Mert Epsileli

Yusuf Samsum

Fırat Yıldız

Burak Korkmaz

Ege Hatırnaz

10/21/2018

BILKENT UNIVERSITY

ANALYSIS REPORT

CS319 | Section 1

Table of Contents

1.0 Introduction

Initially, we wanted to implement the “Katamino" game which is defined as a branch of "Pentomino" games. Historically, these game and other branches like Puzzles and Packings are based on American Professor Solomon W. Golomb, worked on this 1950’s. The game introduced to the public by Martin Gardner in his October 1965 Mathematical Games column in Scientific American. The name of games based on “Pentomino”, are generally finished with “-omino” which means in Ancient Greek “domino” and “kata" means “field”. That is kind of mathematical game (for children especially). The Game first published in 1994 as a board game by French game company Gigamic games.

The first version just includes classical game experience which will also exist in our implementation however we add two additional modes to make game more fun. As it is known, game structure is suitable for object-oriented design. This helps us to help new features and modes as well. The new features may be expressed as:

- Music option which create composition game duration
- Get hints from the system
- Time limitation for missions
- Leader board system which motivates player more
- Nick name usage in the game and rank considering of that
- Settings option
- Refresh the blocks
- Classical Mode (Hint only)
- Challenge Mode
- Dynamic Mode (Flexible game field)

Before players enter the game, they will enter their nickname. They will be allowed the game only if they have unused nickname. After that, game will start according to

their game mode selection. In the game, player will use the mouse arrow to move the blocks and leave the right place to complete the level. Game will progress level by level. If the player tries putting the block into wrong place, the player will not be allowed to leave the block in that area. Player may only leave the piece if the piece is suitable for that part of the game field. Players may refresh the main frame. Otherwise, player continue to play the game from next level. In each level, players will be introduced new game fields which have more spaces to fill. Finally, players will see the leader board which are ranked by considering of their nickname and time score.

In the implementation part, SQL language is taught to use for database implementations. For mock-up, we planned the use “Balsamiq" and "Visual Diagram” for diagrams. The game implementation will be in JAVA because JAVA is one of the best implementation fields of object-oriented programming. However, JAVA and C++ are not that good for GUI implementation. For GUI, Javascript WebGL will be used.

2.0 Overview

2.1 Game Play

Katamino is a children game which improves the visual memory. In the project, Katamino is evolved into a game with more features. Player selects the ‘Start’ button to start game. Before the starting, player needs to choose a nickname which is not used before. After that, the player comes to mode screen. In this screen, player chooses one of three modes. The solution set will be determined after the mode selection.

For “Classic Mode”, the game starts from 3. level which means the stick will be 3. stage of the table and there will be an 3x5 empty area. The player tries to fill the empty part of the game table with given and useful blocks in minimum time as possible. When the given area is filled, player will level up. New level will be 4 and there will be an 4x5 empty area. This time, player will try to fill this empty part with new blocks which are given. The game will go on like this until the last level which is 13. After the end of the game, the player’s

nickname and finishing time will be added to leaderboard. Player will be able to see his/her rank by using leaderboard. This game mode also includes refresh option which offers a new solution set for players.

In “Challenge Mode”, gameplay and player’s aim will not change. Different from classic mode, there will be 2 more blocks which are useless in the block set. Player will not know which are useless. The game will be harder because if the player tries to fill the empty area with these useless blocks, he will waste his/her time.

“Dynamic Mode” will be designed for players to change the dimensions of Katamino for trying new experiences. In this game mode, a player will be able to change the dimensions of the table. After this change, new proper blocks will be given to user with respect to dimensions of empty area. Player will try to fill this empty part with these blocks in minimum time.

2.2 Modes

There will be three different game modes for players to have varied experiences through our game. “Classic Mode” is the Original Katamino game with time extension. Players can see their results and compare their ranking in the game which makes our game even more competitive than the original game. Refresh option offers players a new solution set and building blocks with a penalty. In “Challenge Mode” players face with two additional blocks which are useless in game. It is going to be difficult for players to decide which blocks are useless in the game. “Dynamic Mode” will offer players to determine the dimensions of the board. For all the game modes, players need to solve the Katamino against the clock.

2.3 Leaderboard

Challenges make a person to put an effort to improve oneself. They give meaning to the effort that oneself spend. As challenges become solved, comparisons take the stage. Comparisons are the basis of the challenges. Players try proving themselves in these comparisons with playing repeatedly. To make Katamino even more competitive and attractive “Leaderboard

System" will be used to increase the challenge between players. The scores of the players will be hold according to the time that they spend to finish the game. Players records will be classed according to the game mode that being played. Players have a chance to compare their scores with other players. Also, players can observe their development in Katamino by shoe individual scores. Our game will not let players to solve Katamino once, it will motivate them to improve themselves, and our game by way of our Game's Learning Machine.

2.4 Time

Time will be recorded in terms of seconds. If a player uses hints to solve Katamino, 10 seconds will be added to the time as a penalty. Addition to that refresh option comes with a penalty of 15 seconds. In gameplay, time will be recorded at the right top of the screen. For the challenge mode, significant time will be given to a player to complete the challenge such as different expanding sizes of field in each play. Players who use their time efficiently may transfer their time for the next game. At the end of the game, the time that being spent to solve Katamino will be the result of a player in the leaderboard.

2.5 Hints

If a player gets stuck in solving Katamino, hints will be available to guide. Solution path will be highlighted for player. Hints will be limited for each level. Each good stuff comes with its opposites. Therefore, hints cause for increasing time in gameplay. Each time hint used, time will be added 10 seconds.

2.6 Refresh

Refresh option offers players to a new solution set, and new blocks in the game. It has 15 seconds penalty in time for every new solution.

2.7 Player

Each player in the game must have a unique nickname to compete other players. Their playing time will be recorded during gameplay. At the end of the gameplay, players can see their score and their ranking in the leaderboard. Moreover, players can see their previous results in the game to see their progress.

2.8 Blocks

There are 12 different types of blocks which consist of five 1x1 square. Each block has different colors for identification. In classic mode, number of blocks will be same as the game level for limiting players options. In challenge and dynamic mode extra two blocks will be given to increase possibilities, and at the same the difficulty.

2.9 Board

The board has dimensions 5x13 in classic and challenge mode, and the stick determines game level with limiting the boards dimensions. Players can able to set the dimensions of board in the dynamic mode.

2.10 Stick

The stick will determine the game level. It limits the board with changing dimensions. It starts from 5x3 squares all the way up to 5x13 squares. When the level has been completed, the stick will be removed 1 step and give players 5 new squares in other words a new dimension, and one or more blocks according to the game mode that being played.

2.11 Settings

In settings section player can able to change music, and volume level. Language options are available to make the game global.

2.12 Rules

Katamino has 12 different blocks with unique shapes, a stick to separate the playing field, and a board. In Classic and Challenge Mode, game field will start from 3x5 squares all the way up to 13x5 squares. Challenge Mode offers two

additional useless blocks for a harder experience. In Dynamic Mode, players can determine the board themselves. The games aim is complete the squares with given blocks in the shortest time possible.

3.0 Functional Requirements

3.1 Start the Game

On the main menu, after choosing whether the game will be Normal or Challenge Mode. After selecting the game mode, user should be able to start the game with selected mode.

Normal and Challenge Mode differ by Challenge Mode having extra random pieces which does not fit into solution. (How to play those modes are explained further on 3.3 part.)

View is changed into “Play” view. User will have an option to return to menu on the top left corner of the screen.

On the opposite side of the screen, time will be shown to player on top right corner.

At the bottom, there will be Inventory where the user will select the piece to play.

On the right side of Inventory, user will have the option to get a hint or get a refresh. Hint suggests the player a piece to use, suitable to the rules of the game. Refresh gives the player a different strategy to play if the user does not like it.

On the bottom right part of the game, user will have option to change volume options.

All those controls are utilized by mouse at the player’s control.

3.2 Solutions

On the main menu, there is an option to look at the solutions. This part is just like the real Katamino where a brochure is included with all possible solutions.

On this part, user will have an option to return to menu at the top-left corner of the screen.

At the left side, there will be a scroll, containing the list of levels and their solutions.

At the right side, solutions will be shown to user.

User will be given random tips about the game at the bottom of this screen.

3.3 How to Play

How to Play is located on the main menu and it explains the rules and purpose of the game to user:

- The player should take the guidebook for deciding which level to play.
- Then, the player should start with the first level, which is 3, and he/she should use only given blocks which specified by the guidebook.
- The player should pass each level after he/she finished it.
- The levels' specifications should be followed by the guidebook.

Since the game is controlled by mouse and clicking the pieces & buttons, we won't have to include a controller schematic.

Player will have an option to return to main menu at the top-left of the screen.

3.4 Leaderboards

To keep the spirit of competitiveness going, we will include a Leaderboard.

This Leaderboard will be split into three parts as follows Normal Mode, Challenge Mode and Dynamic Mode.

This leaderboard will include the player's rank in the leaderboard, player's name and the corresponding time standing for player's rank. It will be same for Challenge Mode and Dynamic mode as well, but we will keep track of it on a different leaderboard.

At the bottom of the screen player will be given a random tip about the game.

Player will have an option to return to menu at the top-left corner.

4.0 Nonfunctional Requirements

4.1 Time

We want the stopwatch displayed on the top of the game as accurate as possible. Frame rates play important role on this part. Therefore, we must make sure we are displaying and recording the time quickly and stop it as quickly as much as possible just when the player is finished playing.

4.2 Simplicity

We want our game to be universal as possible as Katamino is not dependent on any language. Just shapes and symbols define the game itself. So, when designing UI, we should try to avoid using any language except for the menus. For instance, “Return to Menu” can be displayed as a “returning arrow” or “home” symbol. This will provide us a simpler user interface design.

4.3 Competitive

User should be able to compete with different players, so game should be accessible and easily playable by many players. The competitiveness will sweeten the game and perhaps making the player more attention to their time and piece management.

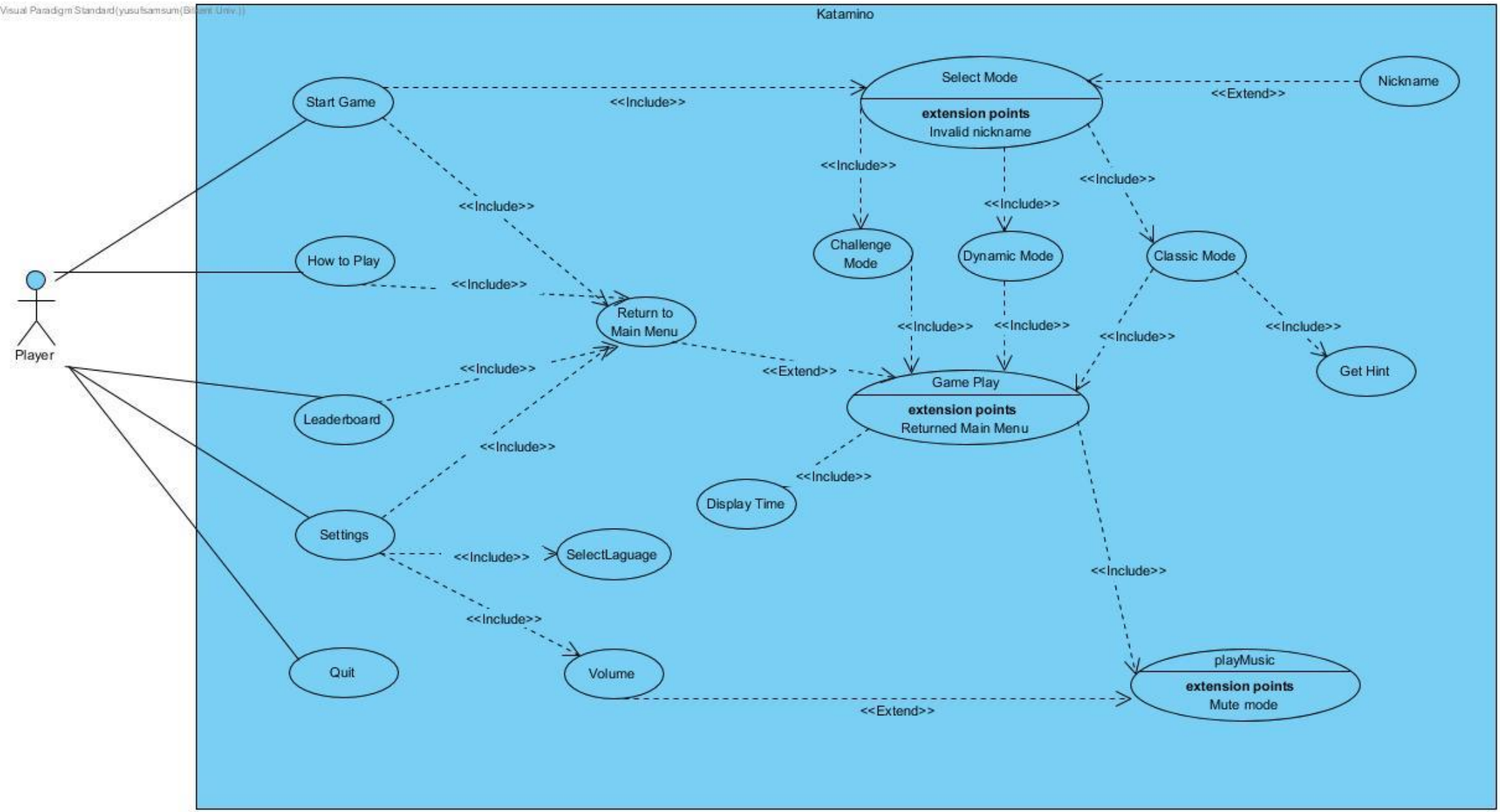
4.4 Game’s Learning Machine

Different valid solutions given by player can improve the list of solutions by expanding the solution list. In short, if a solution is valid and not stated in the solution list, we add it in. This way the game would adapt to different solutions if they are within the rules of the game.

5.0 System Models

Use case model can be found on the next page.

5.1 Use Case Model



There are textual descriptions of the use case with different scenarios.

5.1.1 Use Case Name: Play Game

Participating actors: Player

Flow of events:

1. Player selects “Start Game”.
2. System wants a Nickname from player.
3. Player enters a Nickname.
4. System controls whether the Nickname is proper, or not by searching database. If it is proper (not used before), player is directed to “Select Mode”. If it is not system wants a proper name until it is proper.

Entry condition: Player must be in main menu.

Exit condition: Player selects “Quit”, OR

Player clicks the cross sign on the right upper side.

5.1.2 Use Case Name: Select a Mode

Participating actors: Player

Flow of events:

1. Player selects game mode, which are “Challenge Mode”, “Classic Mode” and “Dynamic Mode”.
2. Game starts, “Game Play” is opened.

Entry condition: Nickname must be proper.

Exit condition: Player selects “Return to Main Menu”, OR

Player clicks the cross sign on the right upper side.

5.1.3 Use Case Name: Starting a Game

Participating actors: Player

Alternative flows of events:

If player selects “Classic Mode”:

1. Player gets the blocks from block part.
2. Useful blocks are given to player.
3. Player tries to place the given blocks to appropriate locations.
3. When blocks are placed appropriately, the level finishes.
4. New level opens automatically.
5. Same operations are done for 13 levels.
6. Game ends.

If player selects “Challenge Mode”:

1. Player gets the blocks from block part.
2. Useful blocks together with 2 more useless blocks are given to player.
3. Player tries to place the given blocks to appropriate locations.
4. When blocks are placed appropriately, the level finishes.
5. New level opens automatically.
6. Same operations are done for 13 levels.
7. Game ends.

If player selects “Dynamic Mode”:

1. Player changes the dimensions of the game table.

2. The proper blocks, which can fill the new game table, are given to player.

3. Player tries to place the given blocks to appropriate locations.

4. When blocks are placed appropriately, the level finishes.

5. Game ends.

Entry condition: Player must choose a game mode.

Exit condition: Player clicks the cross sign on the right upper side.

5.1.4 How to Play

Participation actors: Player

Flow of events:

1. Player chooses “How to Play”.

2. A new screen which includes game instruction is opened.

Entry condition: Player must be in main menu.

Exit condition: Player selects “Return to Main Menu”, OR

Player clicks the cross sign on the right upper side.

5.1.5 Leaderboard

Participation actors: Player

Flow of events:

1. Player chooses “Leaderboard”.

2. The nickname and time information are taken from database by the system.

3. This information is compared.

4. Players’ ranking is determined

5. A new screen which includes ranking of players are opened.

Entry condition: Player must be in main menu.

Exit condition: Player selects “Return to Main Menu”, OR

Player clicks the cross sign on the right upper side.

5.1.6 Settings

Participation actors: Player

Flow of events:

1. Player selects “Settings”.
2. System displays “Volume”, “Voice” and “Select Language” settings.
3. Player chooses the options which he/she wants.
4. Player confirms his/her choices.

Entry condition: Player must be in main menu.

Exit condition: Player selects “Return to Main Menu”, OR

Player clicks the cross sign on the right upper side.

5.1.7 Getting a Hit

Participation actors: Player

Flow of events:

1. Player selects “Get Hint” button which is located on the left bottom of the “Game Play” screen.
2. System find the solution of current level.
3. System place one of the blocks to proper location.
4. Player’s time increases as punishment.
5. Player continue to play.

Entry condition: Player must be in “Game Play” of “Classic Mode”.

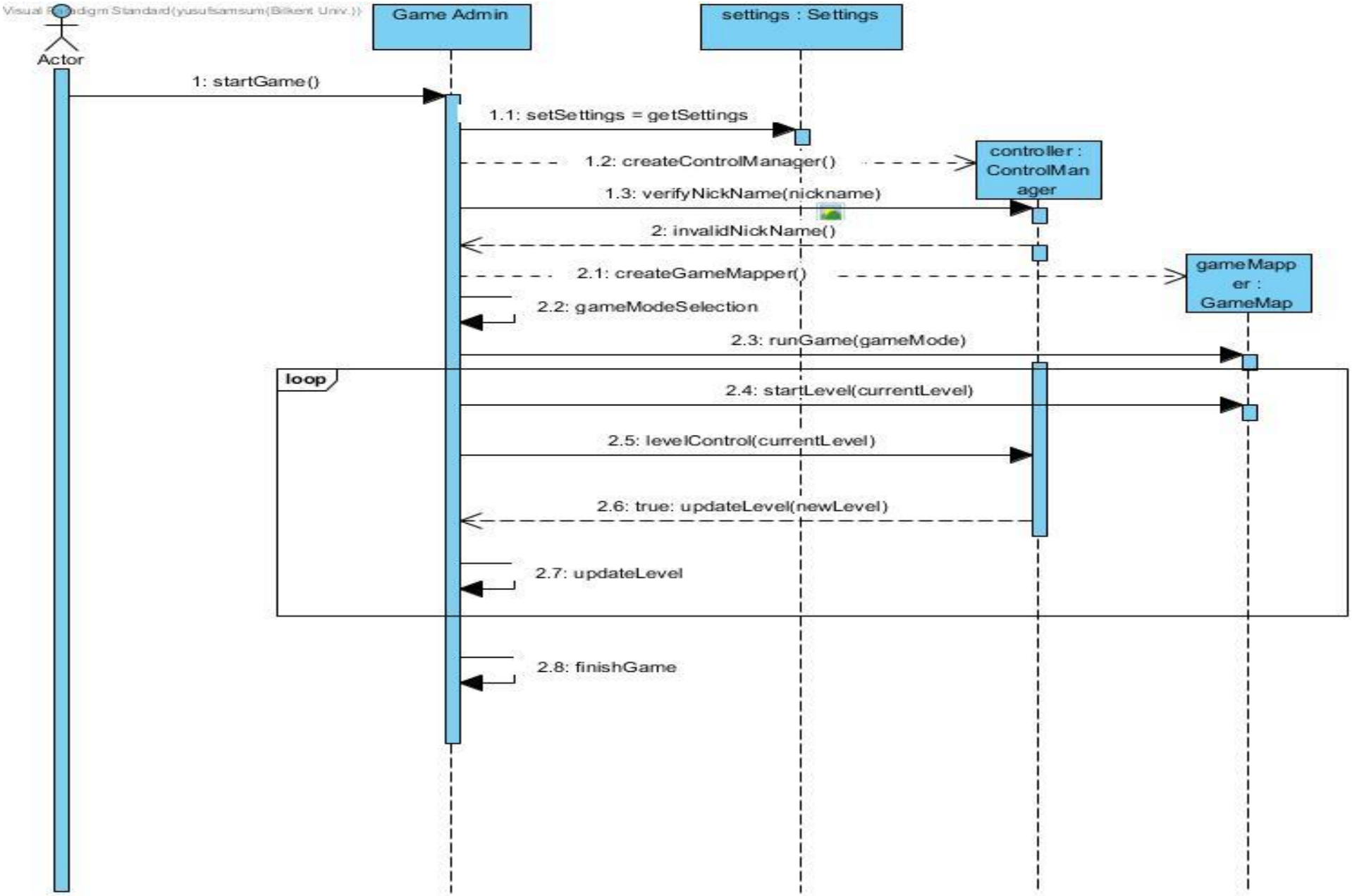
Exit condition: -

5.2 Dynamic Model

5.2.1 Sequence Diagram

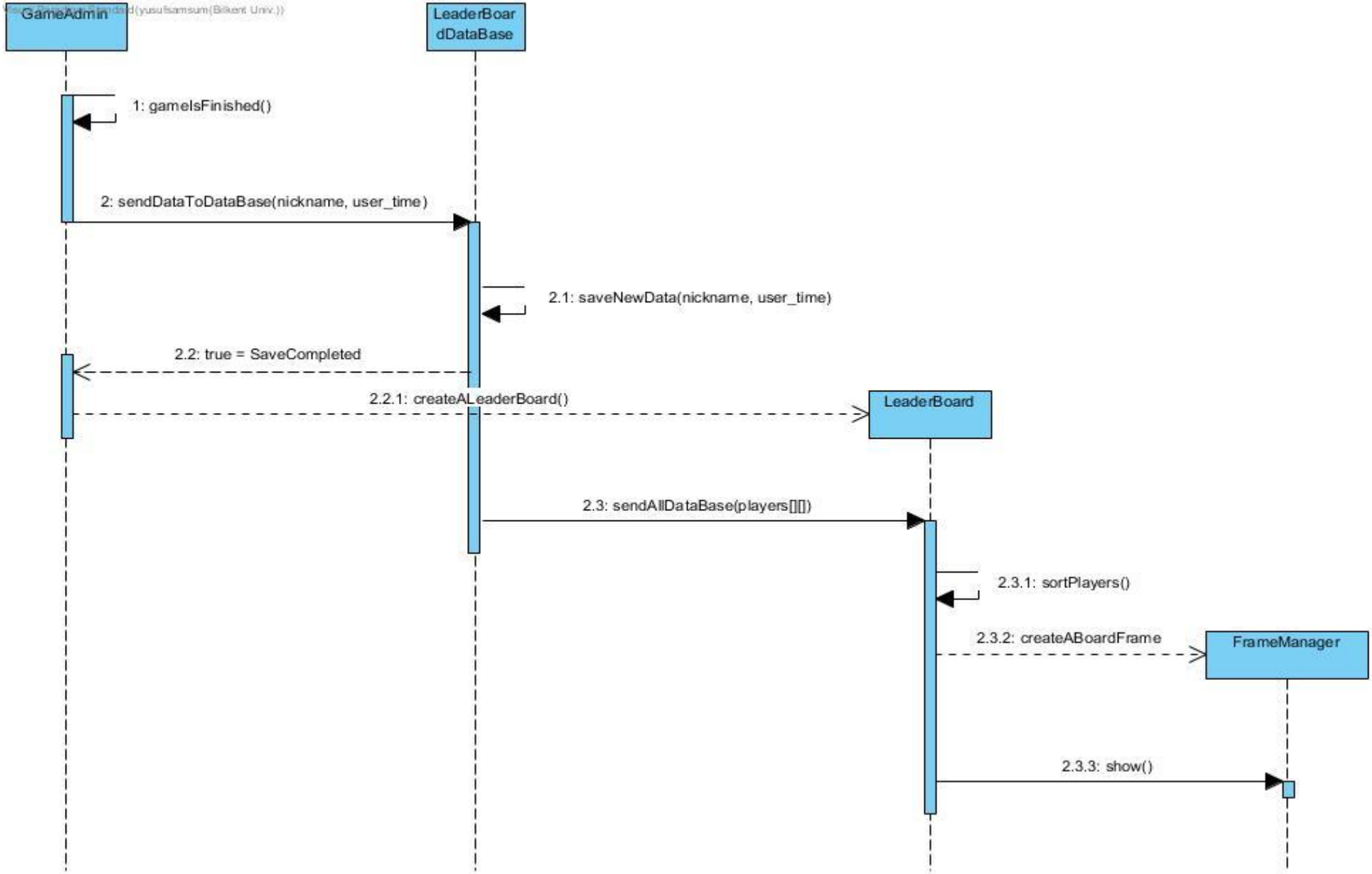
Sequence diagrams are shown with different scenarios at below.

5.2.1.1 Play the Game



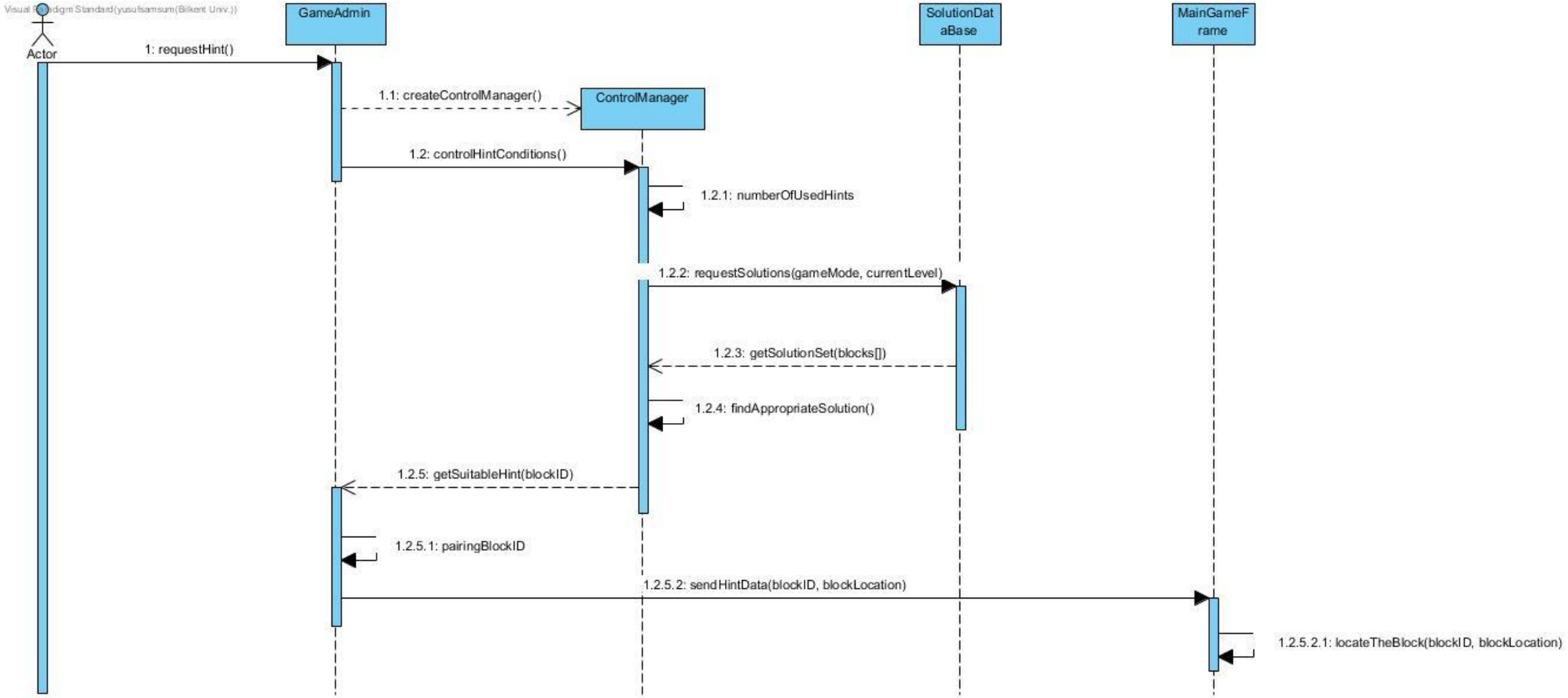
First, actor starts the game from the initial screen. Players have a chance to open the settings from the main screen. There is a quit game option in the main frame too. Game will be managed by the GameAdmin. That is the part which user communicates and controls the game generally so that is a kind of game manager. After, player pushes the start button, he enters his nickname. What's more, the existence of the nick name entered will be checked by Controller. If the name already exists, player will be returned to nickname screen to enter the correct one. Controller will also be responsible all the control cases inside the game such as "is the game finished or is the level completed". GameAdmin is going to create the Controller. Then, player sees the game mode selection screen. By the choosing of game mode, Game admin will create GameMapper to display the game playing. While actor plays the game, the game screen will be renewed considering of the game level, stages and the actions of the game. All these activities have been controlled by GameAdmin which provides the game sustainability.

5.2.1.2 Demonstration of Leaderboard



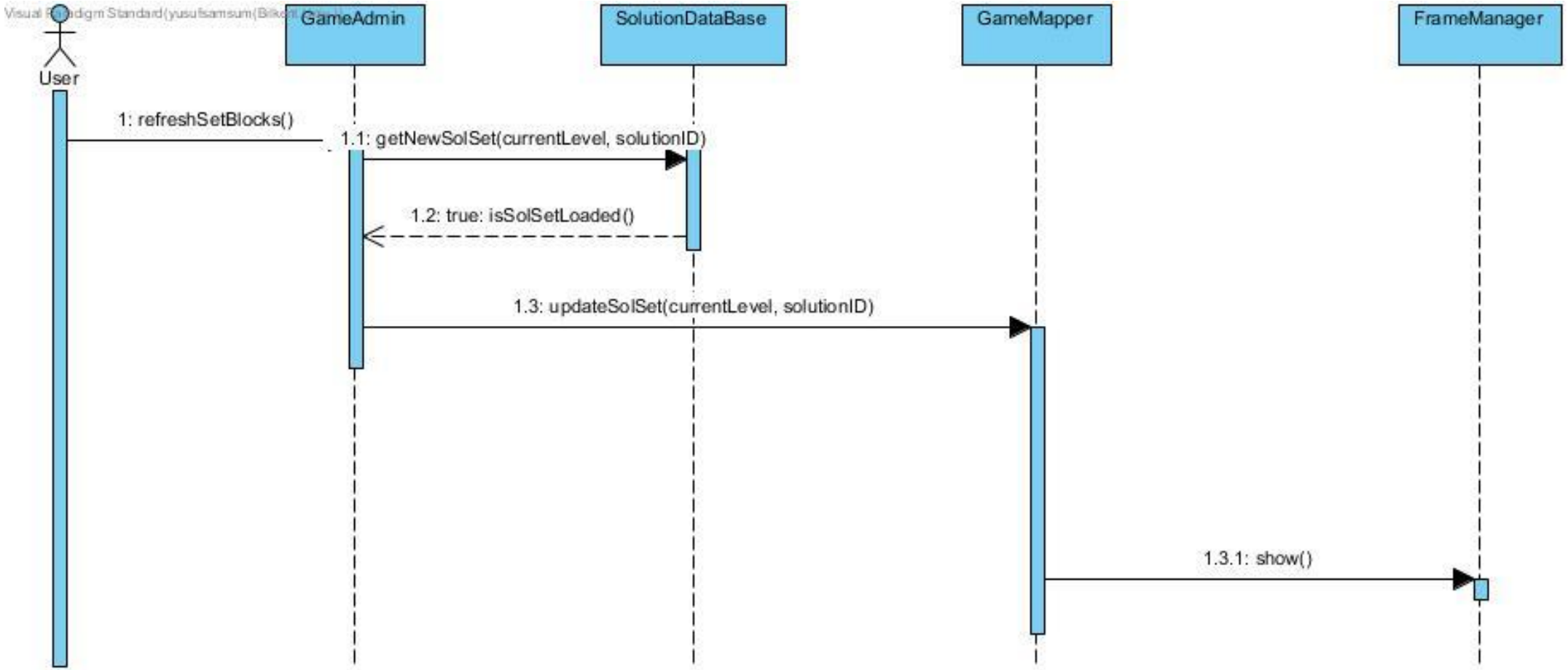
In the end of game, all the players will be ranked depending on their time usage. Players' information which are names of players and time scores of them. This ranking will be held by the "Leader Board Database". It also saves the data each time by the help of saveNewData(nickname, score) method. When a player completes the game its name and time data (as score) will be sent to Leader Board Database. Leader Board Database saves also the new data came from GameAdmin and it checks whether the data saved with isSaveCompleted() method. If isSaveCompleted() returns true, the "LeaderBoard" will be created by GameAdmin. After Leaderboard gets data from database it will sort the score by the help of sortPlayers() method. Finally, the leader board will disappear in leader board frame which is shown at the end of the all games.

5.2.1.3 Getting Hints



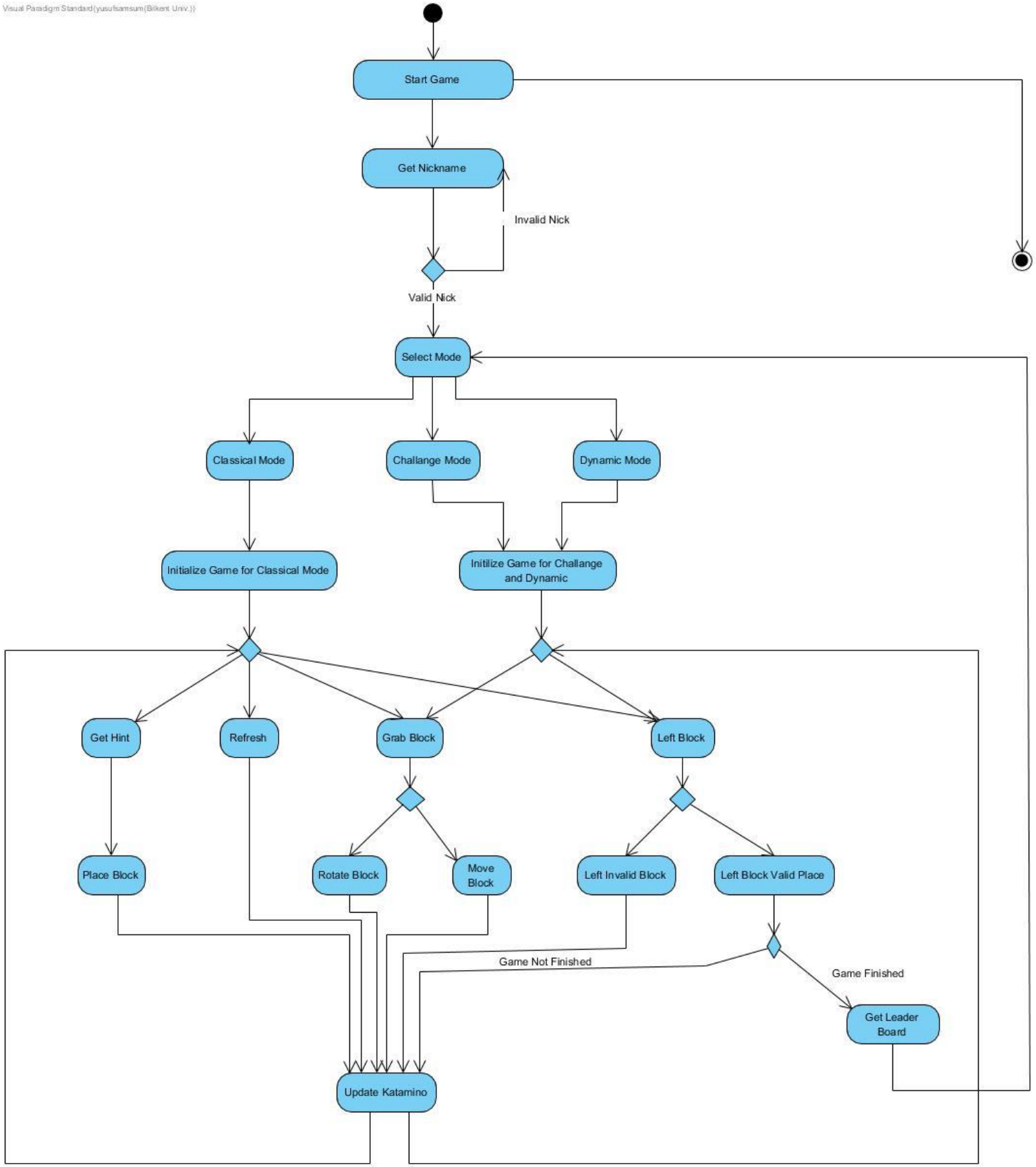
When player was unable to solve the problem during the game, it has chance to get hints from the system which cost a point loss in the classical mode. By the activating hint button, requestHint() method will send a signal to GameAdmin and GameAdmin will create ControlManager(this will be a part of Controller which are already control all datas in the game such as is the nickname valid for playing) directly. It will check that if the conditions in the game are ready for giving hint such as if there is a suitable place or if the field is already full by the help of controlHintConditions() method. If everything fine, it will return true otherwise false. ControlManager will also count the number of hints given in case of giving more hints than the number of blocks. SolutionsDatabase is a place which has all the solutions which are add by designers into the system. In order to give hints, requestSolutions(gameMode, currentLevel) method will be used to get data from SolutionsDatabase with its suitable game mode and level. After the ControlManager decide the suitable solution with findAppropriateSolution() method, that will send the id of the block which will be hint to GameAdmin with getSuitableHint(blockID) method. In the final step, sendHintData(blockID, blockLocation) method will send the hint block's ID and the location it will be placed to the MainGameFrame and that is locate the block considering of locateTheBlock(blockID, blockLocation) method.

5.2.1.4 Refreshing the Solution Set

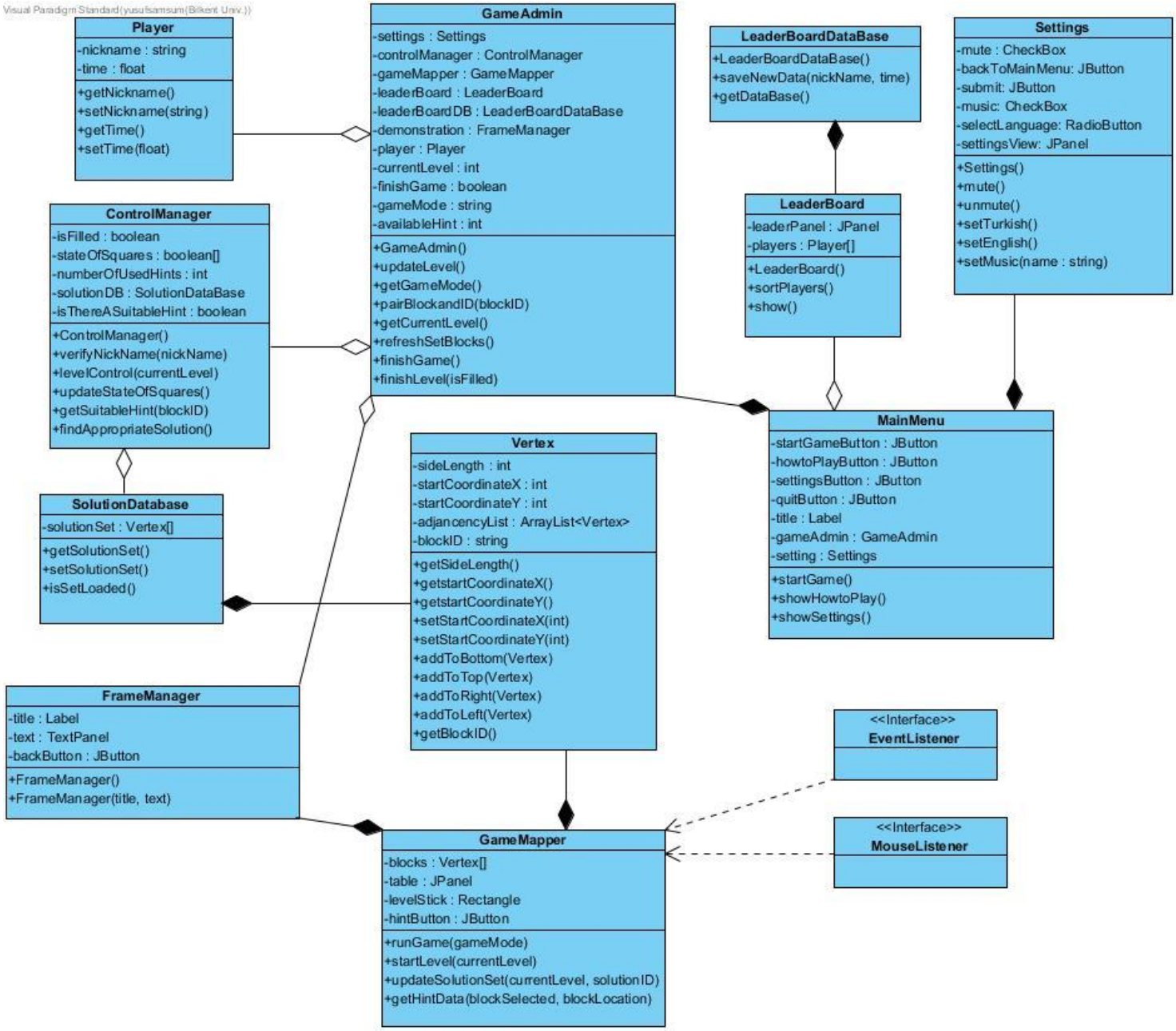


During the game, players also have chance to refresh their pieces used in the game. According to size of the field, players are given blocks to fill the area and there are more than one solution options for a size of field. Players who is not ok with the solution that game offers will have chance to get another random solution pack. By the help of refreshSetBlocks() method, players communicate with the GameAdmin which controls the game. From the “SolutionDataBase”, which contains whole bunch of solution sets, GameAdmin will get the another solution set with getNewSolSet(currentLevel, solutionID) method. It gets the new solution with solutionID because each solution has kept by a specific ID according to level which represents the size of field. After, solution set controlled by isSolSetLoaded() method, updateSolSet(currentLevel, solutionID) method will send the data to GameMapper which controls the game field related with the new set of solution. Finally, that will be displayed in the frame with show() method.

5.2.2 Activity Diagram



5.3 Object and Class Model



GameAdmin

This class will administer the game. This is the place where all selections will be got by the player. Also, game administrator will update the levels, will be responsible of finishing level and game. In addition, game mode will be got from there and send to the game mapper.

Game admin will be a bridge between control manager and user. When a user enters a nickname, control manager gets this information from game admin. Also, for refreshing solution set or getting hints, game admin will get the block IDs from solution database and control manager. After that operation, game admin will pair the block IDs with vertices and send the data to game mapper.

GameMapper

Game mapper is the last connection between Frame Manager and the other classes.

Game mapper determines the board by runGame method. This method determines the mode of the game and create a board with respect to input. If the input is classic or challenge mode, board will be classic Katamino board as mentioned in overview. However, if the input is dynamic mode, board will change dynamically with respect to level and the solution set.

runGame method will also trigger startLevel method. Start level method will update with respect to an input that comes from game administrator. If a level is finished and another will be begun, the level will be started by startLevel and runGame methods.

When a new level starts, solution set will be updated by updateSolutionSet method. This method takes current level and solution set ID from the game administrator. Solution set ID will be matched with solution database and the solution set will be showed after this pairing process.

Also, if user requests a hint from game and control manager will provide the hint data, game mapper takes the block's location and shows on the board as selected block.

Control Manager

This class controls the game and user interactions such as requesting hints, validity of nicknames, whether level is finished or not and state of the board's squares.

Nicknames will be checked by leaderboard database class. If there is a match, nickname will be invalid. Otherwise, nickname will be accepted by the control manager.

Level control will be done by two methods: levelControl and updateStateOfSquares. After every operation which is done by the user, state of the board's squares will be updated. Then, level control will be checked by the ratio of the filling situation of the squares. If the ratio is 100%, the control manager will send new level input to the game admin.

Also, control manager will check applicability of the hints. The user can request only one hit in each level. Control manager will check that if there is remaining hint or not at first. Then, if the player has right to ask a hint, control manager will find an appropriate solution by using solution database. If there is an appropriate solution based on filled squares and solution database, game admin will get a block id from control manager.

MainMenu

This class will be responsible for startup screen. This screen will include 'Start Game', 'How to Play', 'Settings', 'Leaderboard' and 'Quit' buttons. This class will create these buttons. When player chooses one of these buttons, MainMenu class will show the context of that choice.

Settings

This class will contain settings. The player will be able to choose language, volume level and music. Settings class will manage these choices.

Vertex

The blocks will be determined by many direct graphs and there will be a vertex class which includes adjacency list of a vertex. Vertex has block IDs and coordinates of a vertex which will be kept for creating squares for graphics.

SolutionDatabase

SolutionDatabase class will provide connection between IDE and solution database. This database will be prepared by developers and keep the blocks and proper locations of these blocks with respect to levels and game modes. When player selects the hint button, ControlManager class will request a hint from this class for that level. After that, this class will send a solution to ControlManager class and this hint will be demonstrated to player.

Player

This class will manage player features. When a new player starts to game, a new player object will be created to access to this player from other classes. This player will have two properties which are nickname and finishing time.

Leaderboard

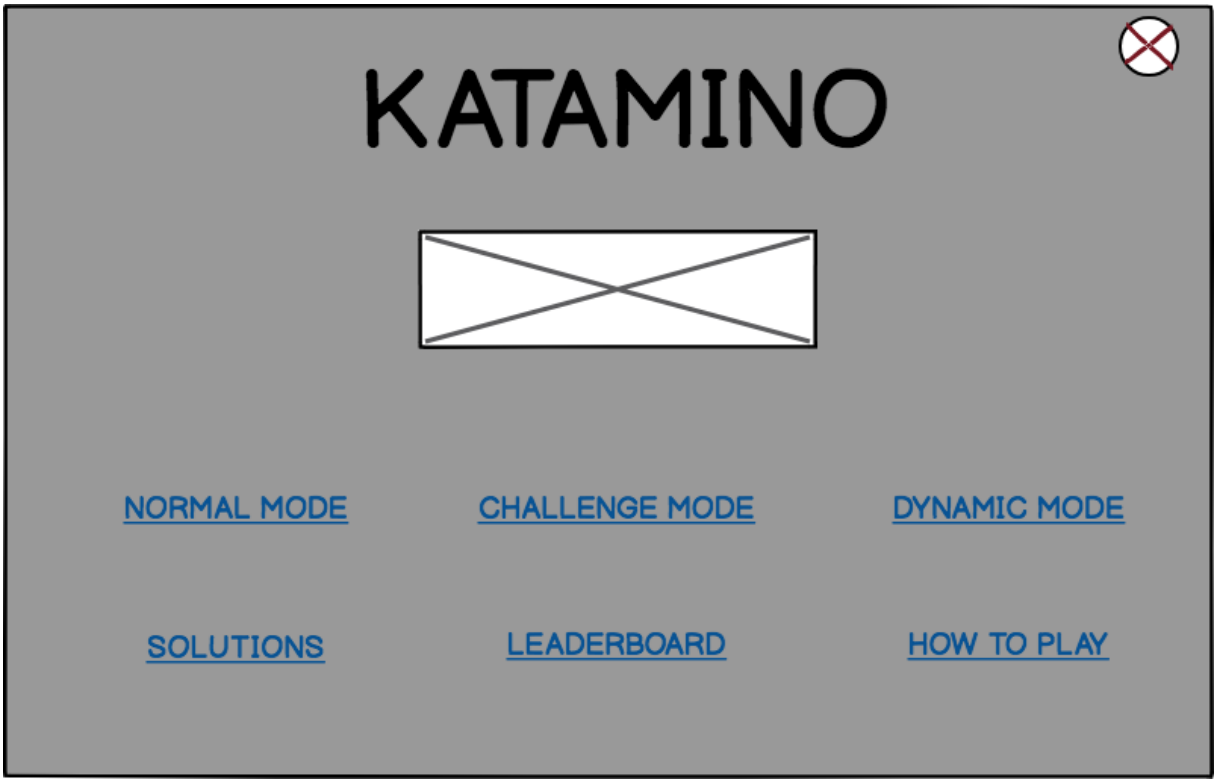
This class will take users' name and time information from the Leaderboard Database class and sorts them. After the sorting operation, it will create a panel which includes sorting nicknames with respect to finishing time, if player selects the leaderboard option in the main menu.

LeaderboardDatabase

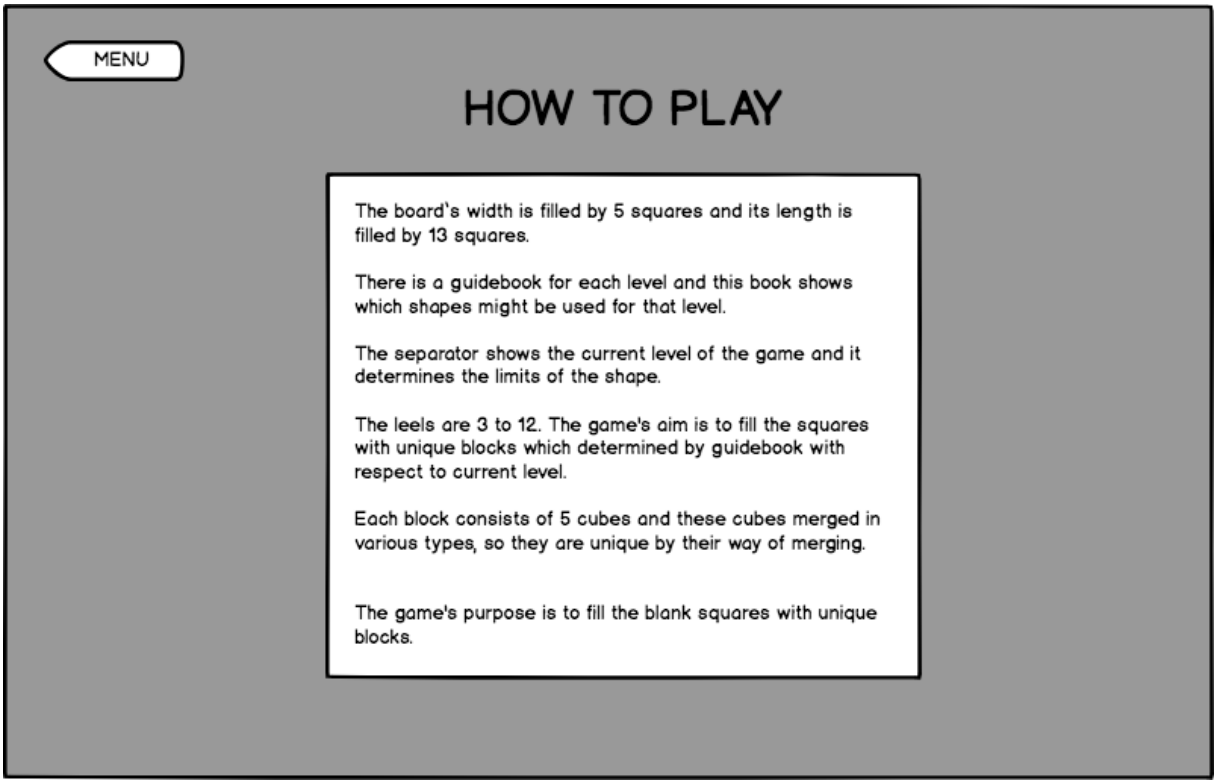
Leaderboard database will be a class which deals with users' nickname and time and their relations with database. Firstly, it will create a connection between the nickname database and IDE. Later, it will take the user's nickname and time from GameAdmin class at the end of the game and save it to the database. After that, it will send all information in the database to Leaderboard class to be sorted and showed.

5.4 User Interface

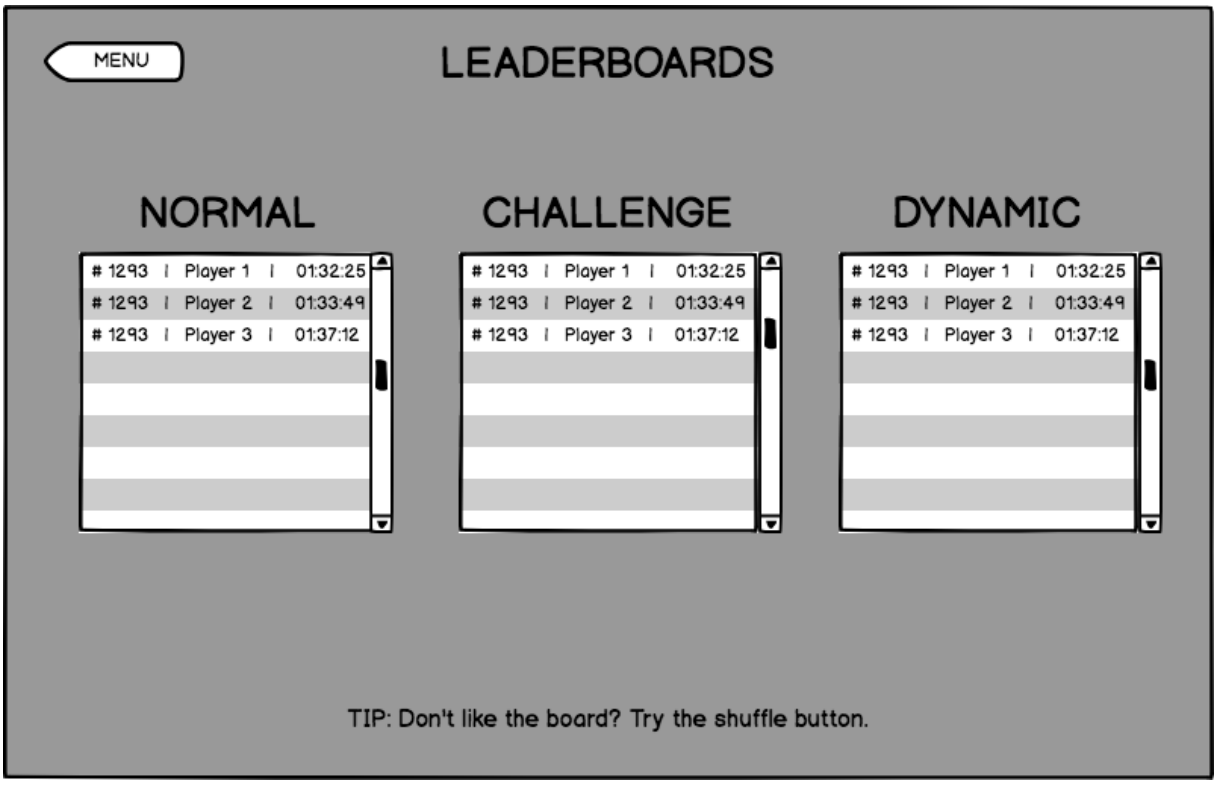
5.4.1 Main Menu



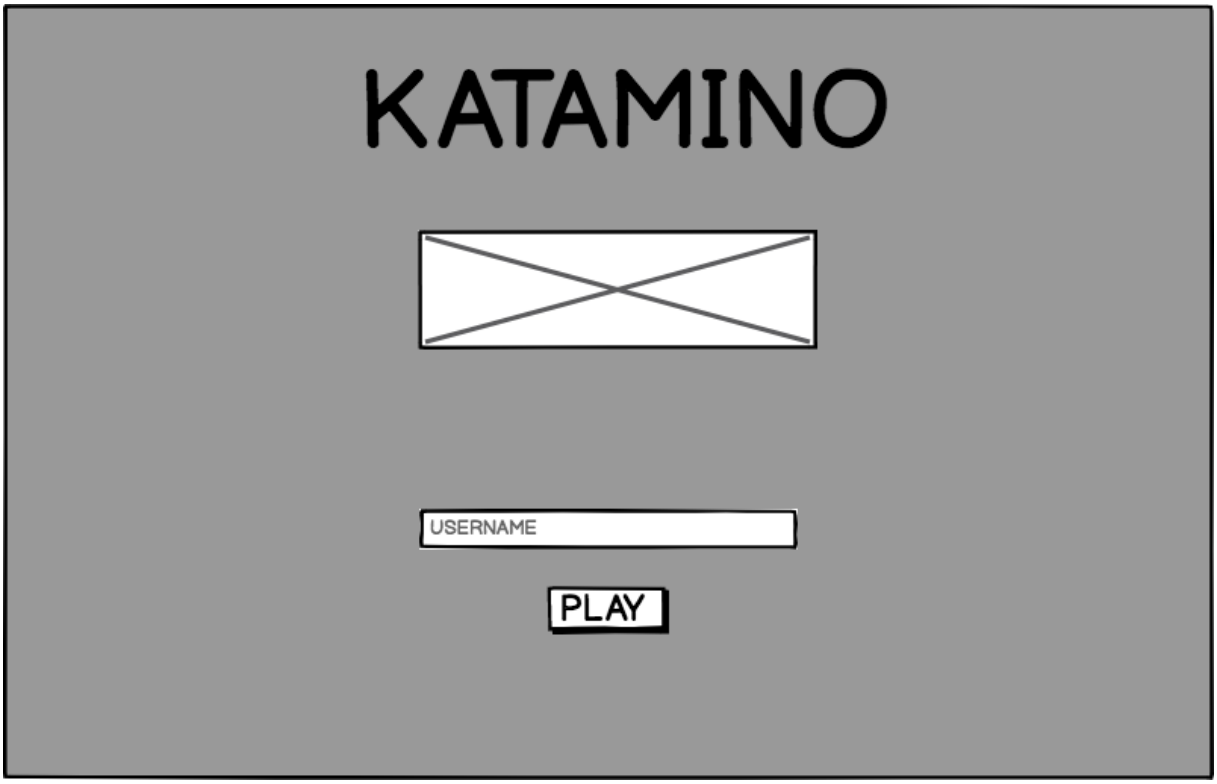
5.4.2 How to Play



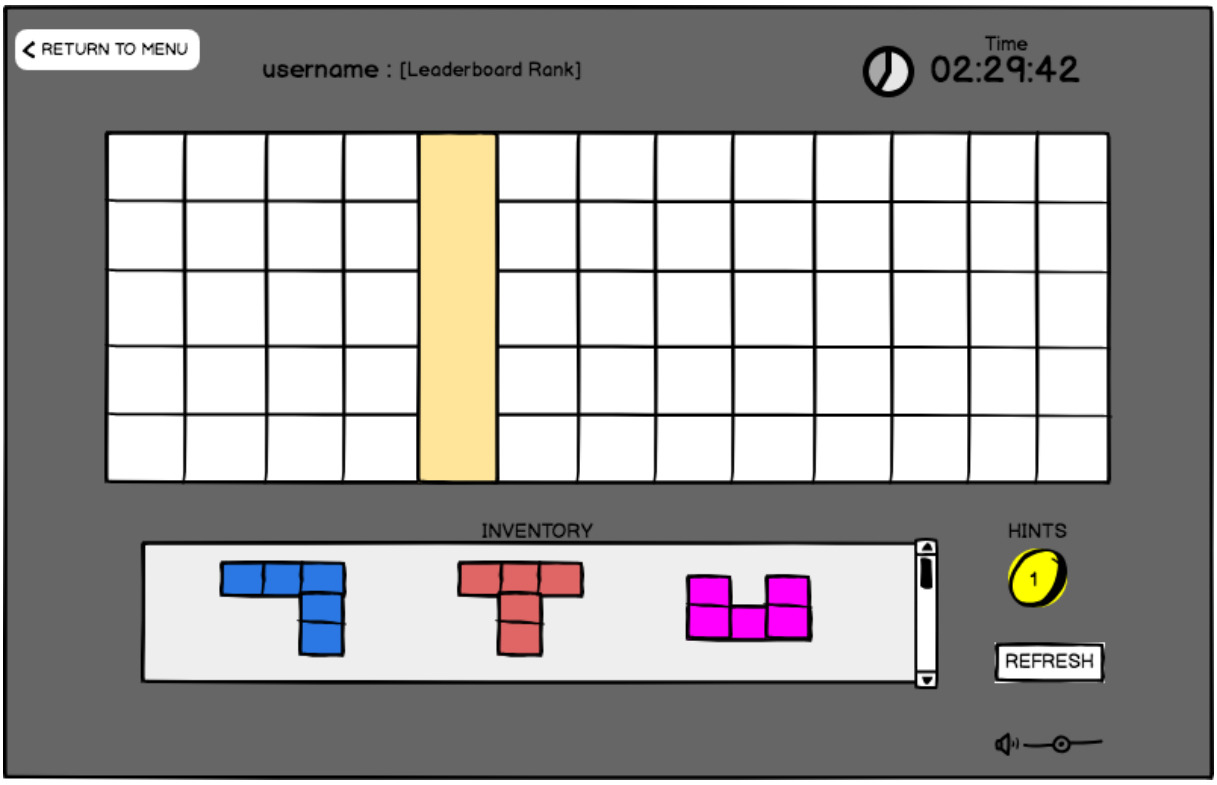
5.4.3 Leaderboards



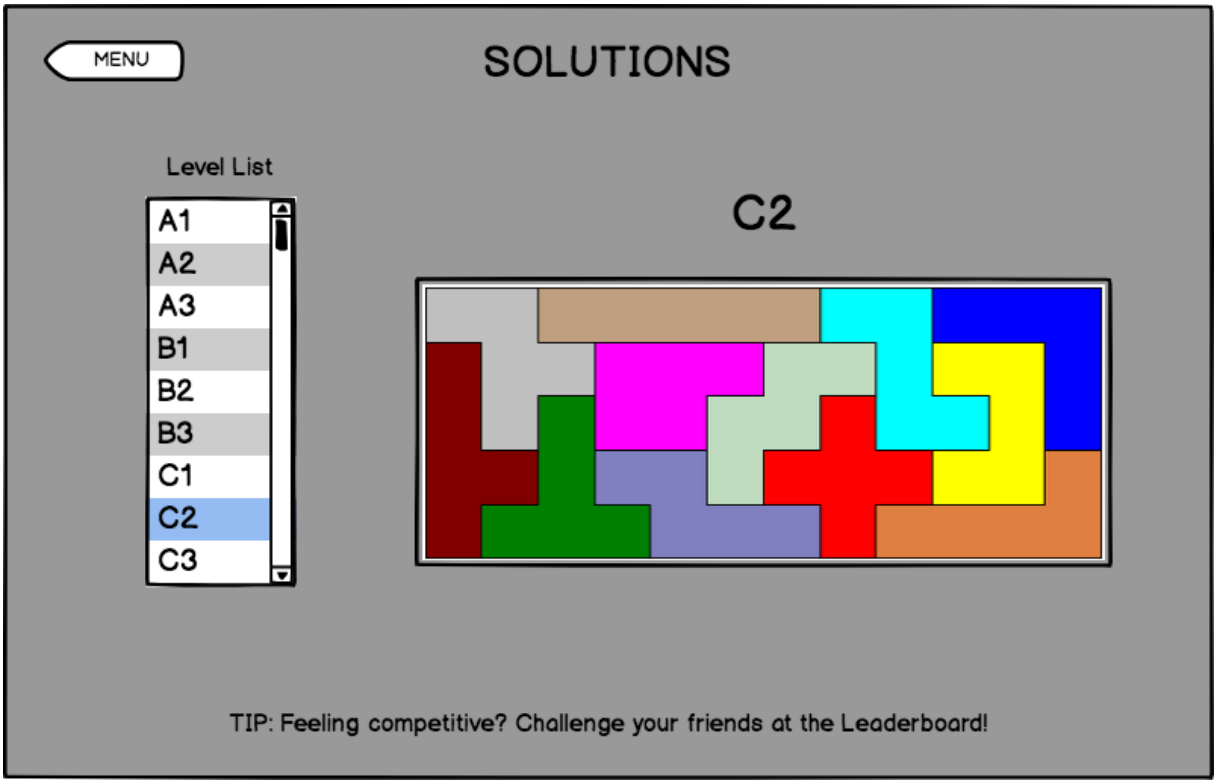
5.4.4 Welcome Screen



5.4.5 Game Play



5.4.6 Solutions



6.0 Conclusion

The analysis report which we prepared is high sketch in the game Katamino, and all the improvements that being thought to make the game more appealing and challenging. Basically, a general looking of the analysis of the system that we developed.

In “Overview”, gameplay of the Katamino that we desired are explained in general terms. The gameplay, modes that we added in game, distinctions, and the general overview of the other features are highly examined in that section.

We tried to explain the essential functionalities of the game, and the features that we are thinking to add in game are in “Functional Requirements” part. We want to make a game that people wishing to play our game every second, so we tried to make our game even more combative, and attractive. “Non-Functional Requirements” of the game highlights the additions that we desired in our game. The offers that we made to make our game more desirable are expounded in that section.

Object and classes clarifies the structure that we formed for our game, and classes. The system models “Use Case Model”, “Dynamic Models”, “Object and Class Model”, and “User Interface - navigational paths and screen mock-ups” clarified down to the last detail. The scenarios, and flow of events describes the structural pathways of our game. Explanation of the classes and object express the relations, process, and hierarchy at the backstage. We spent considerable amount of time to make the best and efficient design can be possible. The design that we explained in the analysis report is composition of all the brainstorms, and roughcasts that we made.

To sum up, the improvements that we developed to make our game more challenging, interesting, and unique are mentioned highly detailed to highlight our difference than the original Katamino game. Object and Classes, Flow of events, and System model are well defined to express the backstage of the game. Hierarchy that we will develop for the classes and objects are designed work productive together. Herewith the structural background of that we will developed for the game has the optimum efficiency possible.

7.0 Glossary & References

- "Description of Katamino". http://en.gigamic.com/game/katamino
- "Gigamic Katamino Classic Puzzle and Game". https://www.tarquingroup.com/gifts/gigamic-katamino-classic-puzzle-and-game.html
- "Visual-spatial Games for Active Brains". https://www.imacs.org/blog/2011/07/visual-spatial-games/
- "The Theory of Tetris". http://liacs.leidenuniv.nl/~kosterswa/tetris/tot.pdf
- "A CLOSER LOOK AT TETRIS: ANALYSIS OF A VARIANT GAME". http://euclid.trentu.ca/aejm/V4N1/Tsuruda.V4N1.pdf
- “PENTOMINOS”. https://www.andrews.edu/~calkins/math/pentos.htm
- “UML Sequence Diagram”. https://www.lucidchart.com/pages/uml-sequence-diagram
- “The UML Basics”. https://www.ibm.com/developerworks/rational/library/3101.html
- “Reading The Sequence Diagram”. http://download.oracle.com/otn_hosted_doc/jdeveloper/j2ee101302/umlmodeling/umlmodeling/mod_notationumlsequencediagram.html
- “CASE EXAMPLES”. https://www.researchgate.net/figure/UML-Use-Case-Diagram-and-Activity-Diagram_fig2_263155723
- “Activity Diagram”. https://www.smartdraw.com/activity-diagram/
- “How to Generate Activity Diagram from User Story?". https://www.visual-paradigm.com/tutorials/user-story-to-activity-diagram.jsp
- “The Class Diagram”. https://www.tutorialspoint.com/uml/uml_class_diagram.htm
- “Class and Object Diagrams”. https://people.cs.pitt.edu/~chang/153/UML/class.html