
Bilkent University

Department of Computer Engineering

# CS 319 - Object-Oriented Software Engineering Project

# IQ Puzzler

## Analysis Report

# Group Name: Violet

# Group Members

Ege Akın
Safa Aşkın
Betim Doğan
Alp Ertürk
Elif Gülşah Kaşdoğan

# 1 Introduction

IQ Puzzler Pro is a brain-teasing board game for children and adults which support the development of cognitive skills. It is a puzzle which has geometric pieces of different sizes and colors; the main purpose is correctly placing these pieces to board or creating a specific shape with the given pieces. Game starts with 2D which corresponds to filling the board and continues with 3D shapes.

For our course project, we will implement a different version of this game. IQ Puzzler is a desktop game which allows users to enjoy both the experience of classical board game and new features. Since IQ Puzzler is a suitable game for object oriented design we were able to add some additional features as we desire. In this report we analyzed the system to develop to implement a fully functioning, well designed game. We designed system models to express the design of our game from different perspectives. We tried to make our class and object models as clear as possible so that we can implement the game by using object oriented principles properly.

We will use Java platform to implement this game. We are planning to use Libgdx or Slick 2D for gaming framework, JavaFX for GUI design. We will apply object oriented design principles we will learn in CS319 during the project.

# 2 Overview

This section describes the features and functionality of our game in general. Components of original board game will be represented together with features we are planning to add to game.

## 2.1 Game Play

IQ Puzzler is a single player game. It has two and three dimensional versions. Player has some puzzle parts with different sizes and shapes and a

board to fill or a 3D pattern to satisfy to finish the game successfully. Pieces will be given such that there will be one unique solution for each level. Level is successfully finished if the given shape is created by given pieces. Players will obtain a score for each successful level. Time and count of moves are not important to finish the game but it will be important for score.

### 2.2 Score and Time

We will add timer and score a system based to help users see their improvement also they can play this game in a competitive manner thanks to this feature. Score will be calculated depending on time spent on finishing the level and number of moves made.

### 2.3 Puzzle Pieces

Pieces are shaped as 4 to 6 balls are connected with different combinations. Each type of piece has a specific color so it can be identified based on color or shape independently. Their structure allows them to form expected 3D shapes.

### 2.4 Board

Pieces will be inserted to the board until board is completely full for 2D. For 3Dversion, board will be used to form the base level of the 3D object.

### 2.5 3D shapes

Expected 3D shape will be given to player and player will try to obtain this shape by using combination of these pieces.

### 2.6 Settings

User will be able to alter the board or color of background. Also user will be able to change the color of pieces

# 3 Functional Requirements

Functional requirements define the functions of our game where functions can be expressed as interaction between user and system, also operations between inputs and outputs. Functional requirements can involve technical details and functionalities that a system need to complete successfully. Our functional requirements are divided into 5 sections as below.

## 3.1 New Game

There will be 2 game modes which are 2D game mode and 3D game mode. If user starts a new game, if there exists any saved game in current mode (2d or 3d), the saved game in current mode will be destroyed. So user will only be able to save 1 game for 2d game mode and 1 game for 3d game mode.

### 3.1.1 2D Game Mode

When user log in to the game, user will see a menu which is composed of play game, load game, leaderboard and tutorial, if user clicks the button new game or load game there, user will see game mode selection screen. If user clicks 2D game mode, user will see the initialized board and pieces. In this game mode the difficulty of levels will be increased. The maps of all levels are generated by the system. The score of user will be increased if user passes the levels and after that there will be a leaderboard according to these scores.

### 3.1.2 3D Game Mode

If user clicks 3D game mode in game mode selection screen 3D game will be initialized. User will be able to see the board and puzzle from various perspectives. Calculating the user's score in 3D game mode will be different than 2D since it is harder than 2D game mode. User will get more points in a

3D level than 2D level. The difficulty of levels in 3D game mode will also increase.

### 3.2 Tutorial

Player will see the tutorial button in the main menu when user login. Tutorial will help user about the buttons and how to play the game. Tutorial will also give a video description which informs the user about game more.

### 3.3 Load Game

Player will be able to load game that user saved before. Load Game button is in the main menu. When user pushes the load game button user will get a 2d or 3d game mode selection window to choose which game mode that user will load from. If user pushes load game button user will see the successfully passed levels and count of stars of a level. There are maximum 3 stars in each level. If user gets 3 stars from one level, it means that user finished puzzle very quick. And if user gets 1 star it means that user finished puzzle scarcely. So in load game feature user will be able to play passed levels again and for example user can increase the count of stars of a level 2 to 3. Load game will also stores the last unfinished puzzle and user will be able to see put pieces in last level that is played.

### 3.4 Sign In

When user starts the game, firstly user will see log in screen. If user had an account user will be able to enter the username and password and immediately start game. In sign in screen, there will be check for validity of username and password. If user does not have an account, game system will not allow user to start game immediately.

### 3.5 Sign Up

When user starts the game, in initial screen there will be a sign up button. If user does not have an account user will pick the sign up button for creating a new account to play the game. In sign up screen, user will create a new username and password. Game system will check if username is taken before, after a successful sign up user will be able to start the game.

## 4 Nonfunctional Requirements

Nonfunctional Requirements are basically define how a system is supposed to be. It specifies criteria that can be used to analyze the features of the system. In our game our three main non-functional requirements are game performance, usability and extendibility. And these three non-functional requirements reflects the quality attributes of our game.

### 4.1 Game Performance

Due to make a game which works in high performance we are planning to have components which will not need much system requirement. In order to increase the game performance we stay away from using high resolution graphics. We are aiming that our game will not allocate more than 1GB memory in RAM and 100MB memory in hard disk.

### 4.2 Usability

The usability of game is user interface is one of the most important points of a game because users have their first interaction with user interface rather than game play or implementation. To provide usability and improve user experience we are planning to use at least 13pt character size in screens. In addition to that anyone whether experienced our game or not can start the game and put his first piece into game board within the 10 secs. The javaFX screen passes will be used to advance visuality.

## 4.3 Extendibility

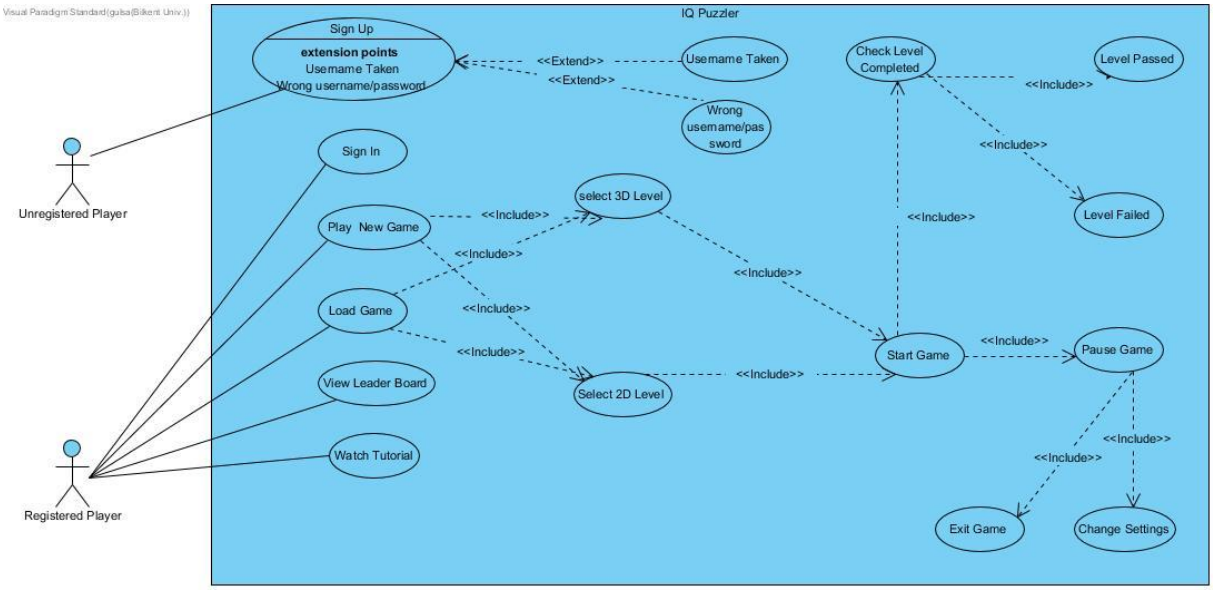
In order to ease the extendibility of our game after and during the development period we are planning to use object oriented principles strictly. Abstraction, encapsulation and inheritance are the main principles that will be used.

# 5 System Models

System models are constructed for abstraction of our game, each model given in this section makes this abstraction from a different perspective. Models are developed to communicate with people who are not familiar with system; it also helps us to develop our game better in terms of software engineering principles.

## 5.1 Use Case Model

Use case diagram represents the functionality of the system from user's view. Use case descriptions explain details of the use case diagram given below.



## Use Case Descriptions

### 5.1.1 Sign Up

**Use Case Name:** Sign Up

**Participating Actor:** Unregistered Player

**Stakeholders and Interests:** Player may create an user ID for saving his/her scores.

**Pre-condition:** User must be in the main screen.

**Post-condition:** -

**Entry condition:** Player clicks on sign up button in main screen.

**Exit condition:**

Player clicks on back button in sign up screen.

**Main flows of events:**

Player chooses an appropriate user name.

Player chooses an appropriate password.

System saves the user name and password and creates the new user ID.

Player is directed into main screen.

**Alternative flows of events:**

Player selects a username that is taken before and gets error.

Player selects an inappropriate password and gets error.

### 5.1.2 Sign In

**Use Case Name:** Sign In

**Participating Actor:** Registered Player

**Stakeholders and Interests:** Player may want to sign in to play the game with his/her account.

**Pre-condition:**

User must be in the main screen.

User must have an account he/she created before.

**Post-condition:** -

**Entry condition:** Player clicks on sign in button in main screen.

**Exit condition:**

Player clicks on back button in sign in screen.

**Main flows of events:**

Player enters the true username and password.

System checks whether the username and the password matches or not.

Player is directed into main screen.

**Alternative flows of events:**

Player enters the wrong username and password combination and system directs he/she to an empty sign in screen.

### 5.1.3 Play New Game

**Use Case Name:** Play New Game

**Participating Actor:** Registered Player

**Stakeholders and Interests:**

Player may want to create a new iq puzzler game.

System creates the game and starts the game.

**Pre-condition:**

User must be in the main menu.

**Post-conditions:** -

**Entry conditions:** Player should choose the game mode (2D/3D).

**Exit condition:**

Player clicks on back button in game screen, OR

Player exceeds the determined game time, OR

Player wins all of the levels.

**Main flows of events:**

Player enters a new nickname and starts the game.

System loads default settings and starts the game.

Player places the pieces at unique spots correctly.

Player completes the one of the levels.

The obtained score is demonstrated on screen at the end of game.

**Alternative flows of events:**

Player does not place the pieces correctly.

Player exceeds the time and game is over.

Player gets bored and may want to close the game.

Player presses the “Pause” button in game screen.

System displays pause menu.

Player presses “Exit Game” button.

System asks verification.

Player presses “Yes” button.

The game instructions are loaded and whenever the user loads the game,

player continues from the position where he/she left.

System exits the game.

### 5.1.4 Pause Game

**Use Case Name:** Pause Game

**Participating Actor:** Registered Player

**Stakeholders and Interests:** Player may want to pause the game.

**Pre-condition:** User must be playing the game.

**Post-condition:** -

**Entry condition:** Player clicks the “Pause” button in game screen.

**Exit condition:**

Player clicks on “Continue” button in game screen.

**Main flows of events:**

Player clicks the “Pause” button in the game screen.

System displays pause menu.

**Alternative flows of events:**

If the player wants to continue the game,

Player presses “Continue” button in pause menu

Game continues.

### 5.1.5 Load Game

**Use Case Name:** Load Game

**Participating Actor:** Registered Player

**Stakeholders and Interests:** Player may want to continue previous uncompleted game.

**Pre-condition:** User must have record that he/she plays this game before and not completed yet.

**Post-condition:** -

**Entry condition:** Player clicks on load game button in main screen. Player should choose the game mode (2D/3D).

**Exit condition:**

Player clicks on “Back To Main Menu” button in 2D/3D mode selection screen.

**Main flows of events:**

Player presses “Load Game” button.

System displays the load game selection screen.

Player chooses the 2D or 3D mode in mode selection screen.

Player starts the game in recorded level.

System loads default settings and starts the game.

Player places the pieces at unique spots correctly.

Player completes the one of the levels.

The obtained score is demonstrated on screen at the end of game.

**Alternative flows of events:**

Player does not place the pieces correctly.

Player exceeds the time and game is over, game instructions are loaded again and waiting for next connection of player.

Player does not want to continue the loaded game.

Player presses the “Back To Main Menu” button.

System displays the main menu.

### 5.1.6 View Leader Board

**Use Case Name:** View Leader Board

**Participating Actor:** Registered Player

**Stakeholders and Interests:** Player may want to view the scoreboard and indicate competition between users.

**Pre-condition:** Player has to be in main menu.

**Post-condition:** -

**Entry condition:** Player clicks on “Leader Board” button in main menu.

**Exit condition:** Player clicks on back button in leaderboard screen.

**Main flows of events:**

Player presses “Load Game” button.

System displays the load game selection screen.

Player observes the scores of different users and the leaderboard of the game.

Player can search for a particular user name to see his/hers score.

**Alternative flows of events:**

Player may want to return to main menu.

Player clicks on “Back To Main Menu” button on screen.

System renders main menu screen.

### 5.1.7 Watch Tutorial

**Use Case Name:** Watch Tutorial

**Participating Actor:** Registered Player

**Stakeholders and Interests:** Player may want to watch the tutorial of the iq puzzler game.

**Pre-condition:**

Player has to be in main menu.

Player must pause the game.

**Post-condition:** -

**Entry condition:**

Player clicks on “Watch Tutorial” button in the main menu.

Player clicks on “Watch Tutorial” button in the pause menu

**Exit condition:** Player clicks on back button in watch tutorial screen.

**Main flows of events:**

Player presses “Watch Tutorial” button on main menu.

System displays tutorial.

Player watches the video tutorial and observes the gameplay.

**Alternative flows of events:**

Player may want to return to main menu.

Player clicks on “Back To Main Menu” button on screen.

System renders main menu screen.

System renders pause screen.

### 5.1.8 Change Settings

**Use Case Name:** Change Settings

**Participating Actor:** Registered Player

**Stakeholders and Interests:** Player may want to change settings of the game.

**Pre-condition:**

Player has to be in pause game screen.

Player must press “Change Settings” button in pause game screen.

**Post-condition:** -

**Entry condition:**

Player clicks on “Change Settings” button in the pause menu.

**Exit condition:** Player clicks on “Back” button in settings screen.

**Main flows of events:**

Player presses “Change Settings” button on pause menu.

System displays settings menu.

Player changes the settings.

**Alternative flows of events:**

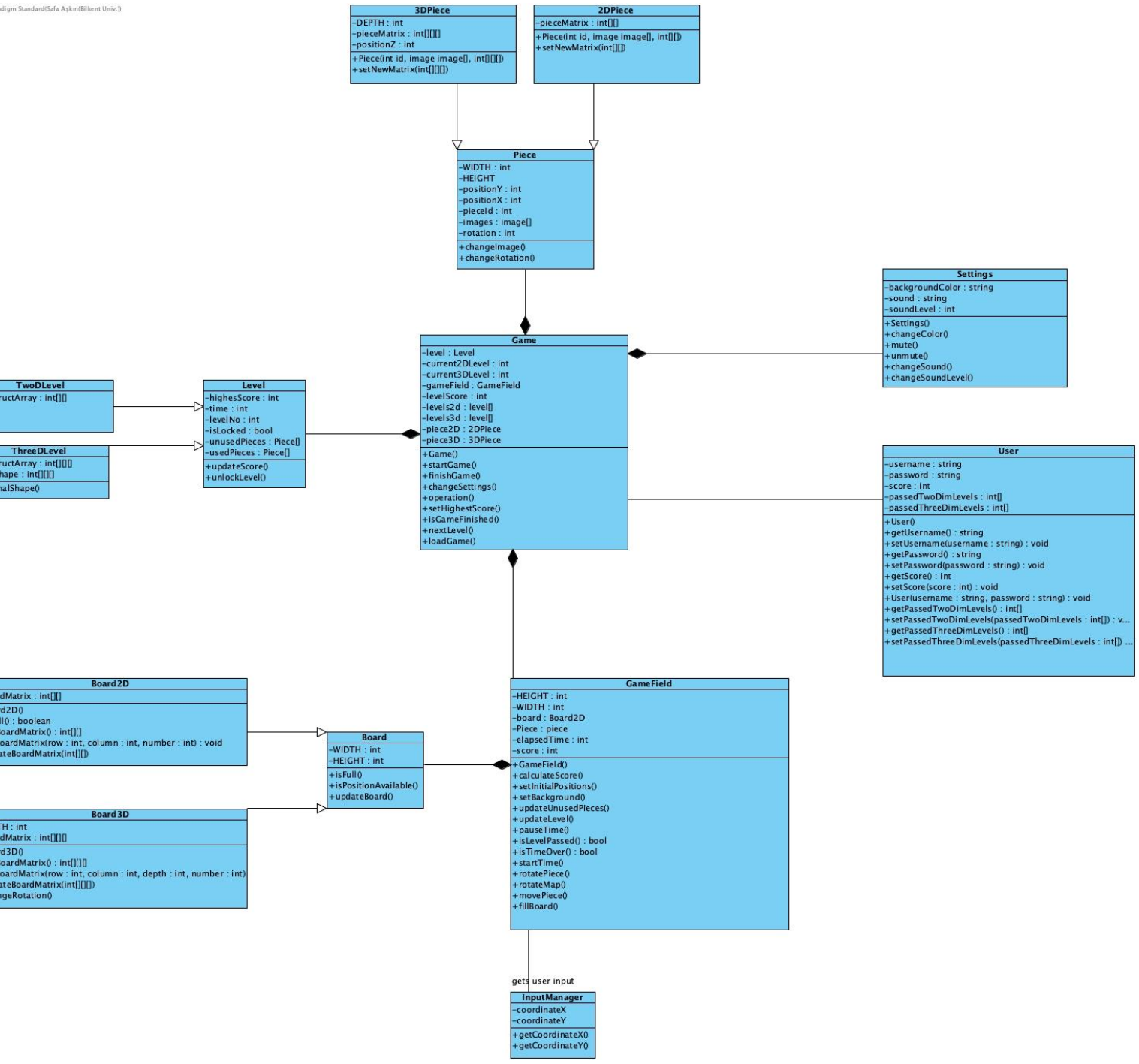
Player may want to return to pause menu.

Player clicks on “Back” button on screen.

System renders pause menu screen.

## 5.2 Object and Class Diagram

Class diagrams represent the structure of the system by showing the classes, their methods, attributes and relations between different classes.



### 5.2.1 Game Class:

Game Class arranges operations like starting or finishing a game, changing settings by accessing User Interface subsystem components, sets high score and loads a game previously played by accessing Database.

### 5.2.2 Piece Class:

Piece class represents individual pieces which will be inserted to board. Piece has id, position, rotation and size attributes, images can be attached to a piece. Determining if a piece is used or not will be done by a boolean attribute isUsed inside a piece object for simplicity. Methods allow us to change rotation of a piece and image attached.

### 5.2.3 GameField Class:

Game Field object has size attributes, a board object, a piece object, elapsed time for game set, current score of game and number of moves made so far. Game Field is the class which sets the game essential. Class organizes and updated the pieces board, time, rotations regularly. Basically each object manages its own data and Game Manager manages objects and all game will be set properly. Score will be calculated in this class and will be recorded to database through Game subsystem.

### 5.2.4 Board Class:

Board is object representing board which player will insert pieces; it is actually the main play field. It has size attributes and functions to check existence of available positions and whether the board is full or not.

### 5.2.5 User Class:

User class represents a user with a username and a valid password and arrays representing passed levels. For each game arrays and score will be updated as needed by provided methods. User’s data will be stored in database system.

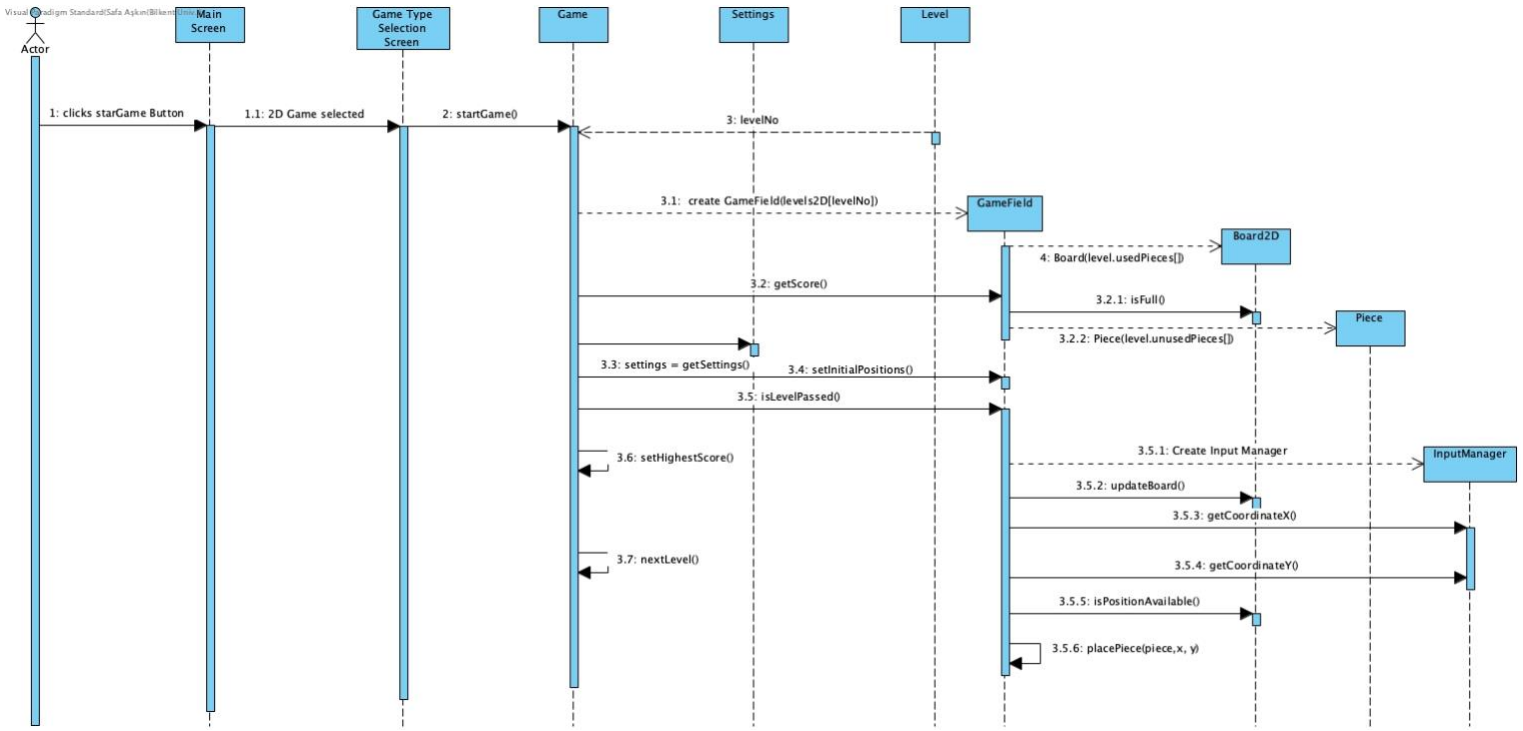
### 5.2.6 Level Class:

Level class is created to represent typical behavior of a level in our game. Each level has a highest score which will be kept as an attribute of each level object to easily compare if the highest score is reached or not by the current player without retrieving data from database for each game. Level can be either locked or unlocked; each level has an assigned time and array of used and unused pieces to determine needed pieces which will be set by Game Manager without extra operations. Level's highest score will be updated after each game, game will be unlocked if the prerequisites are satisfied and determining used and unused pieces will be done within level object.

## 5.3 Sequence Diagrams

Sequence Diagrams describe the dynamic behavior between different objects and system.

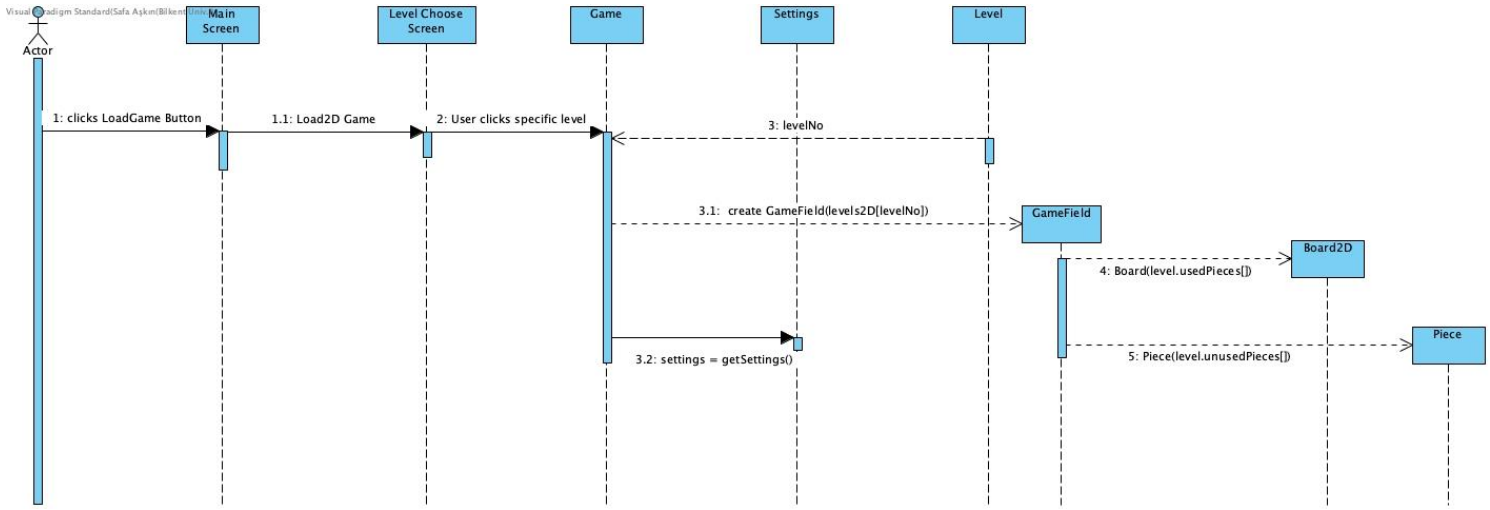
### 5.3.1 Play Game




**Scenario:** User plays the game

User wants to play the game. He enters the Menu and pushes the Start Game button. The Game class initializes all distinct Levels and Pieces. It gets the initial settings from Settings. After that it creates the GameField with proper Level. GameField initializes the game arena by using Level instance's usedPieces and unusedPieces arrays. After placing all Pieces and Board into proper position at scene the game starts. User takes pieces by clicking them and puts them to Board. Game field takes the x and y coordinates and decides which unused Piece is selected. GameField checks whether the game finished or not and it turns the score back to Game when games finishes.

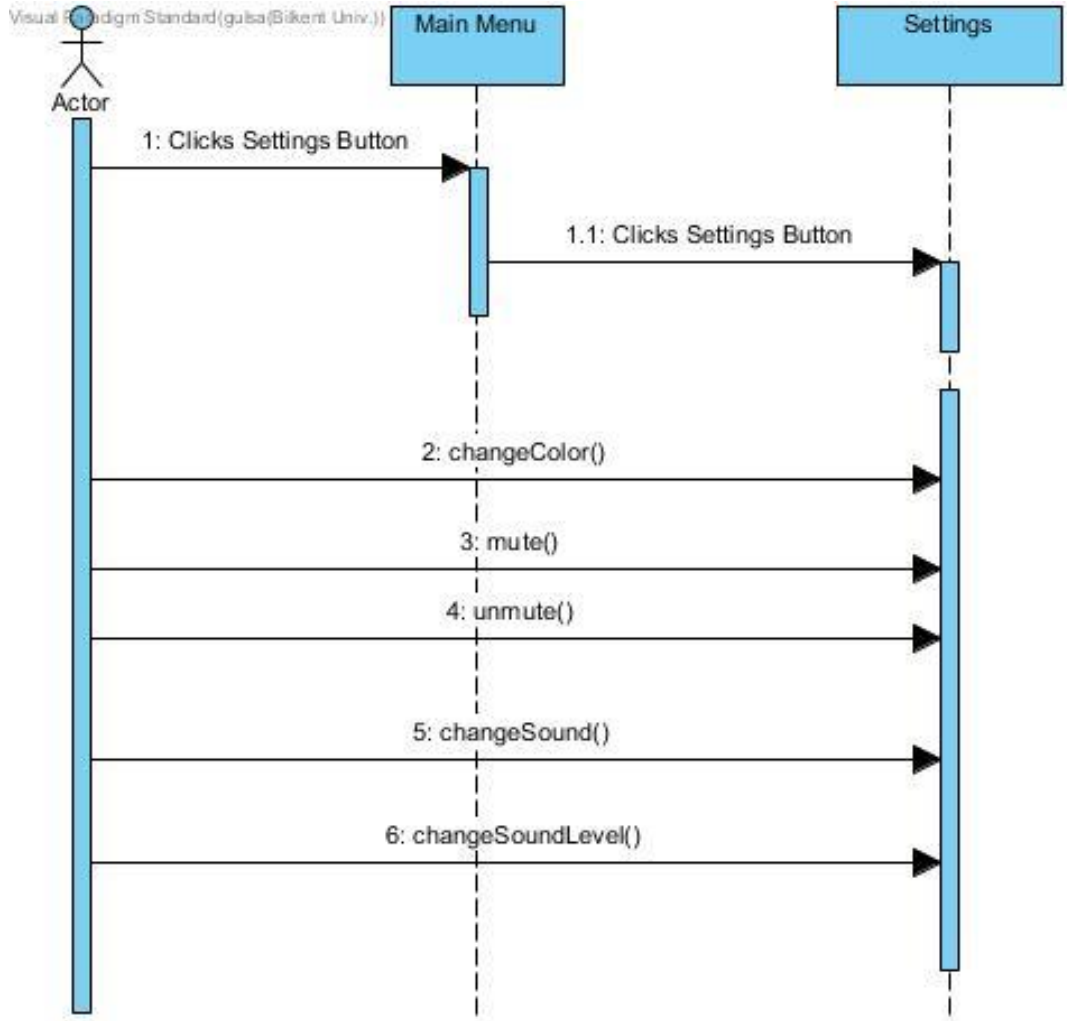
### 5.3.2 Load Game




**Scenario:** Users Loads a game previously played

When user wants to load a saved game, user presses the load button in main menu. When user presses the load game menu, system directs user to game mode selection menu. In this sub menu, system asks user whether user wants to load a 3d game or 2d game. According to his/her preference the level selection screen opens for given game mode and user picks the level that user wants to continue. After that, according to given level, Game class creates GameField object to load all views of current level. After game is loaded user can start to play game.

### 5.3.3 Settings




**Scenario:** User wants to change the background and sound

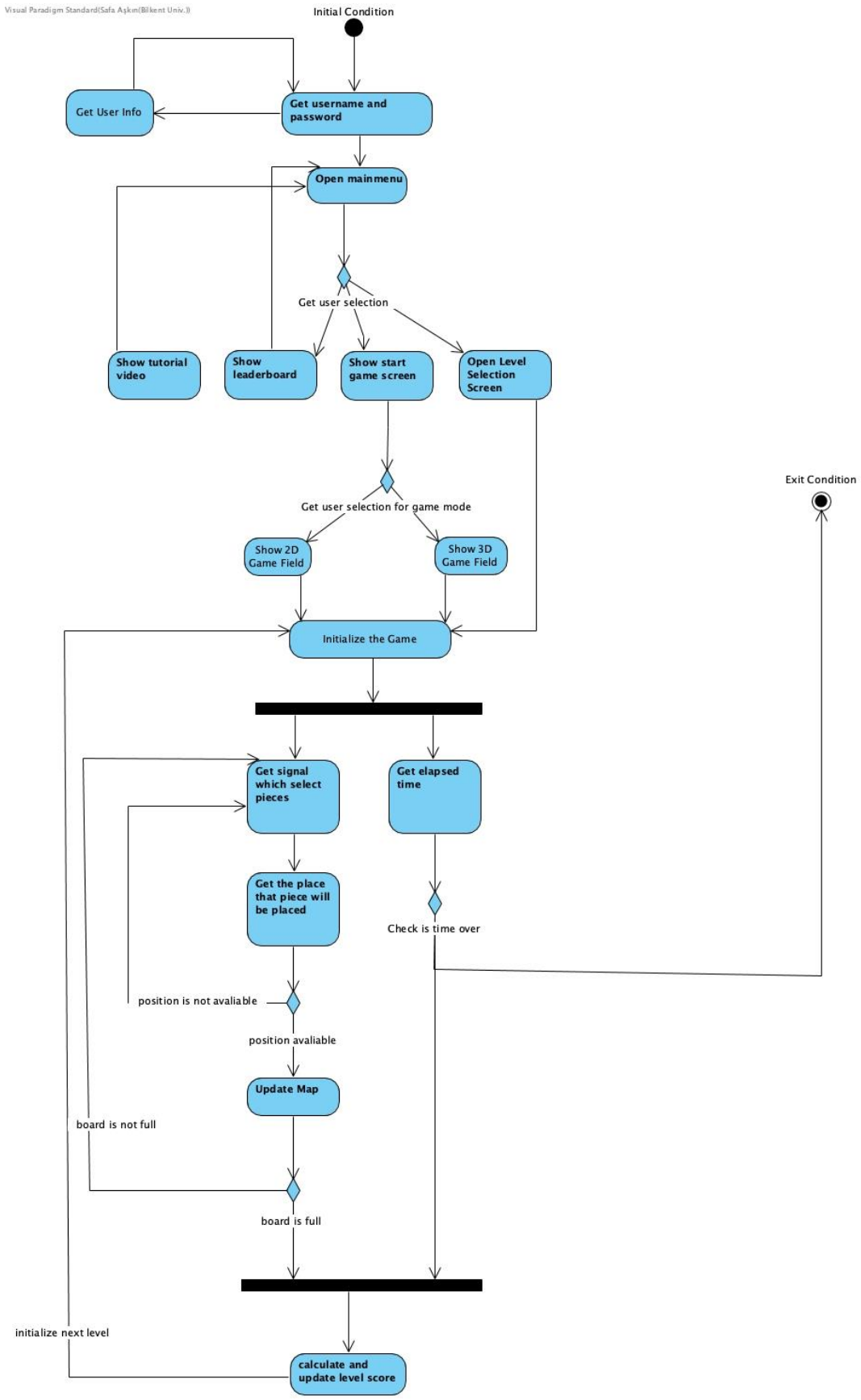
User wants to change the background color, sound level or sound itself. Settings class manages these alterations and provides a better experience to users.

## 5.4 Dynamic Diagrams

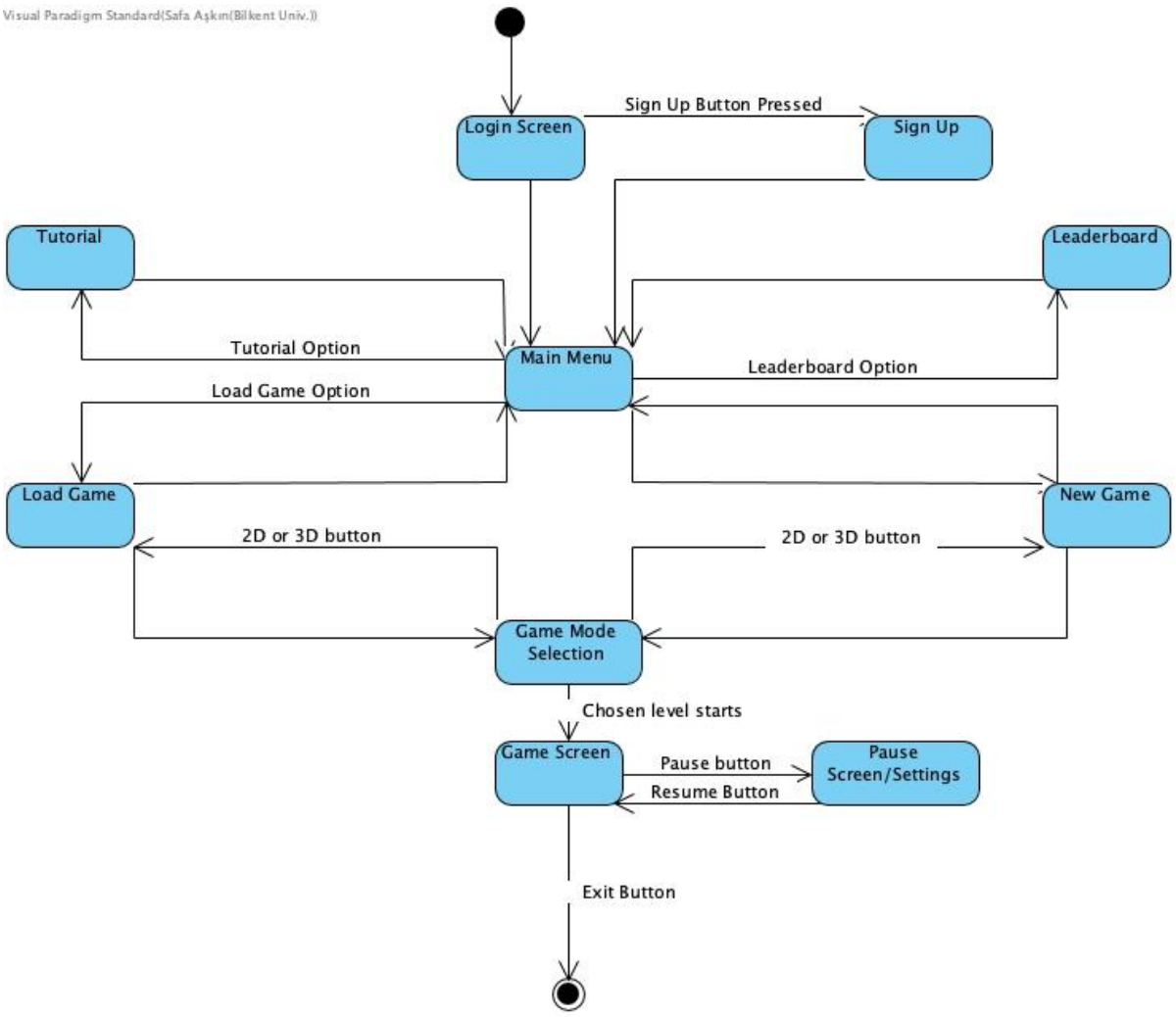
Dynamic diagrams are the diagrams that describe components of the system with interesting dynamic behavior. Generally they demonstrate interactions between objects. The purpose of dynamic diagrams are to detect and supply operations for the object model. Dynamic diagrams are basically state diagrams, sequence diagrams and activity diagrams.

### 5.4.1 Activity Diagram

Activity diagram represents flow of events throughout the game. Diagram starts with the log in screen and goes until the game is finished and score is calculated. Initialization of the game and flow of activities through this process is shown in the figure. We used the elapsed time, used pieces, map's current view to check if game is finished successfully and calculate the score if the game is finished. Each level holds the highest score achieved by a user in this level, we designed it like this so that we can detect if the user has break the record of that level. We update that level's score after game is completed, and we reach exit condition after that.


Visual Paradigm Standard(Safa Aşkın(Bilkent Univ.))
Initial Condition
Get User Info
Get username and password
Open mainmenu
Get user selection
Show tutorial video
Show leaderboard
Show start game screen
Open Level Selection Screen
Exit Condition
Get user selection for game mode
Show 2D Game Field
Show 3D Game Field
Initialize the Game
Get signal which select pieces
Get elapsed time
Get the place that piece will be placed
Check is time over
position is not avaliable
position avaliable
Update Map
board is not full
board is full
initialize next level
calculate and update level score

### 5.4.2 State Chart Diagram

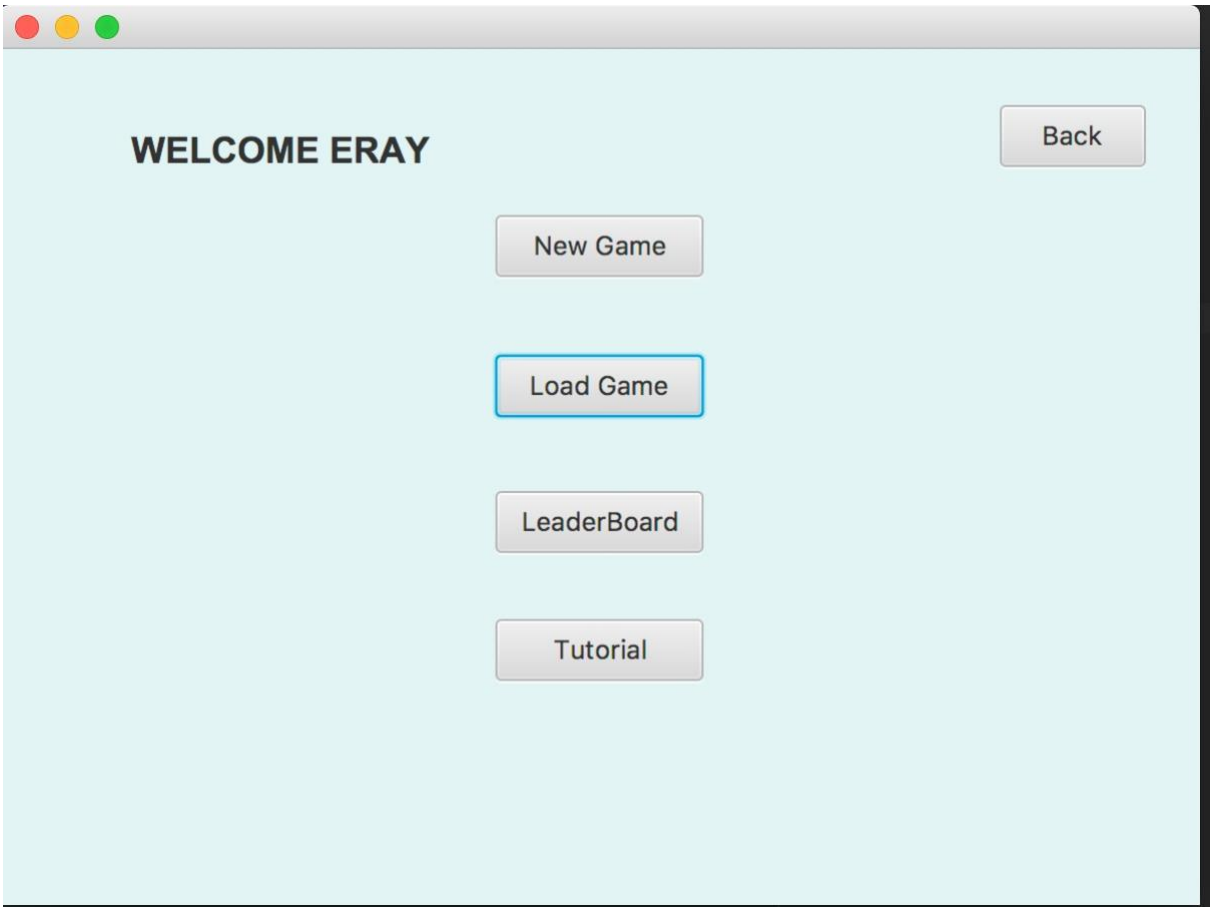



State Diagram shows different states of the game and flow of these states according to user action. State Diagram above describes the changes caused by interaction between user and interface. User invokes transitions by selecting one of the options given in current state.

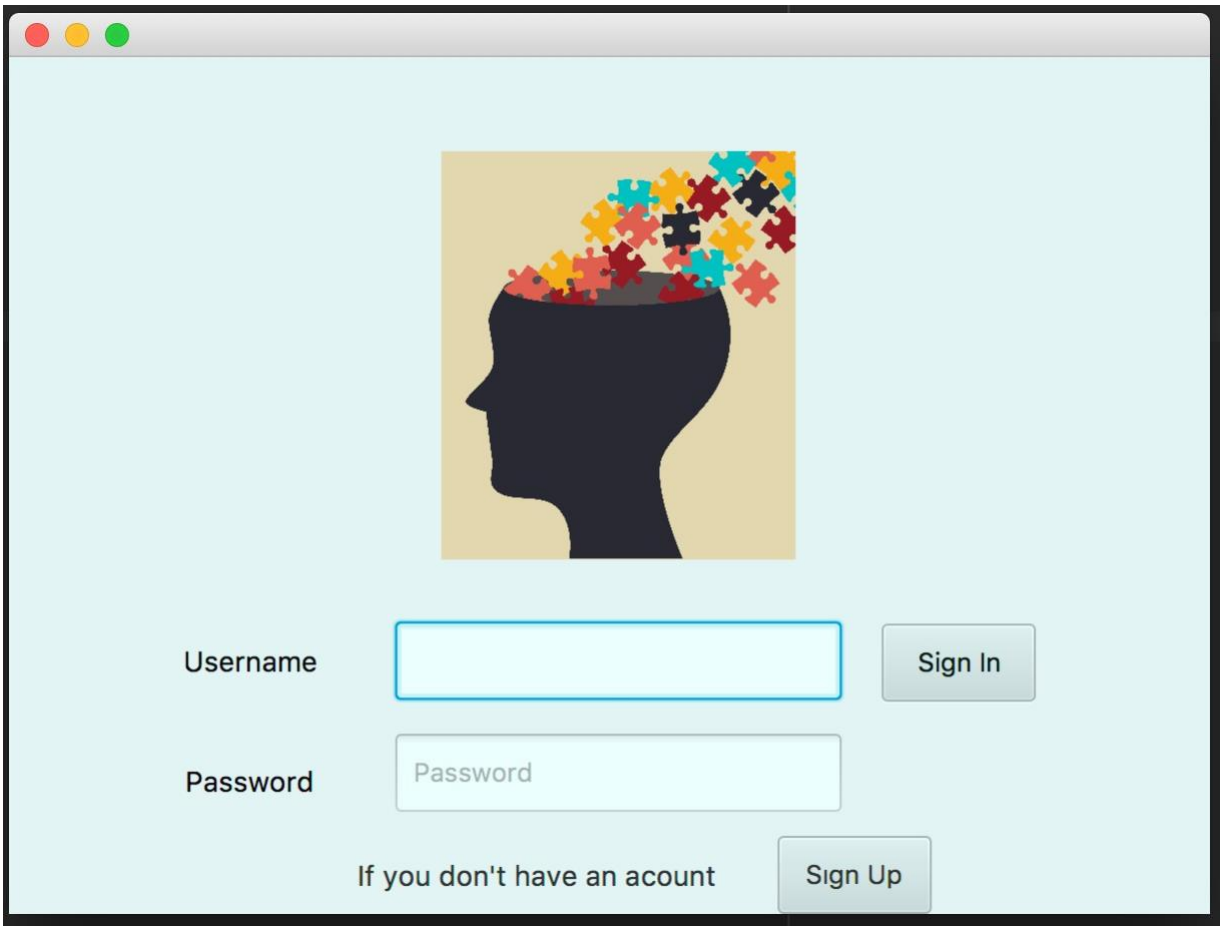
## 5.5 Game Mockup

Game Mockups are basically the view of screens in the real game. By analyzing game mockups customer can give feedback on the screen views of game before implementation. We wanted to design an user interface which is easy to use.

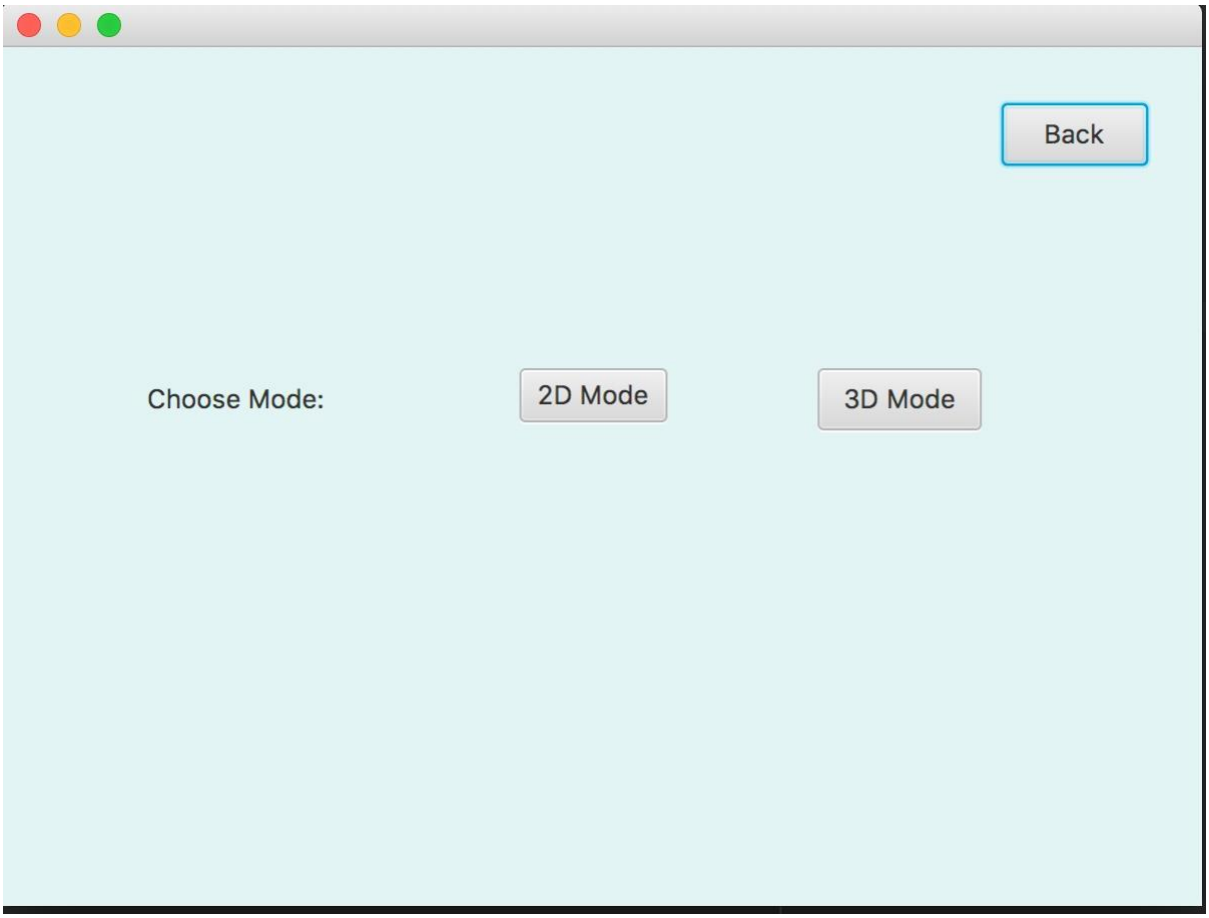
### 5.5.1 Main Screen



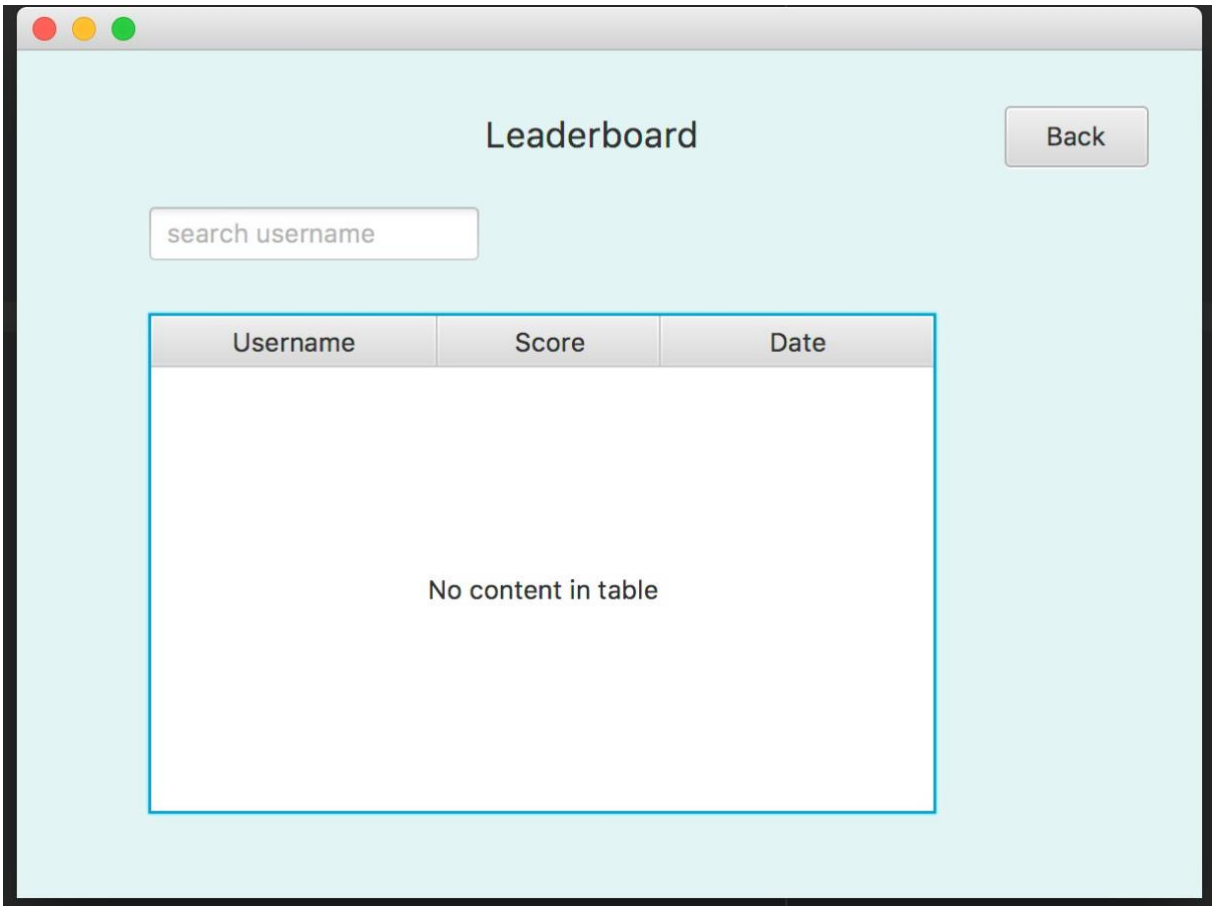
### 5.5.2 Log In Screen



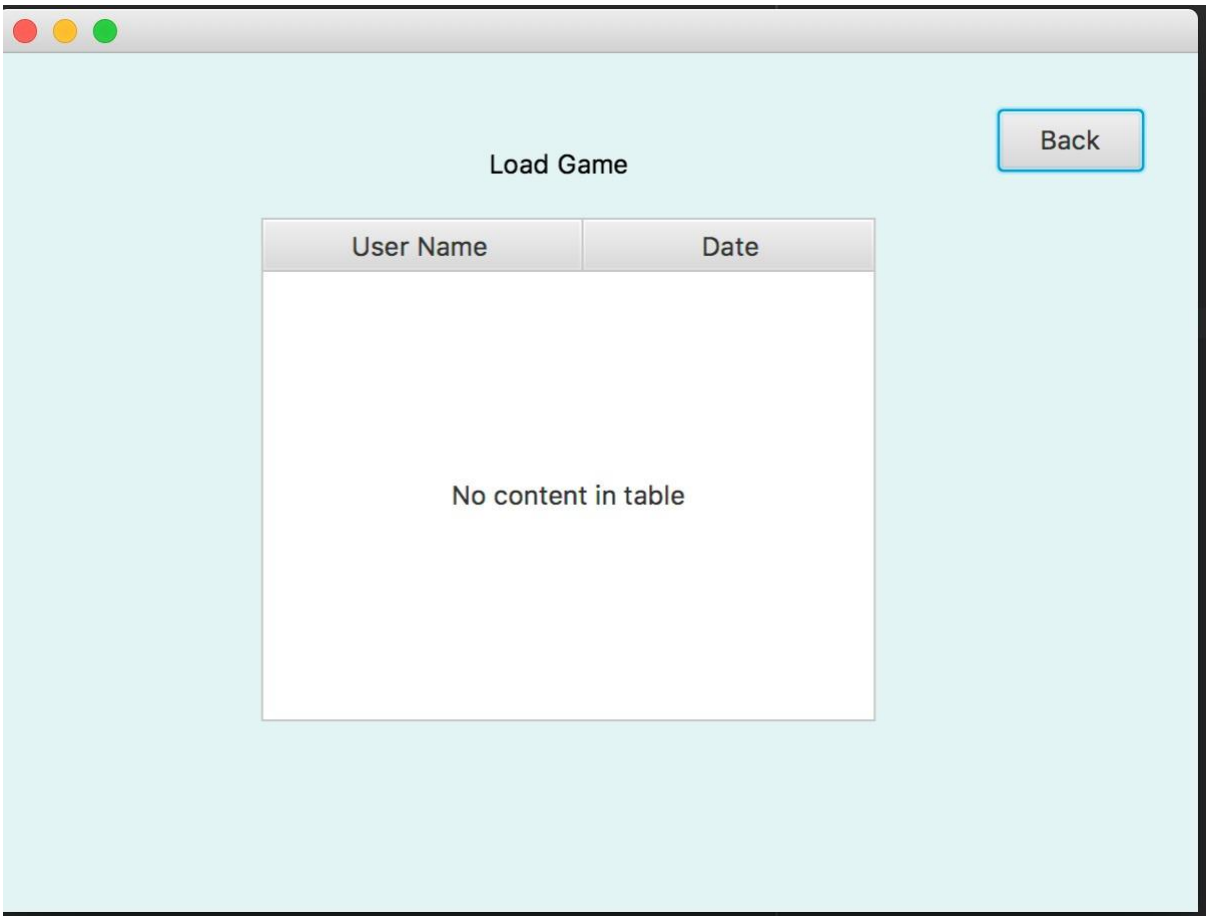
### 5.5.3 Game Mode Selection Screen



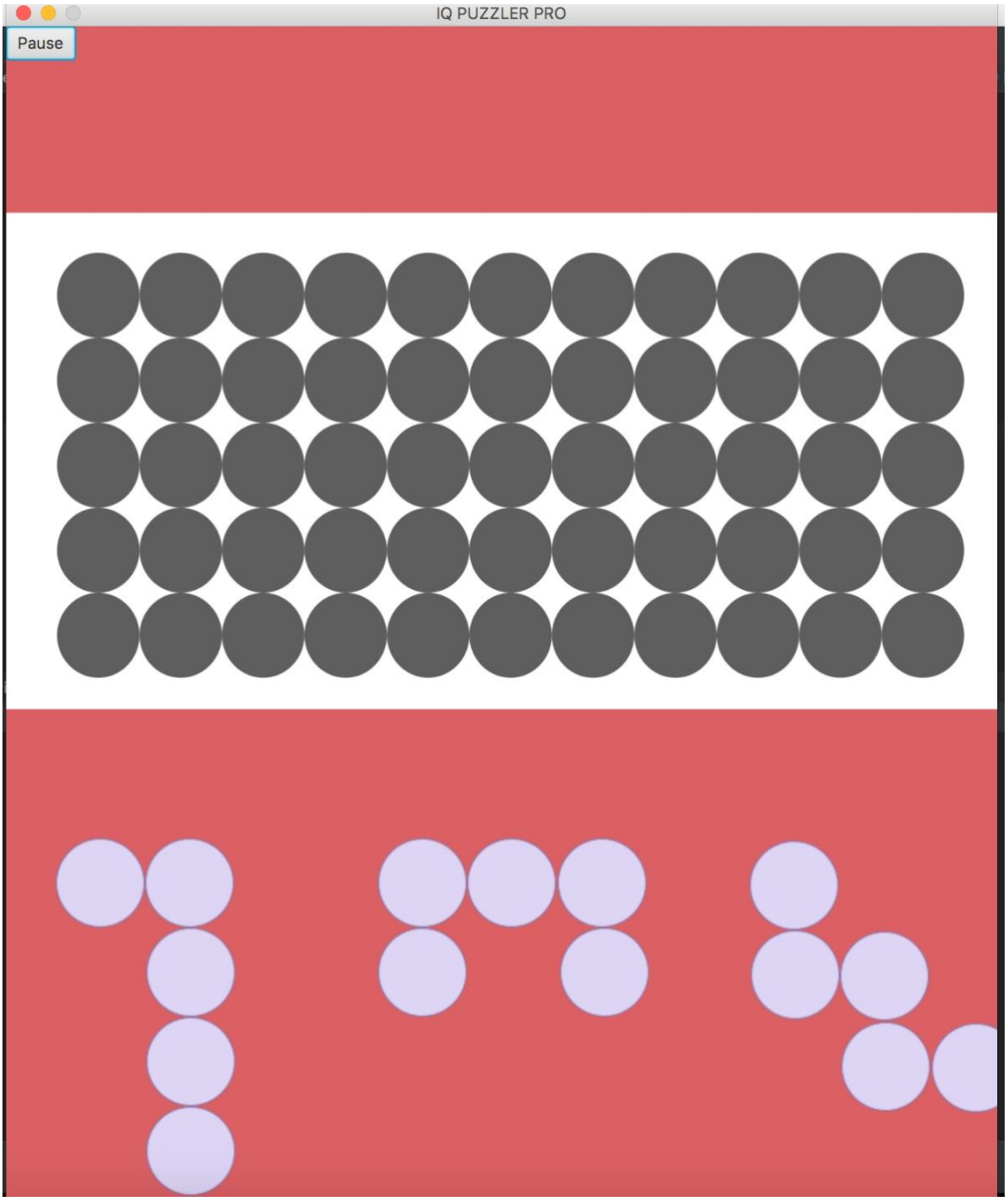
### 5.5.4 Leaderboard Screen



### 5.5.5 Load Game Screen

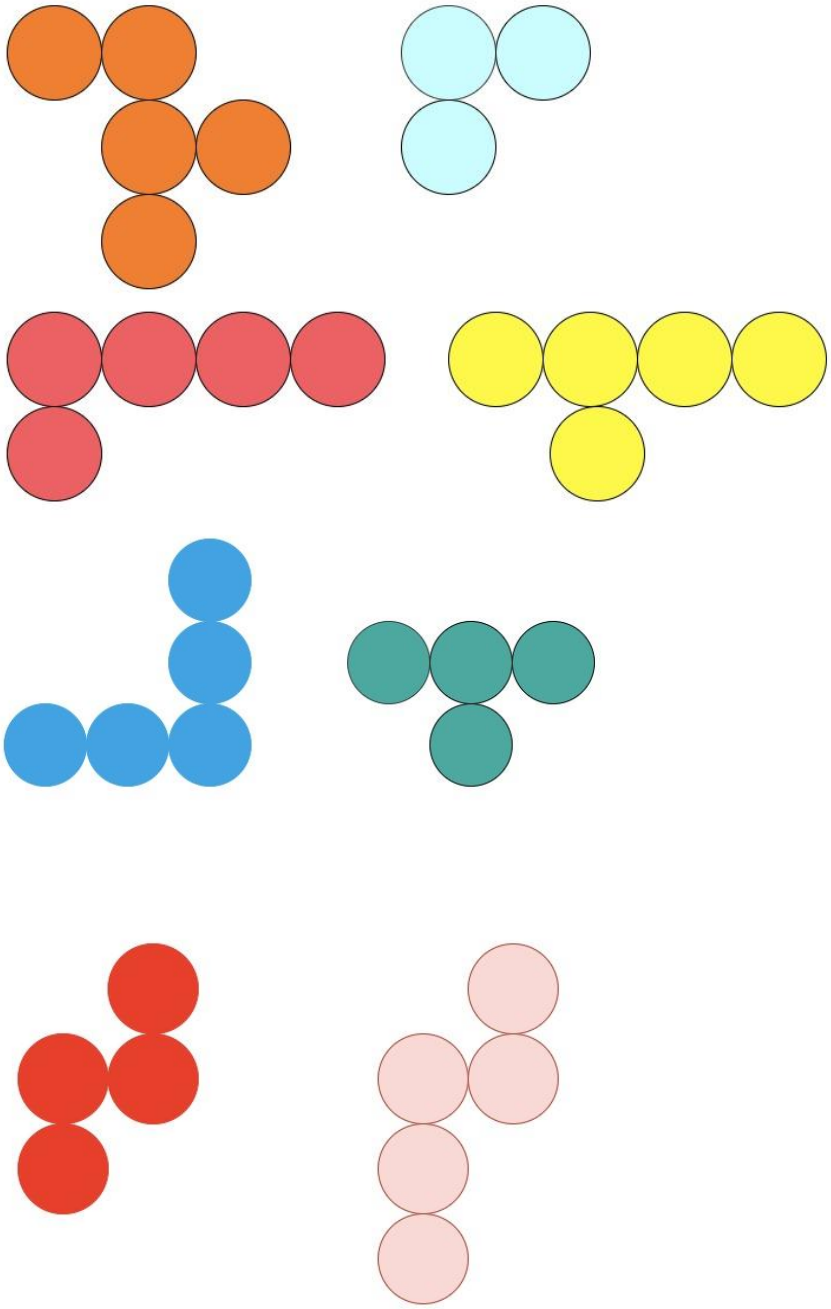


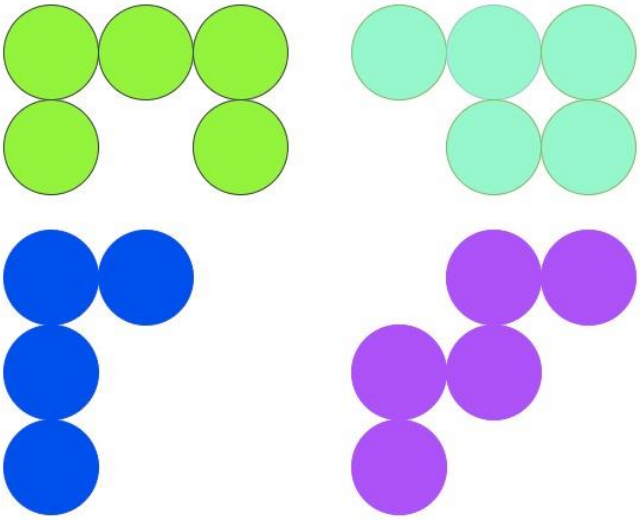
### 5.5.6 Game Screen



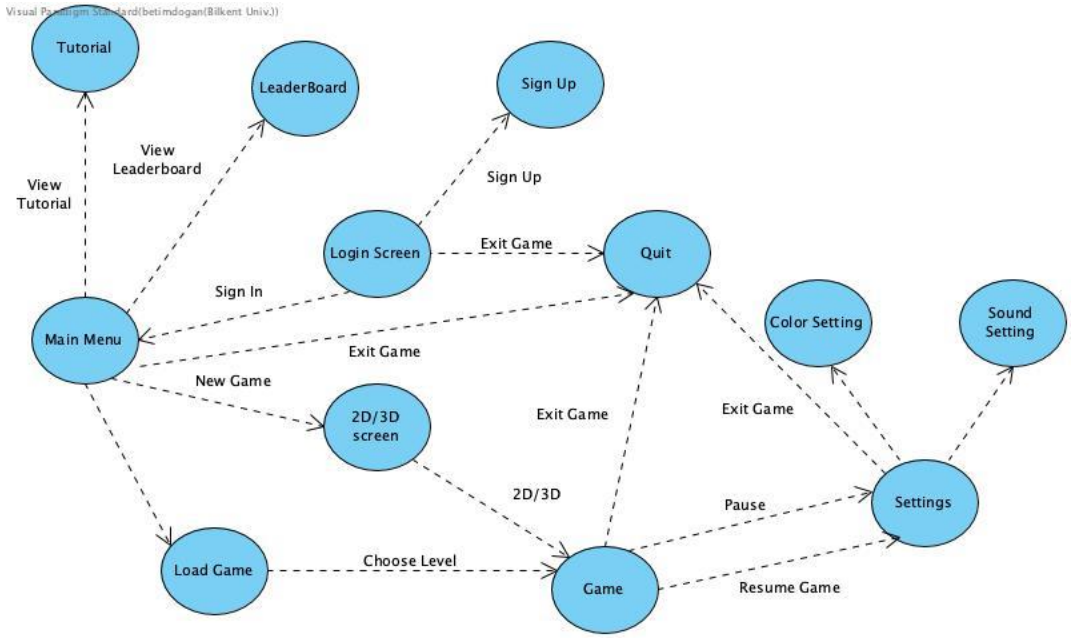
## Icons

**Pieces:**





## 5.6 Navigational Path




# 6 Glossary & References

[1] “Smart Games.” Internet: https://www.smartgames.eu/uk/one-player-games/iq-puzzler-pro, 2018 [Oct. 2, 2018].