Bilkent University

Department of Computer Engineering

CS 319 Term Project

Section 2

Group 2G, Oldies but Goldies

Q-bitz

Analysis Report

Project Group Members

1. Burak Kırımlı
2. Cansu Canan Ceyhan
3. Emre Keskin
4. Mert Çerçiler
5. Yağmur Özkök

Supervisor: Eray Tüzün

Contents

1. Introduction

As a group Oldies but Goldies, we are going to implement a table game called “Q-bitz”. It is an award-winning game that provides visual attention, mental - visual attention, piece - whole and shape - ground relationship, focus development for people over 8 years old. [1] We are going to implement this game as a desktop application which can be useful for all its players in terms of use. Many instances in the game such as cards, maps, and cubes can be implemented as objects.

The aim of Q-bitz is to create the shapes in 80 different Q-bitz cards by using 16 wooden cubes. There are three modes in Q-bitz. First one is a single player mode that players are supposed to create the shapes in cards and they do not contribute against other 3 players. Second mode is a multiplayer mode that all players are supposed to create the same shape in one Q-bitz card and the one who finishes first will win that card. Finally, in third mode, players have limited time to look and memorize the shape in Q-bitz card. Than they have to create that shape by memorizing. The player who makes the right shape the fastest wins the card.

In our Q-bitz, we have three different modes and each mode has three difficulty level which is not a part of original Q-bitz. For every difficulty level, the number of cubes are different and cards are created according to those number of cubes. The first and third game modes are the same as the original Q-bitz, however, the second game mode is different from the original one. It will be explained in detail later on. We also have different features from the original Q-bitz which are timer, time bonus, score, score bonus and some animations will be added as well. In additional features part, it will be explained in detail.

Currently, game will have the following features:

- Beat It (Mode)
- Rolling Stones (Mode)
- The Memory Remains (Mode)
- Variety of Maps (according to different difficulty levels)
- Sound Options
- Background Options
- Bonuses
- Animations

Java platform will be used in order to implement Q-bitz. Moreover, since our goal is to convert a table game into a desktop application, we are planning to use JavaFX to make our job easier for the Graphical User Interface(GUI) part.

2. Overview

2.1 Game Play

Q-bitz is a single player game that the players are supposed to create the same geometric sequence that are differ from in each Q-bitz cards with identical cubes who have different shapes in their sides at a specific time interval. There are three different types of modes in Q-bitz (see 2.2 for more information) and each mode has three different difficulty level. Difficulty levels are different from each other in terms of the cube numbers. However, for each difficulty, cubes are still identical. For each mode and difficulty level, time interval will be decreased as the player starts a new card to play. When the game begins, there will be a score table that shows players score. For every right cube, player will score point and for every wrong cube player will lose point. The player has an opportunity to pause, reset or quit the game.

2.2 Modes

In game there are three different modes which are “Beat It”, “Rolling Stones” and “The Memory Remains”. All these modes have three difficulty levels which are “Easy”, “Medium” and “Hard”. In “Easy” level there are 16 cubes, in “Medium” level there are 25 cubes and in “Hard” level there are 36 cubes. All cubes are identical.

2.2.1 Beat It

In “Beat It” mode, the map is empty. User looks the design which is located on the opened card. With finding the correct side of the cube, user tries to fill the map and complete the design. If user puts the correct cube to correct square, score is going to be increased and if user puts the wrong cube to wrong square, score is going to be decreased. After completing the map with correct cubes, timer is going to stop and shows the elapsed time.

2.2.2 Rolling Stones

In “Rolling Stones” mode, the map is already filled with random cubes. If user rolls the cubes, the patterns of the cubes on the map changes randomly. User can fix the selected cube’s location or swap two cubes. With these actions, if one cube located correctly, score is going to be increased and if one cube located incorrectly, score is going to be decreased. After completing the map with correct cubes, timer is going to stop and shows the elapsed time.

2.2.3 The Memory Remains

In “The Memory Remains” mode, the map is empty. User looks the design for ten seconds which is located on the opened card. After that card is going to be closed. With finding the correct side of the cube, user tries to fill the map and complete the design. If user puts the correct cube to correct square, score is going to be increased and if user puts the wrong cube to wrong square, score is going to be decreased. After completing the map with correct cubes, timer is going to stop and shows the elapsed time.

2.3 Maps

Maps are the places where players put cubes to complete the Q-bitz cards. According to the difficulty level that players selects, maps will be created. We mean that since we have different cube numbers for each difficulty levels, maps will be created according to those number of cubes. Easy level consists of 16 identical cubes, while medium level has 25 and hard level has 36 identical cubes. Moreover, for the “Beat It” and “The Memory Remains” modes, maps will be empty to be filled whereas “Rolling Stones” mode has a filled map with mixed cubes.

2.4 Cards

Cards, we also mention them as Q-bitz cards, have the geometric sequences that will be created by players into the map. Cards are changing according to difficulty levels because each level has different number of cubes. The sequences are fixed into a square and we will divide this square according to the selected difficulty level.

2.5 Options

In the game screen, players have different buttons in order to play the game. First of all, at the top, there will be a “Pause” button which will open the Pause screen and players can reset or quit the game from this screen. If they close this screen without resetting or quitting the game, they will continue to play. For the “Beat It” and “The Memory Remains” modes, there will be two buttons which are “Put” and “Change”. “Put” button provides player to put a cube into the map wherever they want while “Change” button provides them to change the surface of the cube that they want to place. For the “Rolling Stones” mode, players will have three options which are “Put”, “Roll” and “Swap”. In this mode, cubes will be already placed in the map but they will be mixed. Since they cannot change cubes surfaces, they need to roll the cubes by using “Roll” option until a surface which is necessary comes. “Swap” button will be used in order to swap two cubes. After doing this, players can use “Put” button to fix cubes which are in correct places in the map.

2.6 Settings

Players can reach this screen from Main Menu and they will be able to change the music and the background color from this screen.

3. Functional Requirements

3.1 Play Game

After entering the screen user can start the game by pressing Play Game button. As this game single player game. User should only select the mode and difficulty of the game. Basis of the game is same for every mode. Users will construct the the given picture by using cubes. System provides users a picture and cubes. Then system waits users to combine figures on the cubes to construct the picture. After user puts every cube at the map system checks if cubes place is right according to system. However, every mode has differences from each other. In the following there is explanations of every mode.

3.1.1 Beat It

Beat It is classical game where users can construct the given picture by combining the figures on the cubes. To find the necessary figure user can rotate the cube by pressing change button. On every side of the cubes there are different figures. So by rotating cubes, user can reach the necessary figure. When cube is placed on the map system will check if it has right place according to picture. For further explanation about system check 5.2.2 Activity Diagram.

3.1.2 Rolling Stones

In Rolling Stones different from Beat It, users are not able to rotate the cube. Instead they can roll the cubes to get the necessary figures. As users cannot rotate the cube, they will swap the cubes according to the figures on their sides. After putting cube and press put button system will check the figure’s place. When user press roll button, system will roll unfixed cubes when roll button pressed and until they combine all of the cubes by swapping then user will continue rolling and combining.

3.1.3 The Memory Remains

The Memory Remains is same as Beat it only difference between them is, in Beat It players can visualize the picture, every time they want but in The Memory Remains players can visualize the picture for a specific time. While that specific time player should memorize the picture, because while they combining the cubes they will not able to visualize the picture. After they finish combining system will check if combination is right.

3.1.4 Difficulty

After mode selection, user will select the difficulty of the game. Difficulty of the game is based on the size on the map. Difficulty, divided into three as following:

Easy: Picture is divided into 4x4 square. So users will fill 16 squares using cubes.

Medium: Picture is divided into 5x5 square. So users will fill 25 squares using cubes.

Hard: Picture is divided into 6x6 square. So users will fill 36 squares using cubes.

3.2 How to Play

In every mode of the game, how to play varies. So how to play for every mode is in the following.

Beat it:

1. Press change button until you reach necessary side of the cube.

2. Then select the place on the map.

3. Then press the put button.

Rolling Stones:

1. Select two button.

2. Then press swap.

3. Then select the button that you think it has the right place.

4. Then press put.

The Memory Remains:

1. Press change button until you reach necessary side of the cube.

2. Then select the place on the map.

3. Then press the put button.

3.3 Settings

Players can access this from Settings button under the Play Game button. In this section user will be able to change sound settings and change background colors. System will provide user specific colors and user will select among them.

3.4 Credits

Users can access this screen from the Credits button from the menu. In this section, users can see the names of developers of this game.

4. Nonfunctional Requirements

In this section, as a nonfunctional requirements usability, reliability, performance and supportability are going to be explained.

4.1 Usability

In “Q-Bitz” game it is going to be implemented such that user can easily understand and play the game without reading “How to Play” section. Buttons are going to be created and placed in a way that user can understand their functions easily. The game screen will be designed in an understandable manner. Colors on the game screen will be selected in such a way that will not disturb the user’s eyes. These properties are in the usability part of nonfunctional requirements for “Q-Bitz”.

4.2 Reliability

To prevent the possible crashes and bugs that can be occurred in the future “Q-Bitz” is going to be tested many times before released. All possible cases are going to be tested to release a reliable game. This property is in the reliability part of nonfunctional requirements for “Q-Bitz”.

4.3 Performance

System response times are the most important criteria for performance. In "Q-Bitz", all user inputs are going to be acknowledged at most in one second. In our game, there is special peak time case when card is uploading and map is resetting. This case is special peak time for "Q-Bitz" and it is taking at most one second. This property is in the reliability part of nonfunctional requirements for "Q-Bitz".

4.4 Supportability

In supportability, there are two criteria which are adaptability and maintainability. For adaptability, "Q-Bitz", will be designed to keep up with changes in Windows and Mac OS. For maintainability, since "Q- Bitz" is implemented in an object-oriented way, in future its unwanted features can be easily changed and new features can be added easily. These properties are in the supportability part of nonfunctional requirements for "Q-Bitz".

4.5 Additional Requirements

4.5.1 Bonuses

There are two types of bonusses in Q-Bitz, time bonus and score bonus.

In Q-Bitz, player has to finish the card before the time finishes. Also, every successful finish of the card, score will be increased according to the remaining time, and time will be restarted with a lesser amount of time for the new card. However, in each card, time bonus will be placed to the randomly selected area of the map. This means, when player put the cube successfully to the selected area of map, time bonus will be activated which freezes time for 10 seconds. This time bonus will appear only once during each card.

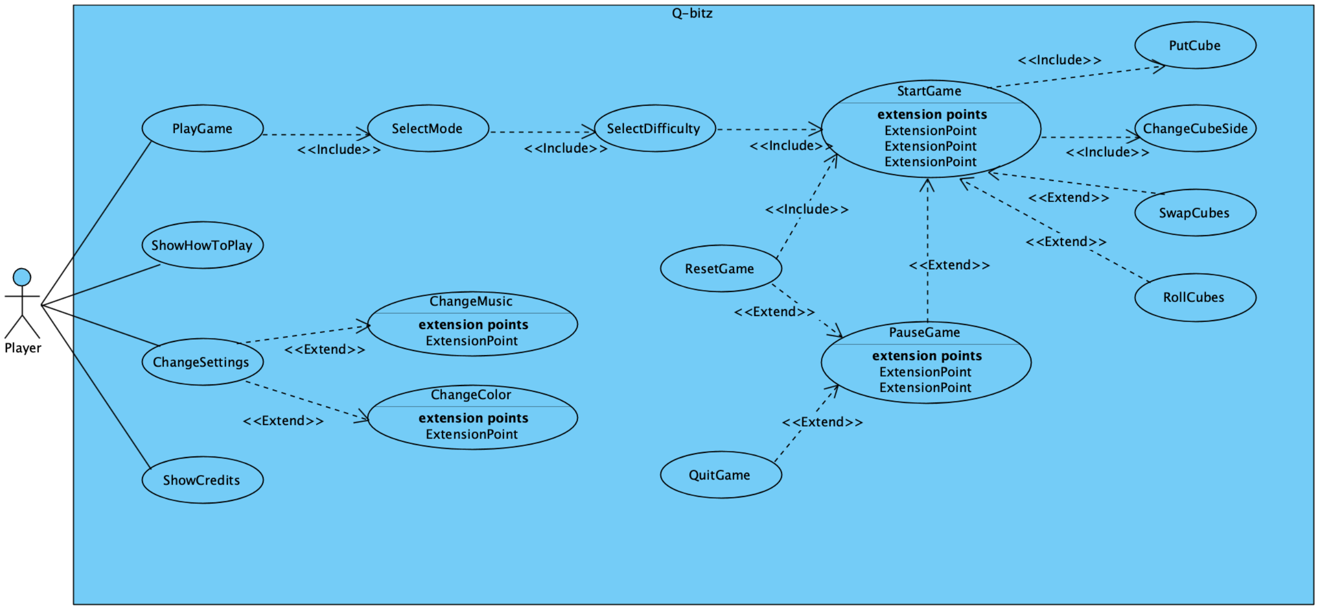
Making the high score is essential purpose of our game. If player put the accurate cube to the map successively 3 times, score bonus will be activated for specific amount of time. During this time period, player will gain the score 2 times than default for each successfull placement.

4. 5. 2. Animations

Animations will make our game more enjoyable and make players more ambitious. We add animations to our game when score bonus and time bonus will be activated. Also, after each successfull finish of card and each game finishes with failure, some funny GIF's will appear.

5. System Models

5.1 Use – Case Model



Use Case #1

Use Case: Play Game

Primary Actor: Player

Stakeholders and Interests:

- Player who wants to play the game
- System chooses random puzzle and starts the game.

Pre-conditions:

- Player must be in the main menu.

Post-conditions:

- No post conditions.

Entry-conditions:

- Player chooses “Play Game” button in the main menu.
- Player must choose the game mode (Beat it, Rolling Stones, The Memory Remains)
- Player must choose the difficulty (easy, medium, hard)

Exit conditions:

- Player pauses the game and click “Exit” button.
- Time limit is up and player can no longer continue playing that current puzzle.

Success Scenario Event Flow:

1. Player chooses “Play Game” button from main menu.

2. Player chooses mod which are “Beat it”, “Rolling Stones” and “The Memory Remains”.

3. Player chooses level among easy, medium and hard.

4. Player is represented a puzzle and cubes.

5. Player presses “Change” button if he/she desires to change side of given cube.

6. Player presses “Put” button then click desired space to put the cube in any place on map.

7. Player can only use ”swapCubes” button if he/she has chosen “Rolling Stones” mode after choosing any difficulty level.

8. Player can only use ”rollCubes” button if he/she has chosen “Rolling Stones” mode after choosing any difficulty level.

9. Player can know he/she is successful if the cube remains in place.

10. Player solves the puzzle.

11. Player is presented with another puzzle.

12. Steps 4-11 until time limit is up.

13. Player sees his/her score on screen.

Alternative Event Flows:

i. If player cannot solve any puzzle in time limit

- Player sees his/her score on screen.

ii. If player wants to return to main menu

- Player pauses the game with “Pause” button.
- Player chooses option ”Back to Main Menu.”

iii. If player wants to restart the game

- Player pauses the game with “Pause” button.
- Player chooses “Reset Game.”

- Steps 4-11 is repeated.

Use Case #2

Use Case: Show How to Play

Primary Actor: Player

Stakeholders and Interests:

- Player who wants to learn how game is played
- How to play screen is displayed by the system.

Pre-conditions:

- Player must be in the main menu.

Post-conditions:

- No post conditions.

Entry-conditions:

- Player presses “How to Play” button in the main menu.

Exit conditions:

- Player chooses “Back to Main Menu” and returns to main menu.

Success Scenario Event Flow:

1. Player chooses “How to Play” button from the main menu.

2. Player reads the description of the game and familiarize himself/herself with the game.

3. Player chooses “Back to Main Menu” and returns to main menu ready to play the game.

Use Case #3

Use Case: Pause Game

Primary Actor: Player

Stakeholders and Interests:

- Player wants to pause the game.
- Pause screen is displayed by system.

Pre-conditions:

- Player must have started the game in order to pause.

Post-conditions:

- No post conditions.

Entry-conditions:

- Player presses “Pause” button in the game screen.

Exit conditions:

- Player clicks exit button in the top left corner of the pause menu and continue to play the game.

Success Scenario Event Flow:

1. Player starts the game choosing game mode and difficulty.

2. Player clicks “Pause” button which is next to score.

3. Player resume playing the game by clicking exit button in the top left corner of the pause menu and resume the game at any desired moment.

Alternative Event Flows:

i. If player desires to restart the game

- Player chooses “Reset Game” in pause menu.
- Player plays a new game with already defined mode and difficulty.

ii. If player wants to exit

- Player chooses “Quit” button from the pause menu.
- Player goes back to Main Menu of the game.

Use Case #4

Use Case: Change Settings

Primary Actor: Player

Stakeholders and Interests:

- Player who wants to change background color or music of the game
- System displays settings screen.

Pre-conditions:

- Player must be in the main menu.

Post-conditions:

- Player must be in the main menu.

Entry-conditions:

- Player presses “Settings” button in the main menu.

Exit conditions:

- Player chooses “Back to Main Menu” and returns to main menu.

Success Scenario Event Flow:

1. Player chooses “Settings” button from the main menu.

2. Player changes background color of the game.

3. Player changes music playing in the background of the game.

4. Player chooses “Back to Main Menu” and returns to main menu ready to play the game.

Use Case #5

Use Case: Show Credits

Primary Actor: Player

Stakeholders and Interests:

- Player who wants to see who make this game.

- System displays credits screen.

Pre-conditions:

- Player must be in the main menu.

Post-conditions:

- No post conditions.

Entry-conditions:

- Player presses “Credits” button in the main menu.

Exit conditions:

- Player chooses “Back to Main Menu” and returns to main menu.

Success Scenario Event Flow:

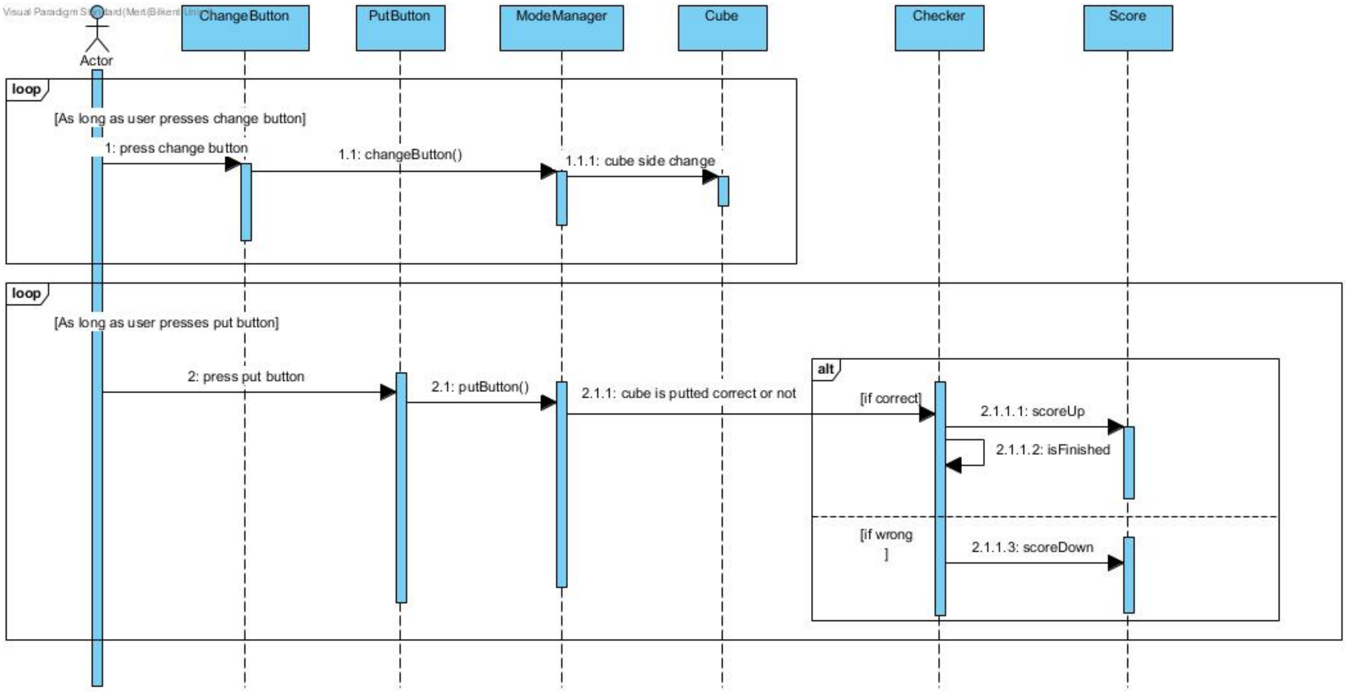
- Player who wants to see who make this game.
- System displays credits screen.

5.2 Dynamic Models

5.2.1 Sequence Diagrams

1. **Scenario:** User plays Beat It or The Memory Remains Game Mode

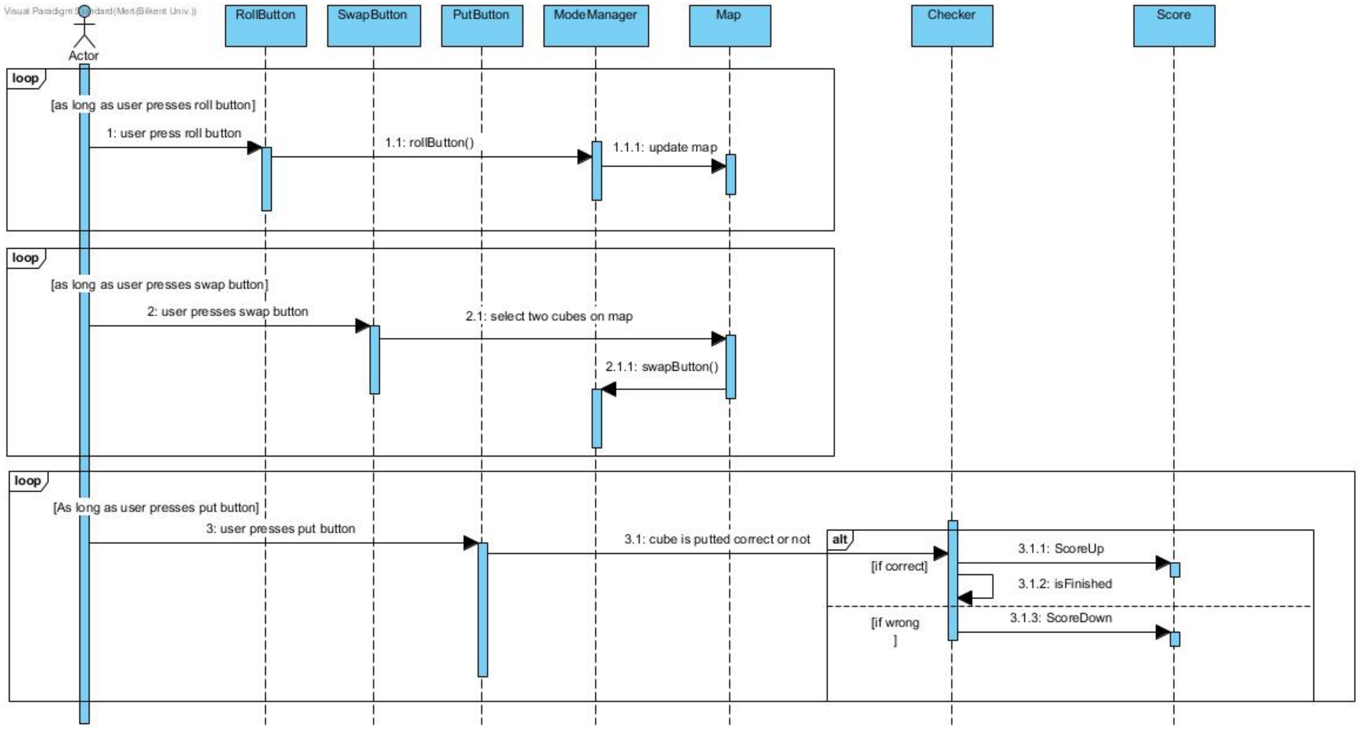
While user is playing the Beat It or the Memory Remains game modes, user has two options, which are putting cube into the map or changing the cube side. If user wants to change the cube side, user clicks the change button. Then changeButton() method, which is in the ModeManager class, will be executed. This method reaches the cube and changes the side of the cube. As long user wants to change the cube side, this sequence will take place. If user wants to put the cube into the map, user clicks the put button. Then putButton() method, which is in the ModeManager class, will be executed, same as the change button. Then if the cube is putted correctly or not will be checked in the Checker class. If the cube is putted correctly, score will be up and if the game is finished or not will be checked in isFinished method, where is also in the Checker class. If the cube is not putted correctly, score will be down. As long as user wants to put cube, this sequence will take the place.



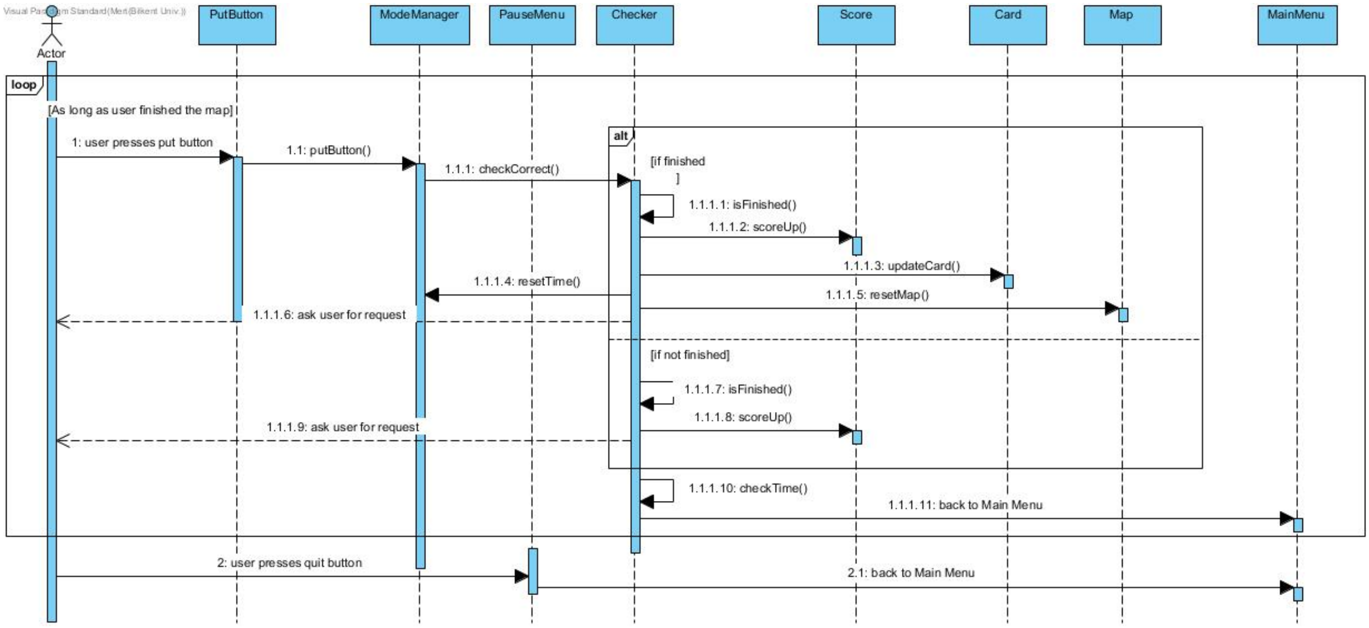
2. **Scenario:** User plays Rolling Stones Game Mode

While user is playing Rolling Stones game mode, user has three options, which are rolling cubes, swap two cubes and put cube into the map. If user wants to roll the cubes, clicks roll button and rollButton() method will be executed, which is in the ModeManager class. This method changes the all cubes except the stabled ones. As long as user wants to roll cubes, this sequence will take the place. If user wants to swap two cubes, clicks swap button and selects two cubes that is wanted to swapped in the map. Then swapButton() will be executed which is in the ModeManager class. As long as user wants to swap cubes, this sequence will take place. If user wants to put cubes, it is same as BeatIt and MemoryRemains modes. User clicks cube and put button, then putButton() method will be executed in ModeManager class. After that, if the cube is putted correctly or not will be checked in the Checker class. If the cube is putted correctly, score will be up and if the game is finished or not will be checked in isFinished method, where is also in the Checker class. If the cube is not putted correctly, score will be down. As long as user wants to put cube, this sequence will take the place. As long as user wants to put cube, this sequence will take the place.

3. **Scenerio:** Game Finish

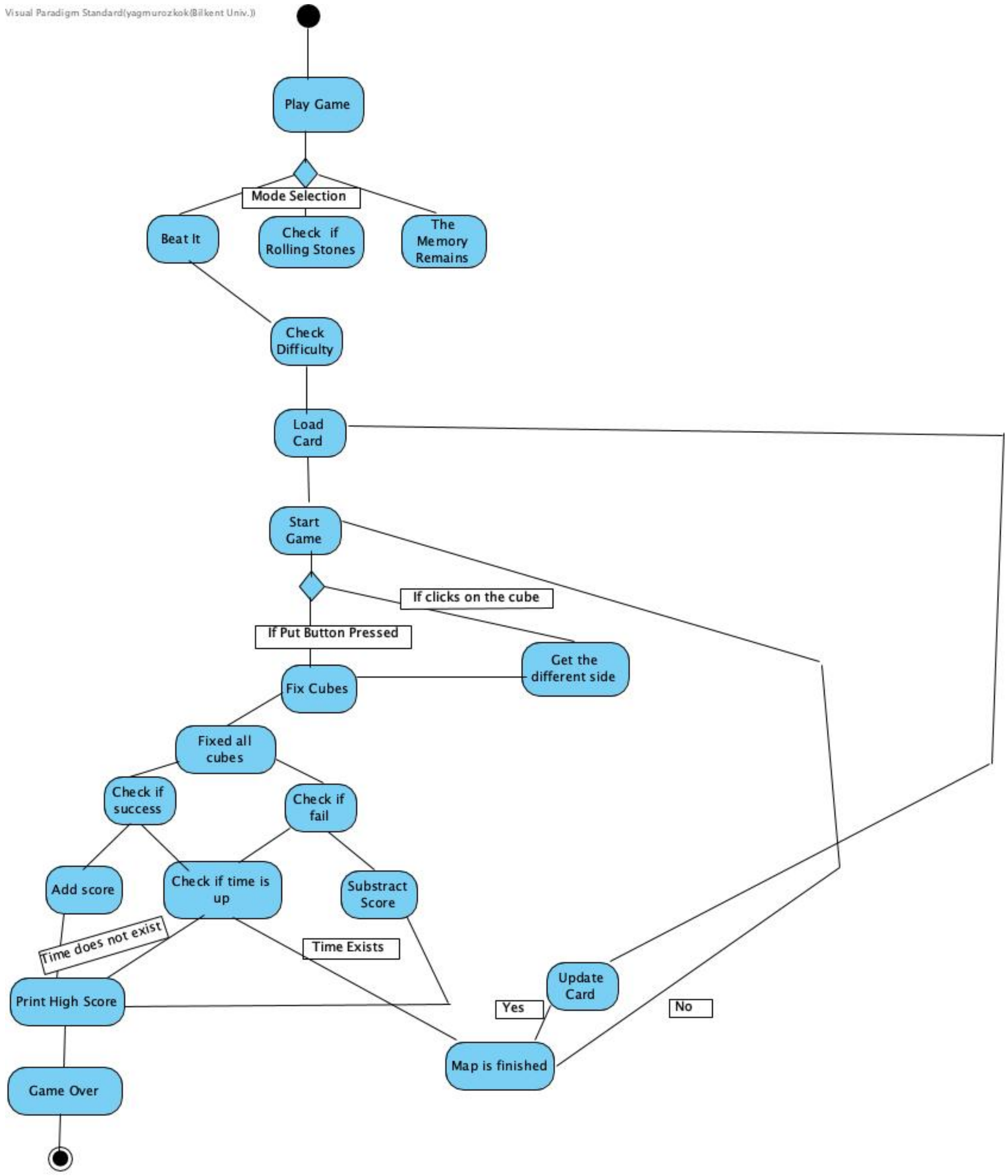

Visual Paradigm Standard(Mert(Bilkent Univ.))
Actor
RollButton
SwapButton
PutButton
ModeManager
Map
Checker
Score
loop
[as long as user presses roll button]
1: user press roll button
1.1: rollButton()
1.1.1: update map
loop
[as long as user presses swap button]
2: user presses swap button
2.1: select two cubes on map
2.1.1: swapButton()
loop
[As long as user presses put button]
3: user presses put button
3.1: cube is putted correct or not
alt
[if correct]
3.1.1: ScoreUp
3.1.2: isFinished
[if wrong]
3.1.3: ScoreDown

After the successfull insertion, if all cubes placed according to the card, isFinished method will return true, which is in the Checker class. Then score will be increased according to remaining time, new card will appear, time and map will be restarted. Then player will play the new card. However, if isFinished will not return true, that means player does not place all cubes according to given card, after successful insertion. So score will be up and player keep playing the existing card. Also, whenever time is up, game will over and user will be forwarded to the main menu. In addition, user can click the quit game button in the pause menu and quit the game manually.

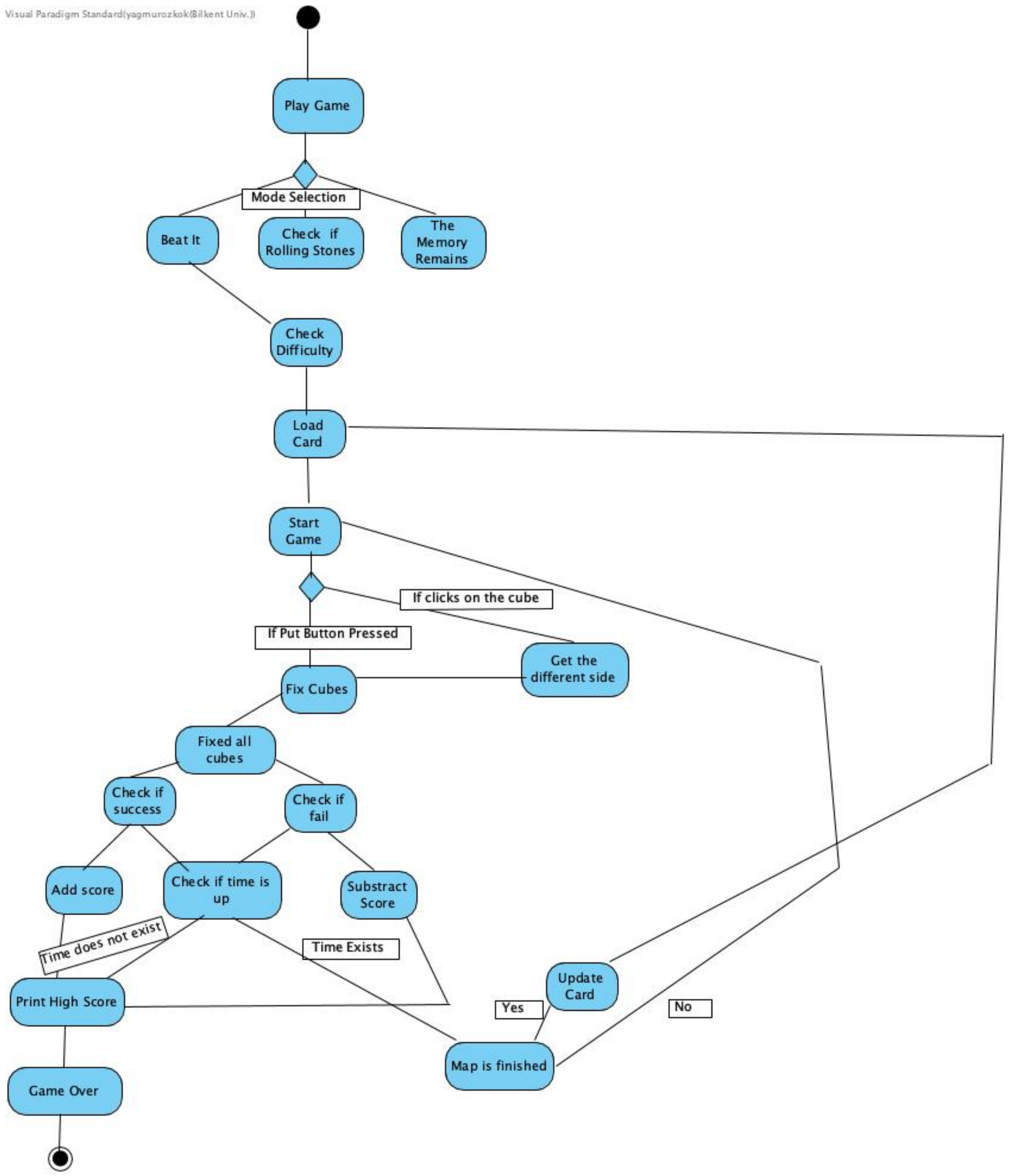


5.2.2 Activity Diagrams

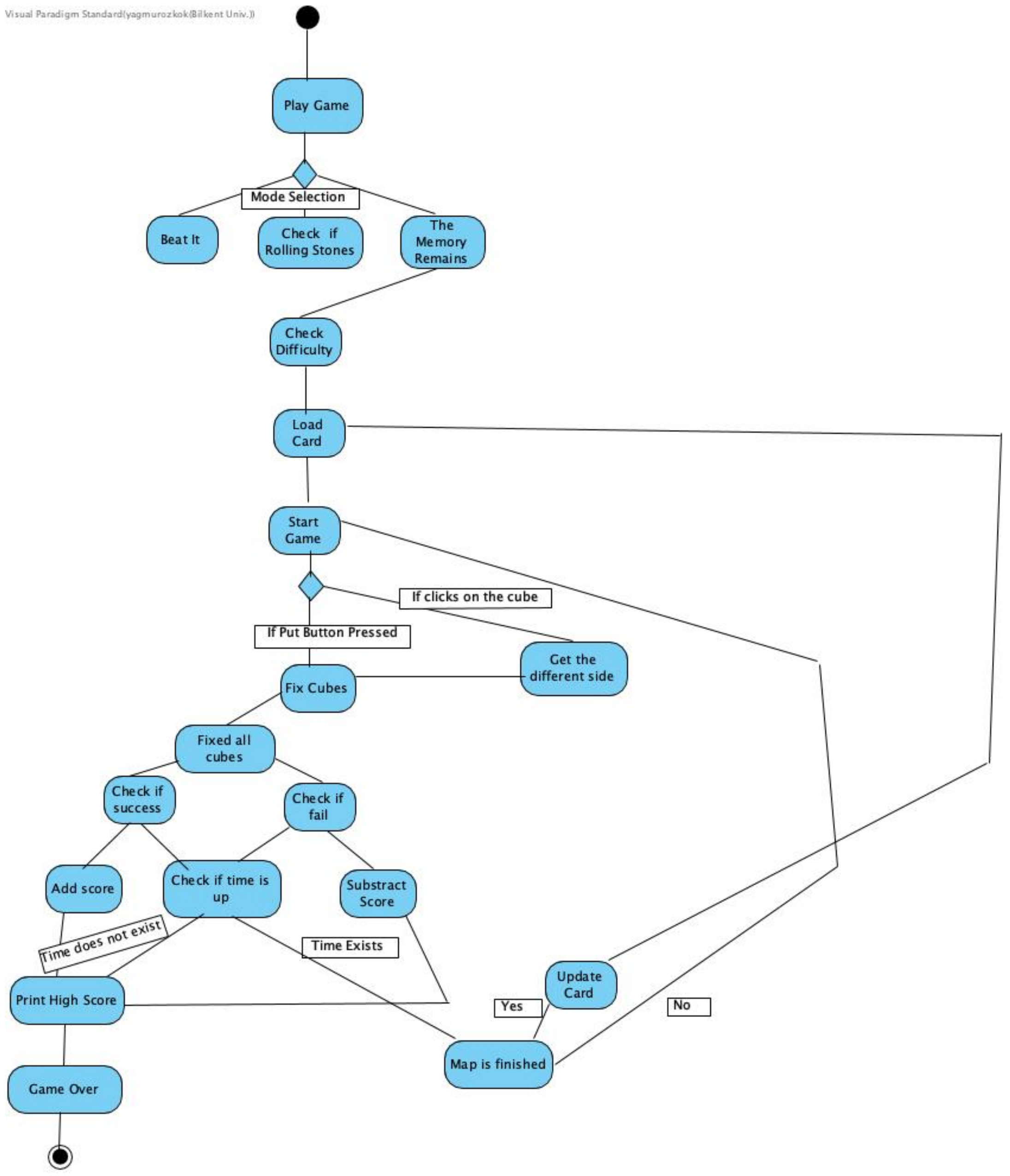
There are three different activity diagrams related to mode selection.



If user selects Beat It mode (Figure 1)



If user selects Rolling Stones mode (Figure 2)



If user selects The Memory Remains mode (Figure 3)

The activity diagrams above, shows how the system runs while playing. Activity diagram explains three paths after starting game:

First of there is three paths because game consists of three different modes;

Before starting playing, user selects the mode of the game. If players select Beat It as shown in the Figure 1, system leads users to a section where they can select the difficulty of the game. Difficulty of the game based on the number of cubes. After selecting the difficulty, user starts the game. After selection of the difficulty, a picture will load for the players to construct the figure by using cubes. Then users will start to play the game. While playing the game, user may have three different interactions with the game.

First interaction: Users can check time while playing. Aim of the game is to reach high score in a specific time. So visualizing the time amount is important. Also when game is over, system will lead users to main game menu where they can start to play game.

Second interaction: To win, users should combine the figures on the sides of cubes and get the picture on the card. So as an interaction, after pressing change button users will be able to rotate the cube to get the necessary figure to construct the picture.

Third interaction: While playing the game, players interact with the system by placing the cube. After they place cube, user press the put button. Then system checks if the figure is in the right place. If it is in the right place system, add points to their score. When all of the placing is done if they have time, a new picture loads. Otherwise system subtracts points from their score and they try to fix their cubes.

Activity diagram of Rolling Stones (Figure 2) resembles mode1 only two interactions of the system is different from Beat It. After users select the second mode, system leads players to the same procedure as above. They can select the difficulty of the game. They start to game and system provides them a picture. After they start game they have three interactions with game.

First interaction is the same with the Beat it, system provides user the remaining time and when time is over system leads user to main menu where they can select modes.

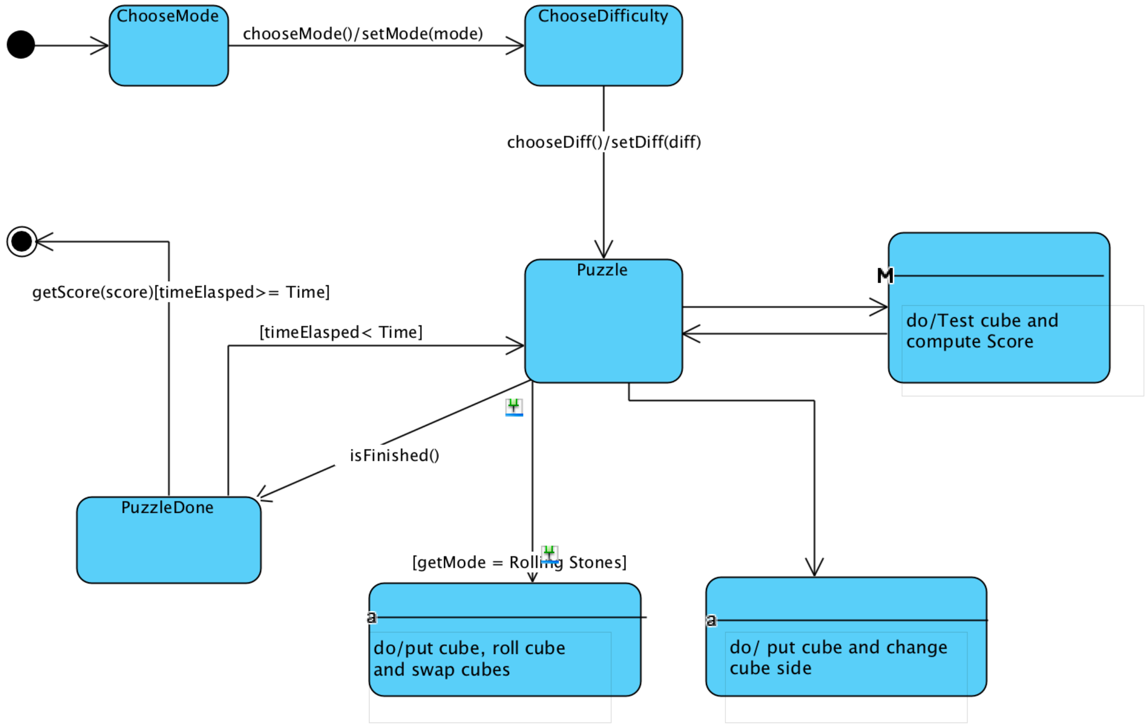
Second interaction: This modes difference from Beat It is, user will roll the cubes but cannot change their sides. So to get ideal sides user should roll the cubes instead of rotating cubes. So as an interaction there is roll button which rolls the unfixed cubes.

Third interaction: After first rolling user will fix the cubes by pressing put button. If the place of the figure is right than system will add points to users' score. Then users will decide if the figures on the cubes are enough to construct the picture. If it is not after putting some cubes, user will press roll button and system will lead user to the second interaction. If there is remaining time, then system will load new picture. If the place of the cubes is not right than system will subtract point from the users' score than player will continue to combine the cubes to construct last picture.

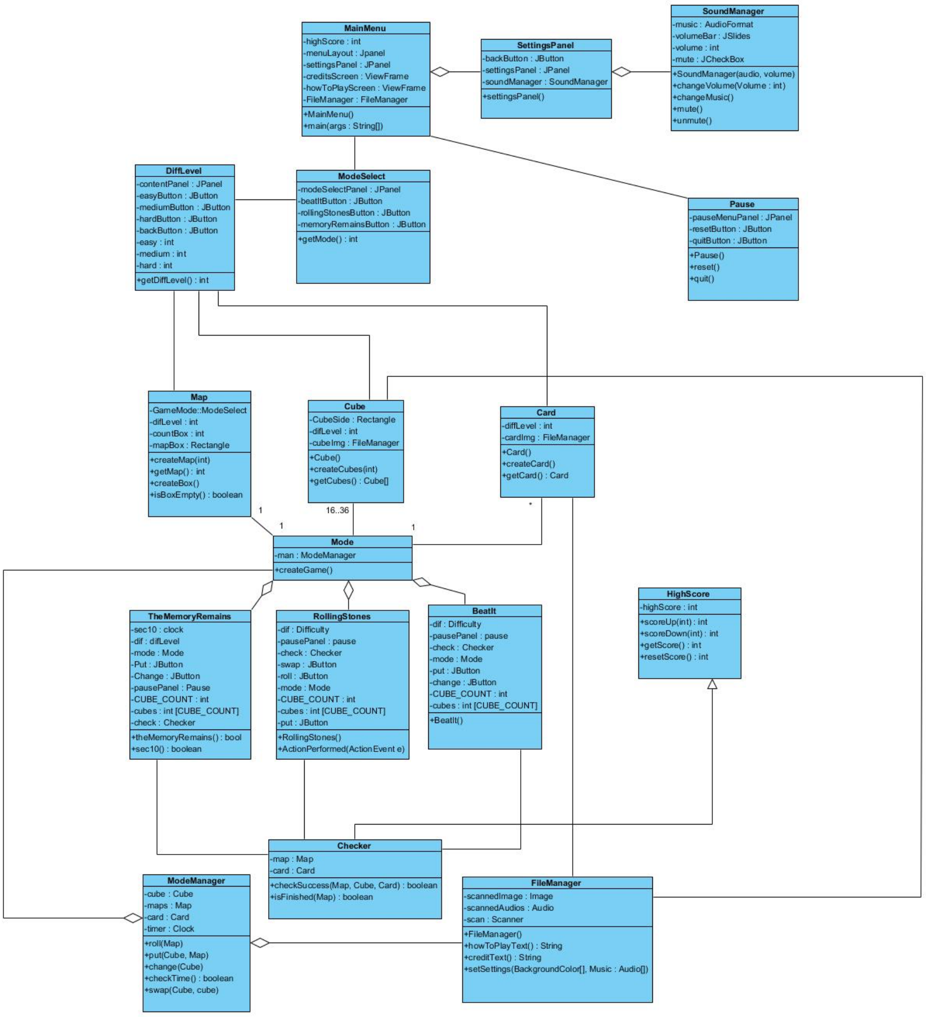
In the The Memory Remains, system procedure is same with the Figure 3. Difference of this mode from Beat It is visualization of the picture is limited. Player must memorize the picture than after a specific time, player could not see the picture and combine the cubes according to users' memory. So the interactions are same with the Beat It.

5.3 State Diagram

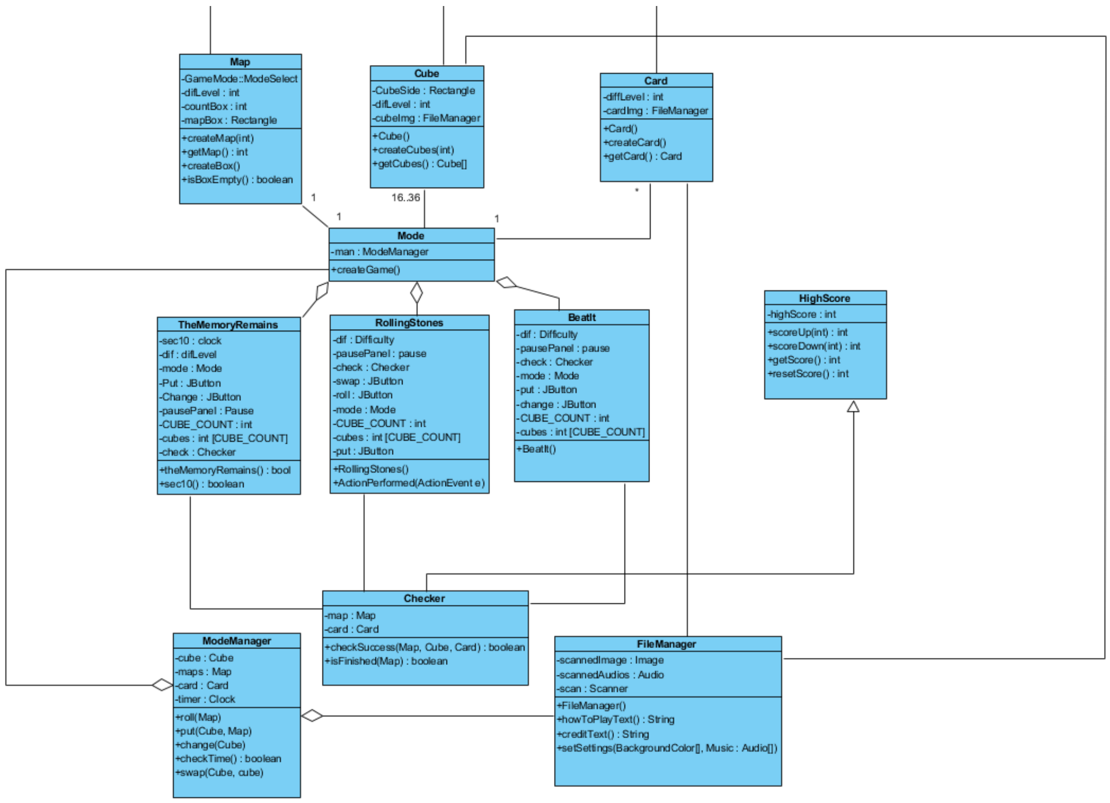
In the given state chart diagram, initial point to start the game is selecting mode and difficulty. After these options are chosen, program is in “Puzzle” state in which the whole game is played. It is connected to three nameless states, first one with activities:” do/put cube, roll cube and swap cubes” are the options for the Rolling Stones game mode. Second one with activities “do/ put cube and change cube side” are options for the other two game modes (The Memory Remains and Beat it). Final nameless state is for checking cube movement and incrementing score or decreasing remaining time depending on the move of cube. After puzzle state is finished, if the time remaining is greater than zero, program goes back to puzzle state and so on until the time is up. When time is up, program stop executing and displays high-score.


ChooseMode
chooseMode()/setMode(mode)
ChooseDifficulty
chooseDiff()/setDiff(diff)
getScore(score)[timeElasped>= Time]
[timeElasped< Time]
Puzzle
M
do/Test cube and compute Score
isFinished()
PuzzleDone
[getMode = Rolling Stones]
a
do/put cube, roll cube and swap cubes
a
do/ put cube and change cube side

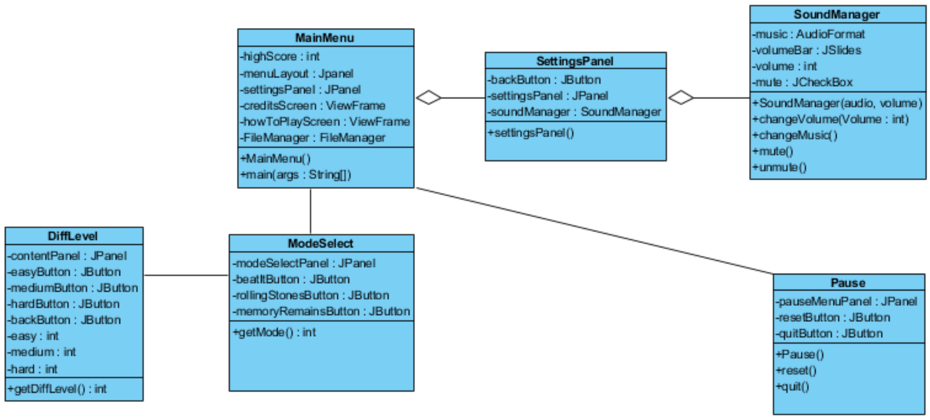
5.4 Object and Class Model



Full Object Model



Detailed Object Diagram 1



Detailed Object Diagram 2

The class diagram of Q-bitz as shown in the figure. We have 17 classes in the current situation.
When user enters the game, there is main menu class, which user can choose four options. Play game, How to Play, Settings and Credits. Settings have 2 specifications which user can change sound and change the background colour. If user chooses Play Game, there is ModeSelect class, which user can choose one of the three modes, and DifLevel class that user can choose difficulty level of the selected mode, which are easy, medium and hard. After the difficulty level selection, there are four classes before the game starts.

Map Class: Basically, map is the area that player arrange cubes. In this class, map is created according to selected mode, and selected difficulty level.
Except Rolling Stones, map is created empty, but in Rolling Stones, map is created with cubes with random faces. Also, amount of the area is specied with the difficulty level.

Card Class: In this class, card is created when game starts and every successful finish of the previous card. It is created according to random order of cubes faces.

Cube class: In this class, cubes are created according to specific amount of selected difficulty level. Each cube has different image in their surfaces, but all cubes are identical.

Mode Manager class: Mode manager class is the essential class of the all modes. In this class, the design of the map, card and cubes will be generated according to selected modes. Put, roll, swap, change commands will be implented. Also, checkTimer is a common specification for all modes, which will be also implented in this class.
1. Put is the command that put cube to the selected area of the map, after player decided the face of the cube.
2. Roll and swap are the commands for Rolling Stones Mode. In roll command, after

map is fullfilled with cubes random faces, player rolls all cubes except placed ones. In swap command, player swaps two cubes in the map.
3. Change command is the command for Beat It and the Memory Remains Modes. In this command, player changes the face of the specific cube.
4. checkTimer checks the time whether it is greater than zero or not. If it is less than zero, time is up and game is over.

Beat It, the Rolling Stones and the Memory Remains: These are different modes of our game. Each mode will be implemented in its own class, using the commands that implemented in Mode Manager class. Player plays the game in these classes.

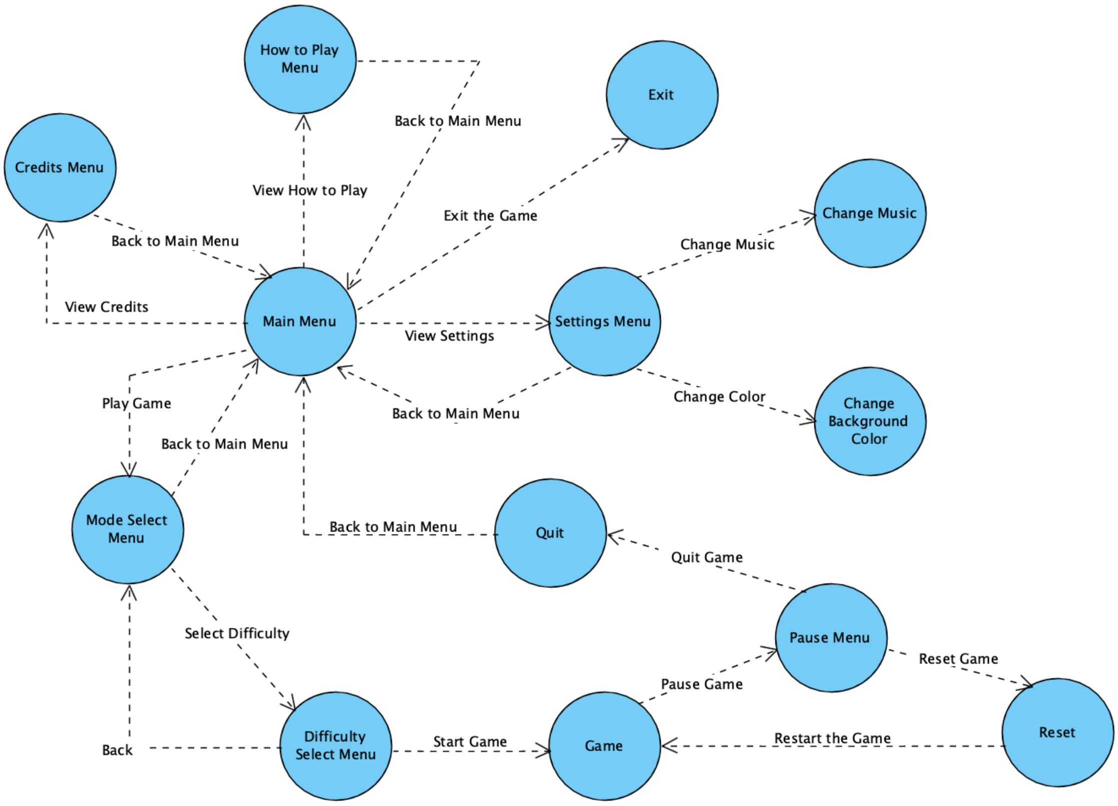
Pause Class: Pause is the user interface menu when player wants to pause the game. It has two specification, first one is reset the game and the other one is quit the game and returns to the main menu.

Check Success: After player puts the cube to specific area of the map, this class checks if player put the cube to the accurate area of the map or not. Also, checker classs has isFinished method that checks if player finish the card or not.

High Score Class: If player successfully put the cube, players score will be up. If not, players score will be down. Also, high score is kept in this class.

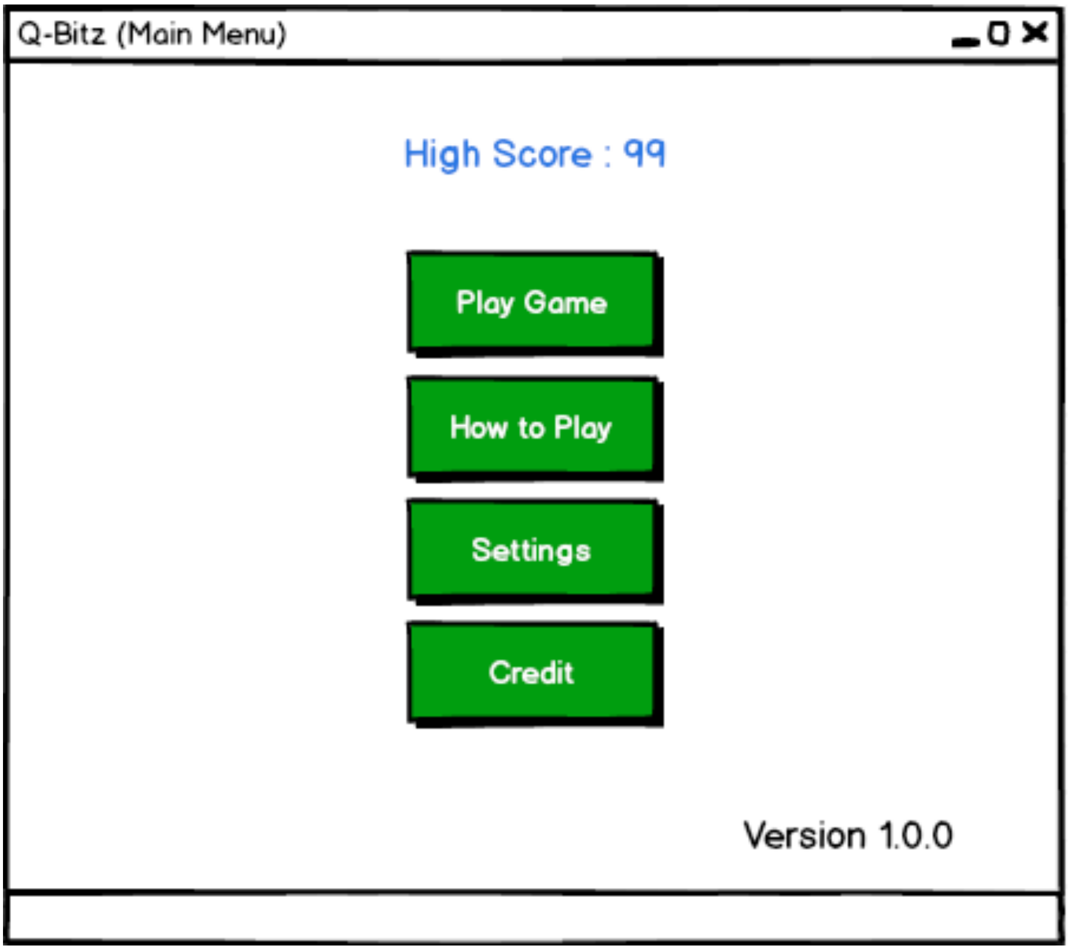
5.5 User Interface - Navigational path & Screen Mock-ups

5.5.1 Navigational Path



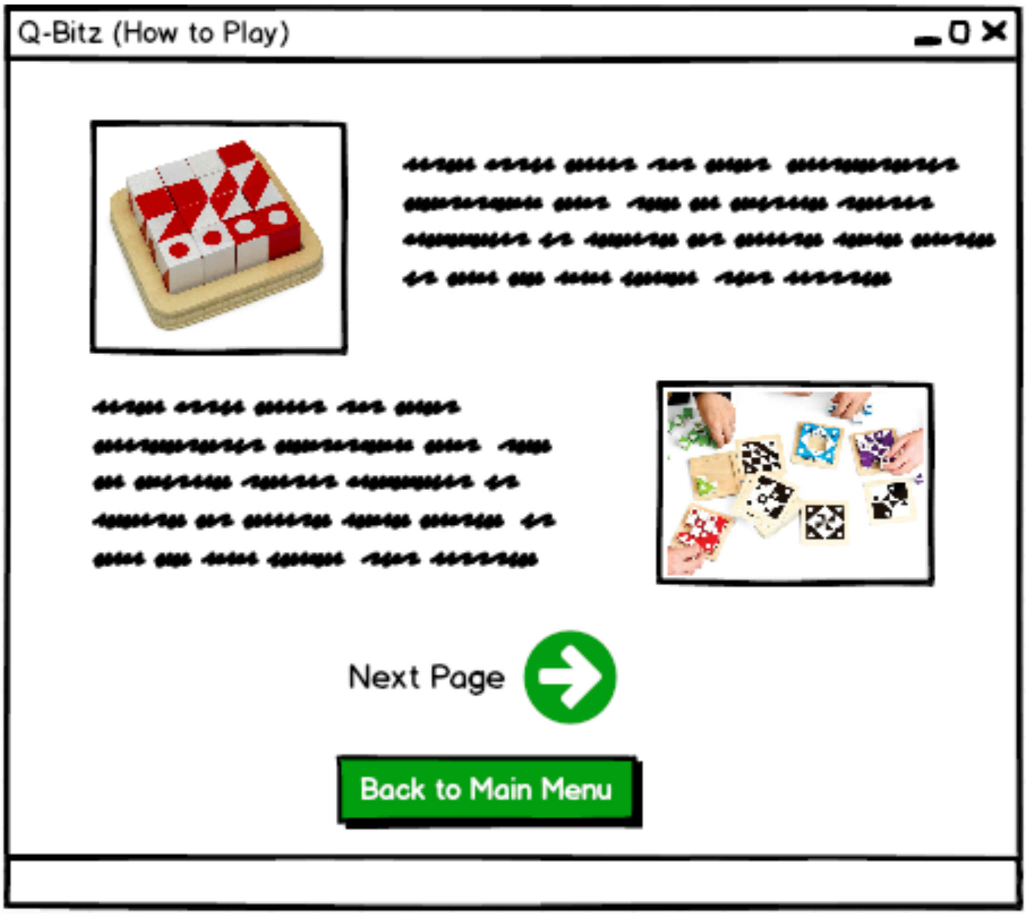
5.1 Screen Mock-ups

5.1.1 Main Menu



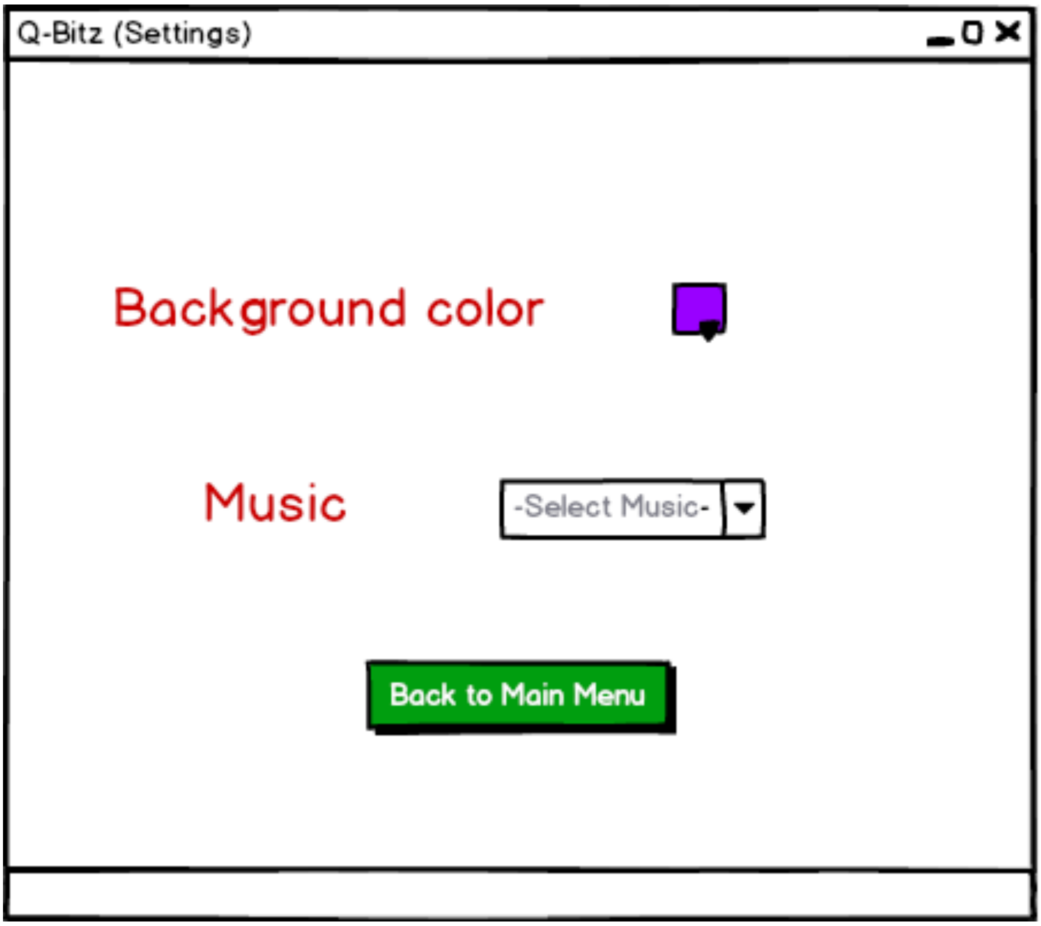
In the main menu screen there are 4 buttons and 1 high score counter. First button is “Play Game” button which redirects the user to the "mode selection" screen. Other buttons are “How to Play”, “Settings” and “Credit”. High score counter shows the highest score that user made before. In the lower right corner, user can see which version of the game that he/she is using.

5.1.2 How to Play



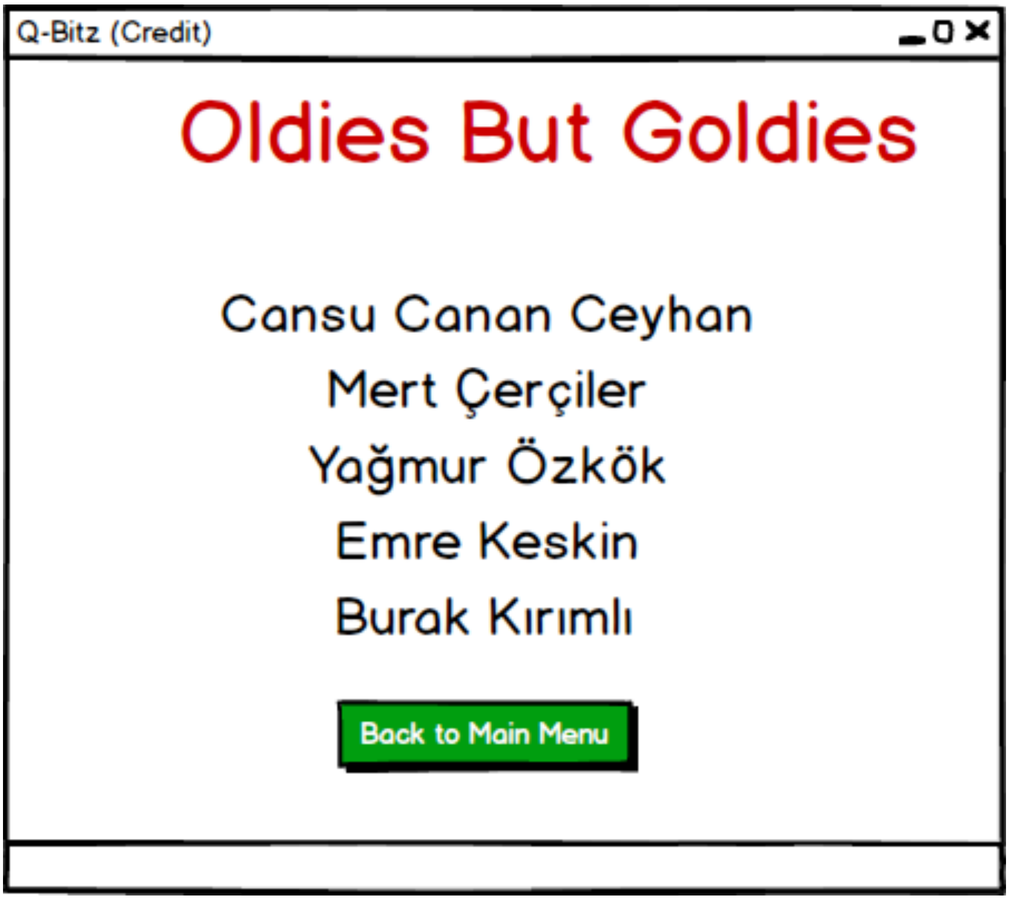
“How to Play” screen gives the user some information about the game. In addition to that, there are some pictures about game. In “How to Play” screen user can learn about game modes and functions of buttons. There is “Next Page” button in this page which allows the user to move to the other page. “Back to Main Menu” button helps user to move back to “Main Menu” screen.

5.1.3 Settings



“Settings” screen helps the user to change the background color of the game. With using “Settings” screen user can change the game music. “Back to Main Menu” button helps user to move back to “Main Menu” screen.

5.1.4 Credit



Oldies But Goldies

Cansu Canan Ceyhan

Mert Çerçiler

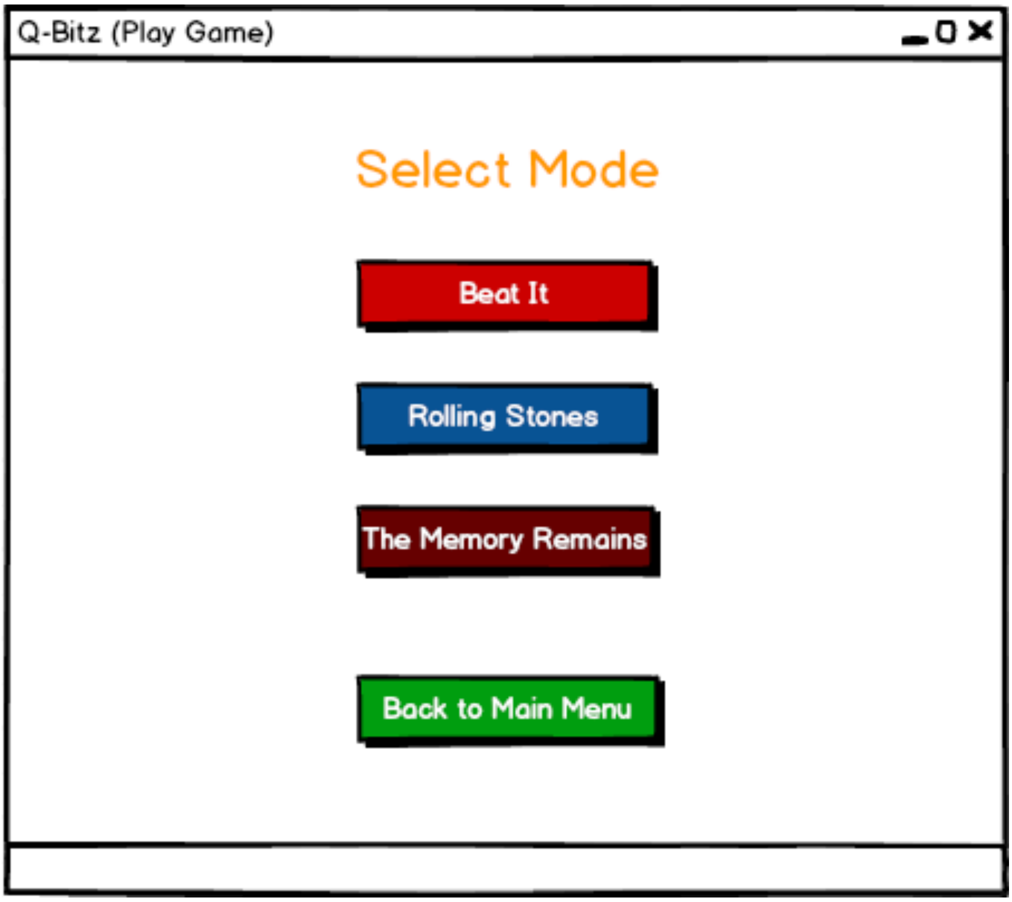
Yağmur Özkök

Emre Keskin

Burak Kırımlı

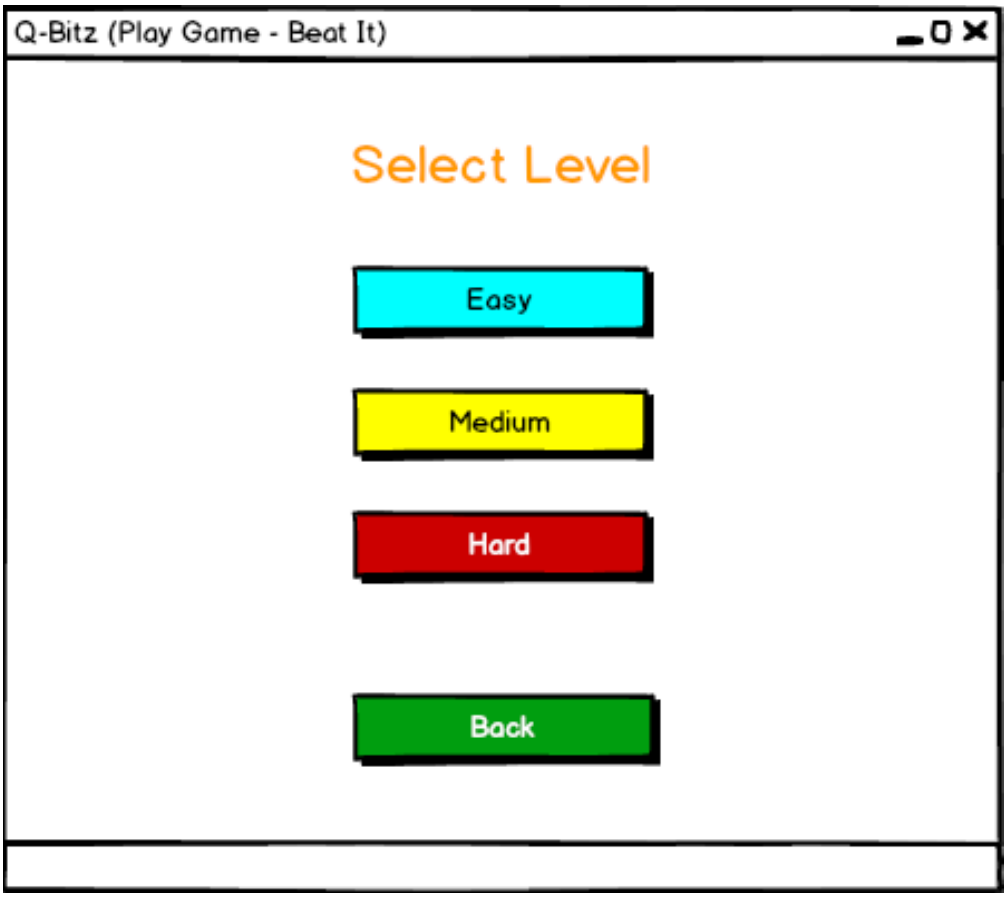
In “Credit” screen, user can see the names of the developers of the game. “Back to Main Menu” button helps user to move back to “Main Menu” screen.

5.1.5 Play Game – Select Mode



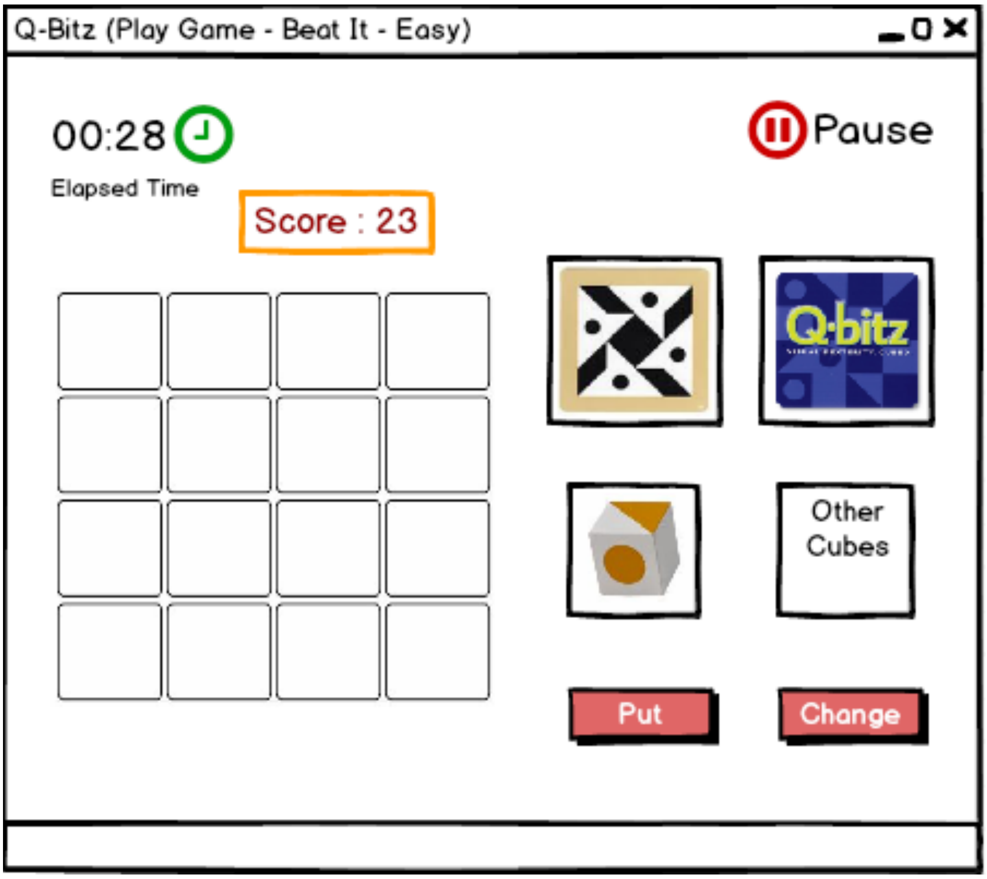
If user presses the button “Play Game” in “Main Menu” screen, the “Select Mode” screen will be opened. In “Select Mode” screen, there are 4 buttons which are “Beat It”, “Rolling Stones”, “The Memory Remains” and “Back to Main Menu”. “Beat It”, “Rolling Stones”, “The Memory Remains” are the names of game modes. If user selects one of them, “Select Level” screen will be opened. “Back to Main Menu” button helps user to move back to “Main Menu” screen.

5.1.6 Play Game – Select Mode – Select Level



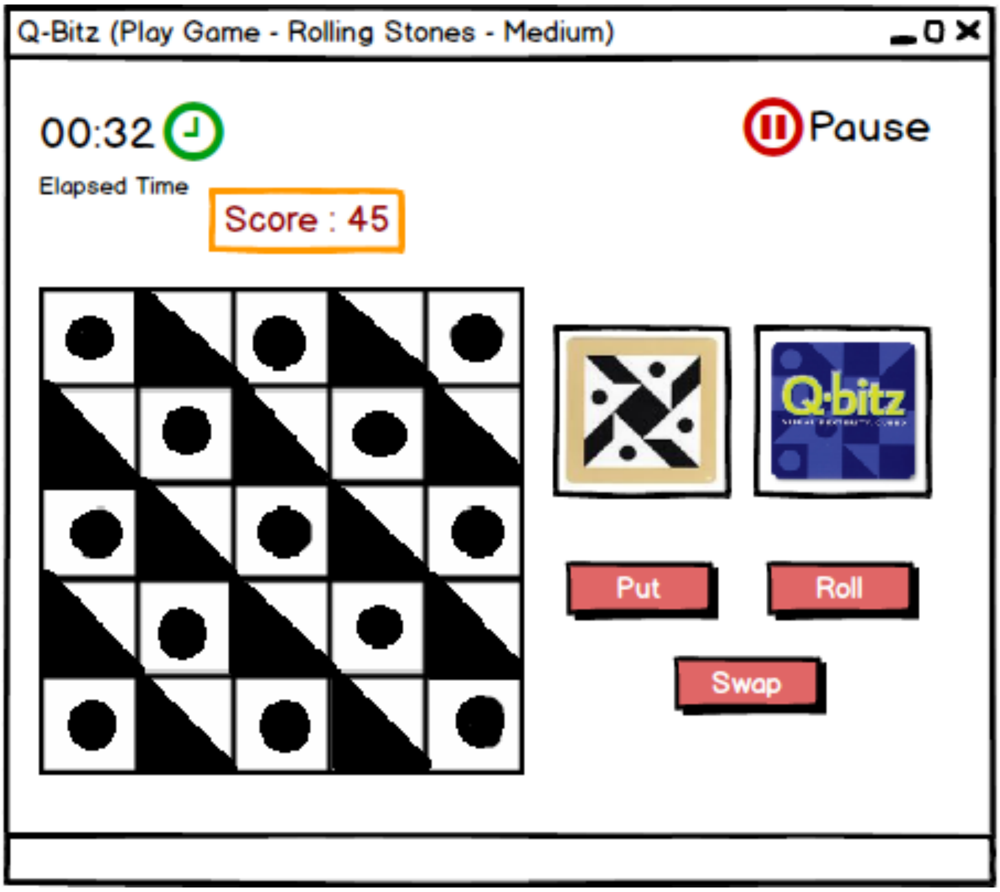
After user selected the game mode, “Select Level” screen will be opened. In this screen there are 3 level (difficulty) choices. In “Easy” level there are 16 cubes, in “Medium” level there are 25 cubes and in “Hard” level there are 36 cubes. All cubes are identical. “Back ” button helps user to move back to “Select Mode” screen.

5.1.7 Beat It



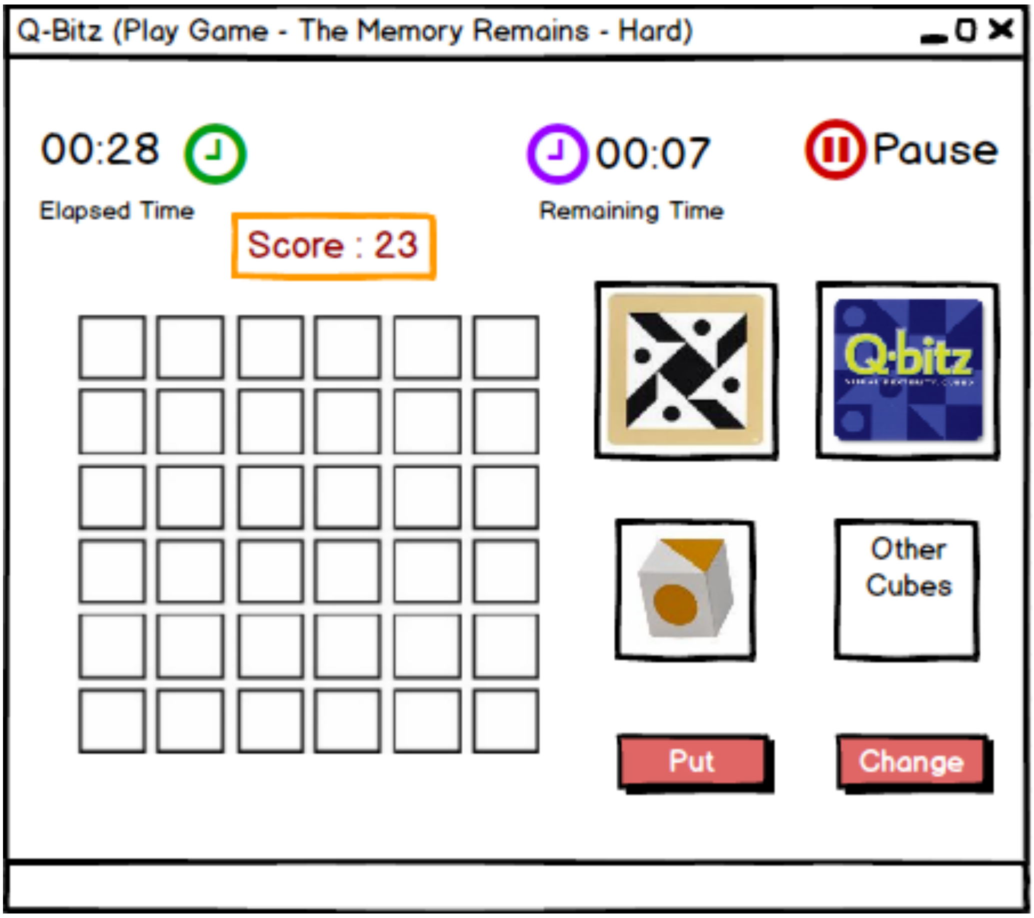
If user selects “Beat It” mode and “Easy” level, this game screen opens. In the top left corner, there is a timer that calculates the elapsed time. In the top right corner, there is a “Pause” button in order to stop the game. In the top there is a scoreboard that counts the user’s points. If user puts the correct cube to correct square, score is going to be increased and if user puts the wrong cube to wrong square, score is going to be decreased. In the left there are 16 squares which are waiting to get filled with correct cubes. We call these 4x4 area as a “Map”. In the right there are many cards, one of them is opened and others are closed. User looks opened card and tries to fill the “Map” like the opened card. Below the cards, there are cubes that user uses to fill the “Map”. If user presses the “Change” button, side of cube will change and if user clicks the one of the squares and then presses the “Put” button, the selected cube is going to placed there.

5.1.8 Rolling Stones



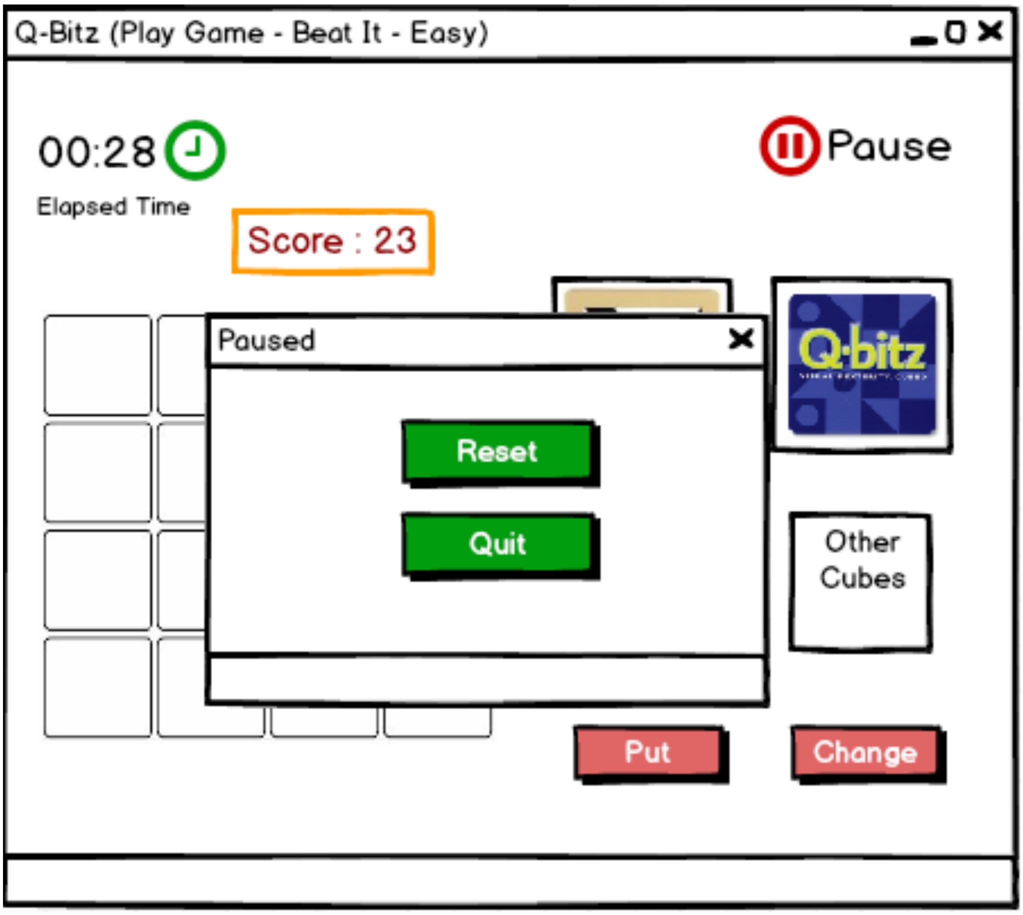
If user selects “Rolling Stones” mode and “Medium” level, this game screen opens. In the top left corner, there is a counter that calculates the elapsed time. In the top right corner, there is a “Pause” button in order to stop the game. In the top there is a scoreboard that counts the user’s points. In the left there are 25 squares which are already filled with random cubes. In the right there are many cards, one of them is opened and others are closed. If user presses the “Roll” button, the patterns of the cubes on the map changes randomly. If user presses the “Put” button and clicks one of the cubes on the map, this cube will be fixed on that location. If user presses “swap” and clicks two cubes, cubes swap. After those actions, if one cube located correctly, score is going to be increased and if one cube located incorrectly, score is going to be decreased.

5.1.9 The Memory Remains



If user selects “The Memory Remains” mode and “Hard” level, this game screen opens. In the top left corner, there is a counter that calculates the elapsed time. In the top right corner, there is a “Pause” button in order to stop the game. In the top there is a scoreboard that counts the user’s points. In the left there are 36 squares which are waiting to get filled with correct cubes. In the top, there is a timer that counts backwards from 10 seconds. User can only look opened card for 10 seconds then card will be closed again. . If user puts the correct cube to correct square, score is going to be increased and if user puts the wrong cube to wrong square, score is going to be decreased. ”. If user presses the “Change” button, side of cube will change and if user presses the “Put” button and clicks the one of the squares, the selected cube is going to placed there.

5.1.10 Pause Menu



While playing the game, if user presses the “Pause” button which is on the top right corner, “Pause Menu” opens. In “Pause Menu” there are two buttons which are “Reset” and “Quit”. If user presses “Reset” button, elapsed time, score, opened card and selected cube is going to be reset and game starts again on the same screen. If user presses “Quit” button, “Main Menu” screen opens. If user presses the “cross” at the top right corner, the game resumes.

6. Conclusion

To sum up what is stated above, our project will basically be a desktop application of the table game “Q-bitz.” Before starting implementing the actual game or writing this analysis report, as a group we meet and discuss several issues regarding our version of the game. First, we agree upon some additions to the game hoping it will make the game more desirable. Then we decide to use Java platform during implementation and research about libraries that can be beneficial to us. Finally, we think of ways to manage our classes and objects so that not only our implementation process will be easier but also our program will be more efficient.

In the end, we come up with this analysis report hoping it will be helpful to us during future reports and actually implementing the game.

7. Improvement Summary

For the second iteration, we added following things:

1. We improve the introduction.
2. Non-functional requirement rewritten.
3. Additional requirements added.
    a. Bonus
    b. Combo
    c. Animations
4. Class diagram fixed. 1 to 1 and 1 to many relations added. Some classes deleted. New classes, attributes and methods added. Divided into subsystems.
5. We redraw the use case diagram. Buttons used in the game added to diagram. Mode types deleted and functions in every mode added.
6. We redraw activity diagram. Dead ends eliminated. End case added to diagram. Relations in the activity diagram changed.
7. Explanations of mock ups corrected.

8. Navigational Path corrected. Dead ends eliminated.

9. State Diagram added.

8. Glossary & References

[1] https://egitimdizayn.com/q-bitz

[2] https://www.mindware.orientaltrading.com/q-bitz-a2-44002.fltr