will not be stored as plaintext, they will be hashed using SHA256 for security reasons.

4.2 Usability

Q-bitz is a user-friendly game that will be easily playable. Easily playable means that it will not take much effort to understand how to play the game. The rules will be simple and the controls of the game will be intuitive. The aim of the project is to preserve this simplicity by providing an easy to use playing interface such that any kind of player will have a comfortable game experience.

4.3 Reliability

Since the game has multiplayer support, the game server should be able to handle the number of users such that there is no lag due to the server and prevent disconnections, and the maximum number of players that the server can handle depends on the hardware specifications of the cloud machine which server runs on. There is no specific limitation for user numbers, since the server program hold different threads for the users, and the performance and reliability of this multithreading design depends on the hardware specifications of the server machine.

4.4 Performance

Cube rotation mechanics should be implemented such that the gameplay is smooth and there is no frame loss while intensive movement. The game should run in at least 30 frames per second.

4.5 Extendibility

The architecture of the software should be designed such that new features can be added in a convenient way. For example, new game modes can be added easily since the core mechanics for rotating the cubes and pattern matching will be modular and can be used when needed.

5. System Models

5.1 Use case model



Figure 1: Use Case Model for Q-bitz Game

5.1.1 Use Case #1

Use Case: Select Cube Face

Primary Actor: User

Stakeholders and interests:

- User who wants to select a cube face for recreating a pattern.

Pre-conditions:

- User must be in a game instance.

Post-conditions:

- No post conditions

Event flow:

1. User left double clicks a cube face from the 3D cube displayed inside the game.

5.1.2 Use Case #2

Use Case: Rotate Cube

Primary Actor: User

Stakeholders and interests:

- User who wants to rotate the cube to select a cube face.

Pre-conditions:

- User must be in game instance.

Post-conditions:

- No post conditions

Event Flow for Rotate Cube:

1. User single left clicks a point on 3D cube display.
2. User drags mouse while clicking on display to rotate.

5.1.3 Use Case #3

Use Case: Place Cube Face

Primary Actor: User

Stakeholders and interests:

- User who wants to place the cube to recreate the pattern.

Pre-conditions:

- User must be in game instance.
- User must select a face on cube.

Post-conditions:

- No post conditions

Event Flow for Place Cube:

1. User enters its mouse on board to select a grid cell on it for placing the face.
2. User selects a grid cell on board by a left click on it to place it.

Event Flow for Place and Rotate a Cube:

1. User enters its mouse on board to select a grid cell on it for placing the face.
2. User selects a grid cell on board by a left click on it to place it.
3. User clicks right button to rotate cube face which is placed on board to recreate the pattern on that grid cell.

5.1.4 Use Case #4

Use Case: Submit the Pattern on Board

Primary Actor: User

Stakeholders and interests:

- User who wants to place the cube to recreate the pattern.

Pre-conditions:

- User must be in game instance.
- User must recreate pattern on board completely.

Post-conditions:

- No post conditions

Event Flow for Submit Pattern on Board:

1. User left clicks the submit button on game instance screen.
2. User sees the post-game scene and return stage/room menu.

Alternative Flow for Submit Pattern on Board:

1. User left clicks the submit button on game instance screen.
2. User stays at the game instance scene until board matches with pattern.

5.1.5 Use Case #5

Use Case: Join Multiplayer Q-bitz Room

Primary Actor: User

Stakeholders and interest:

- User that joins a multiplayer Q-bitz room

Pre-conditions:

- User must be logged in to the system and have an internet connection.

Event flow for joining a public room:

1. User lists the current rooms available.
2. User clicks the desired room to join
3. User enters the public room

Event flow for joining a private room:

1. User lists the current rooms available.
2. User clicks the desired private room.
3. User enters the password associated with the private room.
4. User enters the private room

5.1.6 Use Case #6

Use Case: Create Multiplayer Q-bitz Room

Primary Actor: User

Stakeholders and interest:

- User that creates a multiplayer Q-bitz room

Pre-conditions:

- User must be logged in to the system and have an internet connection.

Event flow for creating a public room:

1. User clicks the create room button.
2. User enters information related to the room such as name and game mode.
3. User creates the room.

Event flow for creating a private room:

1. User clicks the create room button.
2. User enters information related to the room such as name and game mode.
3. User determines the password associated with the room.
4. User creates the room.

5.2 Dynamics models

5.2.1 Sequence Diagrams

5.2.1.1 Select and Place A Cube Face

If the user wants to select a cube face from the cube, user left clicks the desired face. Then the cube face is selected in the game instance controller and cube controllers used for detecting which face of the cube is selected. Then the user hovers the mouse to the board to see the preview of the selected cube face, then the face can be rotated by pressing the right mouse button. Cube can be placed on the board by clicking on an empty grid cell.

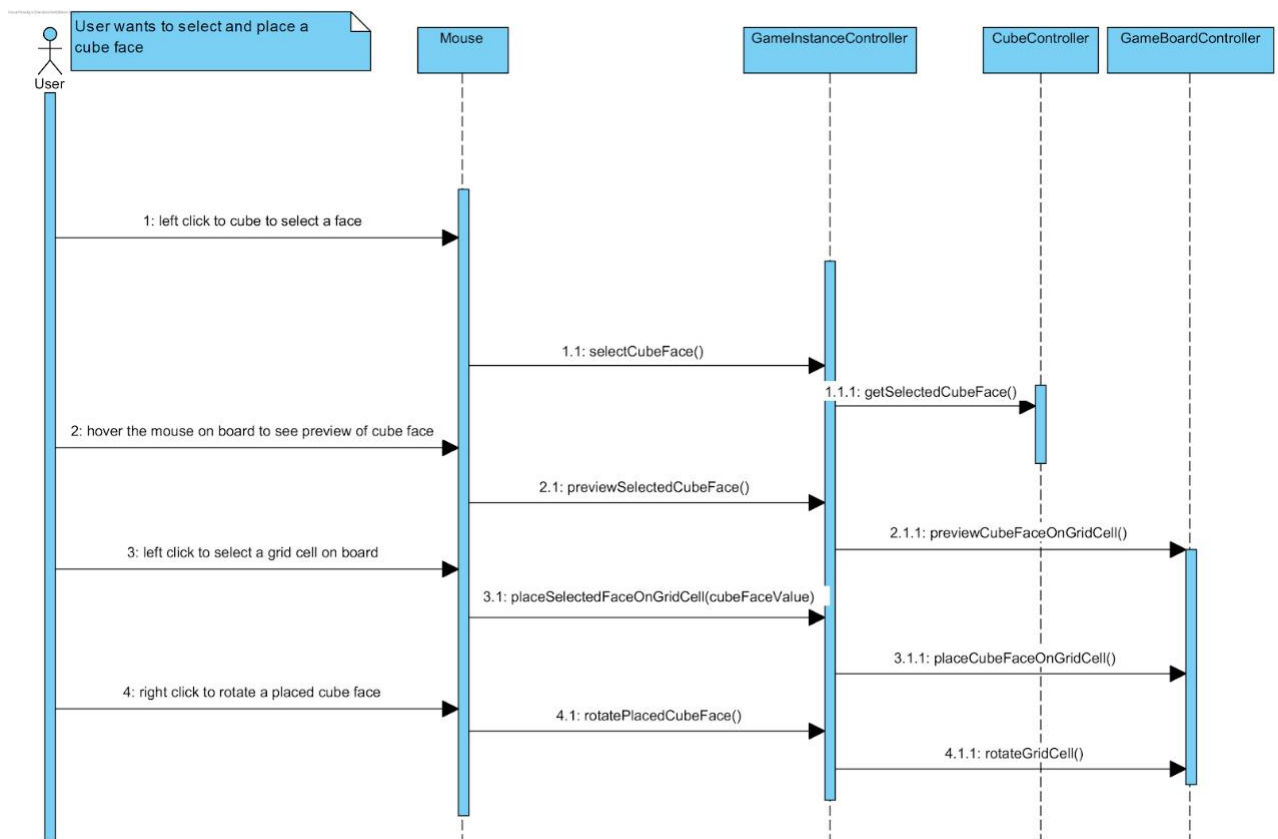


Figure 2: Sequence Diagram for Selecting and Placing a Cube Face

5.2.1.2 Rotate Cube

For rotating the cube, the user just left clicks and drags the mouse the cube is rotated in 3D space using the game instance controller. The cube stays its place after the mouse is released and can be rotated again using the same click and drag mechanics. This gives the user the ability to easily find the desired cube face on the cube.

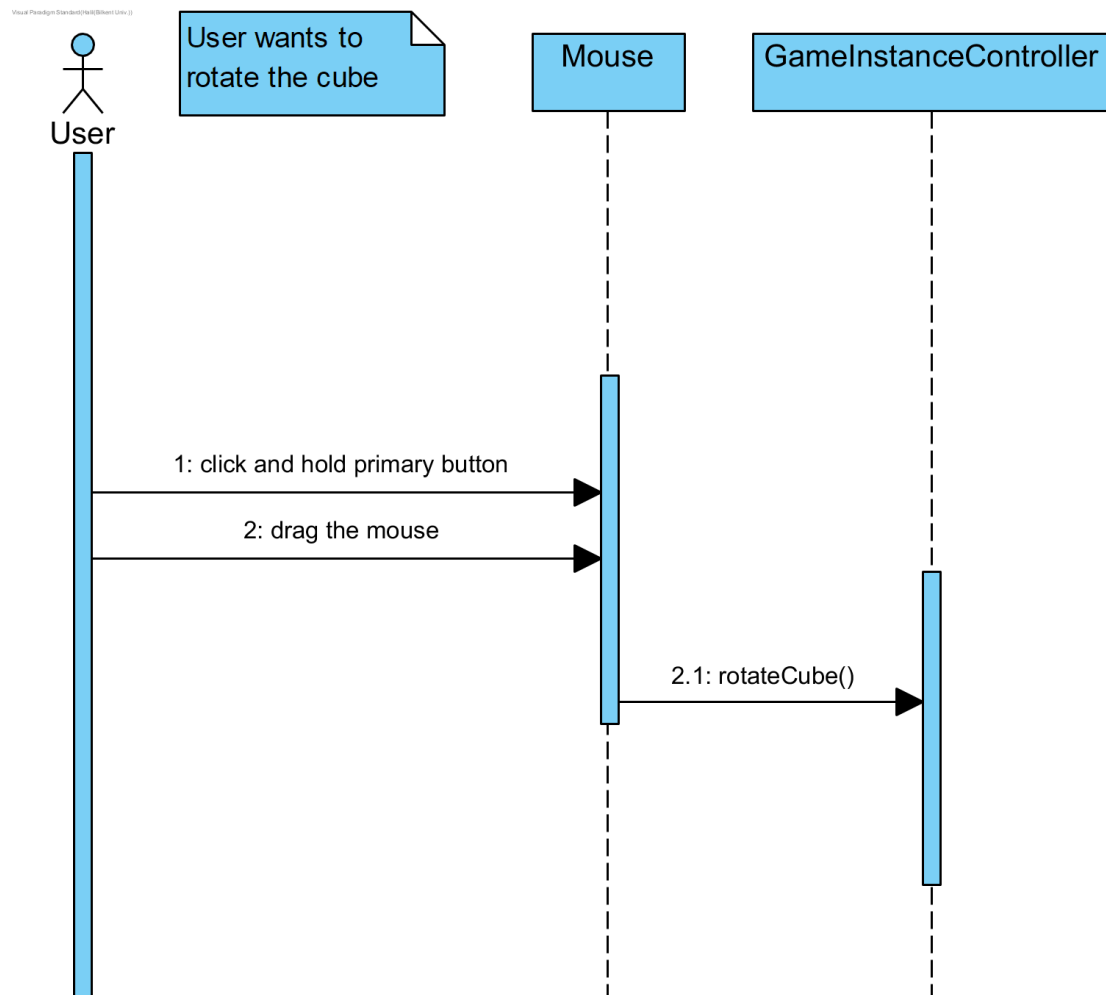


Figure 3: Sequence Diagram for Rotating Cube

5.2.1.3 Single player Game

The users select single player menu from the main menu, in single player menu the user clicks the desired game mode and board size. According to the game mode and board size, the game instance is created using game instance controller which includes creating the board, desired pattern and the cube. After the game ends, the post-game menu appears showing the elapsed time.

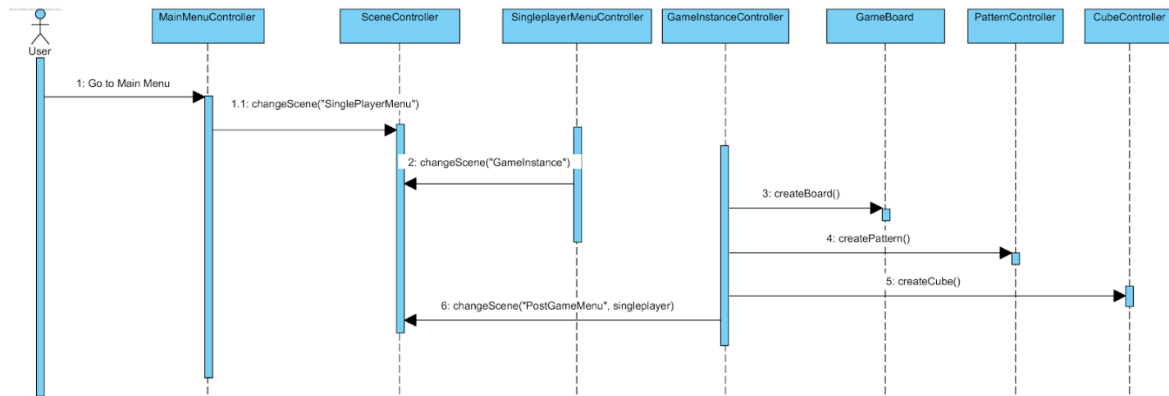


Figure 4: Sequence Diagram for Single player Game

5.2.1.4 Multiplayer Game

The user will login to the system, if the user is in multiplayer mode, the user credentials will be controlled from the server and necessary replies will be sent. After successfully logging in to the system, the user will join to a game room. The details of the room such as information related to other players will be obtained by the server. Then the user joins the game and the game procedure will be the same as the single player one. The game will start with several communications to the Game Instance. First the game mode will be selected. The grid size will be initialized and the pattern will be initialized which is automatically generated by the game. Then the cubes will be created and the timer will start along with the game itself. After the game finishes, the game winner will be detected.

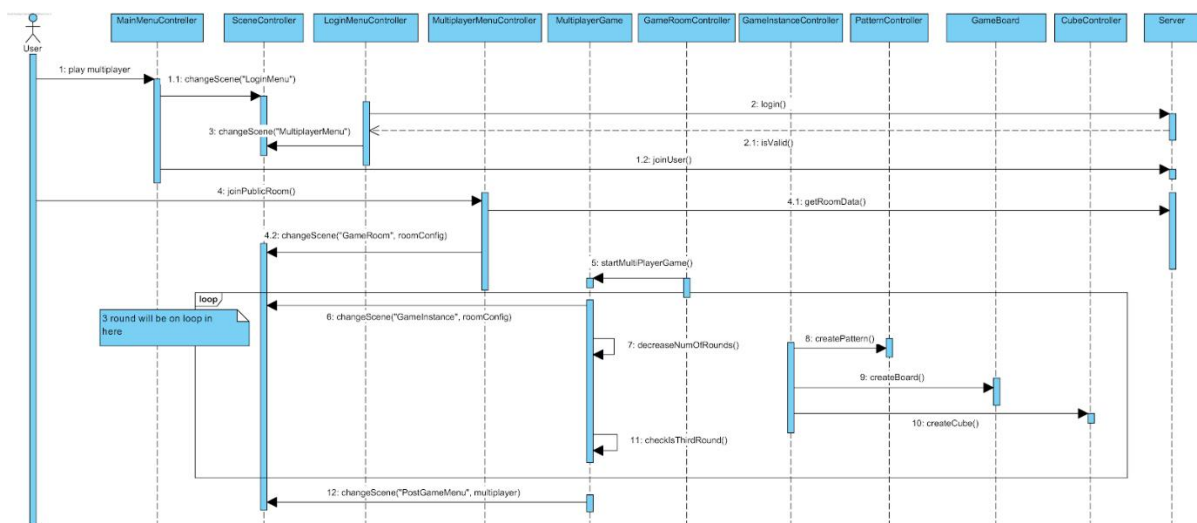


Figure 5: Sequence Diagram for Multiplayer Game

5.2.1.5 Options

To change the configurations of the game, user must enter the options menu from the main menu. In the options menu, music volume, effects volume, cube color and background color can be modified. After applying the configurations, user's old configurations will be overridden with the new modifications.

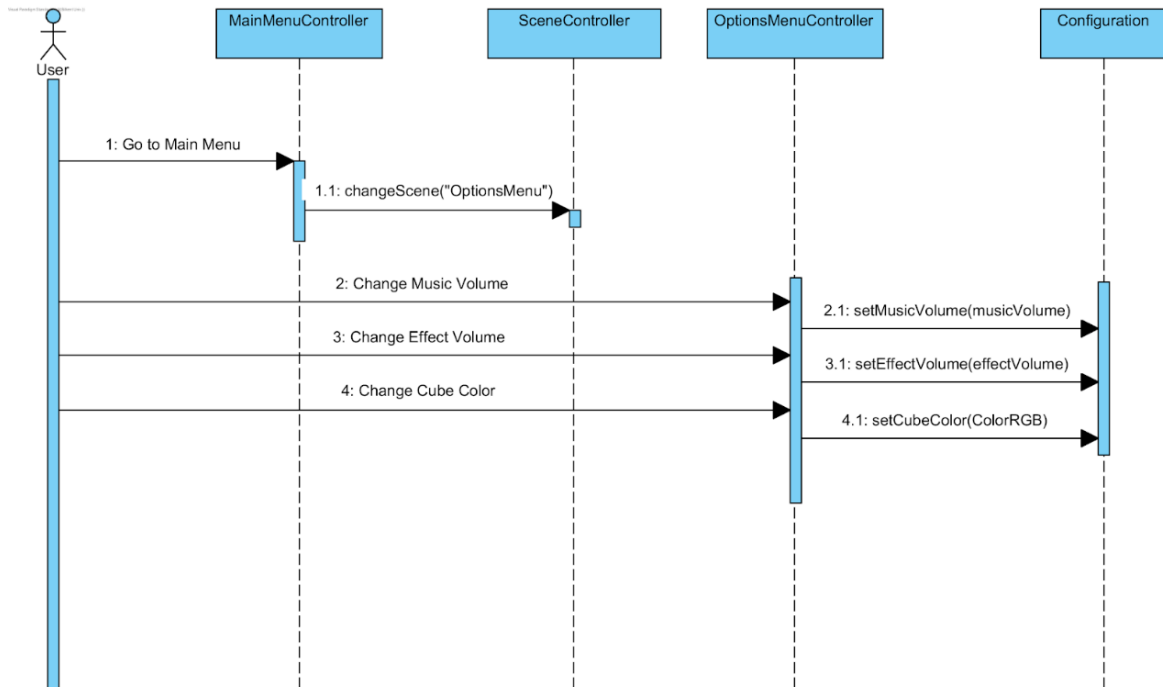


Figure 6: Sequence Diagram for Game Options

5.2.2 Activity Diagram

When the game launches, it will display the main menu consisting of single player, multiplayer, how to play, options and exit choices. If Single Player mode is selected, the system will create a game instance depending on the stage that is selected. If Multiplayer mode is selected, the system will display the login screen which consists of register, login and forgot password options. Then the necessary credential checking procedures will be handled. After registering and logging in successfully, the system will add the user to a room or create a private or public room depending on the request. After the room is full, the game instance will be created. For both playing options the system will change the focus status or orientation of the cube depending on the inputs. If the board becomes full and the pattern is submitted. The system will check the pattern for correctness. Then the post-game screens for single player and multiplayer modes. For single player mode, the post-game screen shows the time elapsed before solving the stage while for multiplayer mode, it announces the winner for the multiplayer game.

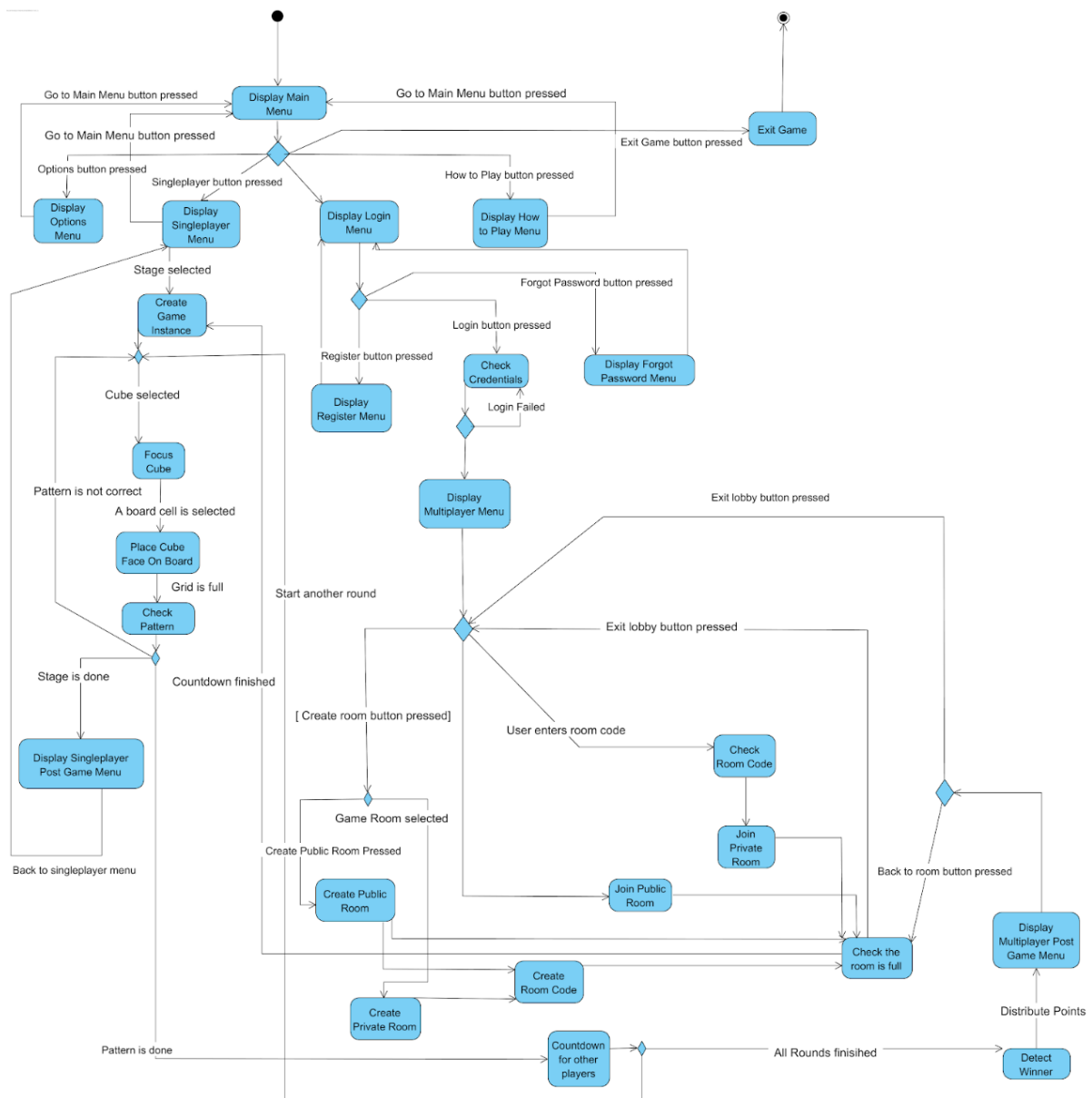


Figure 7: Activity Diagram for Q-bitz Game

5.2.3 State Diagrams

5.2.3.1 Cube

The cubes will be listed in a cube stack, the user can select a cube which will put the cube to a focused state. A focused cube can be rotated and then placed which will put the cube to the Cube on Board state. The user can also remove a cube from the board and this will put the cube to the Cube in Cube Stack state.

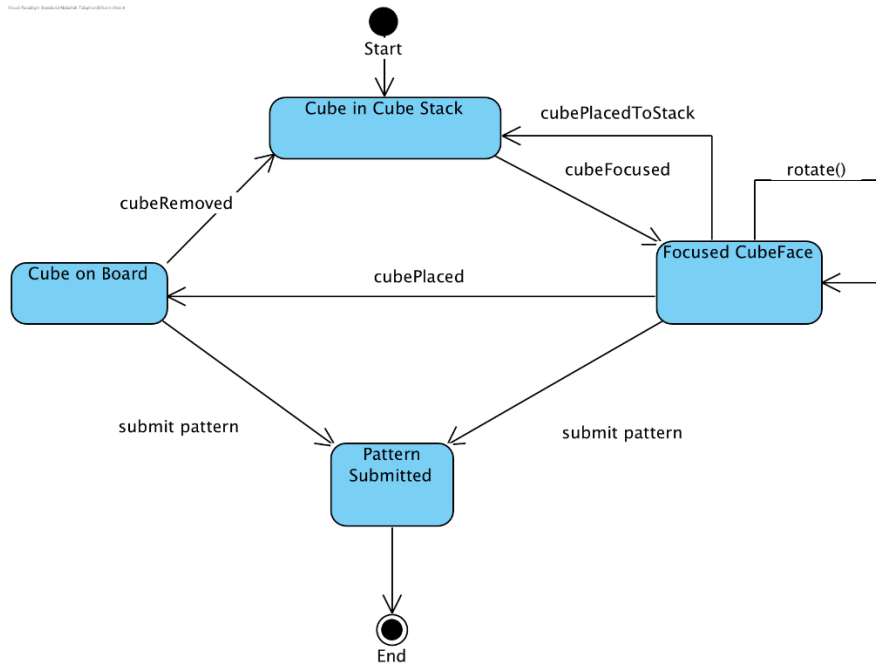


Figure 8: State Diagram for the Cube

5.2.3.2 Game Room

A Game Room is in the “Waiting for Players” state initially. If a user enters the room the user list is updated, if the room becomes full, the room will be in pre-game state. Maximum number of players in a room is decided when the room is created by the room creator. Pre-game state automatically starts the game after 15 seconds and the room will be in “Game in Progress” state. If the room is not full or the game has ended, the room will go back to “Waiting for Players” state.

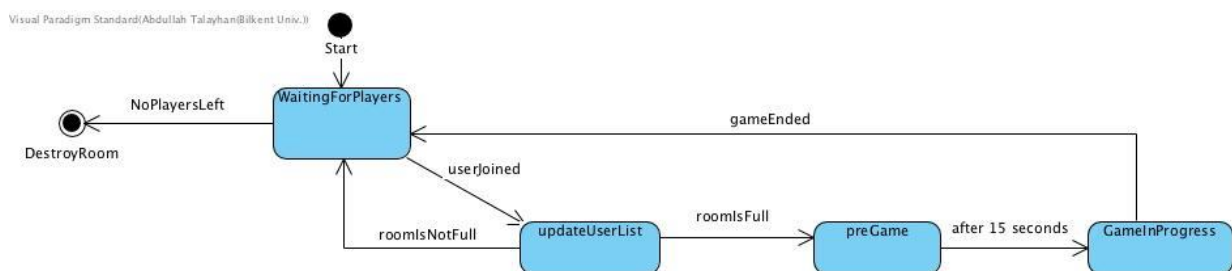


Figure 9: State Diagram for the Game

5.2.3.3 Board

At the start of the game, the board is empty. If the mouse is hovered over the board, the board will preview the face of the cube if a cube is selected. If the mouse is left clicked, the board will preview the face of the cube rotated 90 degrees for each click. After cube is placed, the board can be full or can contain some faces without being full. If the mouse is double clicked, the clicked face on the clicked board cell is removed, after removing a cube face, the board can either be empty or can still contain some faces. The created pattern can only be submitted if the board is full.

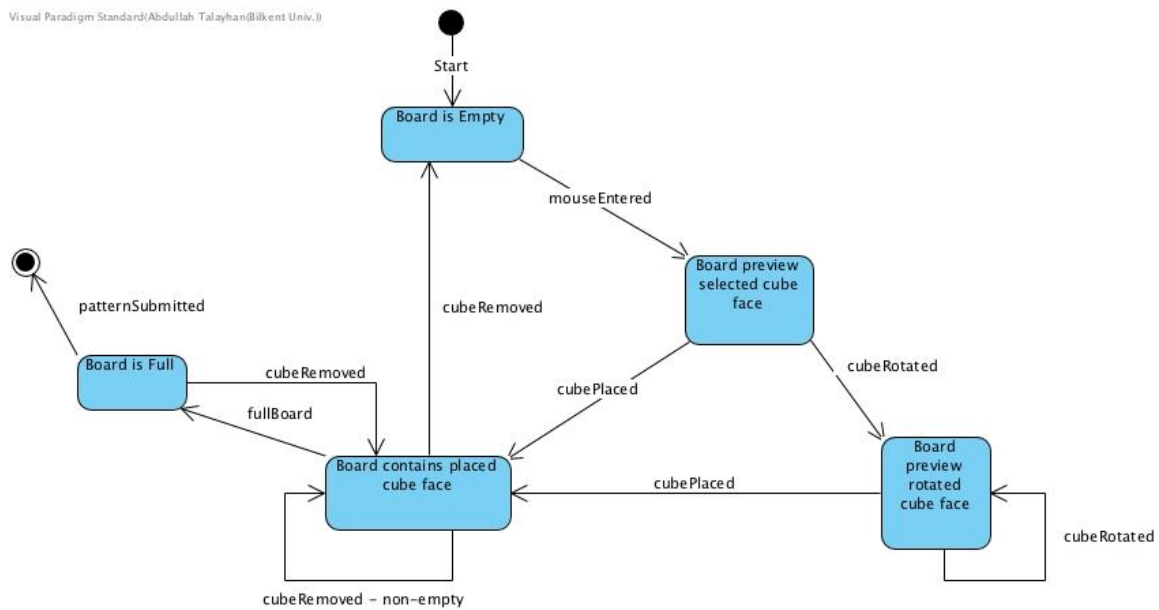


Figure 10: State Diagram for the Board

5.3 Object and class model

5.3.1 Client Class Model

Our game has a root class “Game”, handling the input listeners and the view.

- We have a “Menu” abstract class for all the navigation and non-game screens. User settings menu will display the fields for changing the email and password of the account and options for resetting the progress or deleting the account. Multiplayer menu, will display the room list and an option to create a game room. Room menu, is the menu for creating a new room and selecting its game mode, number of players and minimum required level for joining the room. On the Register menu, user can create an account and on the Login menu, user can log in with an existing account. If the user forgets their password, they can reset their password from the Forgot Password menu. On the Options menu, user can configure the game. Main menu allows navigation between game modes and other screens such as how to play and options. In the single player menu, user can select a stage to play.
- “User” class contains the information about the user such as experience, username, configuration and the online status.
- “Configuration” class contains the volume levels of music and effects, cube color scheme and key bindings for controls.
- “Game Instance” class is the main class that contains all the game components such as “PatternController”, “Pattern”, “CubeController”, “Cube” and “GameBoard”.
- “Pattern” class contains the pattern that will be recreated in the game with cubes.
- “PatternController” class initializes the pattern in the “Game Instance”. If it is a single player game, the pattern will be pre-generated pattern stored locally and if it is a multiplayer game, game will send a request to the server and server will generate a pattern and send it to the “PatternController”.
- “GameBoard” class contains the grid, the game is played on. If the grid is full, user can check if they recreated the pattern correctly.
- “Cube” class provide the functionality for rotation and rolling of the cubes. “rotate()” function provide the rotation of cube one face at a time to a specified direction.
- “CubeFace” class contains the information of the face of the cubes and order of the faces to create the cube object exactly for each creation.
- “CubeController” class contains the functions for creating the cubes and updating the cube places for “GameInstance” class.
- “GameRoom” class contains the data of the game lobby. “GameRoom” has a name, owner, number of players in the room, minimum level to join and visibility status. “GameRoom” can also generate a room code for others to join via code.

- “MultiplayerGame” class contains the attributes for endgame of a multiplayer game such as “winner” of the game and “pointsOfPlayer” to distribute the points to users.
- “SceneController” class allows transitioning from a menu to another menu and sending data between two menus.
- “MenuController” is an abstract superclass of all the menu controllers. It allows scene transitions using SceneController and communication between the server.

5.3.2 Server Class Model

- “QBitzServer” is the main class which holds the “QBitzServer” properties of the game. This class has instances of “DatabaseConnector” and “SocketServer” classes which are the main systems of the server of the game.
- “SocketServer” is used to hold socket operations. It has a collection of “SocketHandlers” and it is responsible for managing the socket connections with the clients. It receives operation requests from clients, processes them, and sends responses. This class is also an extension of Thread class. Therefore, it runs on a different thread other than the main thread of the “QBitzServer” application.
- “DatabaseConnector” is used to handle database operations. It has specific methods for predefined database operations. It also has dynamic query methods which can be used querying DB server easily.
- “ServerSocketHandler” is used to handle socket connections.
- “MailSender” is used to manage and send e-mails to the e-mail addresses.
- “Manager” is an abstract class which is responsible for managing the user account operations which are related to e-mail operations.
- “ResetPasswordManager” is an instance of “Manager” class and it is used to reset the password of the user accounts.
- “VerificationManager” is an instance of “Manager” class and it is used to verify the user e-mail addresses.
- “Room” is a model class to symbolize a Q-bitz game room. It holds the main properties of a room.
- “User” is a model class to symbolize a Q-bitz user. It holds the main properties of a user.

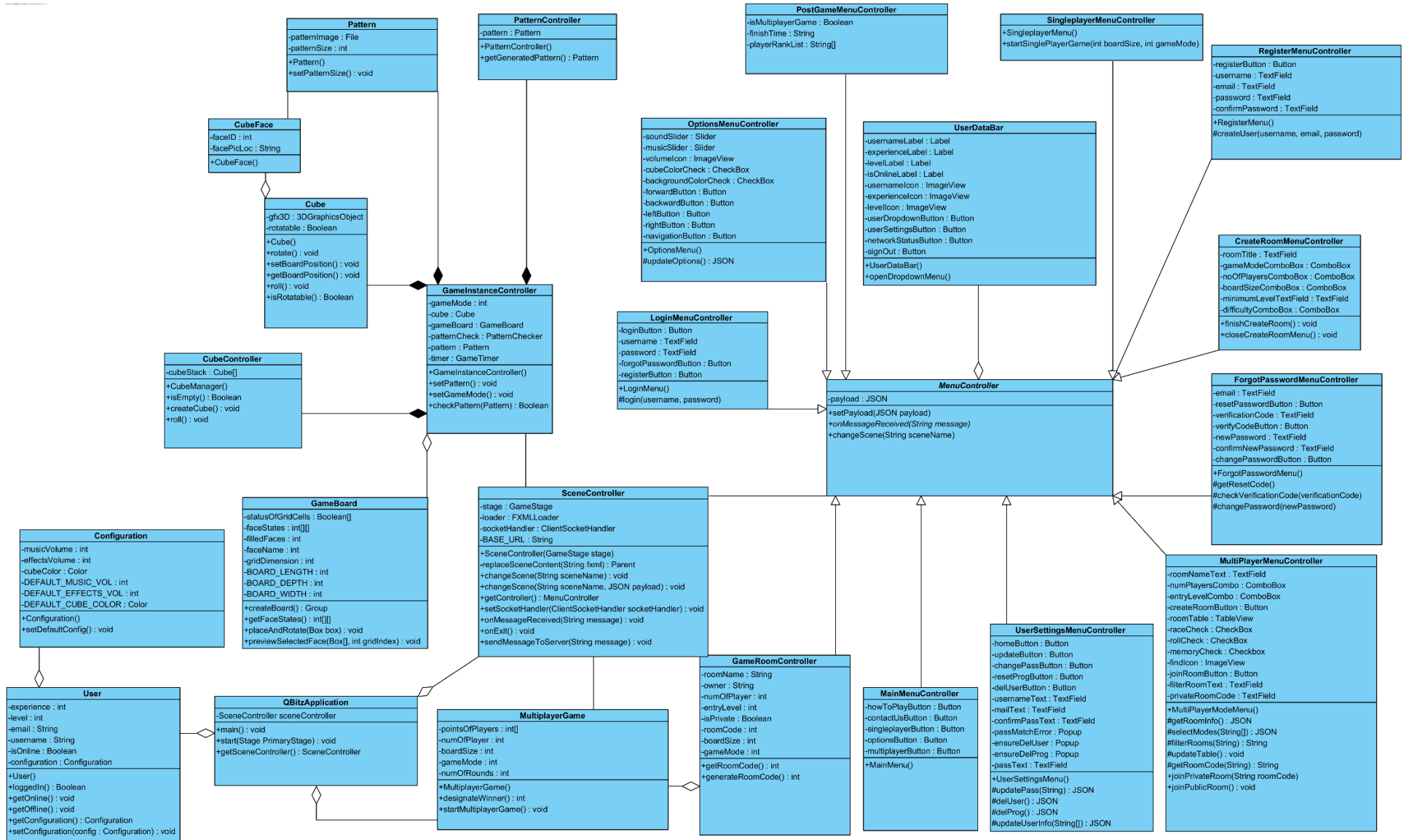


Figure 11: Class Diagram for the Client-side of the Q-bitz Game

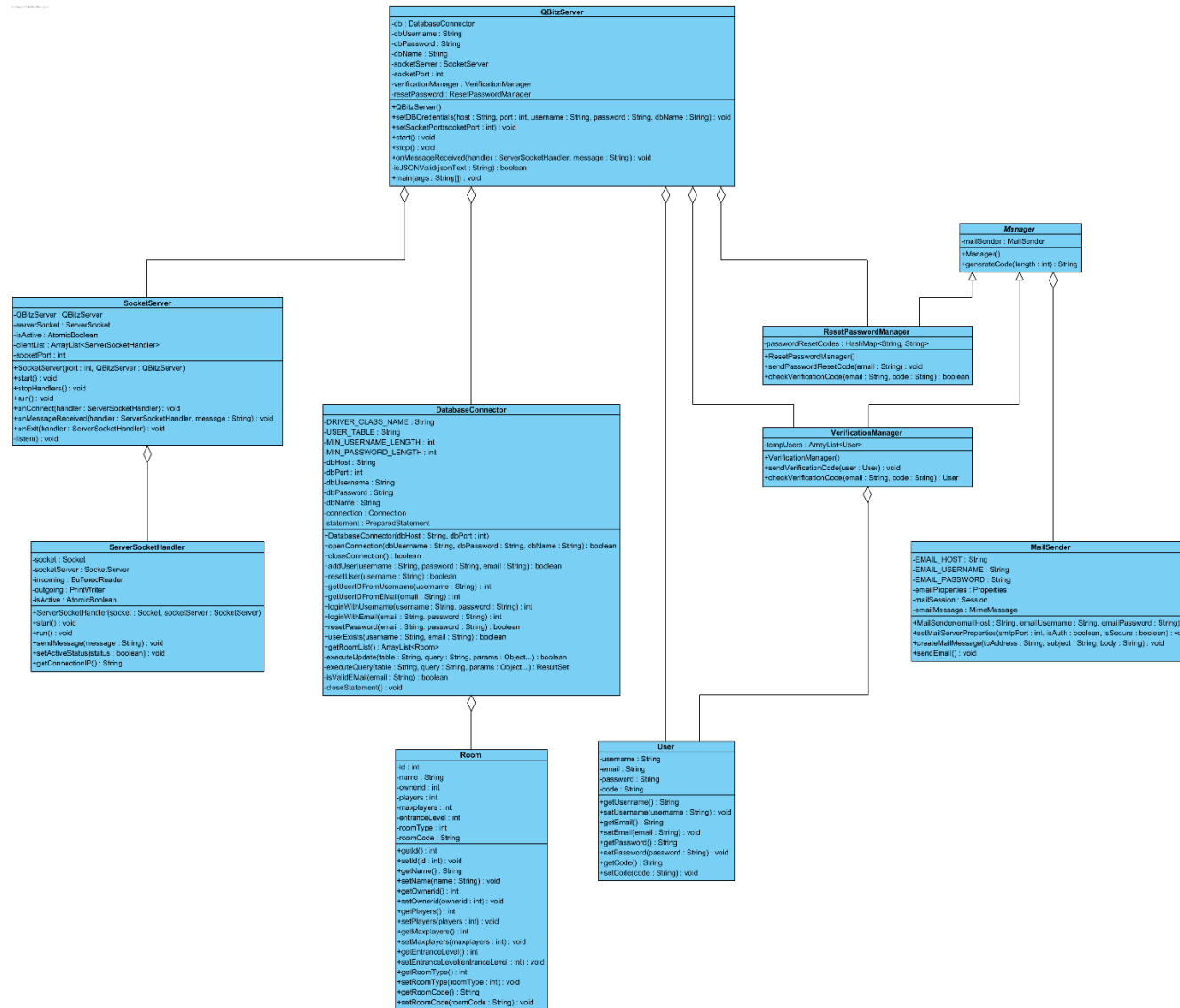


Figure 12: Class Diagram for the Server-side of the Q-bitz Game

5.4 User interface

5.4.1 Login

When the game is launched, user will be presented with the login screen. On this screen, users can login or create a new account or reset their password. If the user has no account, user can create an account by “Register”.

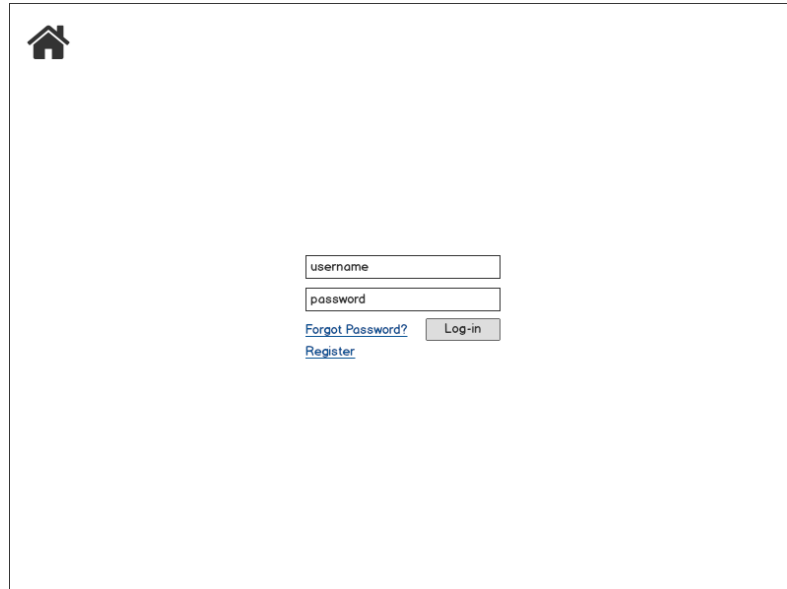
The image shows a login menu interface. In the top-left corner, there is a small house icon. The main area contains a form with two input fields: 'username' and 'password'. Below the 'password' field, there are two links: 'Forgot Password?' and 'Register'. To the right of these links is a 'Log-in' button.

Figure 13: Login Menu

5.4.2 Register

User can create a new account by entering an email, username and a password.


The image shows a register menu interface. In the top-left corner, there is a small house icon. The main area contains a form with four input fields: 'username', 'e-mail', 'password', and 'confirm password'. Below the 'confirm password' field is a 'Register' button.

Figure 14: Register Menu

5.4.3 Main Menu

Main menu provides the user with a selection of choices such as multiplayer, single player, how to play and options. User can also view their progress and other settings by clicking on the username.

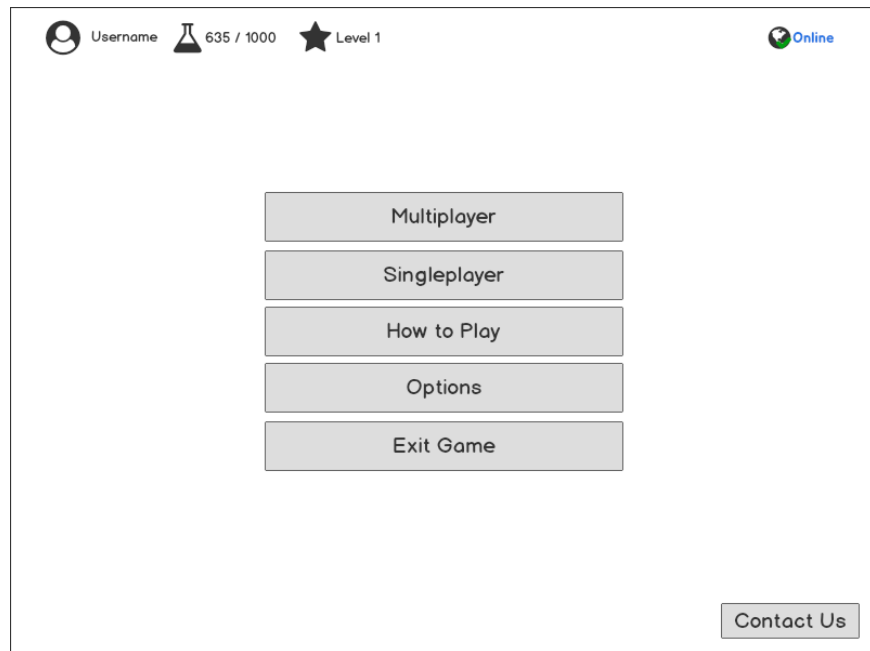


Figure 15: Main Menu

5.4.3 User Settings

User can change their email, username or password on the user settings screen. User can also delete their account or reset their progress on this screen.

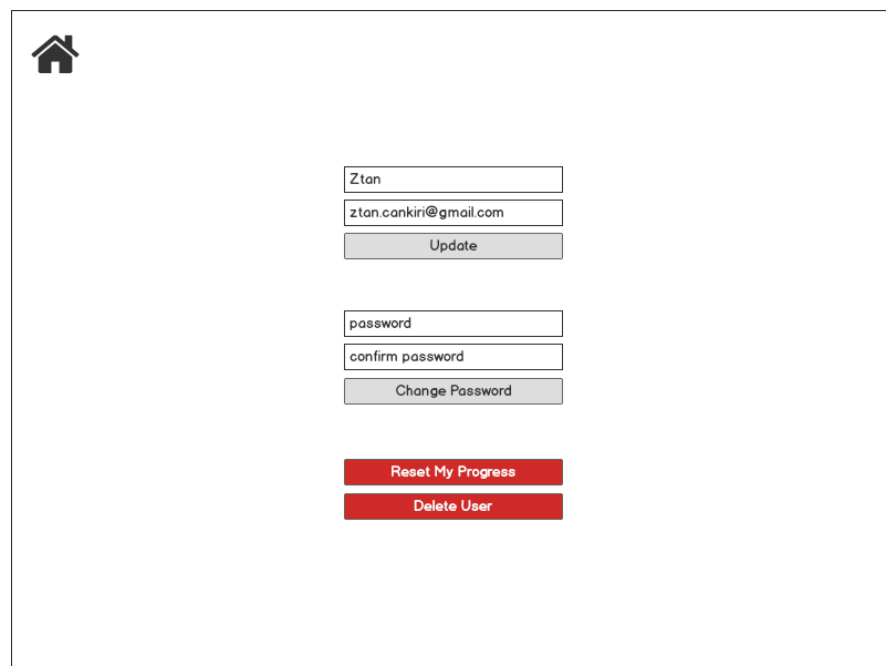


Figure 16: User Settings Menu

5.4.3 Room Selection in Multiplayer Mode

User can select a room from the list to join. User can also enter a private room by entering a room code. Rooms can be filtered with checkboxes or keywords.

The screenshot shows the 'Room Selection Menu' interface. At the top, it displays the user's profile: 'Username' with a profile icon, '635 / 1000' with a flask icon, and 'Level 1' with a star icon. On the right, there is an 'Online' status indicator. Below the profile, there is a search bar labeled 'filter rooms...' with a magnifying glass icon. To the right of the search bar are checkboxes for 'Race', 'Roll to Win', 'Memory', and 'Tournament'. Further right are two buttons: 'room code' and 'Enter Private Room'. Below these is a 'Create New Room' button. The main part of the interface is a table with the following columns: 'Room Name', 'Game Mode', 'Owner', 'Players', and 'Entrance Level'. The table contains several rows of room listings, including 'QBits Masters', 'Last QBits Benders', 'QBits Maestros', 'Game of QBits', 'Obsessive Mathematicians', 'Single Ladies Only !!!', 'QBits Maestros', 'asdasd123124', and 'QBits Gentlemen'.

Room Name	Game Mode	Owner	Players	Entrance Level
QBits Masters	Race	Ztan	6 / 6	5
Last QBits Benders	Roll to Win	con456	2 / 5	-
QBits Maestros	Memory	jeijey	2 / 2	16
Game of QBits	Tournament	eray_06	5 / 8	21
Obsessive Mathematicians	Race	sait	6 / 6	5
Single Ladies Only !!!	Roll to Win	crazy_berkin	2 / 5	-
QBits Maestros	Memory	buffer4head	2 / 2	16
asdasd123124	Tournament	bestPlayer4Ever	6 / 8	21
QBits Gentlemen	Race	david	6 / 6	5

Figure 17: Room Selection Menu

5.4.4 Room Creation in Multiplayer Mode

User can create a room for others to join. On this screen, user enters a room name, number of players and chooses a game mode and a minimum entrance level. A room can be public or private. Public rooms will be listed on the room list, while private rooms will only be accessible via code.

The screenshot shows the 'Room Creation Menu' interface. It features the same user profile and search filters as Figure 17. A 'New Room' dialog box is overlaid on the table. The dialog box contains the following fields: 'Room name' (text input), 'Public Room' (selected radio button) and 'Private Room' (radio button), '# Players' (text input), 'Entrance Level' (text input), 'Game Mode' (dropdown menu), and a 'Create Room' button. The background table is dimmed but still visible, showing the same room listings as in Figure 17.

Figure 18: Room Creation Menu

5.4.5 Game Lobby

On the game screen, all the cubes, a grid to place the cubes, the pattern to recreate, the remaining time and other players are displayed. User can zoom on a cube by clicking on it and then rotate the cube. Cube can be placed on the board by clicking on a tile.

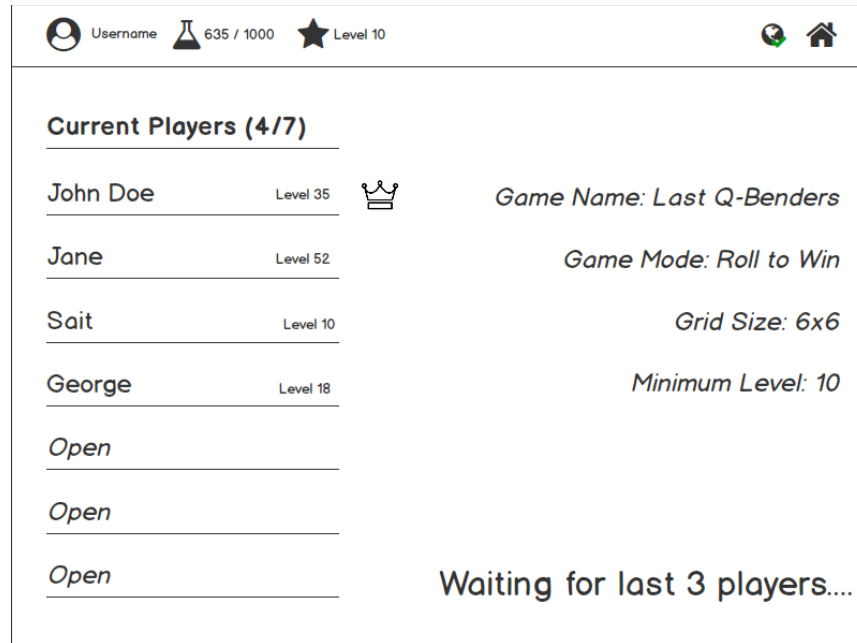


Figure 19: Game Lobby Menu

5.4.6 Game Screen

After joining a game room, user will be directed to the game lobby. In the game lobby, there will be a list of other users that have already joined the same game room and their level. Room name, game mode, the number of players, minimum level to join the room and board size will also be displayed.

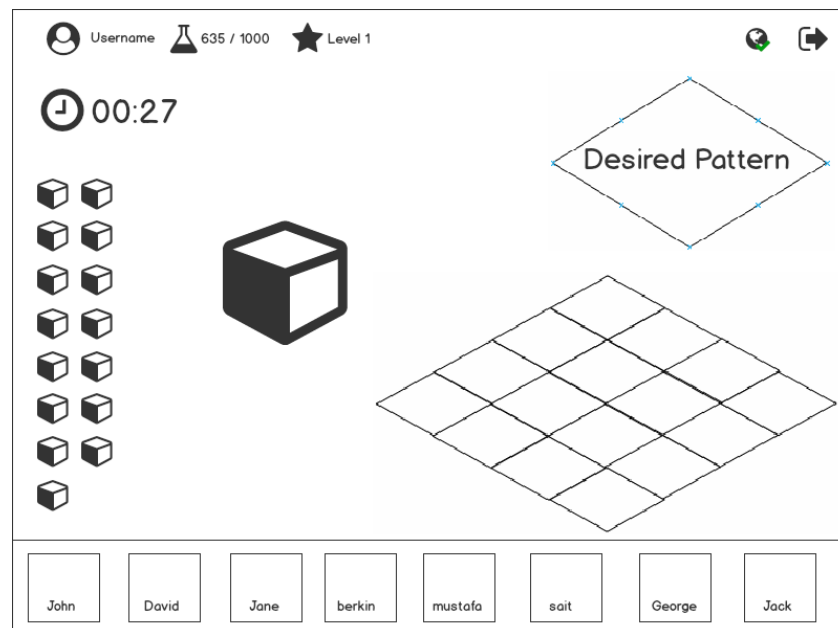


Figure 20: Game Screen

5.4.7 Post-Game Screen

After the game is finished, user will be directed to the post-game screen, where the ranking of users' and the winner will be displayed. User can choose to return to the room selection screen or join a new room with the same players who also wants to rematch.

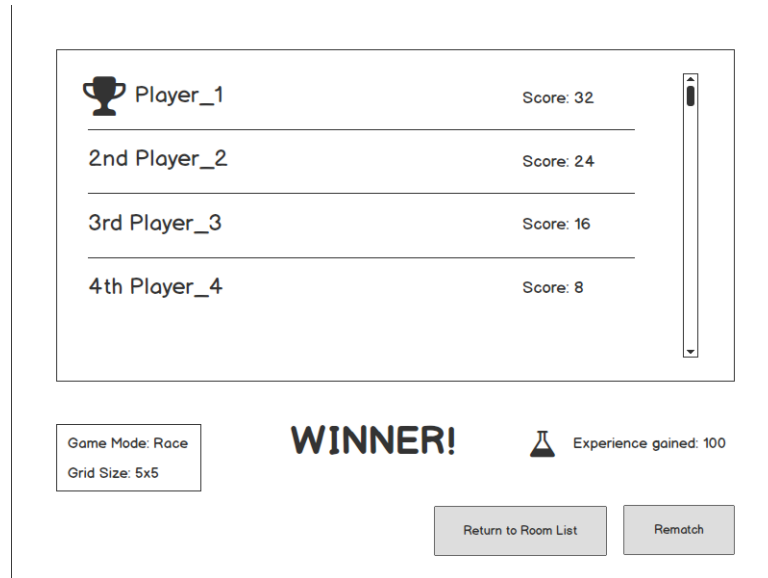


Figure 21: Post Game Screen

5.4.8 Single player Stage Selection Screen

User can play offline in single player mode. User can select a stage which is in different modes and in different board sizes, then can play it.



Figure 22: Single player Stage Selection

5.4.9 How to Play

In how to play screen, there will be illustrations explaining the rules of all the game modes and the controls.



Figure 23: How to Play

5.4.10 Options

In options menu, user can adjust the game's volume, change the color scheme of cubes and key bindings.

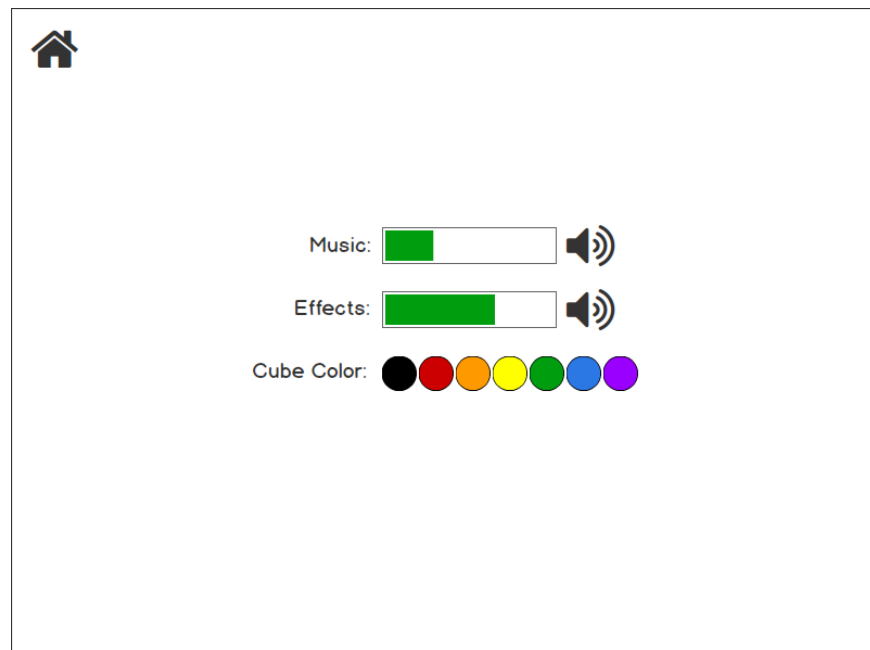


Figure 24: Game Option Menu

6. Improvement Summary

Since the first iteration, we had important changes for the game controlling mechanics that makes the game more playable in terms of placing the cubes faster. This change was to switch the cube rotation controllers from keyboard to the mouse and doing the cube face rotations by right clicking the mouse. We have updated our Use Case and Sequence Diagrams to show this change. We have also updated the mockup user interfaces that contains game controllers.

We revised our use case diagram and textual descriptions to better reflect that the diagrams represent a Q-bitz game instead of a more general game structure.

We revised our class names and names used in Sequence Diagrams to remove the ambiguities in Actor, Player and User or Game, Game Instance, Main Menu. Now we have more consistency between the diagrams and the names that are used inside the diagrams.

The Board is an entity that changes its state many times while placing or removing cubes. In order to better represent it, we have added a new state diagram for the cube class.

We have updated our Activity Diagram to capture the public and private room difference while joining a room.

Overall, we have increased the consistency of our diagrams and revised the details according to the design changes in the game.