Bilkent University

Department of Computer Engineering

CS 319 Project

Project name: Piece12

Analysis Report v2.0

Fantastic 4

Berk Mehmet Gürlek

İsmail Yavuzselim Taşçı

Nursena Kurubaş

Ali Kemal Özkan

November 26, 2018

This report is submitted to the Department of Computer Engineering of Bilkent University in partial fulfillment of the requirements of the Object Oriented Software Engineering course CS319

Contents

1. Introduction

Piece12 is a puzzle kind of mind game that can be played by all the people older than six years old. The idea behind the game is completing a puzzle in the required way. Game has twelve different shaped pieces and different shaped boards. At the beginning of each level user is given a partially filled board with fixed pieces on it and remaining pieces as free for user to fill the gaps on the board with them. If user achieves to fill the board with no holes on it, then it means the level is finished and user may move on to the next level. There are three different game modes: classic, memory and time race, and each mode has different levels that gets harder progressively.

The original version of the game is actually a board game named IQ Puzzler Pro, but we will implement this game as a 2D Android application with new features. We will use Java as a language on Android Studio as IDE since we are considering to release it on Google Play Store, we will also get use of some entities of LibGDX for our user interface.

This report contains an overview, which has many details about the game play. In addition to that, we added some requirements, system models, diagrams with explanations and a sample UI of the game, which gives some idea about how the user will see the game.

2. Overview

In this part, each specifications of the game will be explained.

2.1. Game Play

After launching the game, user will see the main menu, which has the following buttons: “New Game”, “Daily Mission”, “Leaderboard”, “Settings”, “Credits”, “Shop” and “How to Play”. After learning the game from “How to Play” part, user may start playing by clicking “New Game” button. Following page will ask user to choose between single player and multiplayer options. If user chooses single player, then the next page will ask user to choose a game mode: classic, memory or time race mode. Finally, after choosing a mode user will choose the level and start playing the game. Game will simply give user a partially filled board and asks user to fill the board with remaining pieces, which are shown under the board. User should drag the pieces and put them to empty places on the board and board should have no empty hole on it for level to be completed. After completing a level, user can move to next level, go back to level map or jump to the main menu.

2.2. Game Modes

First, there are single player and multiplayer modes. And there will be three game modes in them: classic, memory and time race which has different difficulty levels in each game mode. User will earn coins for each completed level proportional with the level’s difficulty and earn stars to indicate the success level of user at that level.

2.2.1. Single Player

In single player option there are 3 different modes that can be played.

Classic Mode

In classic mode, game will provide user a board, which is partially filled with some of twelve pieces, and remaining free pieces will be settled under the board. User will try to fit these free pieces to the empty places on the board. User should place all pieces on the board -which fills the whole board- to be able to complete the game. At the end of each level, users will earn stars according to their move count. Difficulty is changing according to the given partially filled board at the beginning. At first levels, there are only 2 pieces to place on the board and as the level numbers increase, game will ask user to place a lot more pieces.

Memory Mode

In memory mode, the partially filled board will be shown to user for several seconds at the beginning and then user will have an empty board with 12 pieces under it for user to place them on the board. After that user will try to fill the board following exactly same pattern as shown. If user tries to put a piece to a place other than shown, phone will vibrate and the piece will return under the board. At the end of each level, users will earn stars according to their move count, same as the classic mode.

Time Race Mode

In time race mode, game will give user the same levels as classic mode, but this time user will try to finish the game in a given time limit. Otherwise, level will be failed and no coins and stars earned. If user finishes the game before the given time,

level will be completed successfully and user will earn stars according to the remaining time.

2.2.2. Multiplayer

In multiplayer option, two different players can race against each other. User can choose to play against a random player or a friend and user should choose one of the three difficulty levels: Easy, Medium, and Hard. Faster player will win the game in each mode but there can also be draws. There will be three different game modes in multiplayer option, same modes as the single player's, but the playing styles have some additional features on them, which are written below.

Classic Mode & Memory Mode

In these modes, again the gold is completing the board faster than the opponent. The one who completes the board first will win the game.

Time Race Mode

In time race mode users will be given a time and they will try to fill as many boards as they can in this time. When a user finishes a board, another board will come, and user will try to fill it this time, until the time is up this loop will go on. The one who filled more boards will be the winner.

2.3. Levels

In each game modes, there will be many different levels from easy to hard. User will try to pass the levels one by one. It means, when user completes a level, s/he will be able to play the next level and previous levels too, but no further levels than the next level. At the end of each level user will earn stars according to their

success rate. If user fails on a level, that means there will be no stars in that level and further levels will remain unlocked until user earns at least one star on that level.

2.4. Settings

In settings user will be able to open or close the music, sound effects and notifications separately.

2.4.1. Notifications

User will be sent notifications daily about daily missions and if user does not play the game for long time, some notifications will be sent to remind the game to user.

2.4.2. Music

A smooth music will be played when the game is running.

2.4.3. Sound Effects

Some sound effects will be heard when user clicks a button or when holds and releases a piece.

2.5. Shop

Users can use the coins that they earned from the games to buy some features such as different themes, and unlock the augmented reality option.

2.5.1. Augmented Reality

In augmented reality option, user can use camera to play a game as if it is a real board game. This option will change the game background to the camera's vision. For example, by launching the back camera and pointing it to a table, user can have the feeling that the board and pieces are actually on the table.

2.5.2. Themes

Themes consist of different board shapes and patterns, colorful backgrounds, different piece colors and patterns.

2.6. Leaderboard

Users will be able to see the high scores of each single-player mode and their place on scoreboard.

2.7. Credits

User will be able have some information about our works and us.

2.8. How to Play

There will be some indications and introductory pictures for user to learn the game and play it easily.

3. Functional Requirements

User will be able to access followings from Main Menu:

3.1. New Game

“New Game” button will lead user to choose between “Single Player” (see 3.1.1 below) and “Multiplayer” (see 3.1.2 below) options. After that, user will choose a mode, a level and will be ready to play the game.

In the game, there will be a partially filled board on the screen that has stable pieces on it, and some free pieces under the board that are moveable. User will drag the pieces to the empty places on the board and put them on accurate places one by one. Completing the board is the main aim of the game no matter which mode is chosen. During the game there is always a “Pause” option that blackens the screen

and shows some options with some symbols that means: resume the game, go to main menu, go to settings, restart the level, go back to map.

3.1.1. Single Player

User can choose this option after clicking the New Game button. Game can be played as a single player and offline. Three different modes of “Single Player” option are memory, classic, time race. After clicking one of these modes anew page that shows level map will come up and when user choose the level game will start. User earns coins through the game and stars at the end of each level to indicate the level success.

3.1.2. Multiplayer

User can choose this option after clicking the New Game button. There is also a multiplayer option in game, which allows user to play the game with other online users. After choosing a mode and difficulty level, user can choose to find a random opponent or play with a friend. Main goal is to complete the board before the opponent. Faster player always wins no matter which mode is open. This mode requires Internet connection.

3.2. Daily Mission

There will be a different random-mode medium level game every day. Users will be able to load this game through an Internet connection. These missions will send a notification when they are regenerate. User will be able to earn many coins when the mission is finished.

3.3. Shop

Users will be able to spend the coins they earned from the game here by buying themes and so. They will also use shop page for using their bought products, such as choosing a background between the ones they bought (and the default one). User can unlock and activate or deactivate augmented reality option from here.

3.4. Settings

User will be able to turn on & off sound effects, music and notifications from here.

3.5. Credits

There will be a text about the game's creators and their other works for user to read if requires.

3.6. How to Play

There will be educative illustrations about the game play and brief explanations under each image to make user learn better by not getting bored.

3.7. Leaderboard

Users will be able to see their own rankings and the highest scores as a list from this page.

3.8. Rate Us

This button will be displayed as a star symbol and direct user to Google Play Store to rate the application.

4. Non-Functional Requirements

In this part non-functional criteria of the game will be explained. We construct the game based on these four requirements.

4.1. User Friendly

User-friendliness is one of the most important requirements of our game. To attract the user, user interface of the game will be easy to use and understand, so we focus on the functionality of the game a lot, for instance, we are trying to make as easy as possible for users to reach all options whenever they need.

For a better UI, colors and themes are also made smooth to ease user mind and provide them focus, also we kept our visuals plain and simple for not to distract the user and help them focus on the game better.

It will also be easy for user to learn game by some illustrations and brief explanations and by playing a tutorial -which we call Level 0- rather than reading long paragraphs, so we worked a lot on learning part of the game to make it easy, fast and fun.

4.2. Game Performance

Game performance is one of the most non-functional requirement for efficient game. We will keep performance high to keep user in the game. Therefore, the fps rate of the game will be at least 60. In addition; the game won’t need high system requirements in order to play easily with good performance. User can play

the game with older phones or older android versions. Game will not use much cpu or ram. Average android phone will run the game without problems.

4.3. Extensibility

Since we have game modes, levels, shop in our game, it will extendible in many component. New modes can be added or new level (higher or lower difficulty) can be added easily. We have shop option that we can buy something by coins. Our shop is also extendible, there may be varied in terms of game items. All of these help the increase the extensibility of the game.

4.4. Reliability

The software is built for being reliable in many concepts. Users' data, progress and preferences is kept after they exit the program so that they can continue the game from where they left. We will try to implement a game that has no bugs. When we realize a bug is exist, we are trying to fix it as fast as possible and not delaying it to a later time.

5. System Models

In this part, models and diagrams of the game will be explained in detailed way.

5.1. Use Case Model

In this part every use case is shown by a diagram. Then, every use case will be explained step by step.

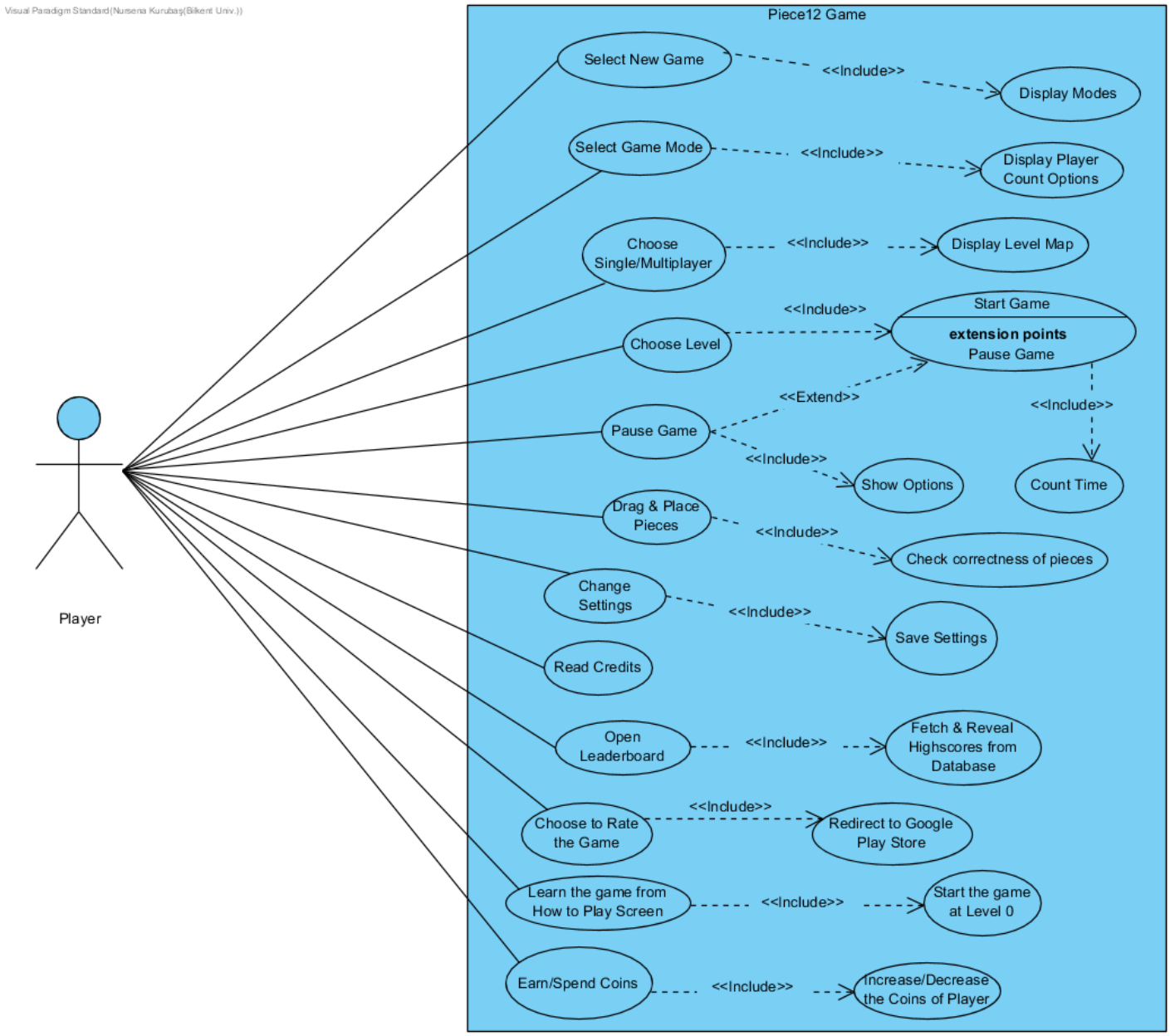


Figure 1: Use Case Diagram

Use Case #1

Use Case: Select new game

Primary Actor: Player

Stakeholders and Interests:

-Player/Players want to play the game

-System creates the game and starts the game.

Pre-conditions:

-Player must be in the new game screen.

Entry-conditions:

-Player selects “New Game” button from main menu.

-Player should choose the Single Player or Multiplayer.

-Player should choose a game mode (Classic/Memory/Time Race)

Exit conditions:

-Player selects “Quit Game” button from the pause menu.

-Player successfully finished the game by complete the level.

Success Scenario Event Flow:

1. Player selects “New Game” button from main menu.
2. Player chooses “Single Player Mode”.
3. Player chooses a game mode “Classic”.
4. Player presses “Start” button.
5. Player drags and drops pieces.
6. Player finish all the map.
7. Player has chance to play next level.
8. Player repeats the steps 3 to 7.
9. Player finishes the game.

Alternative Event Flows:

1. If Player wants to return main menu:

-Player pauses the game by pressing pause button from game screen.

-Player selects “Quit Game”

-System quits the game.

Use Case #2

Use Case: Select game mode

Primary Actor: Player

Stakeholders and Interests:

-Player/Players want to select game mode.

-System show user the game modes.

Pre-conditions:

-Player must be in the game mode screen.

Entry-conditions:

-Player selects “New Game” button from main menu.

-Player should choose the Single Player or Multiplayer.

-Player should choose a game mode (Classic/Memory/Time Race)

Exit conditions:

-Player press “back” button.

-Player select a game mode.

Success Scenario Event Flow:

1. Player select new game.
2. Player select single/multiplayer option.
3. Player select a game mode.

Alternative Event Flows:

1. If Player wants to return main menu:

-Player press back button.

-Player gets back to the main menu.

Use Case #3

Use Case: Choose level

Primary Actor: Player

Stakeholders and Interests:

-Player/Players want to select level

-System show user the levels.

Pre-conditions:

-Player must be chosen a game mode and should be in levels screen.

Entry-conditions:

-Player selects “New Game” button from main menu.

-Player should choose the Single Player or Multiplayer.

-Player should choose a game mode (Classic/Memory/Time Race)

-Player should choose a level from 40 different levels.

Exit conditions:

-Player press “back” button.

-Player select a level and game starts.

Success Scenario Event Flow:

1. Player select new game.
2. Player select single/multiplayer option.
3. Player select a game mode.
4. Player select a level.

Alternative Event Flows:

1. If Player wants to return main menu:

-Player press back button.

-Player gets back to the main menu.

Use Case #4

Use Case: Pause Game

Primary Actor: Player

Stakeholders and Interests:

-Player/Players want to pause the game

Pre-conditions:

-Player must be in the game screen.

Entry-conditions:

-Player plays the game.

-Player press the pause button.

Exit conditions:

-Player press “play” button after pause screen is showed.

-Player go back to main menu.

Success Scenario Event Flow:

1. Player plays the game.
2. Player press the pause button.
3. Game is paused and pause screen is showed.
4. **Alternative Event Flows:**

1. If Player wants to return main menu:

-Player press home button.

2. If player wants to keep playing:

-Player press the play button.

3. Player wants to change setting:

-Player press the view setting button.

<u>Use Case #5</u>

Use Case: Settings

Primary Actor: User

Stakeholders and Interests:

-User wants to change sound settings, sound effects and notifications.

-System changes sound settings and notification settings.

Pre-conditions:

- User must be in main menu.

-User must be in pause menu.

Post-conditions:

-Sound settings are updated.

Entry-conditions:

-User selects “View settings” button from main menu.

-Player selects “Change settings” button from pause menu.

-User changes sound settings by choosing on/off.

-Player changes notification settings by choosing on/off.

Exit conditions:

-Player selects “Exit” button from the pause menu.

-Player presses “Esc” from keyboard.

Success Scenario Event Flow:

-User selects “View settings” button from main menu.

-User turn off the sound effect “off” button.

-System mutes the game sound.

Alternative Event Flows:

1. If User wants to return main menu:

a. Player presses “back” button.

b. System displays main menu.

2. If Player wants to change settings:

a. Player selects “Show Options” from Pause menu.

b. Player changes sound settings.

3. If Player wants to return pause menu:

a. Player presses “back” button from Settings menu.

b. System displays pause menu.

Use Case #6

Use Case: View Credits

Primary Actor: User

Stakeholders and Interests:

-User wants to see credits of the game

-System displays credits.

Pre-conditions:

-User must be in the main menu.

Entry-conditions:

-User selects “View Credits” button from main menu.

Exit conditions:

- User selects “back” button from the credits screen.

Success Scenario Event Flow:

-User selects “Credits” button from main menu.

Alternative Event Flows:

1.If User wants to return to main menu:

-User selects “back” button.

-System displays main menu.

Use Case #7

Use Case: How to Play

Primary Actor: User

Stakeholders and Interests:

- Player wants to learn how to play the game.

-System display how to play screen

Pre-conditions:

- User must be in the main menu.

Entry-conditions:

-User selects “How to Play” button from main menu.

Exit conditions:

-Player selects “back” button from the how to play screen.

Success Scenario Event Flow:

-Player selects “How to Play” button from main menu.

-System displays How to Play screen.

Alternative Event Flows:

1. If User wants to learn how to play:

-User selects “How to Play” button from main menu.

-System displays How to Play screen.

2. If User wants to return to main menu:

-User selects “back” button.

-System displays main menu.

Use Case #8

Use Case: Leaderboard

Primary Actor: User

Stakeholders and Interests:

- Player wants to see leaderboard.

-System display leaderboard screen.

Pre-conditions:

- User must be in the main menu.

Entry-conditions:

-User selects “Leaderboard” button from main menu.

-User select a game mode.

Exit conditions:

-Player selects “back” button from the leaderboard screen.

Success Scenario Event Flow

-Player selects “Leaderboard” button from main menu.

-Player select a game mode such as “classic”.

-System displays leaders of the classic mode.

Alternative Event Flows:

1. If User wants to see leaderboard.

-User selects “Leaderboard” button from main menu.

-System displays modes screen.

-User chooses a mode such as “memory”.

-System displays leader of memory mode.

2. If User wants to return to main menu:

-User selects “back” button.

-System displays main menu

<u>Use Case #9</u>

Use Case: Rate Us

Primary Actor: User

Stakeholders and Interests:

- Player wants to rate the game.

- System display “rate us” screen.

Pre-conditions:

- User must be in the main menu.

Entry-conditions:

-User selects “Rate Us” button from main menu.

Exit conditions:

-Player selects “back” button from the rate us screen.

Success Scenario Event Flow:

-Player selects “Rate” button from main menu.

-Player select a rate 1-5.

-Player submit the rate.

Alternative Event Flows:

1. If User wants to return to main menu:

-User selects “back” button.

-System displays main menu

<u>Use Case #10</u>

Use Case: Spend coins in shop

Primary Actor: User

Stakeholders and Interests:

- Player wants to buy something in shop.

- System add items to users account.

Pre-conditions:

- User should have coins to spend.

-User should enter the shop.

Entry-conditions:

-User selects “Shop” button from main menu.

Exit conditions:

-Player selects “back” button from the shop screen.

Success Scenario Event Flow:

-Player selects “shop” button from main menu.

-Player press a background to buy it.

-Player coin balance is decreased.

-New background is ready to use.

Alternative Event Flows:

1. If User wants to return to main menu:

-User selects “back” button.

-System displays main menu .

5.2. Dynamic Models

In this part, activity, sequence and state diagrams of the game will be explained.

5.2.1. Activity Diagram

This Activity Diagram shows that which main concepts are based on to build our application.

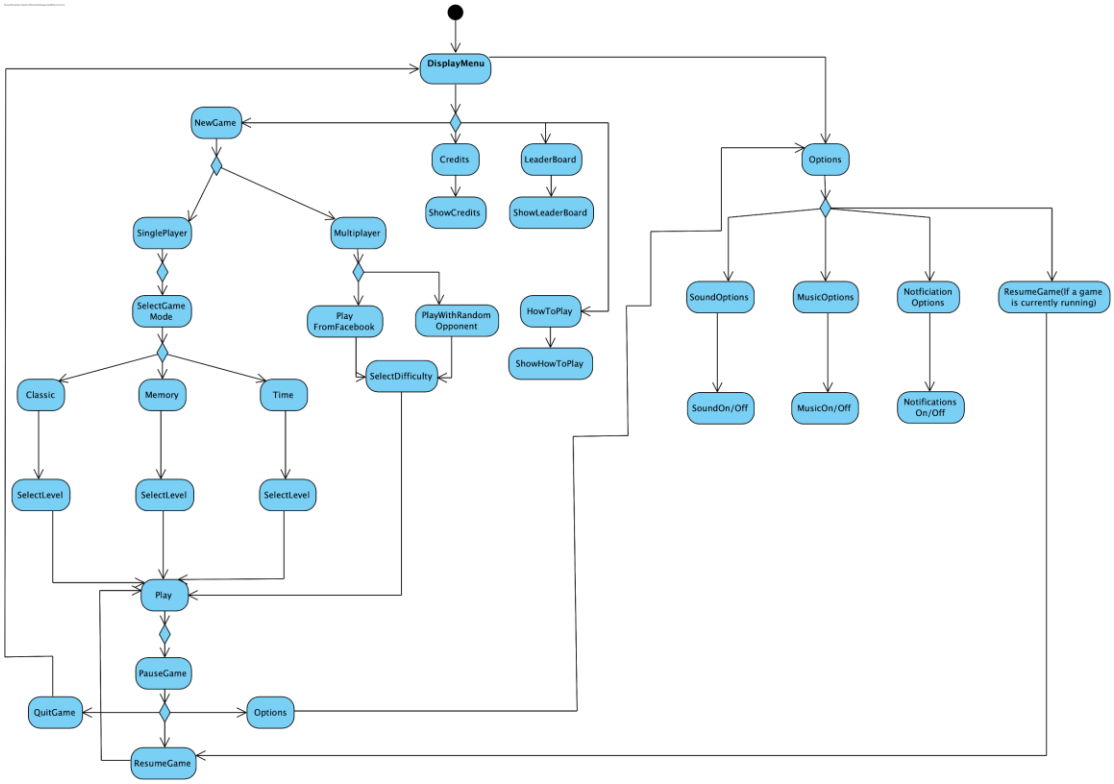


Figure 2: Activity Diagram

The figure represents the hieratical representation of each piece in our application. Initially when the application is opened, the menu shows up which is represented by DisplayMenu. Moreover, each step in the application is reversible, means that you can go back to any screen in our app step by step, except the menu page. Simply, the menu is the first and the last step in our application.

NewGame - Phase1: When the NewGame option is selected, the user faces up with two options: Singleplayer and Multiplayer

Singleplayer - Phase1.1: When Singleplayer option is selected user selects from the three gameplay options that are represented: Classic, Memory, Time

Classic - Phase1.1.1: When classic gameplay is selected, user selects levels that their hardness grows from first to last. When the level is selected, (Initially only

the first level is opened and the rest is locked. When user completes a level, then the next level will be opened.) user starts to play the game. However we have provided a PauseGame option in order to allow the user to QuitGame and return to the main menu, change the settings of the game by pressing on Options button, or resume the game.

Memory - Phase1.1.2: When memory gameplay is selected, user again selects levels but in this case, the solution appears on screen for a limited time and disappears. Then, user tries to memorize the solution and place the pieces into right places. Same specifications are provided in this part as in the classic gameplay.

Time - Phase1.1.3: When time gameplay is selected, user has to complete the puzzle in a given time else, he/she fails. Same specifications are provided in this part as in the classic gameplay.

Multiplayer - Phase1.2: When Multiplayer option is selected user is given two options which are: Connect Facebook to play and Random Opponent. In multiplayer, two users tries the finish the puzzle. Game ends when a user completes the puzzle. Basically the first to complete the puzzle wins the game.

Facebook - Phase1.2.1: When Facebook option is selected, user needs to connect his/her facebook account to play with his/her friends. While playing the game, PauseGame and its features are available to user.

Random - Phase1.2.2: When Random Opponent option is selected, user faces up a random Google Play user who is simultaneously plays the game with the user. PauseGame and its features are available to user.

Options - Phase2: When user enter the options, there are 3 options to select from: SoundOptions, MusicOptions, and NotificationOptions

SoundOptions - Phase2.1: User is provided either choosing to turn the sound on or off. Also, user can go back to options menu.

MusicOptions - Phase2.2: User is provided either choosing to turn the music on or off. Also, user can go back to options menu.

NotificationOptions - Phase2.3: User is provided either choosing to turn the notification on or off. This will trigger the devices notification options. Also, user can go back to options menu.

HowToPlay - Phase3: How to play screen is like a tutorial screen that shows the directives that how to play the game. User can go back to main menu after he/she reads the how to play instructions.

Credits - Phase4: Credits section shows the name developers of the game. This screen also provides to go back to main menu.

5.2.2. Sequence Diagrams

At this section we illustrated three sequence diagrams that are about navigating between states, initialize game and game play.

Navigating Between States

At this diagram we illustrated how the navigational path works in our game.

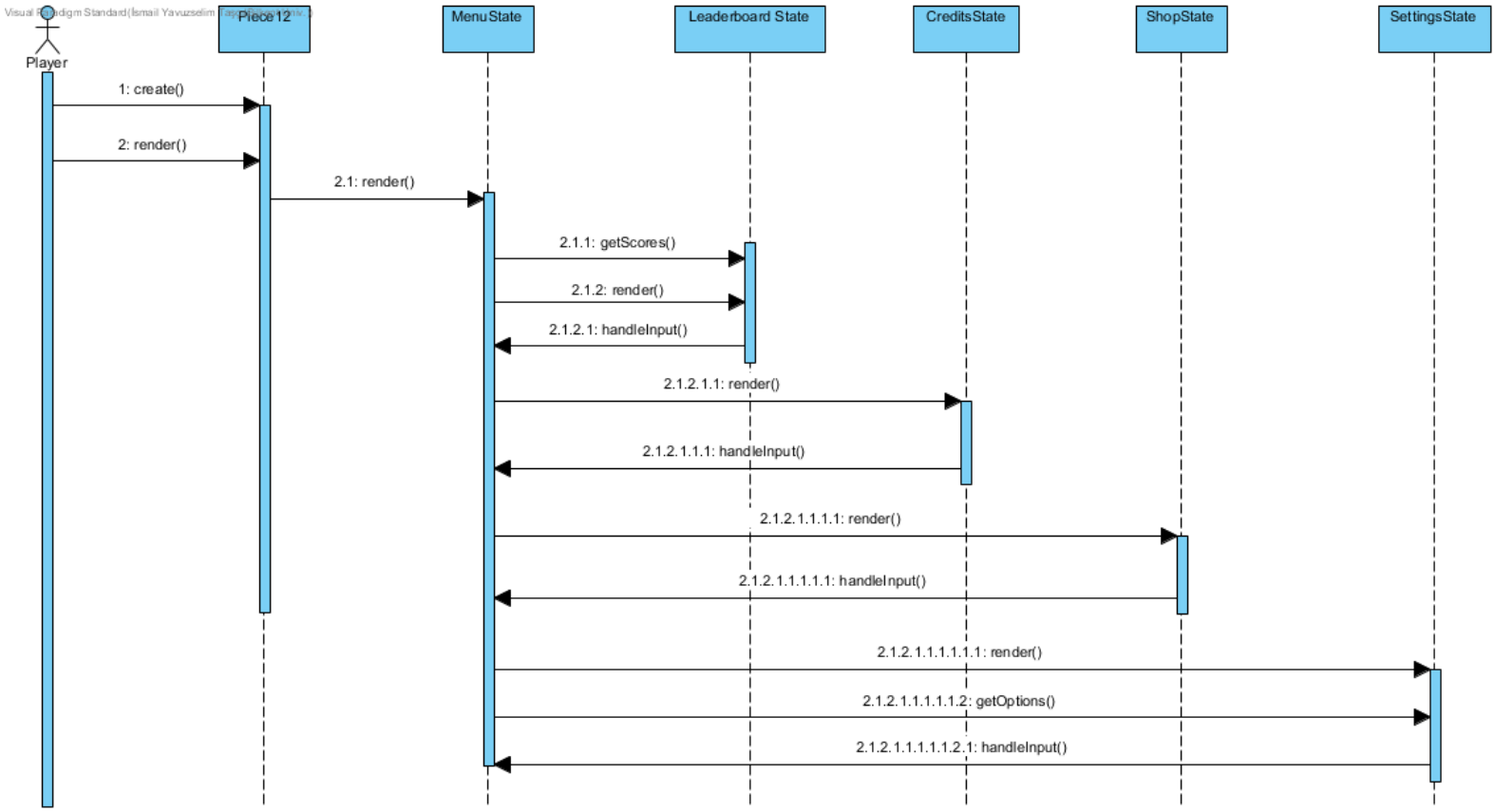


Figure 3: Sequence Diagram #1 (Navigating Between States)

Scenario: Player wants to navigate between game states

In the MenuState Player can choose where s/he wants to navigate and this will result to render() function to create that state. For example, if Player presses Settings button, render() function will initialize SettingsState, and then Player will see the SettingsState. In addition to that, getOptions() function will get current options for Player to see. handleInput() function will take care of the gui inputs all the time so that when Player wants to go back handleInput() function will do that and Player will see the previous screen.

Initialize Game

This diagram explains the game initialization from the first running of the application.

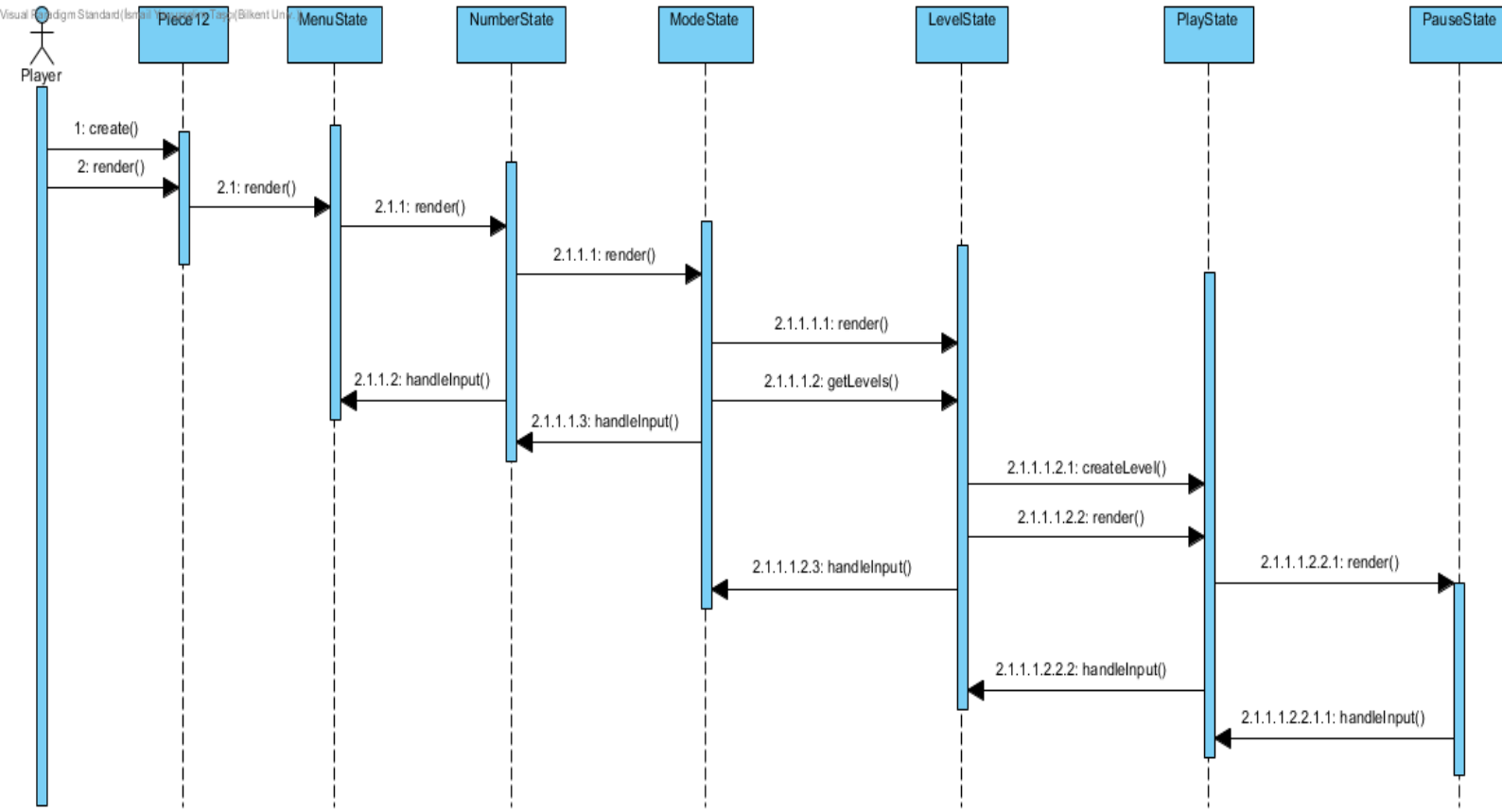


Figure 4: Sequence Diagram #2 (Initialize Game)

Scenario: Player starts the game

When Player wants to start game first s/he opens game and this situation will create our Piece12 class. Then this class will initialize MenuState class by render() function which will be our main menu state. Then when Player choose new game option NumberState will be initialized by render() method for Player to choose singleplayer or multiplayer. When Player made choice ModeState will be created by render() method so that Player can choose which mode s/he wants to play. After that Player will see LevelState screen whis is created by render() function to choose level. When all of these states over user will be taken to the PlayState which is the main game state. In this state Player can play and pause the game. In all of these states handleInput() method takes care of the gui input so that Player can go back or make a choice between choices.

Game Play

After player starts the game by following the previous scenario, the Player and PlayState of the game will have interactions as following sequence diagram.

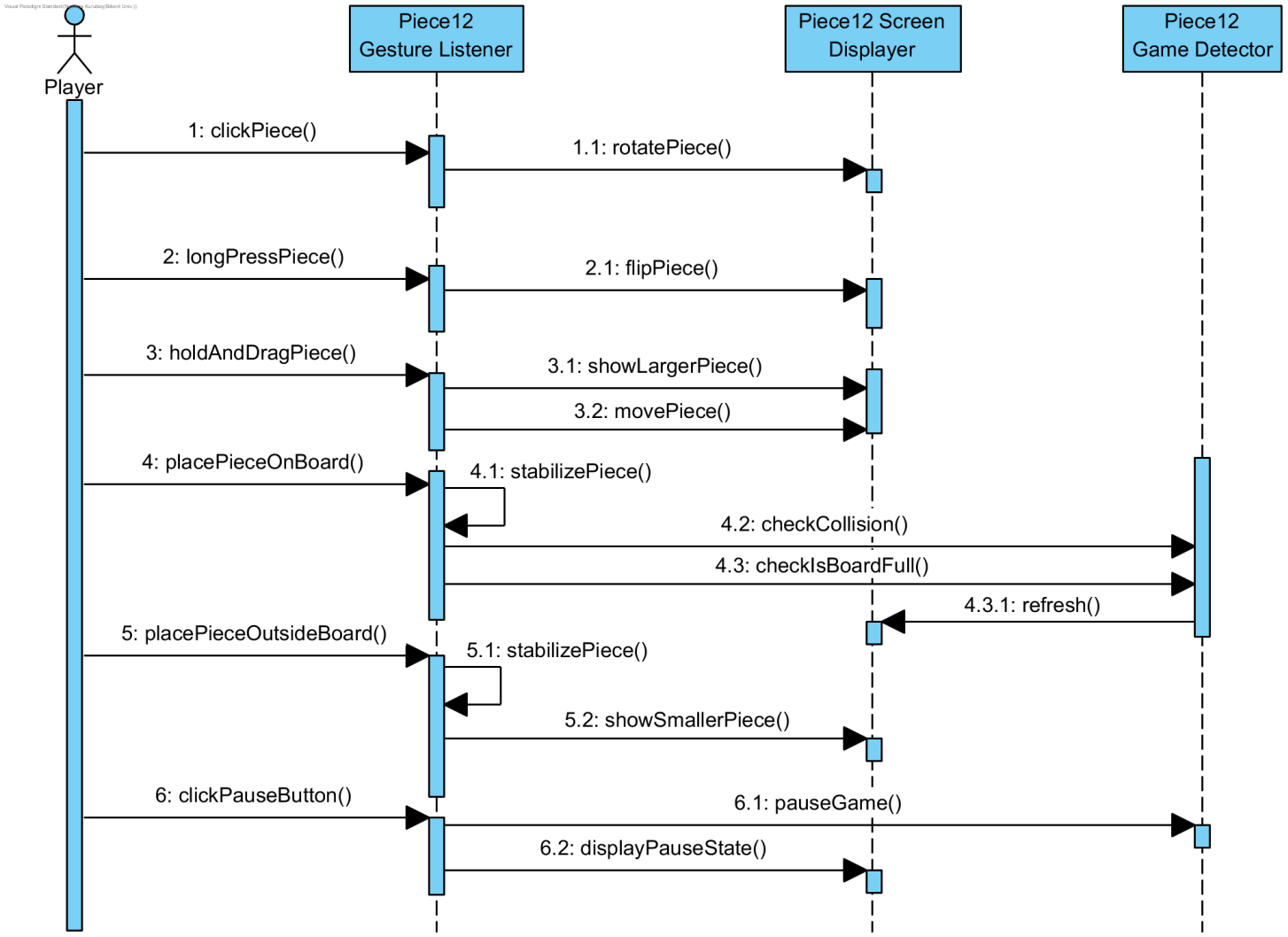


Figure 5: Sequence Diagram #3 (Game Play)

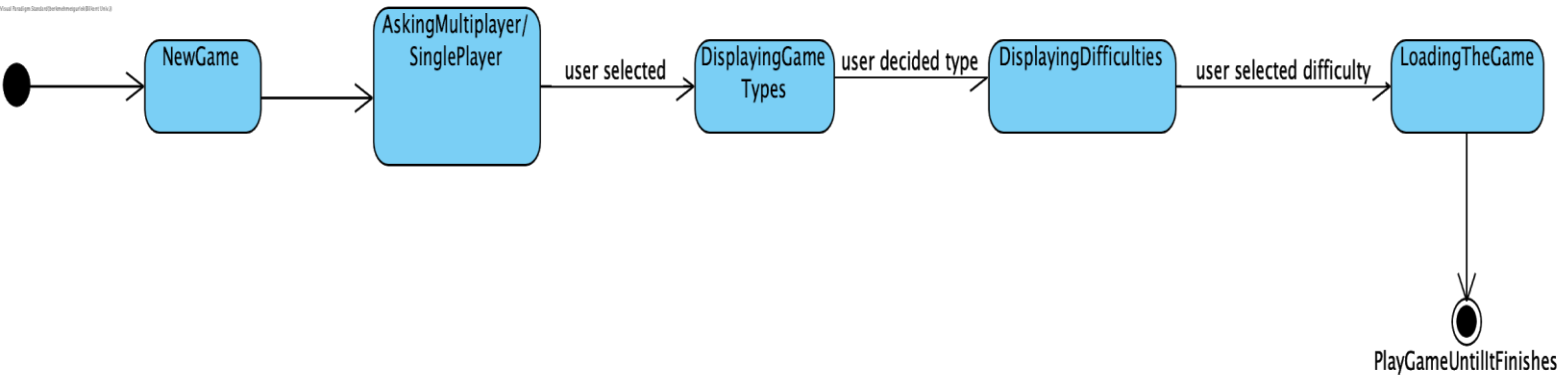
Scenario: Player is playing the game

When Player clicks on a piece, the Gesture Listener of the game will take this input and the piece will rotate 90 degrees and displayed. If Player presses long to a piece, the Gesture Listener of the game will take this input and the piece will be flipped and displayed. When Player wants to hold and drag a piece, the Gesture Listener will take the input and move the piece accordingly and piece will be shown

on the screen larger. When Player places a piece on the board, the Gesture Listener will understand that the holding is stopped and stabilize it to that place until Game Detector decides there is a collision, which means if that place is empty. Game Detector first checks the collisions and then checks if the board is full and screen is refreshed accordingly. If Player place the piece back to outside the board, the piece will be shown smaller again and stabilized to the place it is dropped. While playing the game, if Player pauses the game by clicking pause button the game detector will pause the game, which may include pausing the time counter etc., and screen will display the Pause State of the game.

5.2.3. State Diagrams

At this section, we illustrate three state diagrams to provide easier understanding for



our game.

Figure 6: State Diagram #1 (NewGame)

Figure above represents our application's new game state. It starts after the user selects the new game option from the main menu and each state is processed with user's selections. The states end with the last selection of the user.

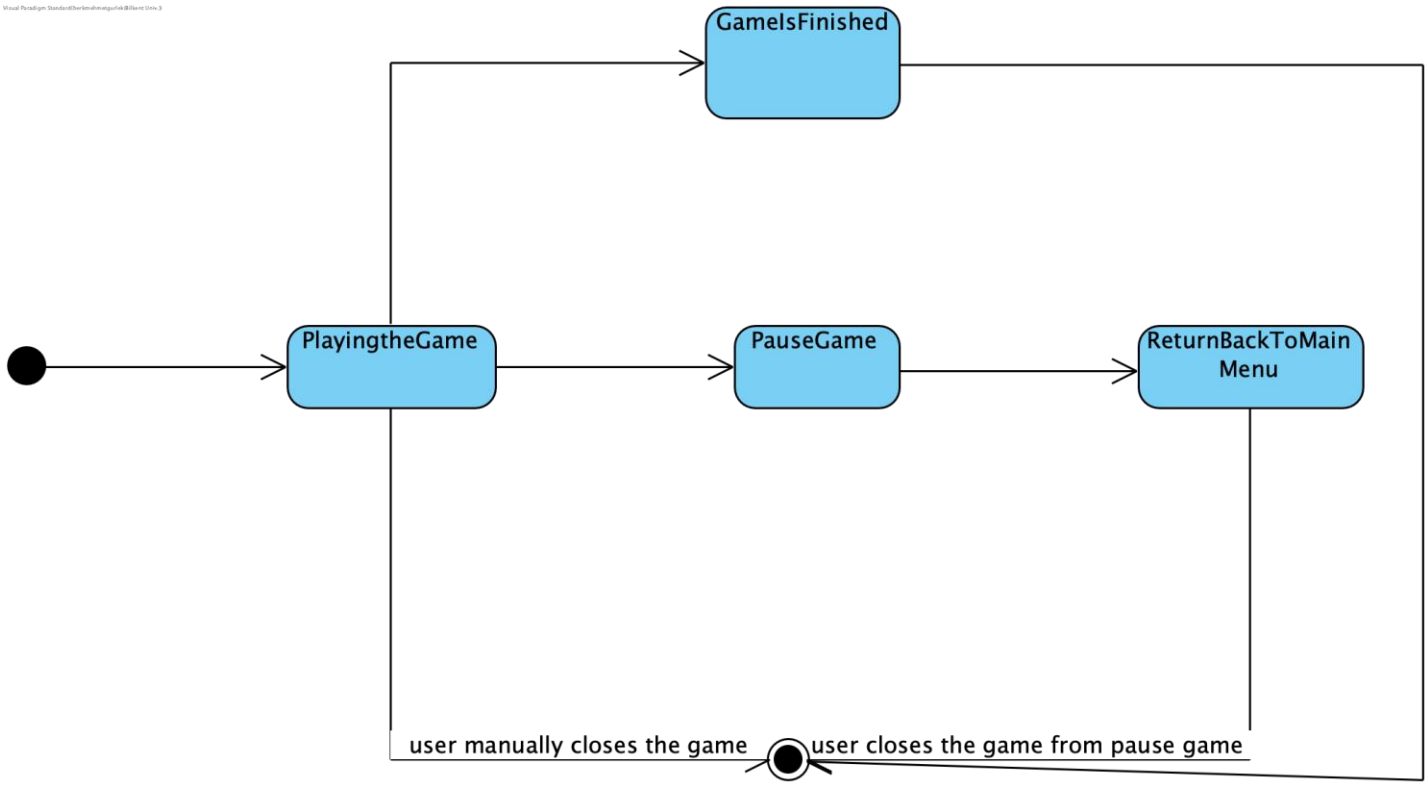


Figure 7: State Diagram #2 (GameState)

In game state basically represents how the user can exit from the game. It starts with after user does all the selection from the previous state diagram shown above. Then user is provided with two different options to quit the game. Initially, user can manually exit from the game via pressing the home button of his/her device. Secondly, user may close the game by selecting return back to main menu option from the pause game option.

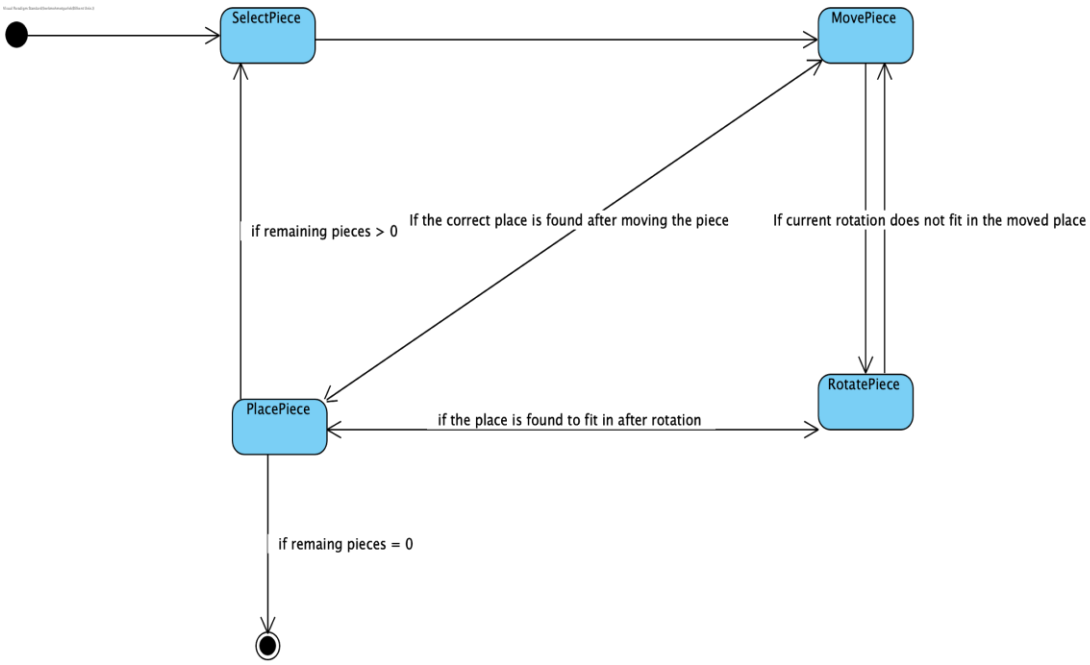


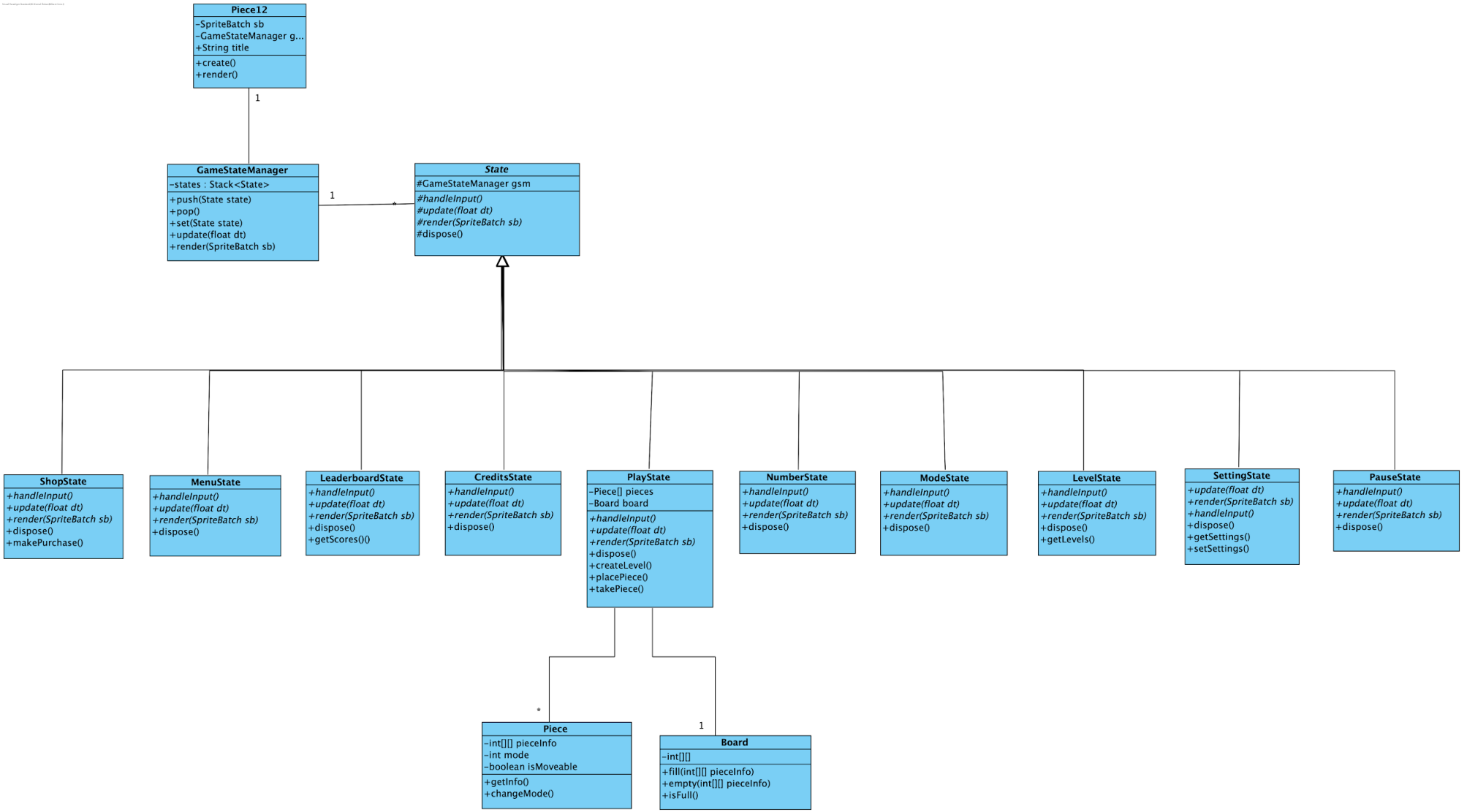
Figure 8: State Diagram #3 (Play Game)

The figure above represents the states while the user is playing the game. User looks to the pieces and selects 1 among 12 of them. Then moves the piece and checks whether the piece is placable. If it is not, user rotates the piece and either tries to place the piece to the previous location or move the piece to another location to place. After each placement, it is checked that are there any remaining pieces. After the 12th placement, the game ends.

5.3. Class Diagram

Class diagram of Piece12 is illustrated below. Currently we have nineteen classes that creates whole game.

Figure 9: Class Diagram (above) (click here to see larger)


Piece12
-SpriteBatch sb
-GameStateManager g...
+String title
+create()
+render()
1
GameStateManager
-states : Stack<State>
+push(State state)
+pop()
+set(State state)
+update(float dt)
+render(SpriteBatch sb)
1
*
State
#GameStateManager gsm
#handleInput()
#update(float dt)
#render(SpriteBatch sb)
#dispose()
ShopState
+handleInput()
+update(float dt)
+render(SpriteBatch sb)
+dispose()
+makePurchase()
MenuState
+handleInput()
+update(float dt)
+render(SpriteBatch sb)
+dispose()
LeaderboardState
+handleInput()
+update(float dt)
+render(SpriteBatch sb)
+dispose()
+getScores()()
CreditsState
+handleInput()
+update(float dt)
+render(SpriteBatch sb)
+dispose()
PlayState
-Piece[] pieces
-Board board
+handleInput()
+update(float dt)
+render(SpriteBatch sb)
+dispose()
+createLevel()
+placePiece()
+takePiece()
NumberState
+handleInput()
+update(float dt)
+render(SpriteBatch sb)
+dispose()
ModeState
+handleInput()
+update(float dt)
+render(SpriteBatch sb)
+dispose()
LevelState
+handleInput()
+update(float dt)
+render(SpriteBatch sb)
+dispose()
+getLevels()
SettingState
+update(float dt)
+render(SpriteBatch sb)
+handleInput()
+dispose()
+getSettings()
+setSettings()
PauseState
+handleInput()
+update(float dt)
+render(SpriteBatch sb)
+dispose()
*
Piece
-int[][] pieceInfo
-int mode
-boolean isMoveable
+getInfo()
+changeMode()
1
Board
-int[][]
+fill(int[][] pieceInfo)
+empty(int[][] pieceInfo)
+isFull()

Piece12 Class: This is the main class of our game. It initializes game itself this is the only purpose of this class.

State Class: This is an abstract class for gathering all game states together. As normal abstract classes, it has no instances. It has four abstract methods which are handleInput for handling all gui inputs in a one method, render for rendering gui elements, dispose for delete gui elements when not needed, update for updating screen or state every delta time.

GameStateManager Class: This class is one of the core classes of our game since it is the class that organize all other State classes. It has a stack of State classes so it can push a new state or pop old state or push and pop at the same by set state.

PlayState Class: This class is the core of the gameplay since this is the class where user play the game. It has array of Piece class and one Board so we can make calculations on them. It has handleInput method for handling gui operations such as pressing back button or pause button. It also have update method as other State classes have for updating gui. It has render for rendering gui elements. It has createLevel function that sets initial situations of levels. Lastly it has two core function such that placePiece and takePiece which are self explanatory.

Piece Class: Piece class is for representing real life puzzle piece in digital environment. It will have 2D integer array so we can visualize piece on the code. It will have isMoveable attribute so user cannot change initially given pieces on the board and it will have mode attribute so when user change its direction we can make calculations on that. It has getInfo method so we can know what piece is this in the

PlayState and it has another method named changeMode so when user wants to change its direction this method will change mod of the piece.

Board Class: This Board class is for representing real life board to the computer. It will have a 2D integer array such that this array will indicate where is empty on the board and where is occupied by 0's and 1's. It will have three methods. One of them will be fill method, which will change array element values 0 to 1 by looking given piece. Another is empty method that checks if board's given portion is empty or not. So that we can decide to call fill function. Another method is the isFull function. This function will check if board is full and pieces correctly placed.

ShopState Class: This state for showing shop menu and letting players to buy skins for their game. It has the all methods that other states have and additional makePurchase method so users can pay coins for skins.

MenuState Class: This state is the main menu state and its purpose is letting player to navigate other states and menus. It has all state methods.

LeaderboardState Class: This state shows the leaderboard for time race mode. It has all methods other states have and additional method called getScores to connecting database and getting scores.

CreditsState Class: This state will show credits of the game and has no other methods than other states.

NumberState Class: Purpose of this state is letting player to choose between multiplayer and singleplayer options. It has all methods that other states have.

ModeState Class: This state for choosing between three game modes. It has methods that other states have.

LevelState Class: This state for choosing level and showing stars to user that he e3arns in each level. It has all state methods and getLevels method getting all levels.

SettingState Class: This state for changing game settings. It has all state methods plus getSettings for getting current settings and setSettings for saving current settings.

PauseState Class: This state is pause screen of the game. It will be showed if user pauses game. It has all state methods.

6. User Interface

Here we illustrated some mockups of the game to give an idea how it will be shown to user.

6.1. Navigational Path

This diagram below, basically shows the navigational path of Piece12 game. It represents the transitions between different states by user's actions, like clicking a button or so.

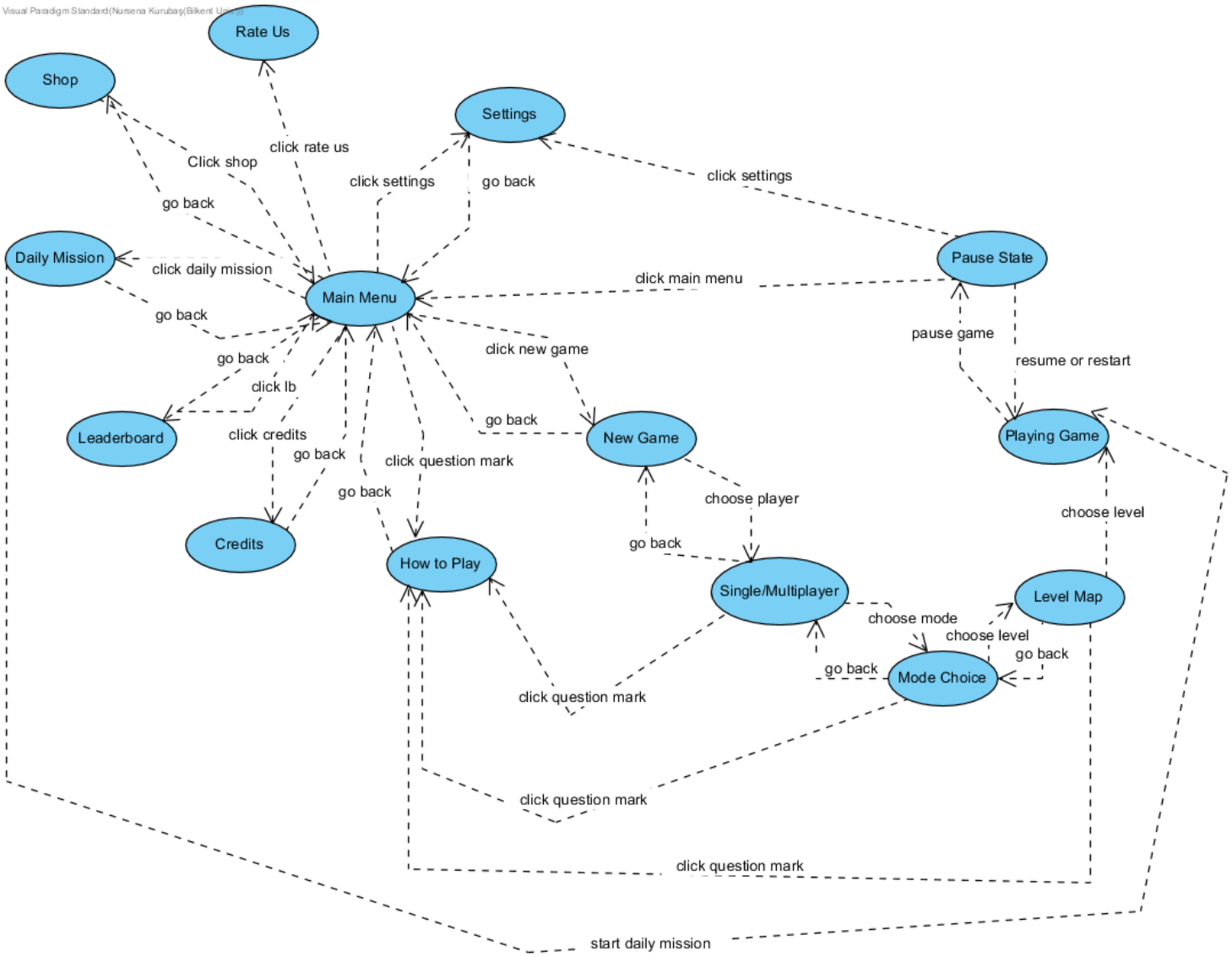


Figure 10: Navigational Path of The Game

This capture represents all actions that a user can perform through the game.

6.2. Screen Mockups

In this section we reveal our aimed User Interface by illustrating some captures we created by using Mockup tools.

6.2.1. Loading Page

This screen will be shown to user until the game loads.

Figure 11: Loading page of The Game

At top we show our logo and right behind it there is the name of our game.

Progress bar will show how much of the game has loaded.

6.2.2. Main Menu

Main menu will be shown to user when the game finishes loading.

Figure 12: Main Menu

This capture represents the main menu of the game. Four boxes at the middle leads the page that are written on them. Star symbol leads to “Rate Us”, hourglass leads to “Daily Mission”, shopping car leads to “Shop” and gear wheels lead to “Settings” page.

6.2.3. New Game

When user clicks “New Game” at the main menu, this page will be open.

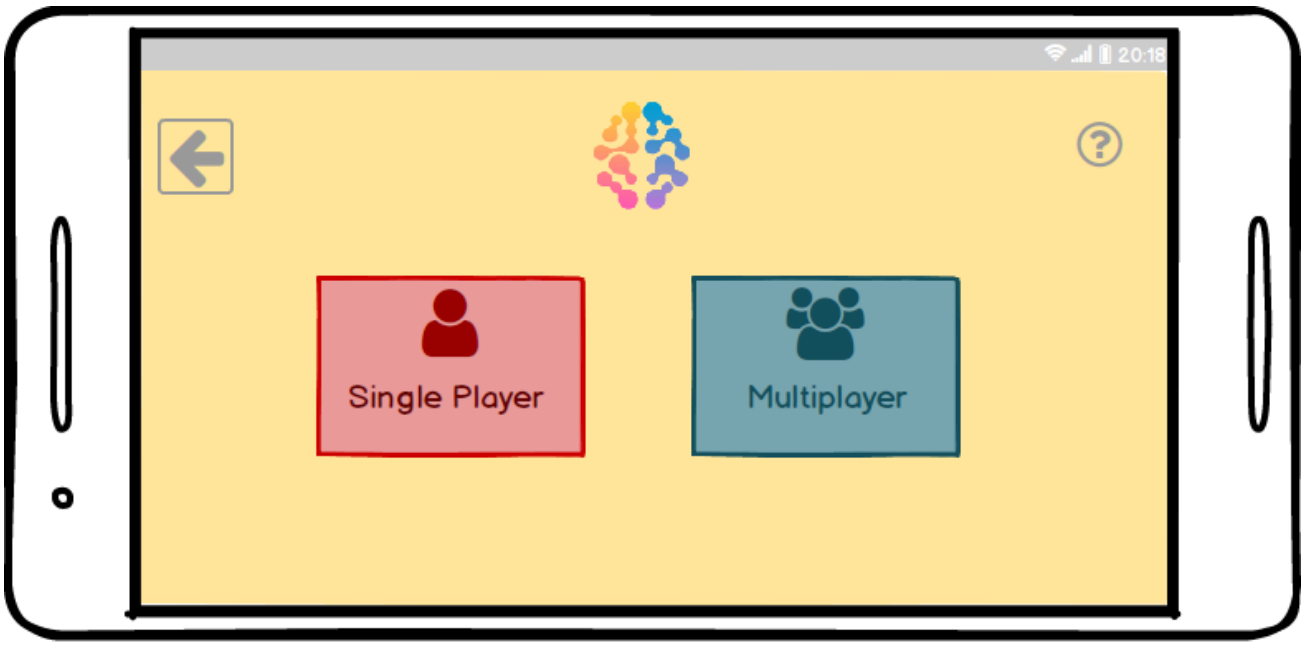


Figure 13: New Game Screen

User can choose one of the “Single Player” or “Multiplayer” options from here, go back to main menu by clicking the arrow or open “How to Play” by clicking question mark.

6.2.4. Mode List

This screen opens when user chooses between single or multiplayer.

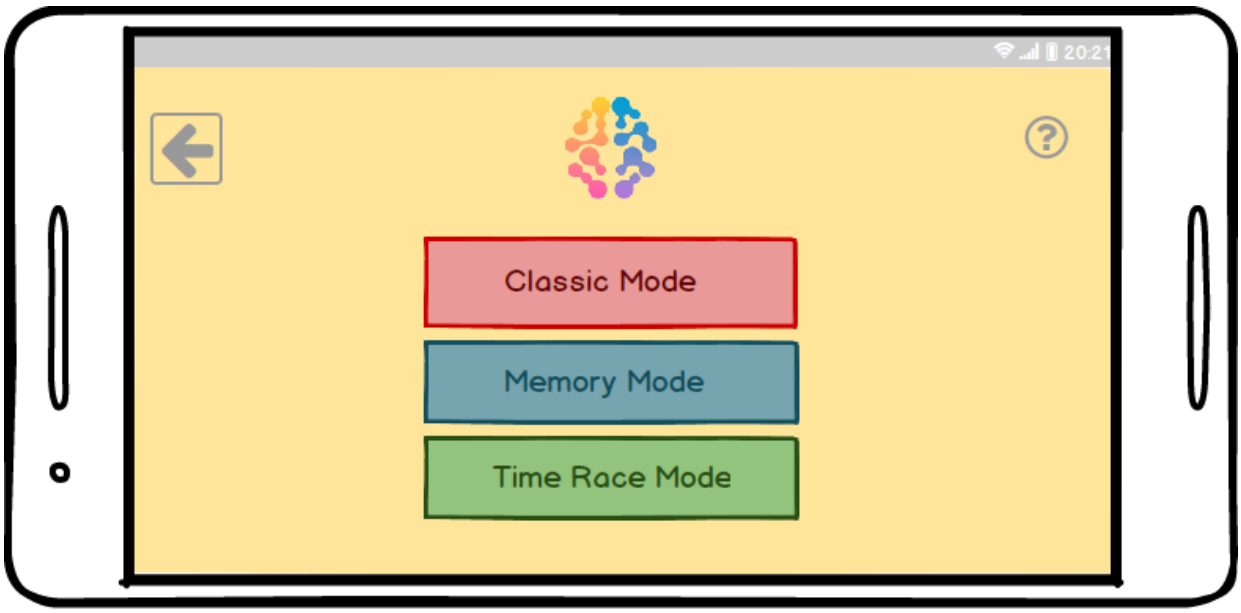


Figure 14: Mode List Screen

From this page user can choose the desired game mode, go back to previous page or click on how to play.

6.2.5. Level Map

After choosing a game mode this screen occurs for user to choose a level.

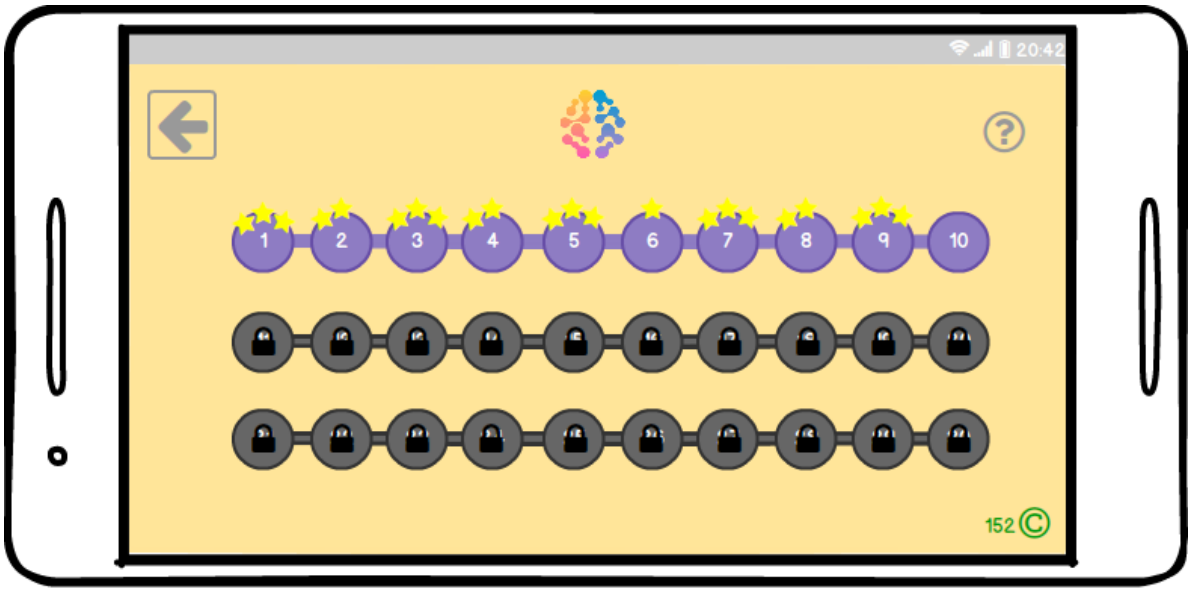


Figure 15: Level Map

Lock on the levels mean that user cannot play those levels until they success previous levels. Stars on the levels indicate success rate. User can also go back to mode list or how to play page from here.

6.2.6. Game Play

When player chooses a level a new game starts and this game play screen occurs. This screen is the core of our game.

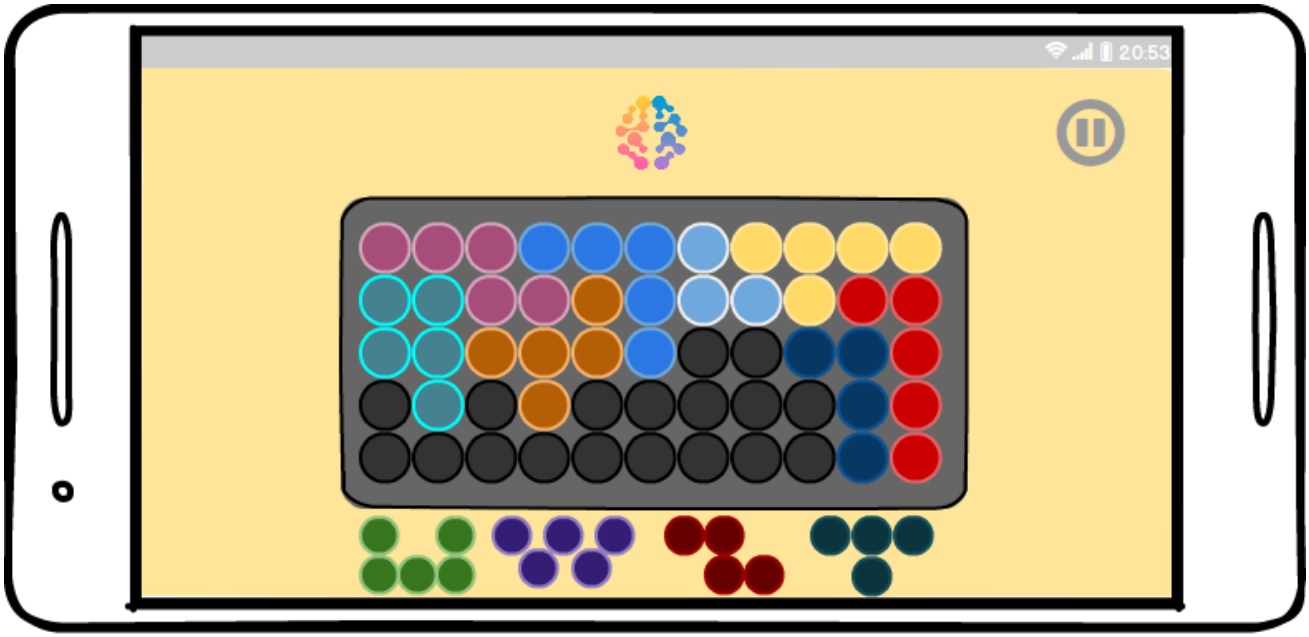


Figure 16: Game Play Screen

This is what our game will look like at the beginning. Blackened places on board indicate empty places. User can drag, rotate, flip and place the pieces below to the empty places on the board. The pieces that are already on the board cannot be moved by player. User can pause the game by clicking the button at right top.

6.2.7. Pause State

If user pauses the game this screen with options will appear.

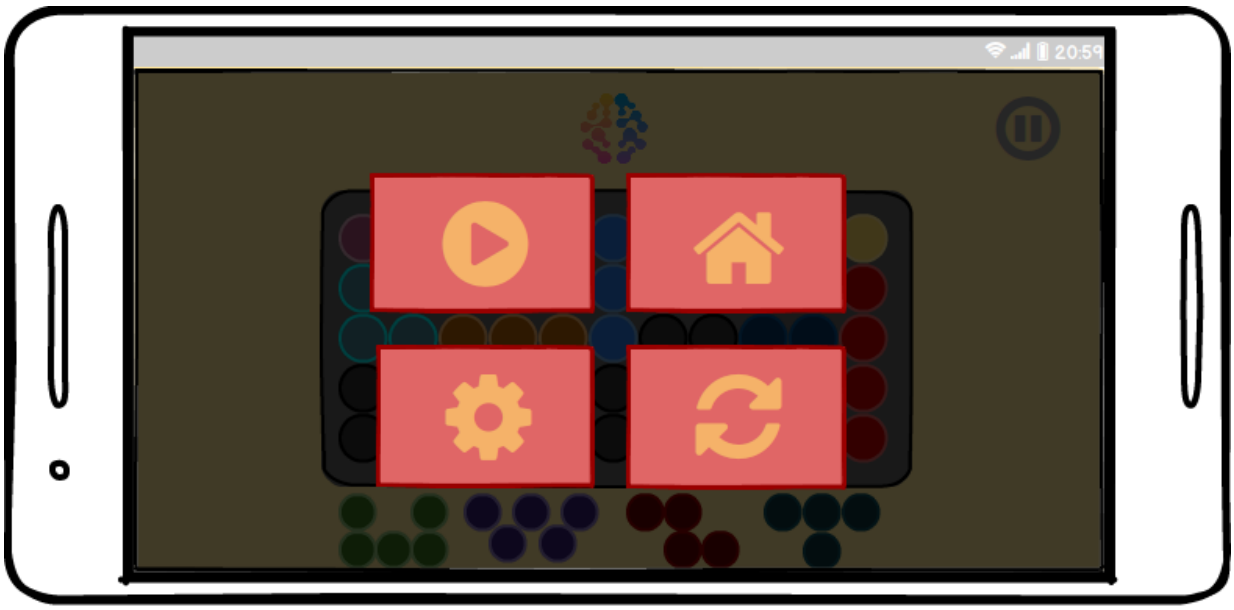


Figure 17: Pause Menu

Left top button resumes the game, right top button leads user to main menu, left bottom button opens settings and the button at right bottom restarts the level.

6.2.8. Shop

The shop screen of our game is illustrated below.

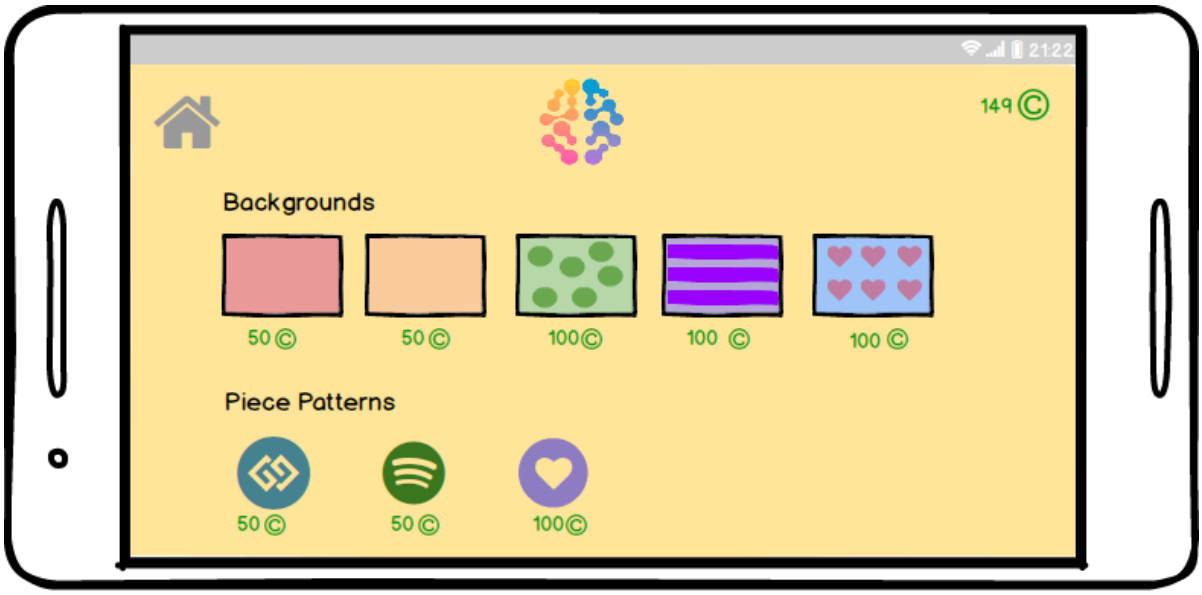


Figure 18: Shop

At right top, user's coins are shown. Each item in shop has different price that written under them. User can double click on the items to buy them.

6.2.9. Credits

This screen occurs when user clicks on the credits button from main menu.

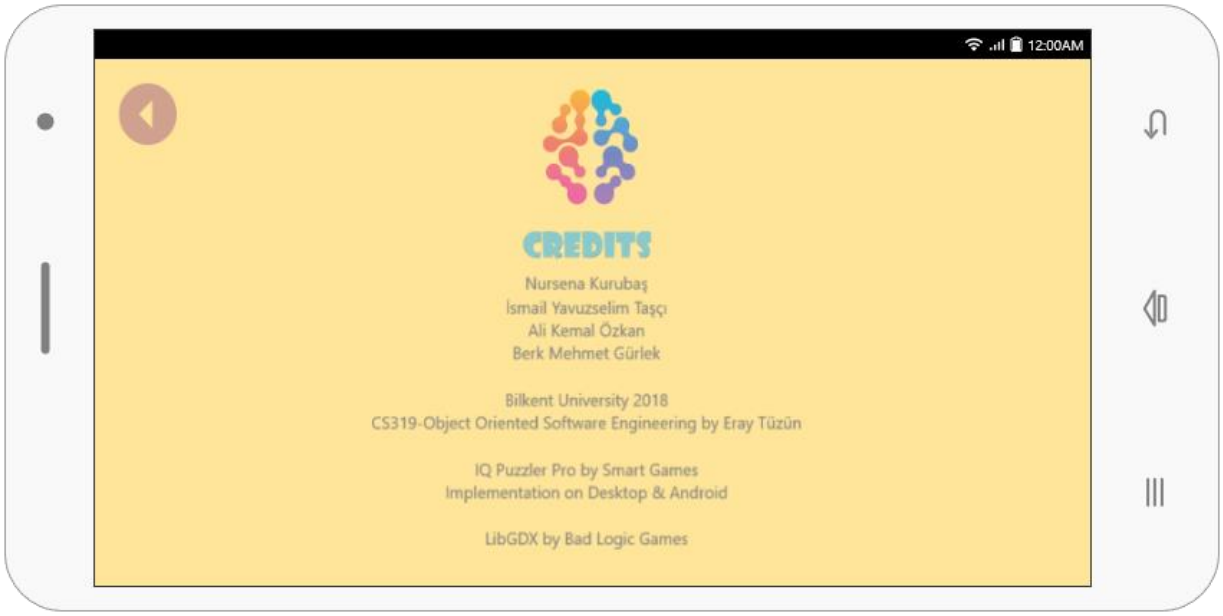


Figure 19: Credits

Credits page includes some information about developers and some references.

6.2.10. Find Opponent

We showed the single player game play user interface and multiplayer is almost the same except for this screen.

Figure 20: Choose Opponent Screen

From this page user can choose to play with an opponent from his/her Facebook friends or can simply play against a random person.

6.2.11. Settings

This screen shows the game settings that can be reached from Main menu or Pause State.

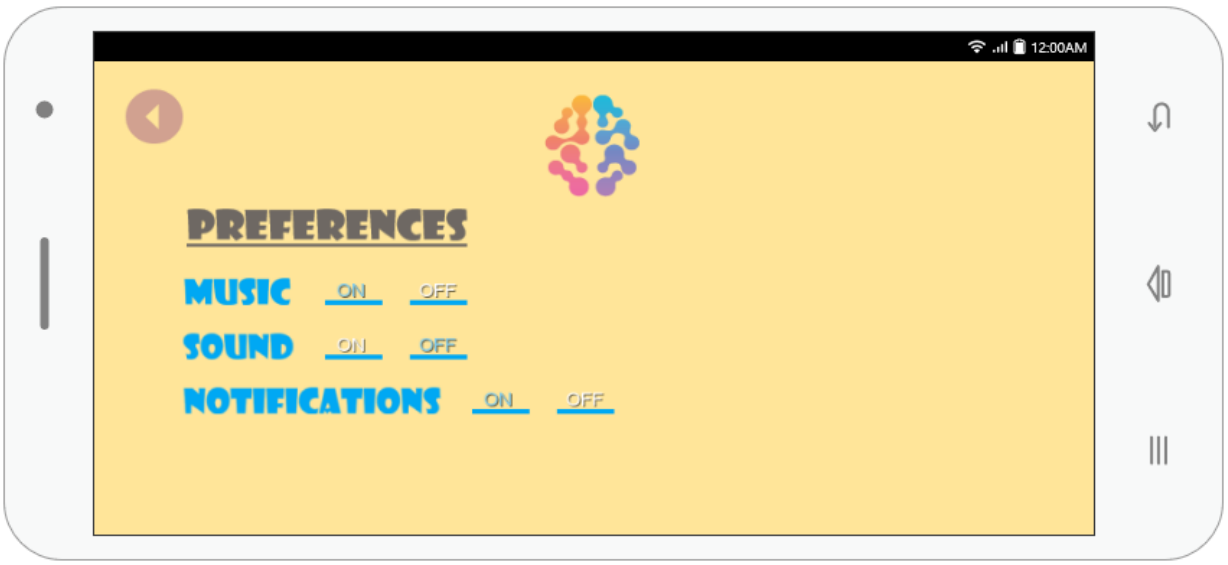


Figure 21: Settings

From this page, user can choose to open or close the music, sound effects and notifications. User can go back by clicking the arrow.

7. Improvement Summary

We implemented all classes almost completely except for PlayState and database-related functions. Our current PlayState has moveable, placeable, rotatable and flippable pieces and an empty board but no win perception and collision detection. We did not yet implemented the multiplayer function. We almost finished all of the user interface related implementation.