CS 319 TERM PROJECT

Analysis Report

Section 3
Group 3B
Project Name: RISK 101

Group Members

Ramazan Melih DİKSU - 21802361
Aleyna SÜTBAŞ - 21803174
Yiğit ERKAL - 21601521
Baykam SAY - 21802030
Berk TAKIT - 21803147

Contents

1. Introduction

Risk is one of the most famous strategy conquest games. This game has been a favorite game for the strategy players since 1957. Various versions of this game have been made and played by players. Risk requires a good strategy, diplomacy, and luck to capture the world. Luck is crucially important in this game due to dice. Besides a good strategy and diplomacy cannot win the game solely if dice are not on your side. Furthermore, Risk has a turn-based game structure. Therefore, thinking and strategy development is the most important part of the game. Every turn players get the new soldiers according to his or her territories and they try to complete the mission and also they try to take over the world.

In Risk101, we decided to create the modification of the Risk game. Our game will be Bilkent University themed. Players will feel this theme in the map, in the character selection part, and also in the missions. Risk101 will be an offline desktop game that can be played multiplayer on the same computer. As a group 3B, we chose to use Java to implement our adapted Risk game. While implementing the game, we will aim to create a themed Risk game that is more fun for Bilkent students to play, and besides, we will make the Risk101 following the object-oriented programming principles.

2. Overview

2.1. Gameplay

In RISK the objective of the game is to achieve complete domination of the game map by eliminating all other players. At the start of the game the players will choose the faculty they will play as and will be given a number of troops according to the player count, and then will take turns claiming territories until all of them are occupied. All players will be given a bonus objective card, which will be kept secret from the other players. After the initial deployment phase, the game will proceed as normal. A turn consists of acquiring and placing new armies, attacking if the player chooses to do so, and finally moving armies around the game board. The amount of armies a player acquires at the start of the round is determined by the value of the territories they currently control, the value of the continents they currently control, and any objective cards they have accomplished. After deploying these armies, the player can choose to attack any enemy territory that neighbors one of their own. This action can be repeated as many times as the attacker chooses, as long as there is more than one troop in the attacking territory. The attacking and defending is done via dice rolls, with both the attacker and the defender rolling dice equal to their number of troops in either the attacking or defending territory, respectively. The attacker can roll a maximum of three dice, while the defender can only roll two. The dice rolls are then compared die by die, with the highest rolls of both sides going against each other, and the outcome of each of the rolls are determined separately. In the event of a tie, the defender wins. The attacker captures the enemy territory by eliminating all defender armies. In this case, the attacker has to move a number of troops that is at minimum equal to the number of dice they rolled. A territory must never be empty in the course of the game, so the attacker must leave at least one soldier behind. The defender wins when the attacker refuses to attack again, or if the attacker goes down to a single army in the attacking territory. After the attack phase, the player can choose to move their troops between any friendly territory that is connected. After this last phase, the turn passes onto the next player. The winner of the game is the player who manages to capture all the territories and eliminate all players.

2.1.1. Objective Cards

The objective cards contain a secret objective that when completed, rewards the player with the number of troops written on it. The objective card may ask the player to capture a certain continent, hold a territory for a number of turns, eliminate a player, capture a certain amount of territories in a single turn or control a specific group of territories. When an objective is completed, the objective card is revealed to the other players and is placed next to the player who has completed it. In the player's next turn, in the deployment phase, the card is discarded and the number of troops shown on it is added to the player's deployment pool. The player then draws another objective card.

2.1.2. Faculties

In this version of RISK, there will be a pool of faculties that the players choose from at the beginning of each game. These faculties will act as their armies for the remainder of the game. Each faculty represents a faculty from Bilkent University, with a unique ability that can be used once a turn or once a game, depending on the ability's specifications. The tentative list of faculties and their respective abilities is as follows:

1) Faculty of Art, Design, and Architecture: Everything by Design
    a) Once a turn, the player may choose to swap the result of one of their die rolls with the opponent.
2) Faculty of Applied Sciences: Eastern Affinity
    a) When attacking a territory in the Eastern Campus region, the player wins in the case of a tie.
3) Faculty of Economics, Administration, and Social Sciences: The Moderate Depression
    a) Once a turn, the player may choose to reduce 1 from the dice rolls of the enemy.
4) Faculty of Education: Reeducate
    a) When the player draws a new objective card, they may choose to discard it and draw a new one instead. This can be done only once per draw.
5) Faculty of Science: Knowledge Above All
    a) Once a game, the player may choose to reveal all of the other players' objective cards.
6) Faculty of Humanities and Letters: The Pen is Mightier

    a) The player gains an extra die if the territory they are being attacked from has more armies than the territory they are defending.

7) Faculty of Law: Lawyered
    a) Once a turn, the player may choose to redo any dice roll. Both the attacker and the defender must roll their dice again.

8) Faculty of Business Administration: Hostile Takeover
    a) Once a game, in the deployment phase of their turn, the player may choose to take over an enemy occupied territory along with the enemy troops. This action also ends the deployment phase for the player.

9) Faculty of Engineering: Over-engineered
    a) The player can't roll a 1.

10) Faculty of Music and Performing Arts: Smoke and Mirrors
    a) Once a game, the player may choose to play an extra turn right after their normal turn.

2.2. Map

Our game map will be based on the map of Bilkent University. There will be three main areas: Main Campus, East Campus, and Bilkent Island. Cities in the original Risk map will be replaced with some of the main buildings from the campus of Bilkent University and the surrounding area. The detailed list of the main areas and the buildings they will include is given below.

1. **Upper Main Campus Area**
    - 1.1. Faculty of Economics, Administrative, and Social Sciences
    - 1.2. Faculty of Law
    - 1.3. Faculty of Engineering
    - 1.4. Mithat Çoruh Auditorium and Classrooms
    - 1.5. Electrical-Electronic Engineering
    - 1.6. Faculty of Art, Design, and Architecture
    - 1.7. Faculty of Education
    - 1.8. Faculty of Humanities and Letters
    - 1.9. Faculty Housing
    - 1.10. Main Campus Library
    - 1.11. School of Applied Languages
    - 1.12. Faculty of Business Administration
    - 1.13. Faculty of Music and Performing Arts
    - 1.14. Bilkent Concert Hall
    - 1.15. Faculty of Science
    - 1.16. Nanotechnology research center
2. **Lower Main Campus Area**
    - 2.1. AB-MikroNano Technologies Industry and Trade Inc.
    - 2.2. Lecture Halls (Building V)
    - 2.3. Main Campus Student Dormitories
    - 2.4. Pharmacy
    - 2.5. Main Campus Health Center
    - 2.6. ODEON
    - 2.7. Main Campus Entrance
    - 2.8. Cafe In
    - 2.9. Main Campus Cafeteria
    - 2.10. Main Campus Sports Center
    - 2.11. Starbucks

2.12. Cyberpark
2.13. Main Campus Mozart Cafe
2.14. Main Campus Coffee Break
2.15. Sözeri Kebab
2.16. Mescit
2.17. Meteksan Bookstore
2.18. Mayfest

3. East Campus Area

3.1. East Campus Entrance
3.2. East Campus Cafeteria
3.3. East Campus Mozart Cafe
3.4. East Campus Coffee Break
3.5. East Campus Sports Center
3.6. East Campus Student Dormitories
3.7. School of Applied Technology and Management
3.8. East Campus Library
3.9. East Campus Health Center
3.10. English Language Preparatory Program Buildings

4. Bilkent Island Area

4.1. Bilkent Center
4.2. Ankuva Shopping Center
4.3. Bilkent Hotel
4.4. Sports International
4.5. Bilkent 1-2 Apartments

2.3. Rules

- Every player rolls a die, the one with the highest number starts by placing an infantry to a territory and claiming it. Whoever placed the first army opens the game.
- At the beginning of each turn, the number of new armies that a player can add is calculated based on how much control they have on the map and the value of their cards.
- A player can only attack a territory that is adjacent to one of their own territories which has at least two armies on it.
- The winner of a battle is decided by die rolls. If the attacker's die is higher, the defender loses an army. If the defender's die is higher, the attacker loses an army. In the case of a tie, the defender wins the battle.
- If a player completes an objective card, they will discard that objective and gain the amount of armies it specifies on the following turn. Then the player will draw a new objective card.
- If a player defeats an enemy, they get the defeated enemy's unfinished objective card.
- To win a game, a player must occupy every territory in the map.

  A more detailed rulebook can be accessed here:
  https://www.hasbro.com/common/instruct/risk.pdf

3. Functional requirements

1. **New Game:** The player has the ability to create a new game. They can then select the name of the save file they want for this game, and select the number of players. Each player selects a character to represent themselves before starting the game. When the game starts, the player sequence is determined and players place their armies on the board one by one by clicking on an empty territory on the map.
2. **Load Game:** The player can choose to continue a game from the save file instead. Upon clicking on this option the player can choose which save they want to load from. Then, the game is loaded and players can continue from where they left off.
3. **Settings:** Clicking on the settings button directs to a settings page. There are two essential settings on this page:
    a. **Sound:** The volume of the game sounds can be set here. There are two different options for the volume of the music and the sound effects. There is also a mute option.
    b. **Display:** The player has the option to play the game on a bordered window or at fullscreen.
4. **Help:** The help menu first explains the basics of Risk. It also explains the specific features of RISK 101. There is also a hyperlink to the official rules for the game Risk.
5. **Credits:** This option shows the creators of the game and has a link for the GitHub page for the game.
6. **Gameplay Requirements:** Each player takes turns playing the game. A turn is composed of a series of stages:
    a. **Army Placement:** The player gains several troops at the start of their turn according to the number of areas they own. If they have accomplished their objective the previous turn, they gain additional troops according to their objective card and draw a new card. They send these troops to the owned areas they want by clicking on the area. When the troops are finished the game automatically proceeds to the next stage.
    b. **Attacking:** The player can wage war on another area as many times as they want at this stage. To wage war, they first need to click on an area they own to determine the attackers. This area needs to have at least 2 troops in it. Then they click on an adjacent area to determine where to attack. A fight menu appears where the attacker selects the number of troops (between 1 - 3) they want to in the attack. The player has an option to cancel the attack before selecting the number of troops from this menu. Then, the war

sequence happens. After the war, the player can select to proceed to the next stage or continue fighting.

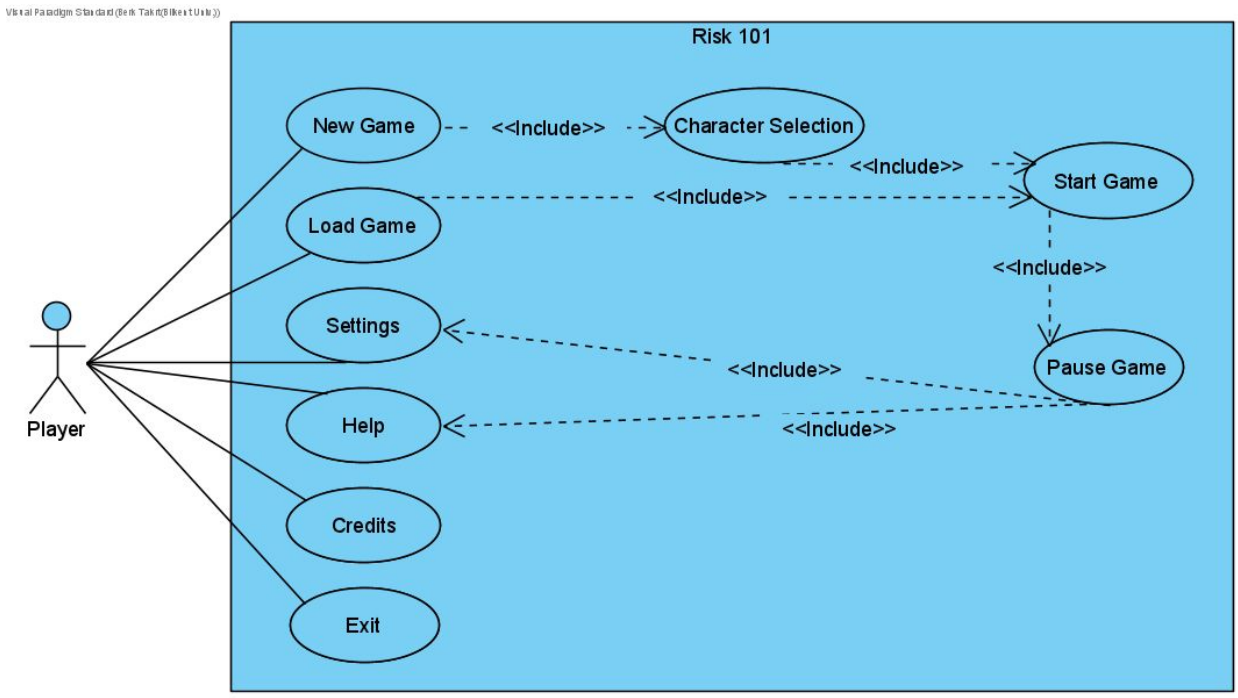
c. **Fortifying:** The player can move their troops from one owned area to another as long as these areas are connected (there is a path consisting of adjacent owned areas connecting the areas). To move the troops, the player first selects the area they want to move the troops from, then the number of troops (they have to leave one troop behind to protect the area), then the destination area. If the destination area is not accessible, the game displays an error message and does not move the troops, if not, the troops are moved to the destination area. The player can choose to end their turn at any time when they are satisfied with their troop formation.

4. Nonfunctional requirements

1. **Supportability:** Implementation of RISK101 will be handled in a particular manner. All contributors who are responsible for introducing new concepts to the game will be able to adapt to the previously conducted components and designs. To sustain RISK101's adaptability, the necessary information will be given in the shape of comments. Comments will be kept short and clear so that the participating programmer would not lose time or get more confused. This comment system will function as a map that will make enhancing certain parts, update implementations, or bug fixing considerably easier. Therefore, the maintainability and extendibility of RISK101 will be preserved with great importance.
2. **Usability:** The user interface of RISK101 will focus on understandability, brevity, and translucency rather than offering complex basics and exhausting lengthy instructions. Cartoonish images will be preferred. The color palette will consist of intriguing and alluring colors while considering the brightness and contrast issues regarding the users with discomforts related to eyes. Game audio will be arranged in terms of realism so that the user can have a realistic experience and sound intensity balance issues will be evaluated carefully. Elements such as icons, tags, and other game basics related components will be kept at a self-explanatory level.
3. **Reliability:** RISK101 enables users to save an unfinished game and load it at any time. No further data will be asked from users nor will be stored in a database. So, privacy concerns are limited to a minimum.
4. **Performance:** Response time will be kept at under 0.5s so that the user experience will progress smoothly. RISK101 will run at the same speed without taking the qualities of the operating machine into consideration. All the game-related operations presented will be available to the user during the gameplay. Any conflicts that can result in these operations to not function will be evaluated and solved. A RISK101 player will be able to accurately use all the preferable tools at any time with the preferable timestamp efficiency.

5. System models

5.1. Use case model



Use Case #1

Use Case: New Game

Primary Actor: Players (2 to 6)

Stakeholders and Interests: Players want to start a new game.

Entry Condition: Player chooses the New Game option from the main menu.

Exit Condition:

- Players exit the game from the pause menu, OR
- The game ends.

Main Flow of Events:

1) Players select New Game from the main menu.
2) System prompts the players to select a save slot.
3) The players select a save slot to override.
4) The system prompts the players to select characters.
5) The players select between 2-6 characters.
6) The players select the Start Game option.
7) System initializes the game with the selected characters.

8) The players take turns playing the game.
9) The game ends, the victory screen shows the player that won.

Alternative Flow of Events:

1. If players want to leave the game before it ends.
    a. Players pause the game by pressing the Escape button.
    b. Players choose the exit game option.

Use Case #2

Use Case: Load Game
Primary Actor: Players (2 to 6)
Stakeholders and Interests: Players want to load an existing game.
Entry Condition: Player chooses the Load Game option from the main menu.
Exit Condition:

- Players exit the game from the pause menu, OR
- The game ends.

Main Flow of Events:

1. Players select Load Game from the main menu.
2. System prompts the players to select a save slot.
3. The players select a save slot to load from.
4. System loads the saved data and continues the game from that point.
5. The players take turns playing the game.
6. The game ends, the victory screen shows the player that won.

Alternative Flow of Events:

1. If players want to leave the game before it ends.
    a. Players pause the game by pressing the Escape button.
    b. Players choose the exit game option.

Use Case #3

Use Case: Settings
Primary Actor: Player
Stakeholders and Interests: Player wants to change the settings.
Entry Condition:

- Player chooses the Settings option from the main menu, OR
- Player chooses the Settings option from the pause menu.

Exit Condition:

- Player exits back to the main menu by choosing the back option, OR
- Player goes back to the pause menu by choosing the back option.

Main Flow of Events:

1. Player chooses the Settings option from the main menu or the pause menu .
2. System displays settings.
3. Player makes changes to the settings.
4. Player chooses the back button.
5. System applies setting changes to the game.

Use Case #4

Use Case: Credits
Primary Actor: Player
Stakeholders and Interests: Player wants to view the credits.
Entry Condition: Player chooses the Credits option from the main menu.
Exit Condition: Player exits back to the main menu by choosing the back option.
Main Flow of Events:

1. Player chooses the Credits option from the main menu.
2. System displays credits.
3. Player chooses the back button.

Use Case #5

Use Case: Help
Primary Actor: Player
Stakeholders and Interests: Player wants to view the help page.
Entry Condition:

- Player chooses the Help option from the main menu, OR
- Player chooses the Help option from the pause menu.

Exit Condition:

- Player exits back to the main menu by choosing the back option, OR
- Player goes back to the pause menu by choosing the back option.

Main Flow of Events:

1. Player chooses the Help option from the main menu or the pause menu.
2. System displays the help page.
3. Player chooses the back button.

Use Case #6

Use Case: Exit
Primary Actor: Player
Stakeholders and Interests: Player wants to exit the game.

Entry Condition: Player chooses the exit option from the main menu.
Exit Condition: None.
Main Flow of Events:

1. Player chooses the Exit option from the main menu.
2. System prompts the user with an “Are you sure?” prompt.
3. Player chooses Yes.
4. System shuts down the game.

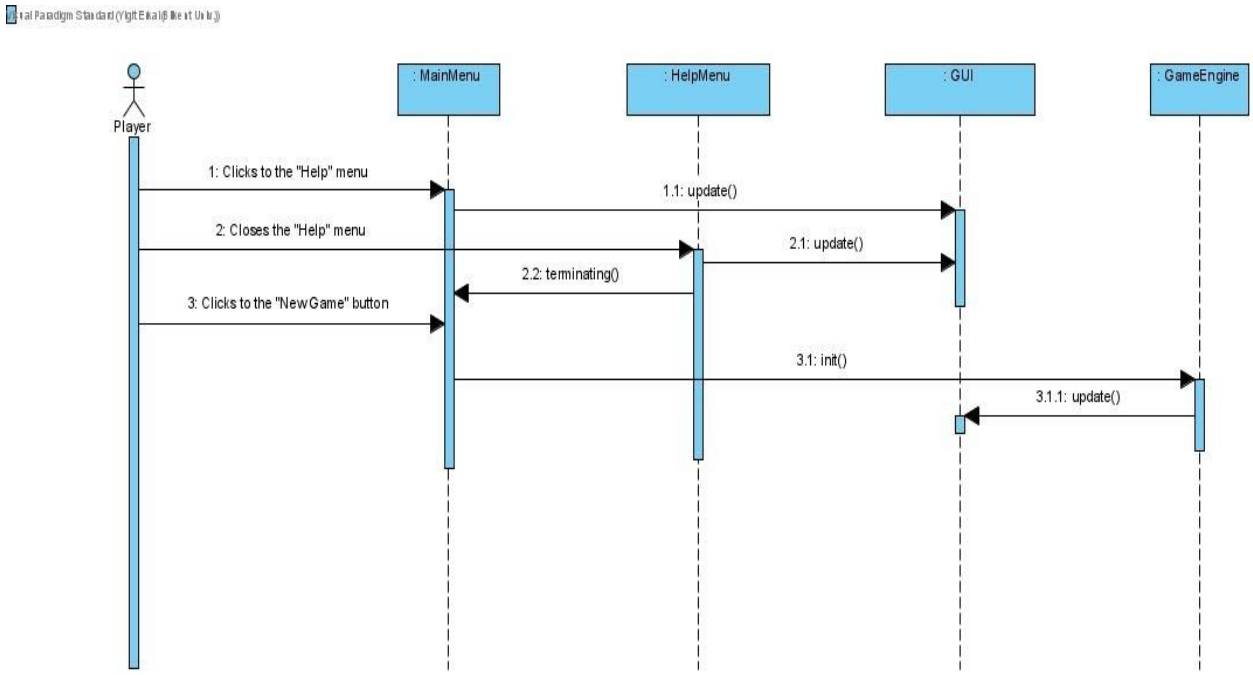
Alternate flow of events:

1. Player chooses the No option on the prompt.
2. Player goes back to the main menu.

5.2. Dynamic models

Scenario name: Execute New Game

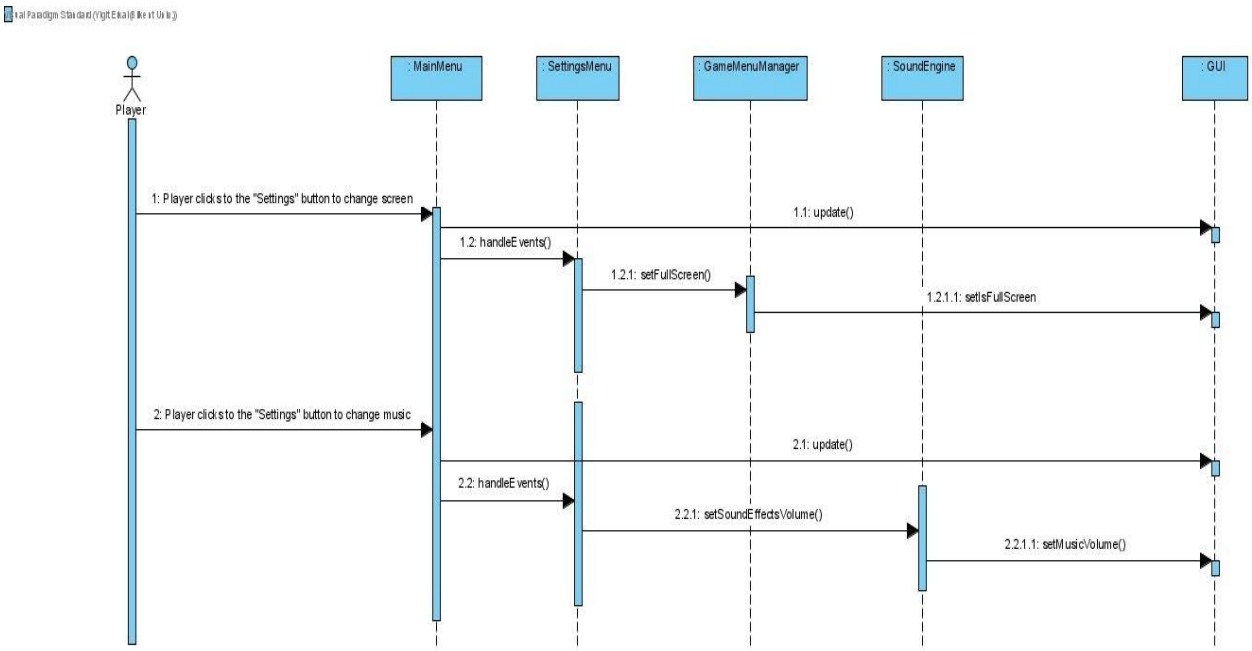
Scenario: Player clicks the game icon to initialize the game. Then, the player clicks the “Help” button to get information about the game. When the player learns how to play Risk101, he returns to the main menu. Lastly, the player clicks the “New Game” button to start playing a new game.



Description: When the player clicks the game icon, the game panel opens and it shows directly the main menu. After that game waits until the next instruction received from the player. When the player clicks the “Help” button, the game opens the new small frame that explains everything about the game. After the player closes this frame, he or she clicks the “New Game” button to start the game. This process continues until the player clicks the “Exit Game” button. After this process, the game program stops and closes the window.

Scenario name: Change Settings

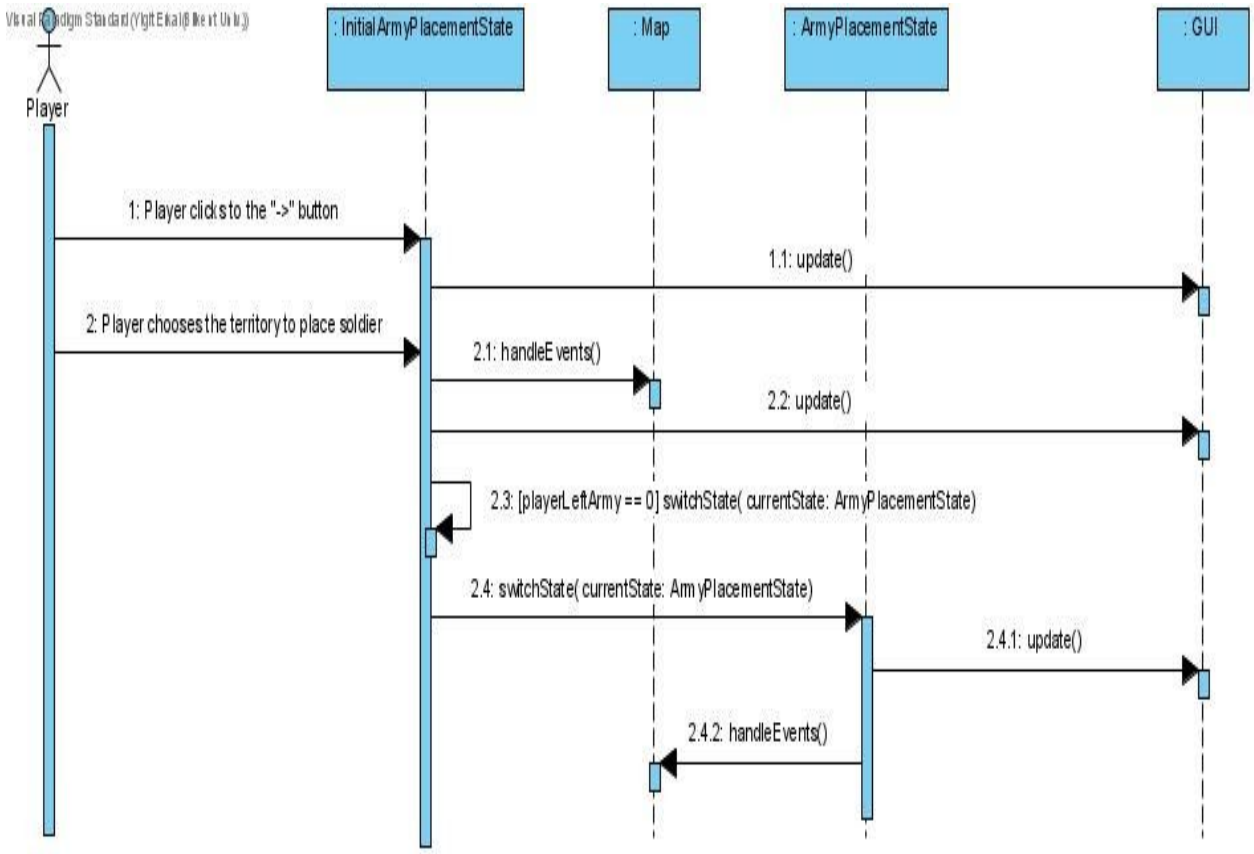
Scenario: Player has already opened the game and learnt every information that he needed to know from the “Help” section. However, this player does not want to play the game with bordered screen and game music. Then, this player clicks the “Settings” button to change it. In the settings part, the player chooses the full screen option rather than the bordered screen option and closes the game music.



Description: When the player clicks the “Settings” button, the settings panel opens. In this section, the player sees the two options section. One of them is to change the screen size and the other one is to set the action sound and game music. In this scenario, the player sets and clicks the full screen and closes the game music. When the player clicks this button, this information is kept in one place. Then, the screen becomes a full screen and game music stops until the player wants to change it again.

Scenario name: Deployment

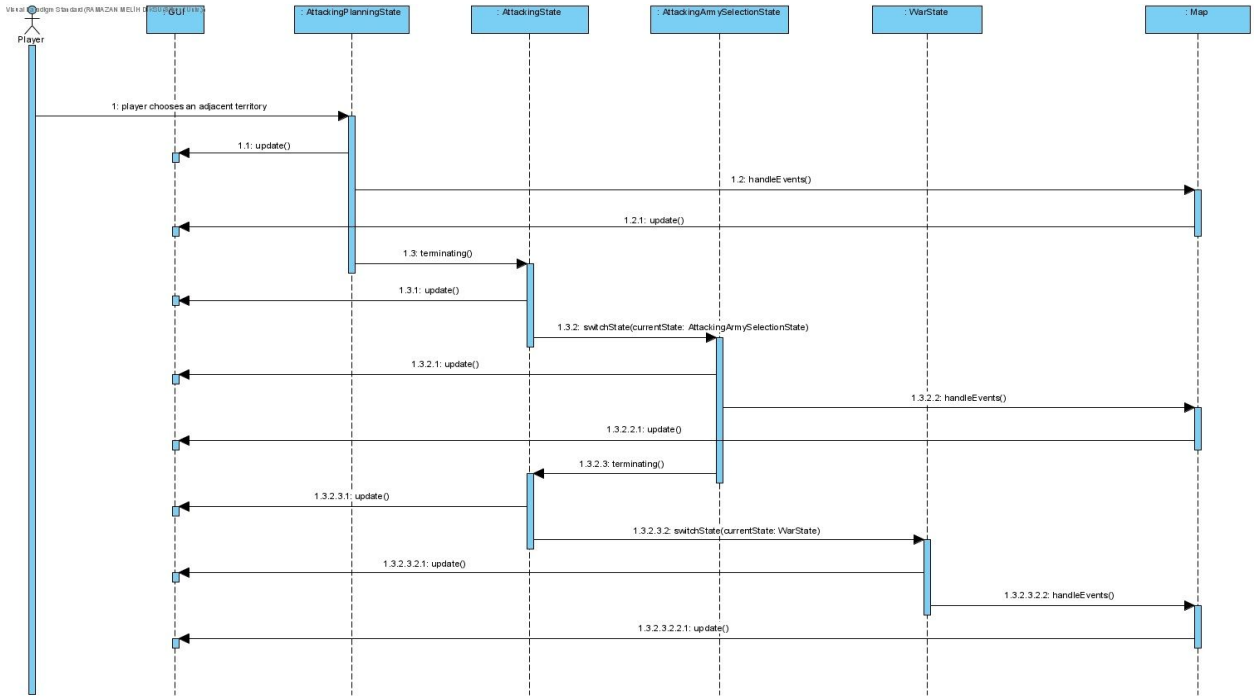
Scenario: Player has already started the new game. Initially, the player picks the territories according to his or her tactics. Then, the player passes to deployment phase. In this phase, the player sees a screen that shows his or her territories brighter and the player starts to place the soldiers one by one.



Description: When the player clicks the "->" button to pass the territory selection phase, the game advances to the next phase which is the deployment part. In the deployment part, the player only chooses his or her territories and when the player clicks that icon, the game engine increases the count of soldiers in that territory. Player repeats the same action until the player's soldiers are finished.

Scenario Name: Attack

Scenario: After the deployment stage, players will have a chance to attack a territory. An adjacent territory will be chosen. Players will choose the amount of dice they want to roll. If the attack is successful, the territory will be captured. Color of the territory will become the color that represents the player.



Description: After the player chooses a territory to attack, the input is received by the AttackPlanningState. Then AttackingState will enable moving through the other attack related states. AttackArmySelectionState will manage the amount of troops used to attack. WarState will function as an animation stage that will show the attacking process. During these processes, the map will be updated continuously. These updates mainly will be:

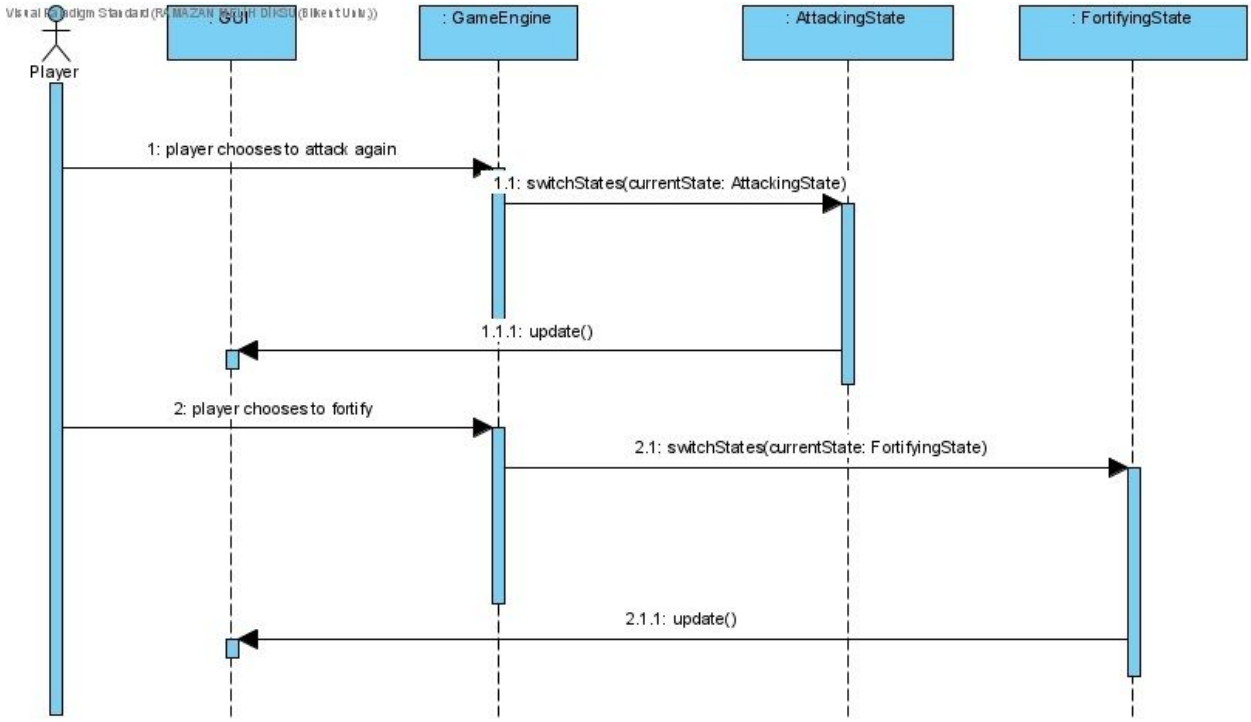
-Chosen territory to attack will be highlighted during the AttackPlanningState.

-Number of troops and dice will be shown during the AttackArmySelectionState.

-Color of the territory will change depending on the result of the WarState.

Scenario Name: Attack or Fortify Choice

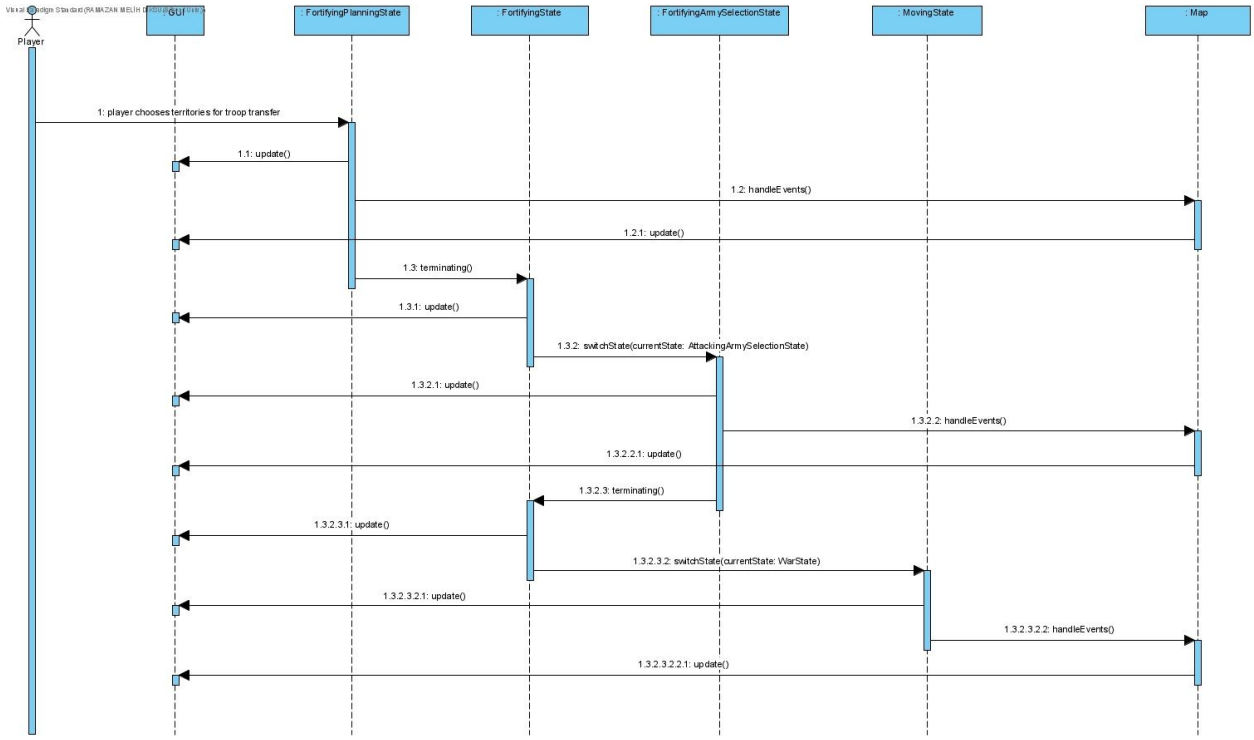
Scenario: Player will choose either to attack further territories or move to the fortifying state.



Description: After a single attack sequence, players can choose to move to the fortify state by pressing "->" button. If a different territory is chosen, the attack sequence is repeated. GameEngine will manage the states according to the received input.

Scenario Name: Fortifying

Scenario: After the attacking stage, if the player chooses to fortify, troops will be transported between the territories depending on the player's choice.



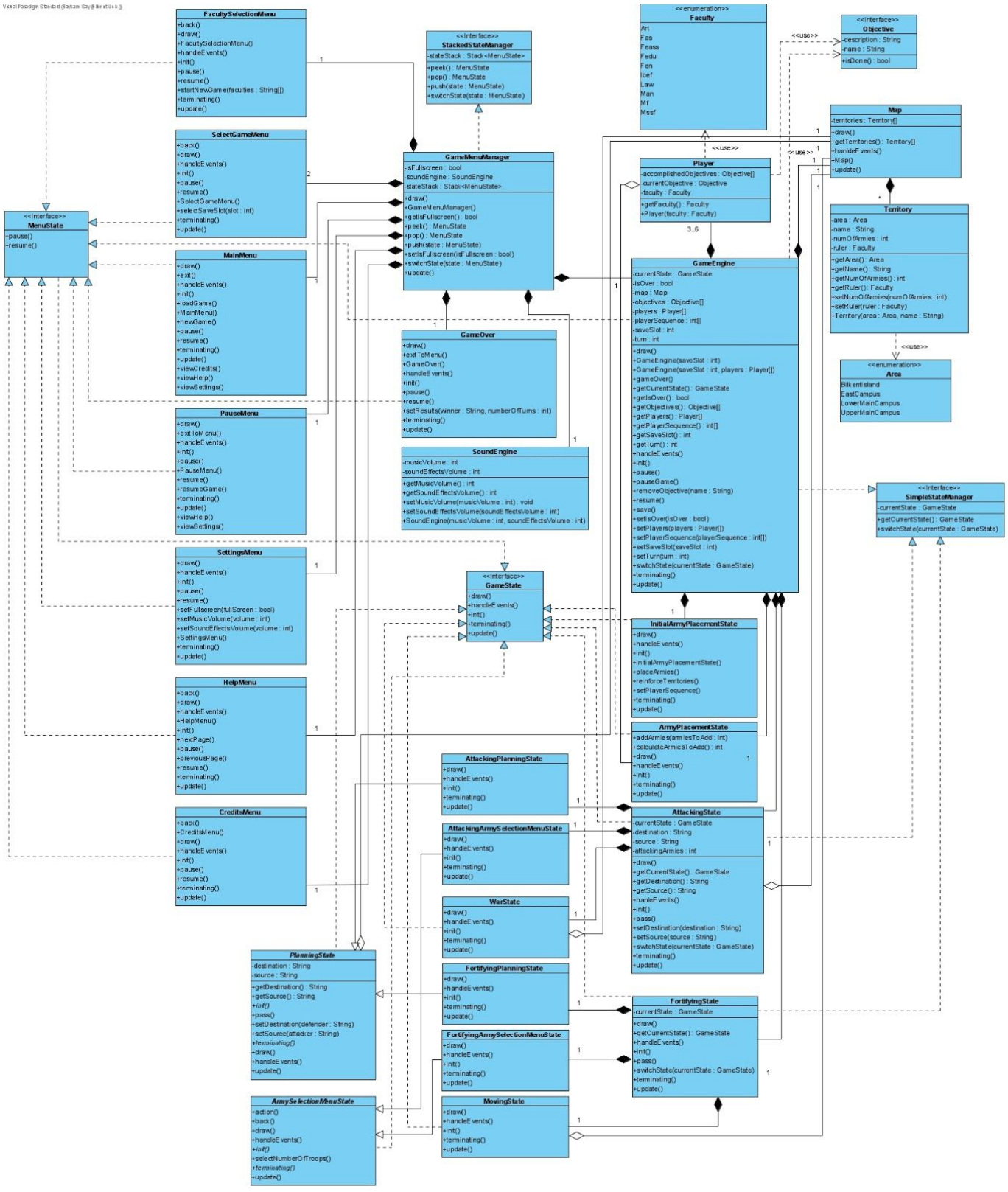
Description: After the player chooses to fortify, the input is received by the FortifyPlanningState where the input is the territory the troops will be transported from. Then FortifyingState will enable the movement between the other fortify related states. FortifyArmySelectionState will manage the amount of troops that will be transported and the choice of the targeted territory. MovingState will function as an animation stage that will show the transportation process. During these processes, the map will be updated continuously. These updates mainly will be:

-Chosen territory to attack will be highlighted during the FortifyAttackPlanningState.

-Number of troops will be shown and the targeted territory will be highlighted during the FortifyArmySelectionState.

-Number of the troops on the selected territories will change depending on the result of the MovingState.

5.3. Object and class model

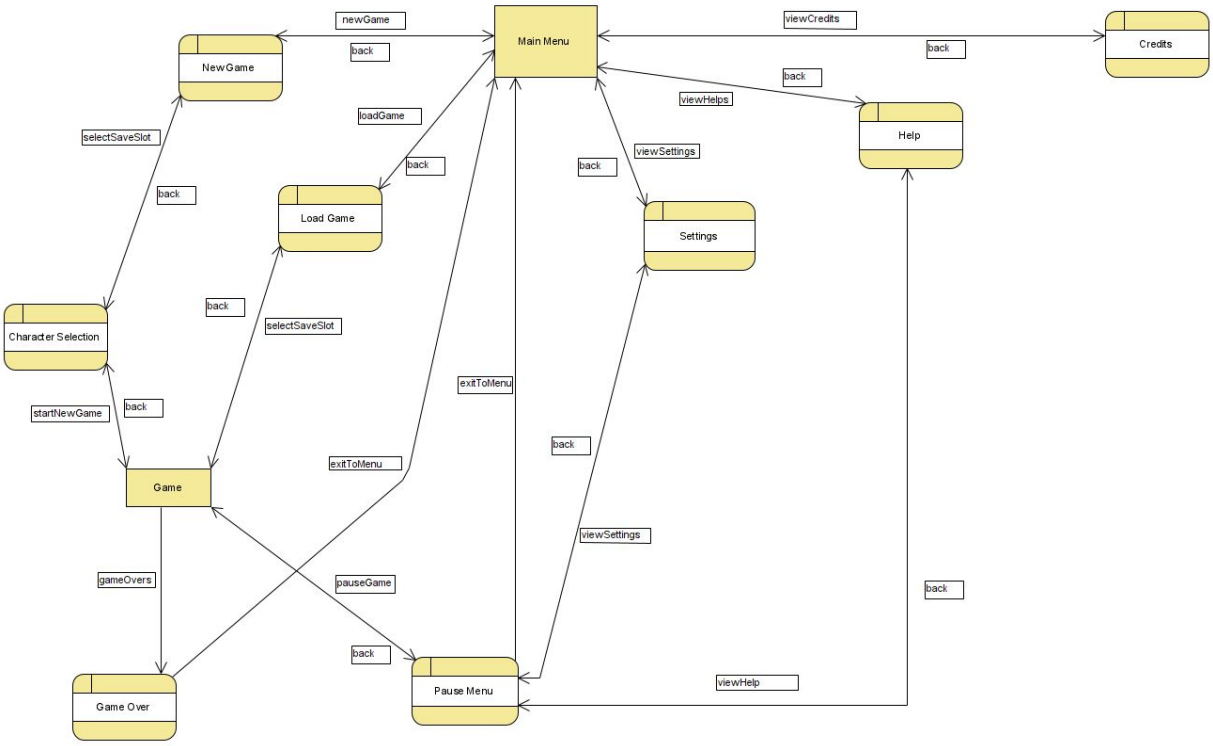


5.3.1. Explanation

The game implements the state pattern to manage the game menus and different game states [1]. **GameMenuManager** is the topmost state manager and manages all the different menu screens. It implements the **StackedStateManager** interface which keeps the active states in a stack. This allows going back between menus using the back button. It also allows pausing the game as the gameplay screen is also a menu state called **GameEngine**. Each of these states implements the **MenuState** which implements the **GameState** interface. **GameState** has 5 operations: init(), terminating(), draw(), handleEvents(), and update(). Init() is called when the **GameState** is first placed in the active states and terminating() is called when the **GameState** is leaving the active states. Draw() draws on the screen, handleEvents() handles the inputs and button clicks, and update() updates the game world. **MenuState** adds pause() and resume() functions to the states which are designed to work with **StackedStateManager**. Pause() is called in the topmost state when another state is pushed to the stack and resume() is called when that state becomes the topmost state again. This way, no screen continues to run in the background and draw to the display [2]. **GameMenuManager** also has the **SoundEngine** which manages the sound in the game. The gameplay screen **GameEngine** is also a state manager that manages different game states. These states are the game stages specified in the functional requirements section. The gameplay has linear state progression: there is only one possible next state and one possible previous state. Therefore, **GameEngine** implements the **SimpleStateManager** interface that holds only the current game state instead of a stack of active states. All states in the **GameEngine** implement the **GameState** interface. Some also have substates which also progress linearly, so they implement the **SimpleStateManager** interface too. **GameEngine** also keeps the **Player**s, the **Map**, and all the **Objective**s in the game.

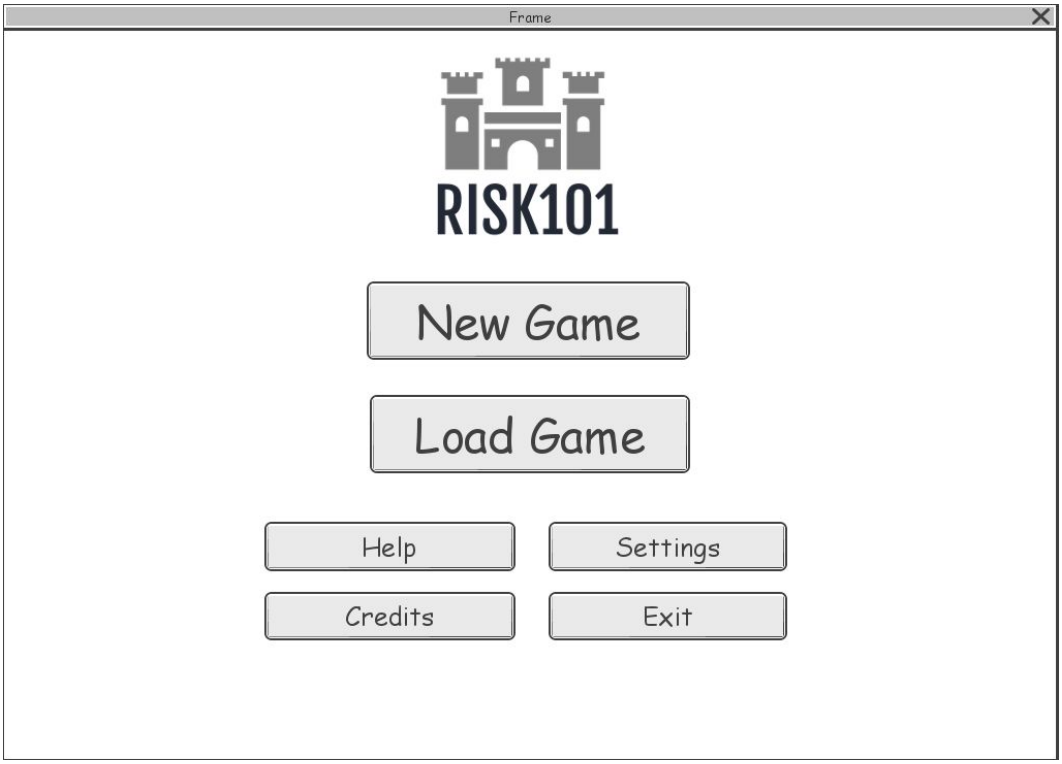
5.4. User interface

5.4.1. Navigational Path

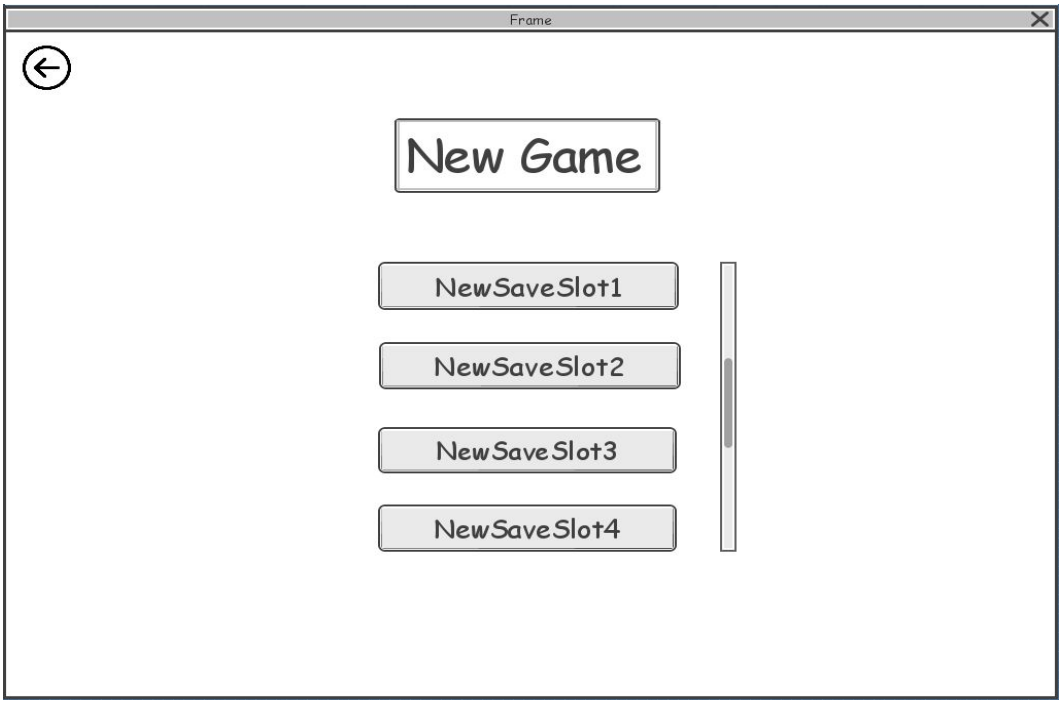


5.4.2. Screen Mock-ups

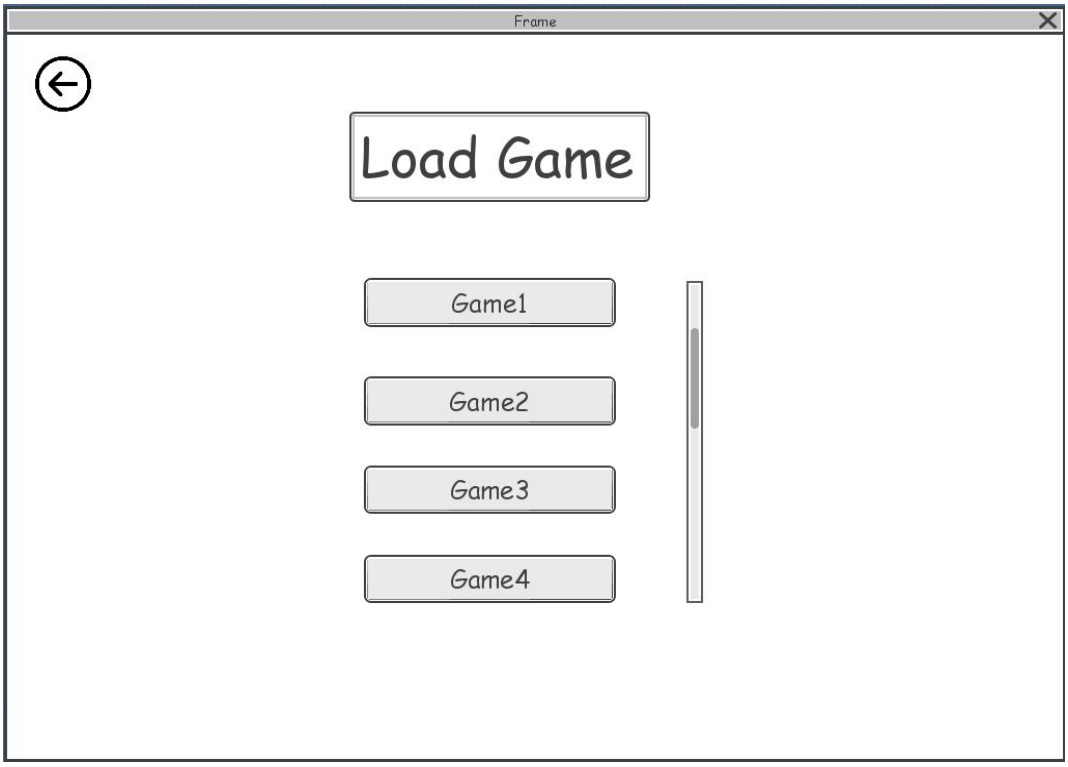
a.Main Menu Screen



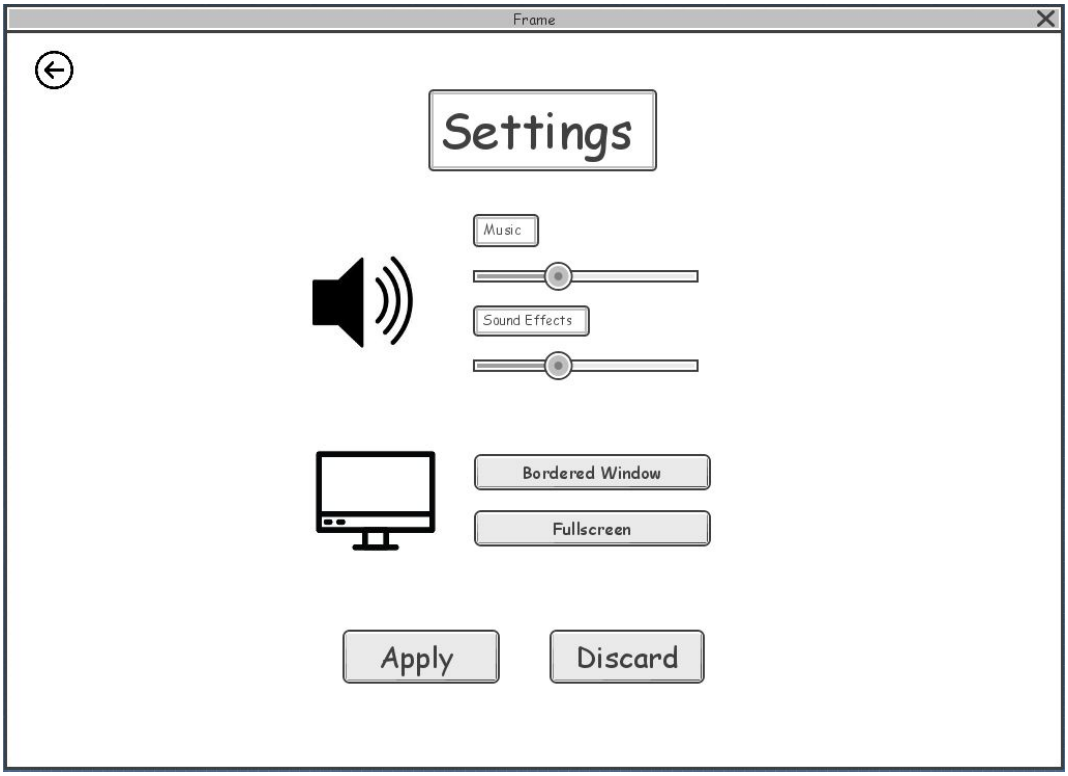
b.New Game Screen



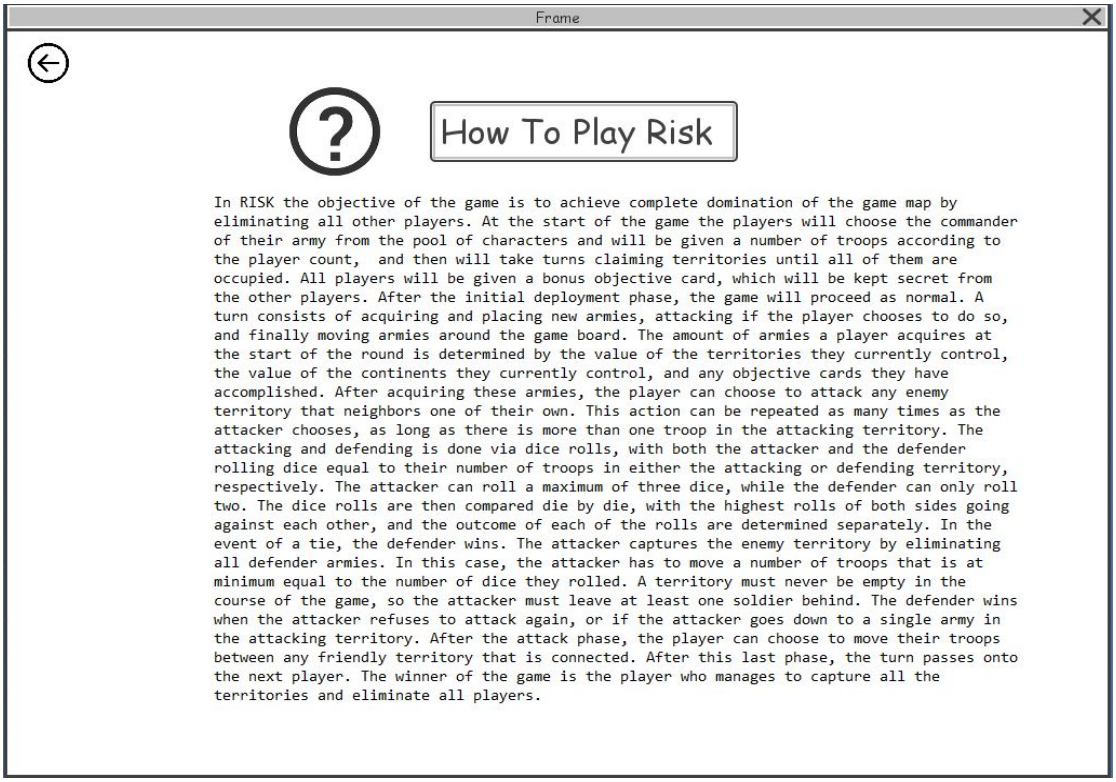
c.Load Game Screen



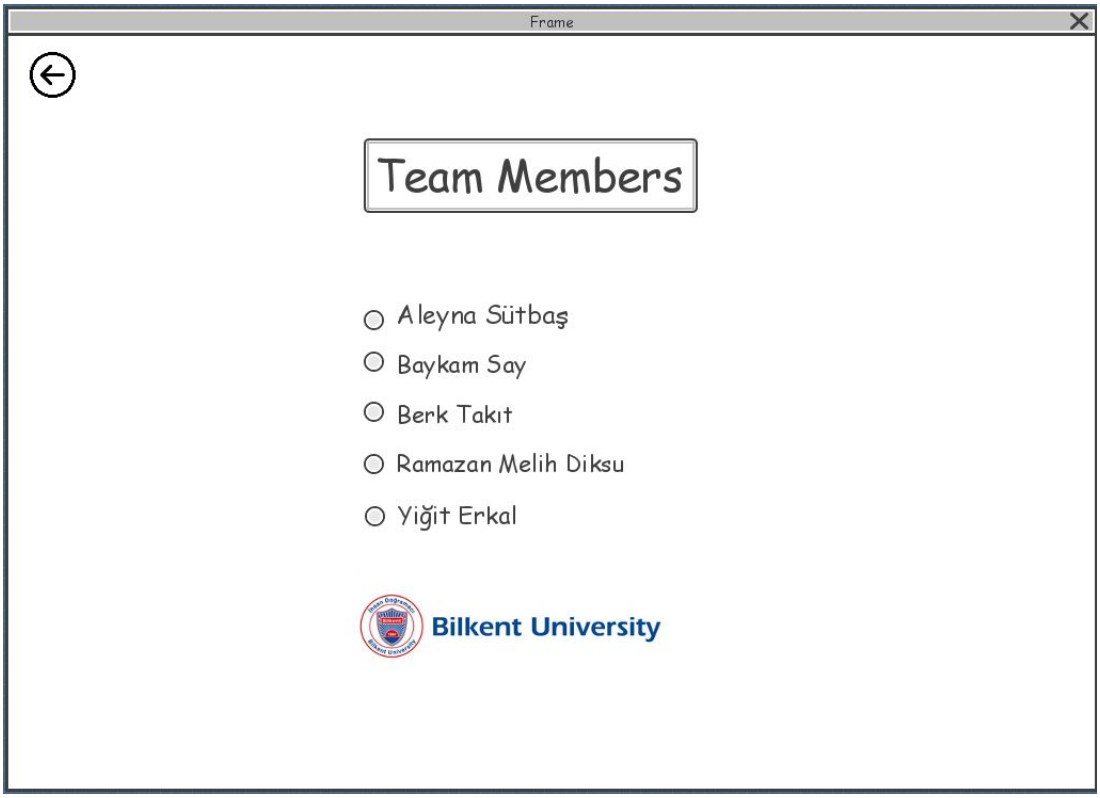
d.Settings Screen



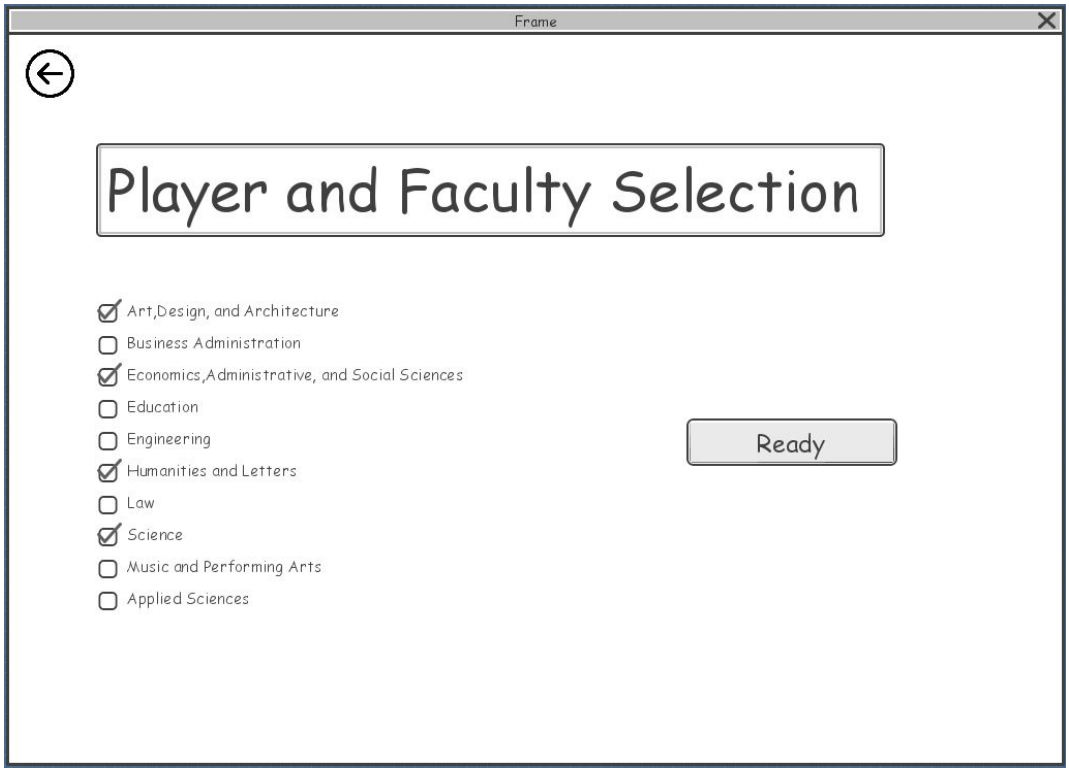
e.Help Screen



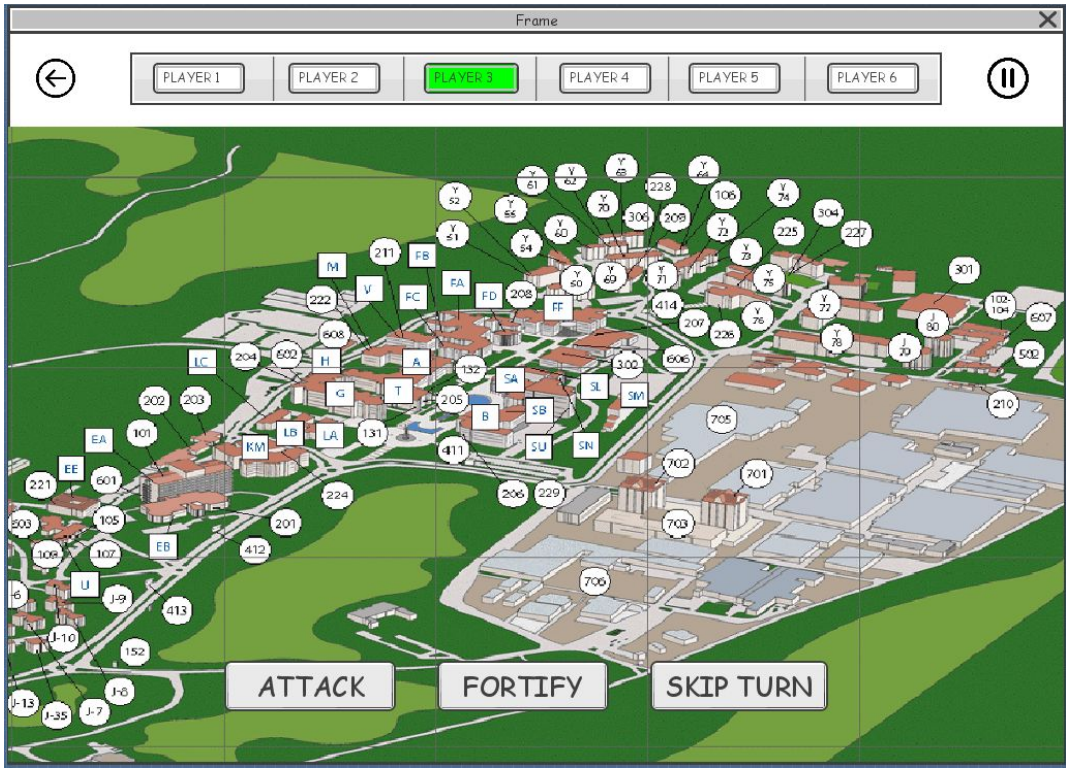
f.Credits Screen



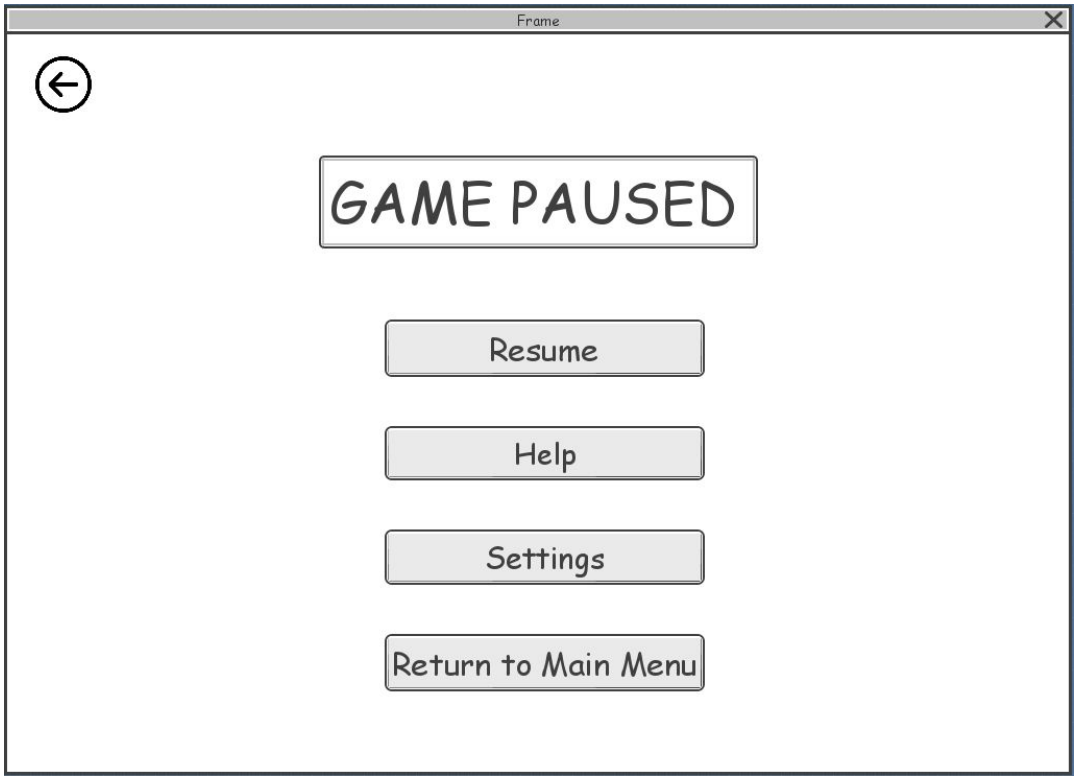
g.Character Selection Screen



h.Main Game Screen



i.Pause Game Screen



j.End Game Screen

6. Glossary & references

[1] “State Pattern”. https://en.wikipedia.org/wiki/State_pattern. [Accessed: Oct 27, 2020].
[2] “Game State Management”. http://blog.nuclex-games.com/tutorials/cxx/game-state-management/. [Accessed: Oct 27, 2020].