Bilkent University

Department of Computer Engineering

Object-Oriented Software Engineering Term Project

CS 319 3G: RISK

Final Report

Group Members:

Işık Özsoy 21703160

Defne Betül Çiftci 21802635

Burak Yetiştiren 21802608

Alp Üneri 21802481

Mustafa Hakan Kara 21703317

Instructor: Eray Tüzün
Teaching Assistant(s): Elgun Jabrayilzade, Emre Sülün, Barış Ardıç

Design Report

Dec 20, 2020

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.

Introduction

Risk is a strategy board game aimed for two to six players of diplomacy, conflict, and conquest. The standard version of the game consists of a board in which a map of the world is depicted. The map is divided into forty-two territories which are grouped into six continents. Turns rotate among players in control of armies with which they attempt to capture territories from other players via battles whose results are determined by dice rolls. The number of dice a player is able to attack or defend with is dependent on the number of soldiers that player controls in the applicable territories. The classic goal of the game is to occupy every territory on the board and achieve total global domination. Both formal and informal alliances may be formed during the course of the game.

Our project is based on RISK Global Domination, however some certain features will be added/removed, such as:

• We plan on allowing players to be able to leave territories in their control unguarded should they choose to.

• We plan on allowing players to set the goal of capturing a continent before the game starts and receive rewards should they be able to do so.

• We plan on implementing a rock-paper-scissors based combat system where more skill is involved as opposed to dice rolls.

• We plan on implementing an alliance mechanism where the two players allied will be allowed to go through their ally's territories to attack an opponent's territory. The allies will not be able to attack each other for the duration of their alliance. This implementation will not be available if only 2 players are playing.

• We plan on implementing an airplane mechanism. A player will be able to build an airport on his territories by paying a cost.

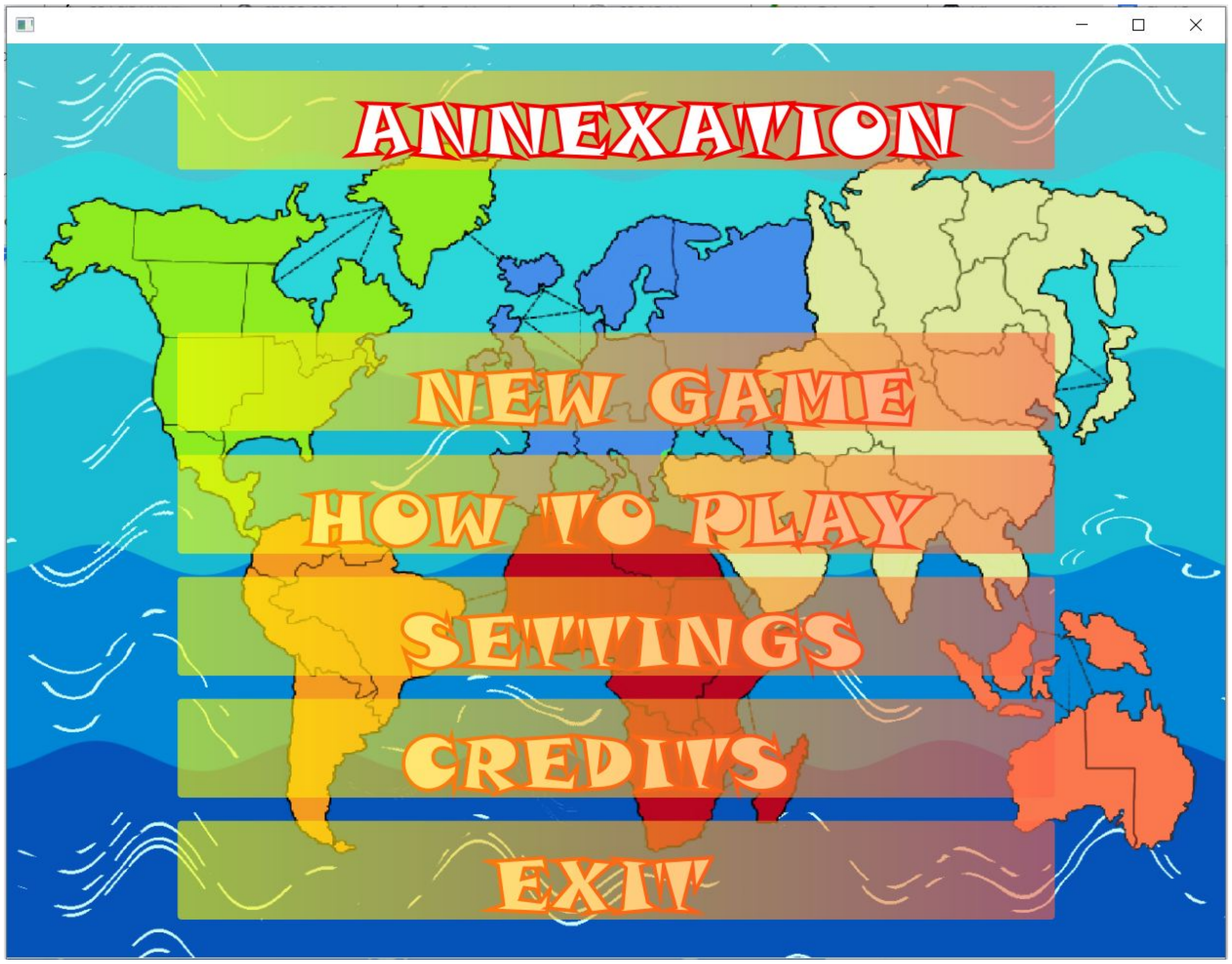
We have not been able to implement our Local File Management subsystem. Therefore our game does not currently have the ability to save or load a game into a .txt file and parse said file to create a game when loading it from the save file. This was due to time constraints and the fact that we had deemed those features as low priority in our analysis report. Our GUI is likewise not as pretty as we would have liked it to be, yet it is in a functional state. We did not have much time to work on prettying up our GUI and instead focused on making sure it was usable. We as well did not implement the settings screen on the main menu, currently nothing happens when the user clicks that button. Other than these however, our game is in a fully functional state.

Lessons Learnt

- We learnt about the importance of designing a system beforehand as opposed to jumping straight to coding.
- We learnt how problematic it is to write software with a GUI as opposed to one that is run on console.
- We had written some functionalities such as rock-paper-scissors to take inputs from the console, thinking that we could easily adapt them to work via the GUI when we wrote the GUI. However, this process was not quite as simple as we had thought, we had to essentially rewrite some whole methods.
- We learnt the importance of scheduling many check-in meetings to see what everyone is doing/what everyone has done regarding the project between the meetings. We saw that the more frequent we had our meetings, the more likelihood that people would work more on the project.
- We learnt the importance of noting down decisions we have made regarding implementation details, sometimes we would vote on an issue in a previous meeting but not remember the outcome of our vote in the next meeting. This is also indicative that the issue may not have been so important to require a vote on anyways.
- We learnt that there are far more corner cases than expected, when writing the code we would constantly find corner cases and have to ask our teammates on what to do.
- We learnt that being comfortable with an IDE is an important skill to have. In the beginning phases of the project we spent almost as much time actually coding the project and trying to use our IDE to access GitHub or making sure all the assets we wanted was properly given a filepath et cetera.
- We learnt just how frustrating it is creating a GUI using only code (as in without using any visual aid tools). It was difficult aligning buttons or getting text to stand right where we wanted it to et cetera.

User’s Guide

Upon launching our game, the user will be met with the following screen.

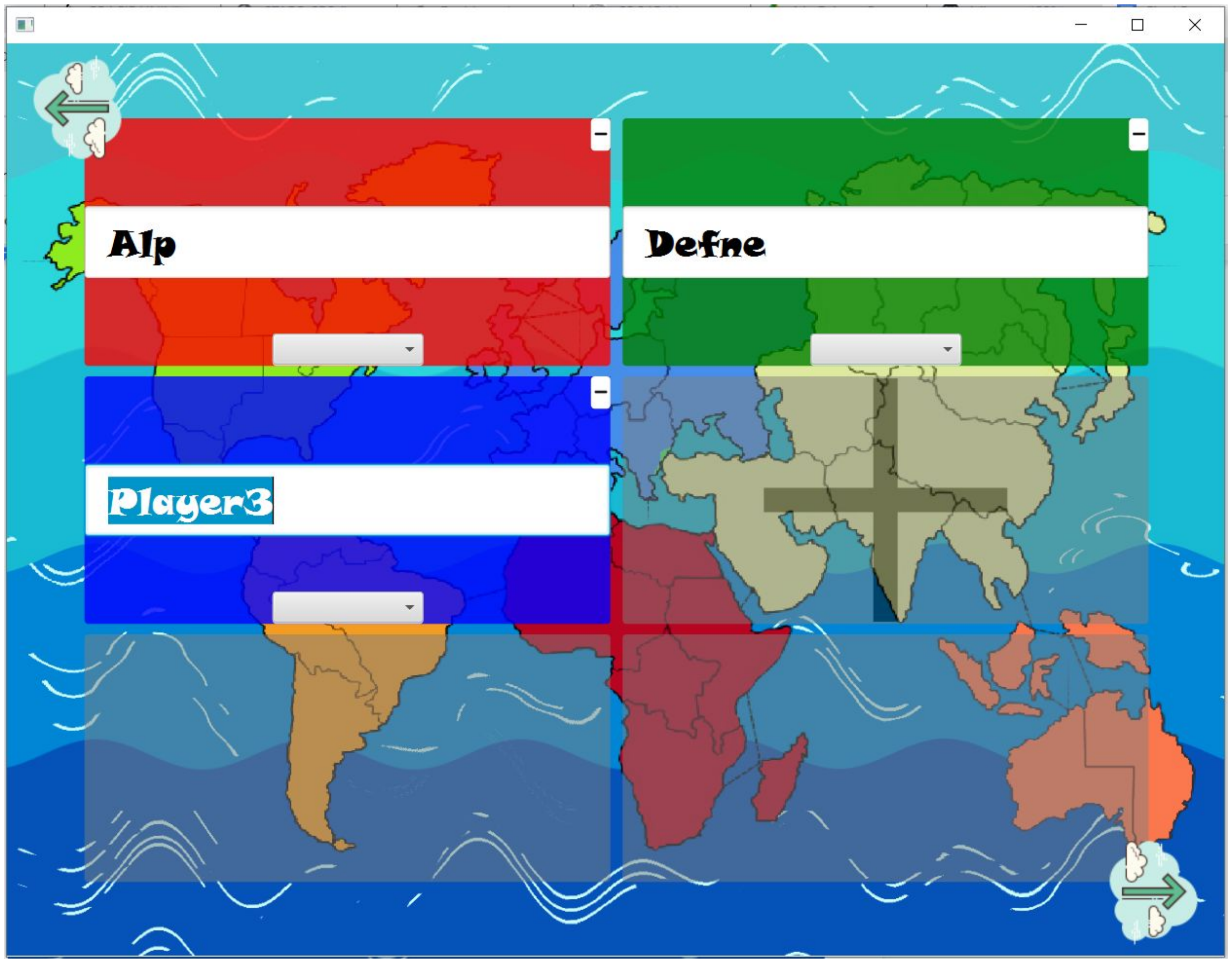


From this screen the user will be able to click on the buttons which will do the following:

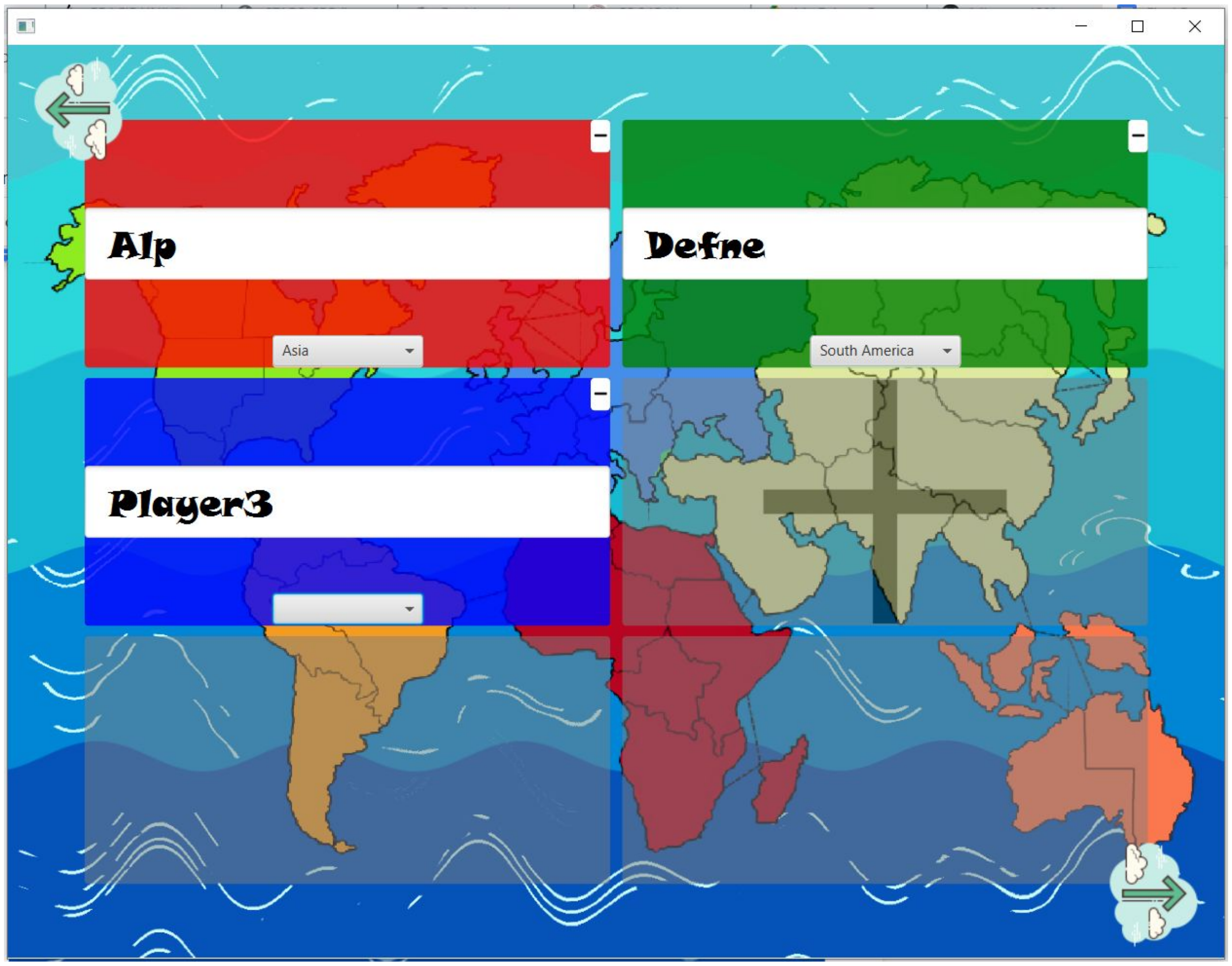
- New Game: Advance into the player number and color selection screen to start a new game.
- How to Play?: Advance into the help screen where the user will be able to learn how to play the game.
- Settings: We have not currently implemented this screen. Therefore, this button does nothing right now.
- Credits: Advance into the credits screen where the user will be able to see the creators of the game.
- Exit: Quit the game and close the application.

The user will only be able to click the buttons, our menus will not be navigable via the arrow buttons.

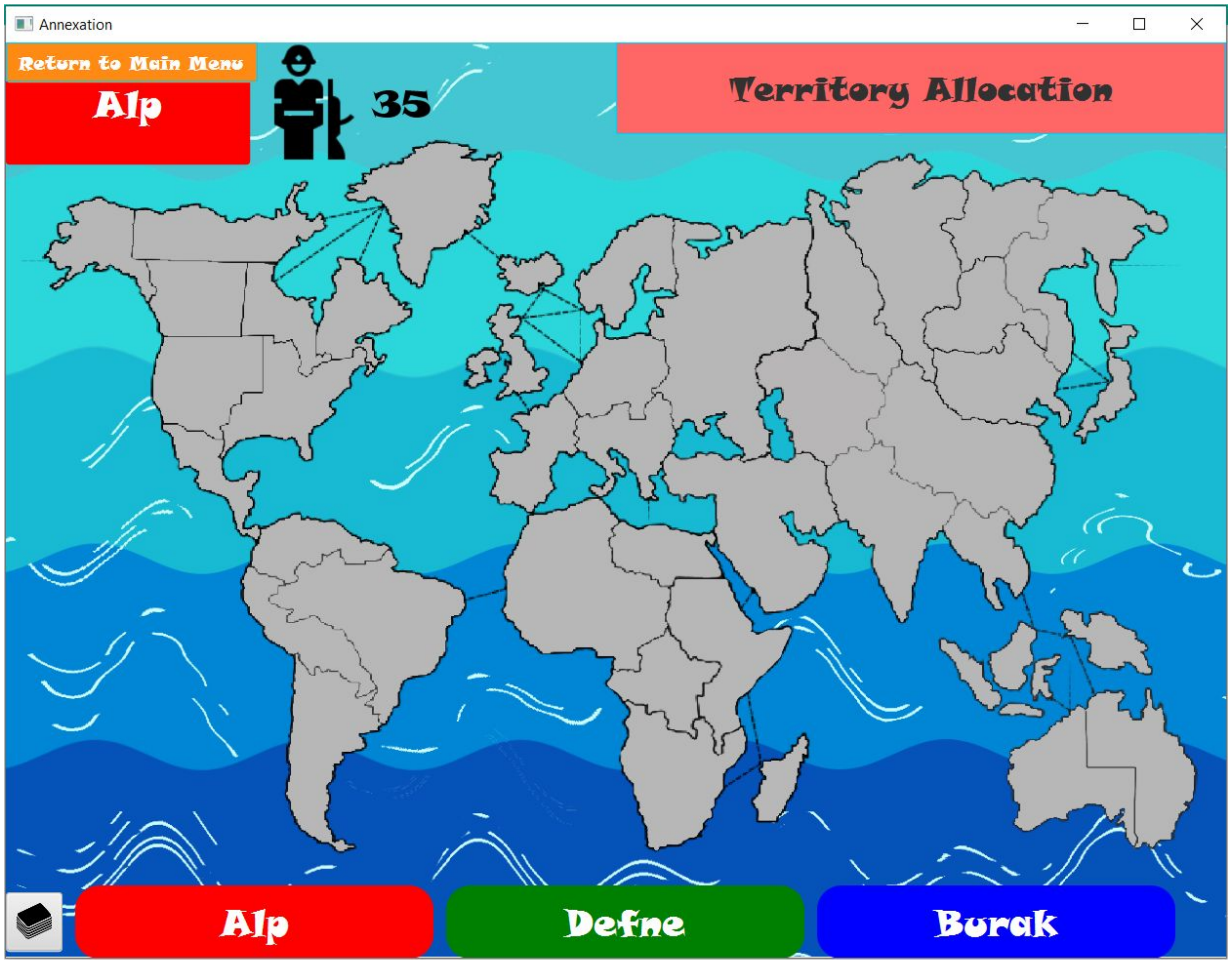
Once the user clicks the New Game button, they will be met with the following screen.



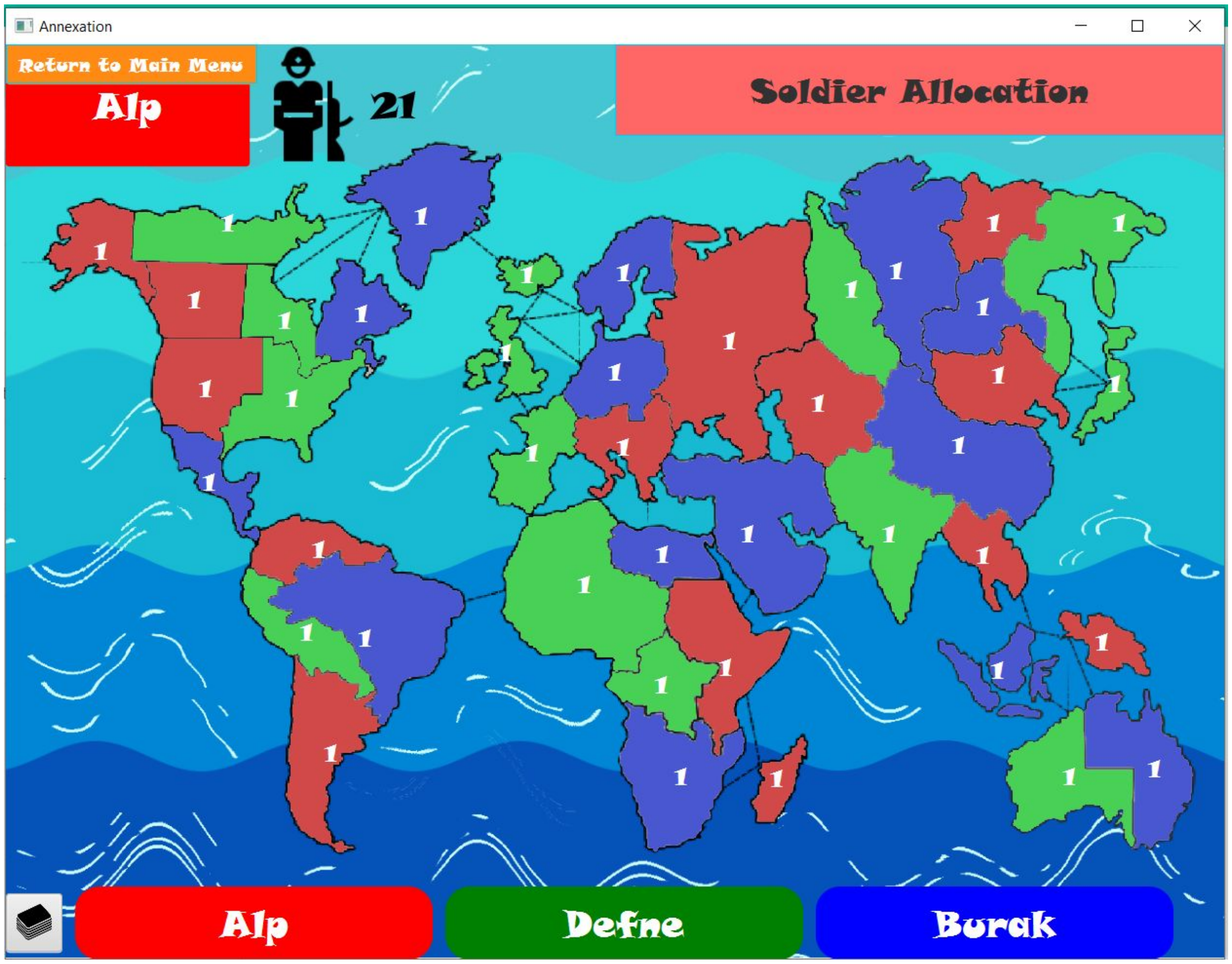
Here, the user will be able to click on the + icon to add more players. They will also be able to choose a name for the current players by clicking the name of the user and entering a name via the keyboard. They will also be able to change the color of the player by clicking the color colored part of their box, and the target continent using a drop down menu. They can also press the arrow on the upper left corner to return back to the main menu or the Play button at the lower right corner to advance into the game.



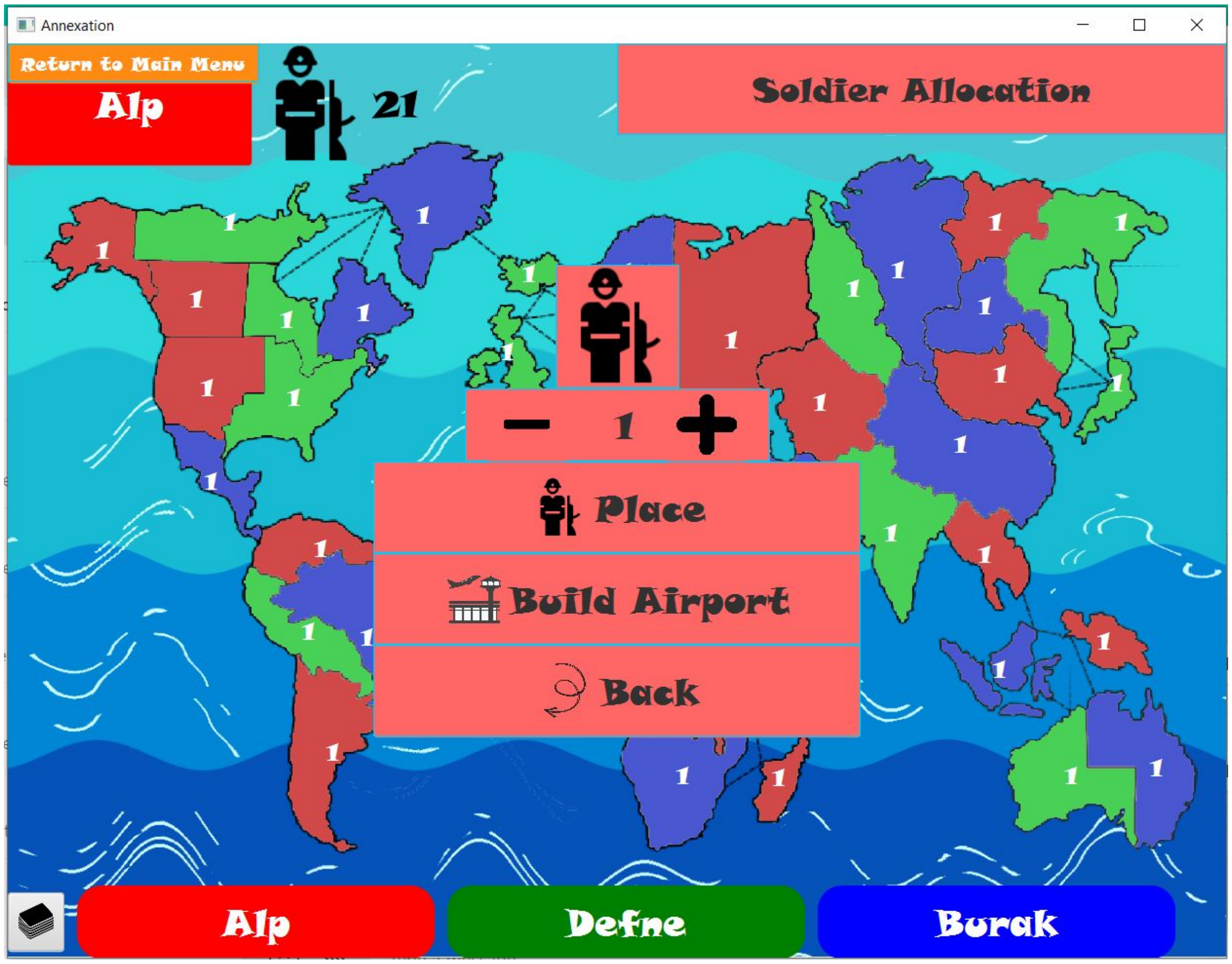
This screen shows how target continents are seen before the start of the game.



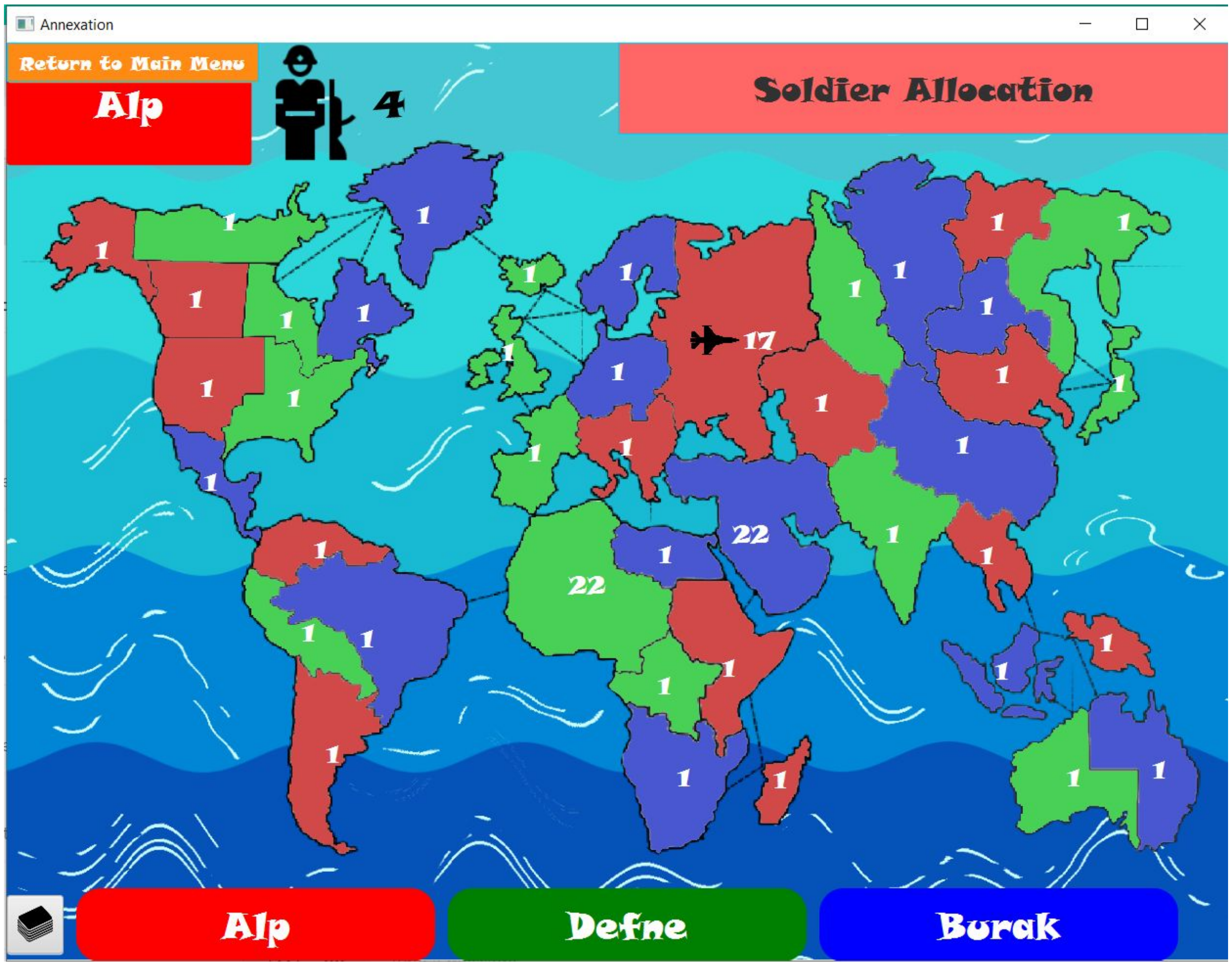
Once they start a game, they will be met with the following screen. This is the territory allocation phase, users will be able to click on an empty territory one by one to claim that territory as one of their starting territories. Doing so will as well automatically place one troop on that territory. The user will be able to see how many troops they have remaining on the upper left side of the screen, next to their name under the soldier icon. All the players will be shown at the bottom center of the screen with their name and color. At any time during play, players will be able to press the orange Return to Main Menu button at the top right hand side of the screen to return to the main menu. The phase the player is in will always be shown on the upper right hand corner of the screen.



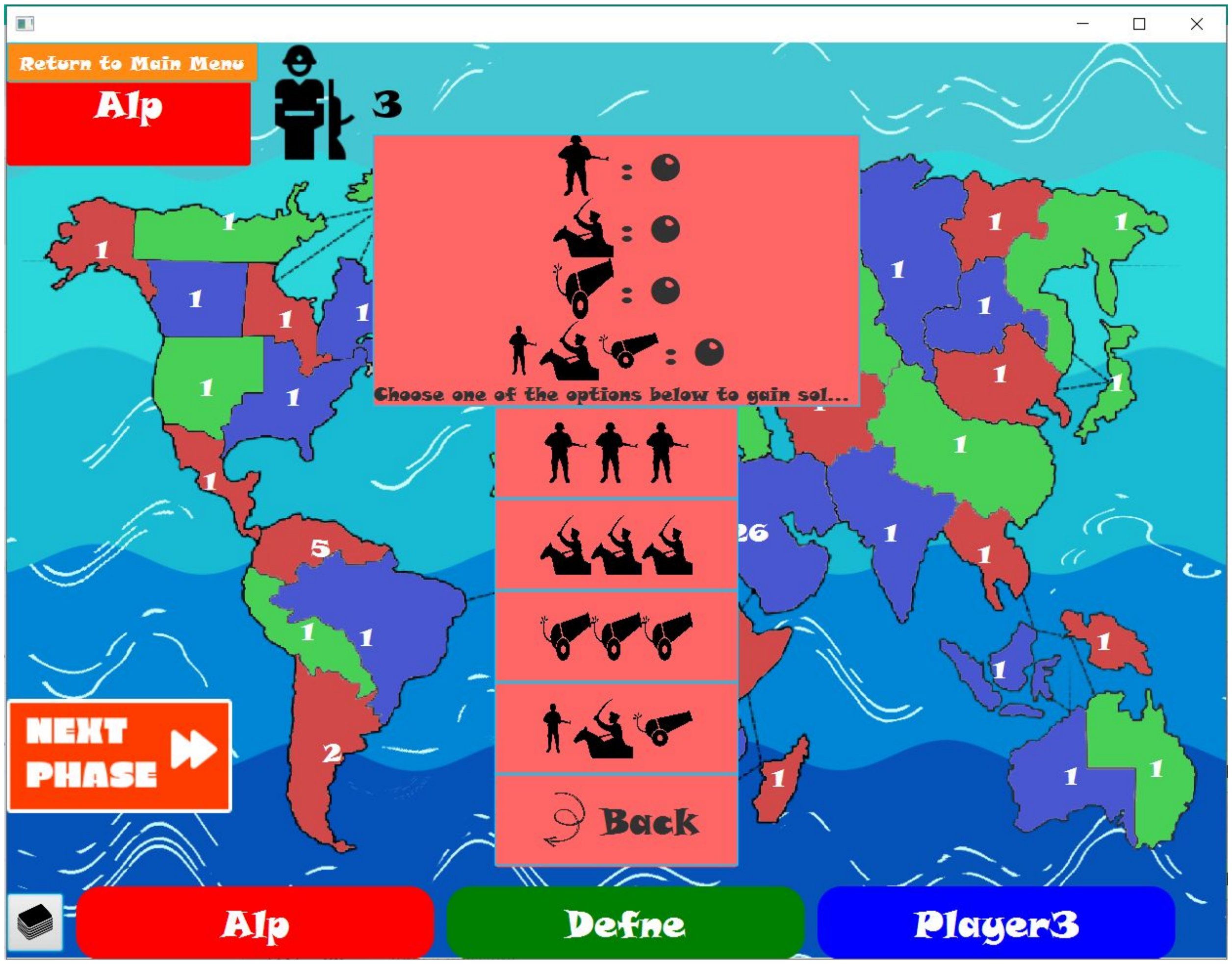
Once territory allocation is complete will come the initial soldier allocation phase. Here, players will be able to place their soldiers on territories which they have chosen as part of their starting territories. How this is done is shown in the next screenshot.



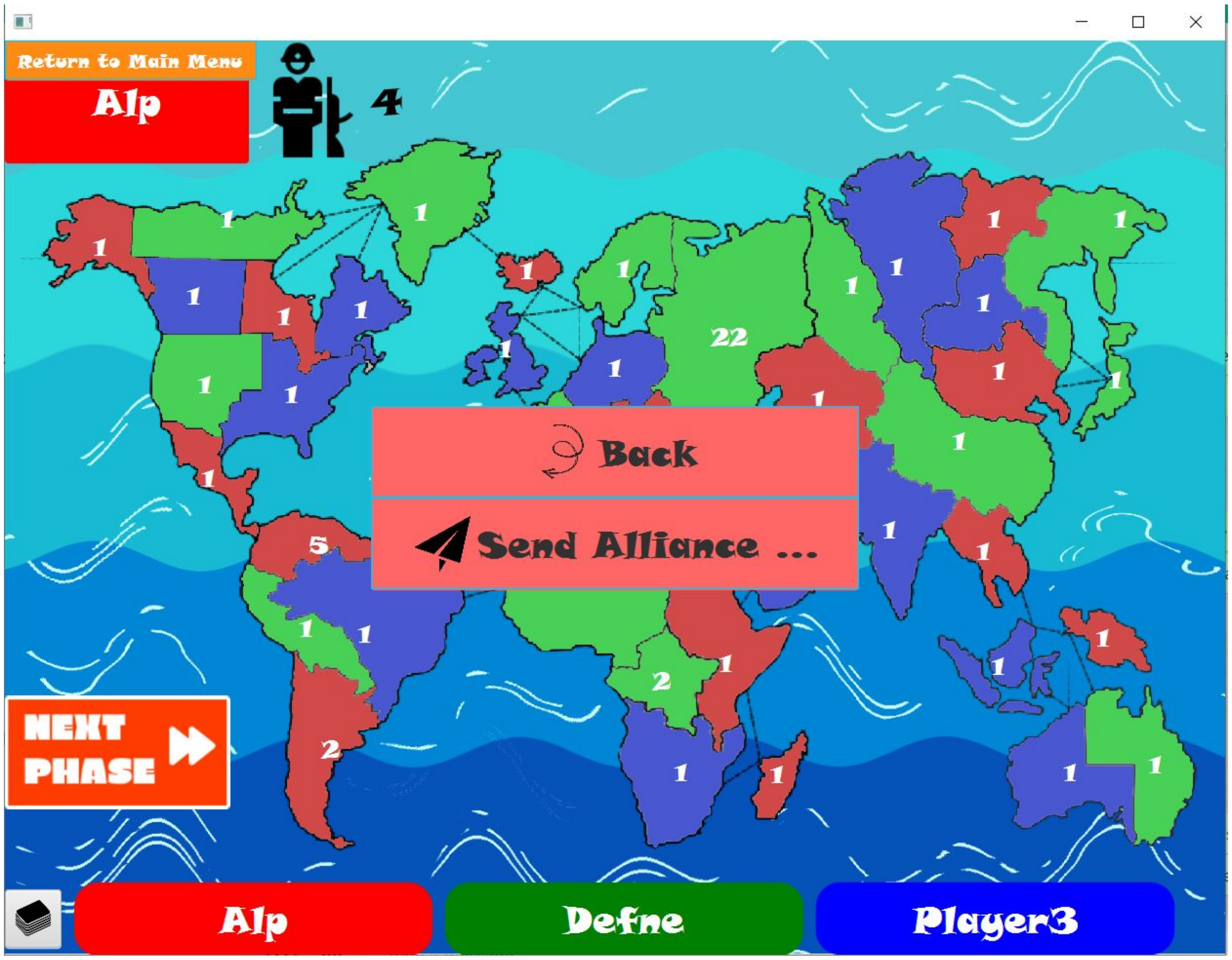
This will be the soldier allocation phase of a player's turn, which is their first phase. Here, they will be able to click a territory and place as many soldiers as they wish onto that territory, up to the number of soldiers that they currently have which will be shown on the upper left side as before. They will also be able to build an airport on a territory after clicking on it using the build airport button.



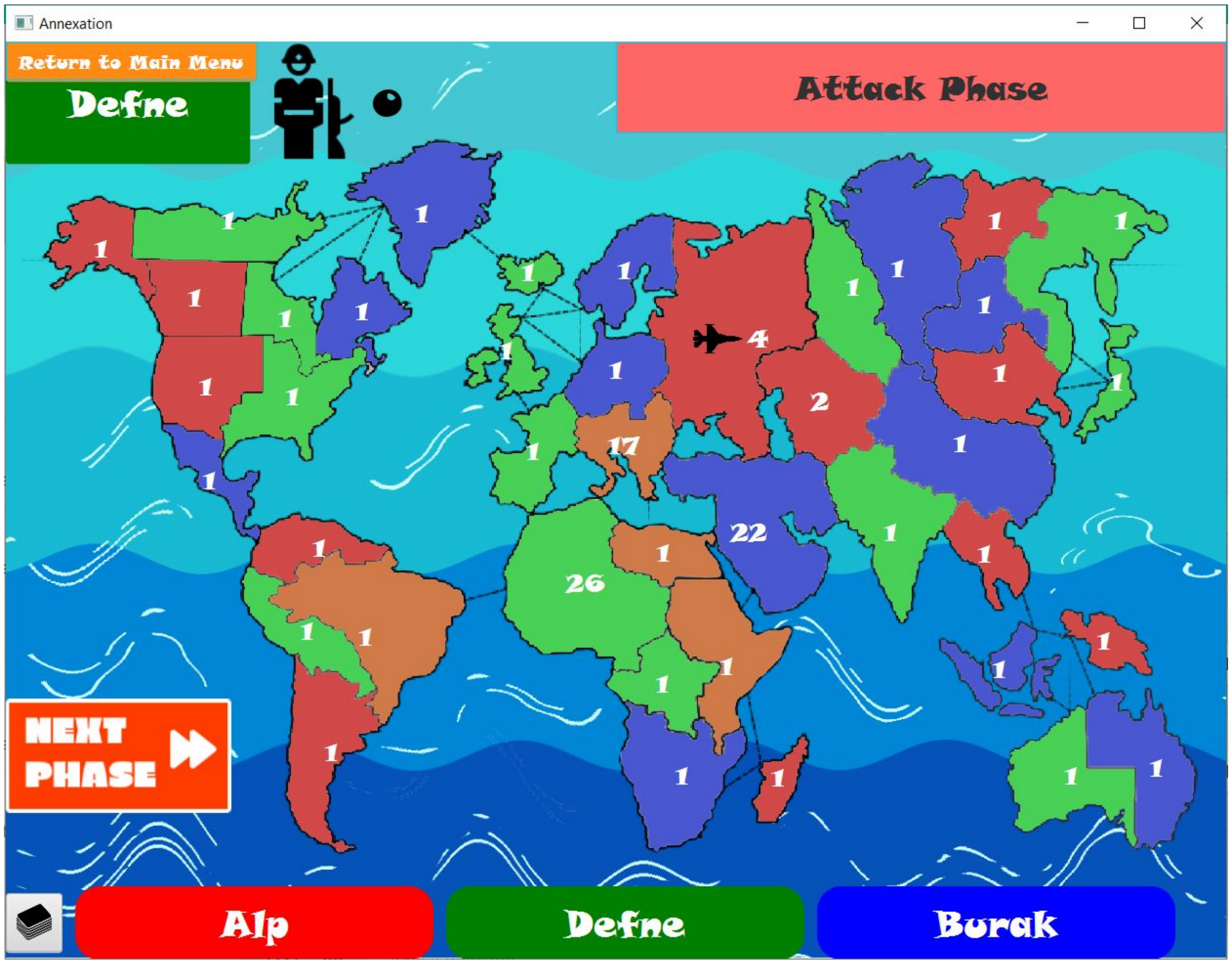
Once a player builds an airport on a territory, that territory will have the airport icon on it, next to its soldier count. On the above screenshot, the territory belonging to Alp (the red player) with 17 troops has an airport, for instance.



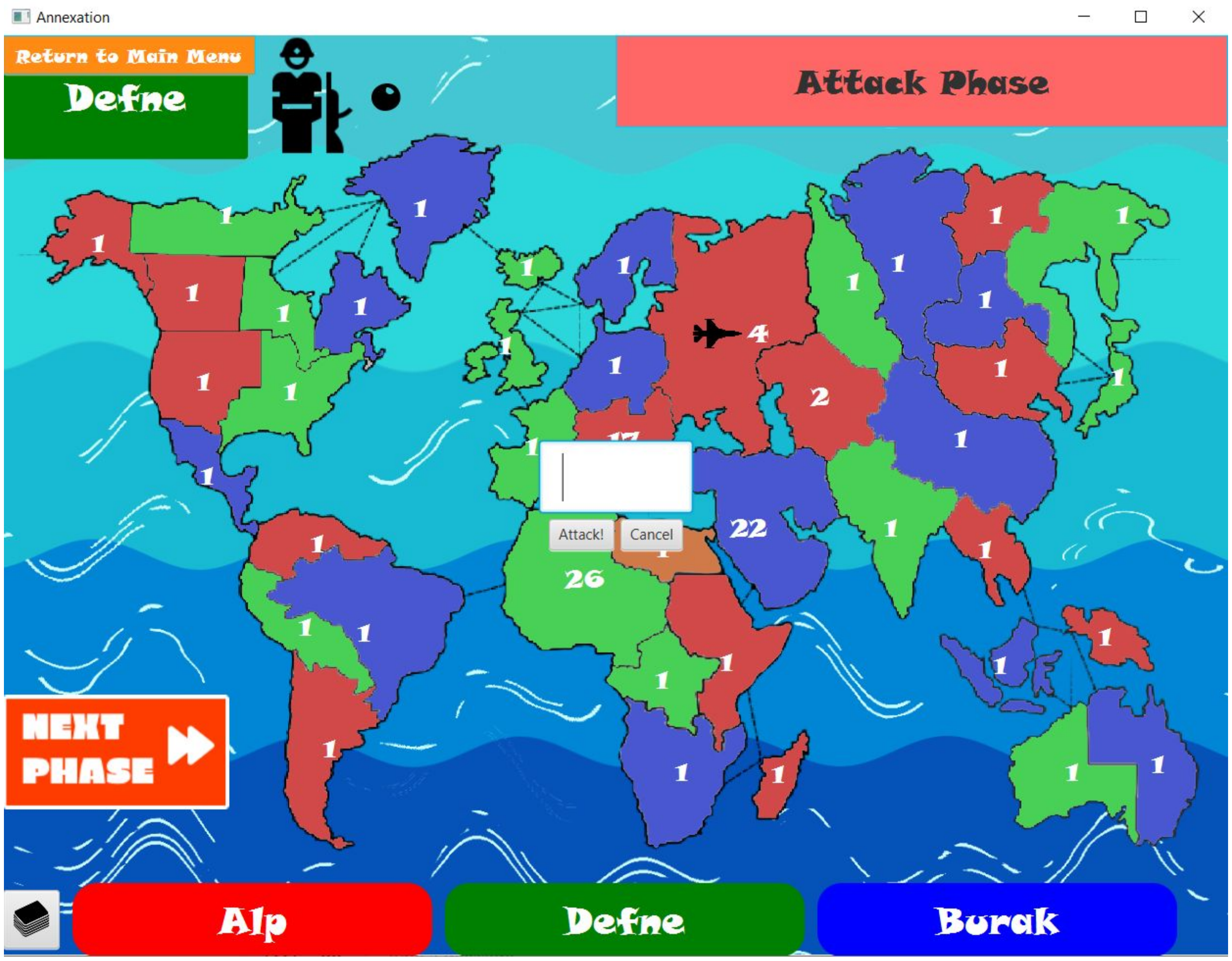
During play, a player will be able to click the cards icon found in the lower left hand side of the screen to open the card exchange window. From this window the player will be able to see how many cards they have of each type, and will be able to click one of the four buttons found below to exchange them for soldiers. These buttons will always be active, if the player presses a button representing an exchange for which they do not have the right cards, nothing will happen.



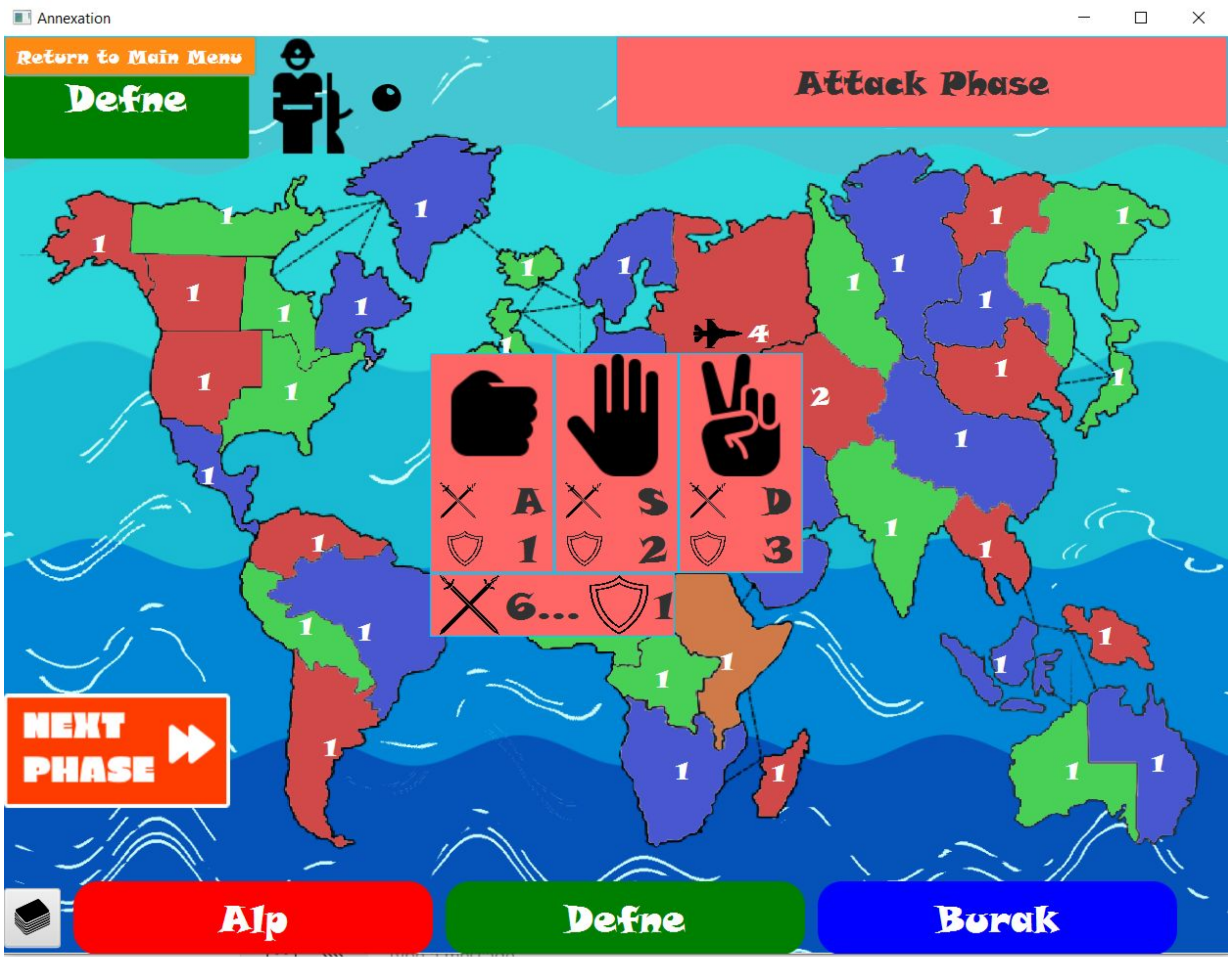
A player will be able to send an alliance request to another player by clicking on their icon and clicking the send alliance request button. Also, once a player has allocated all of their soldiers they will be able to click the next phase button on the lower left to get to the next phase.



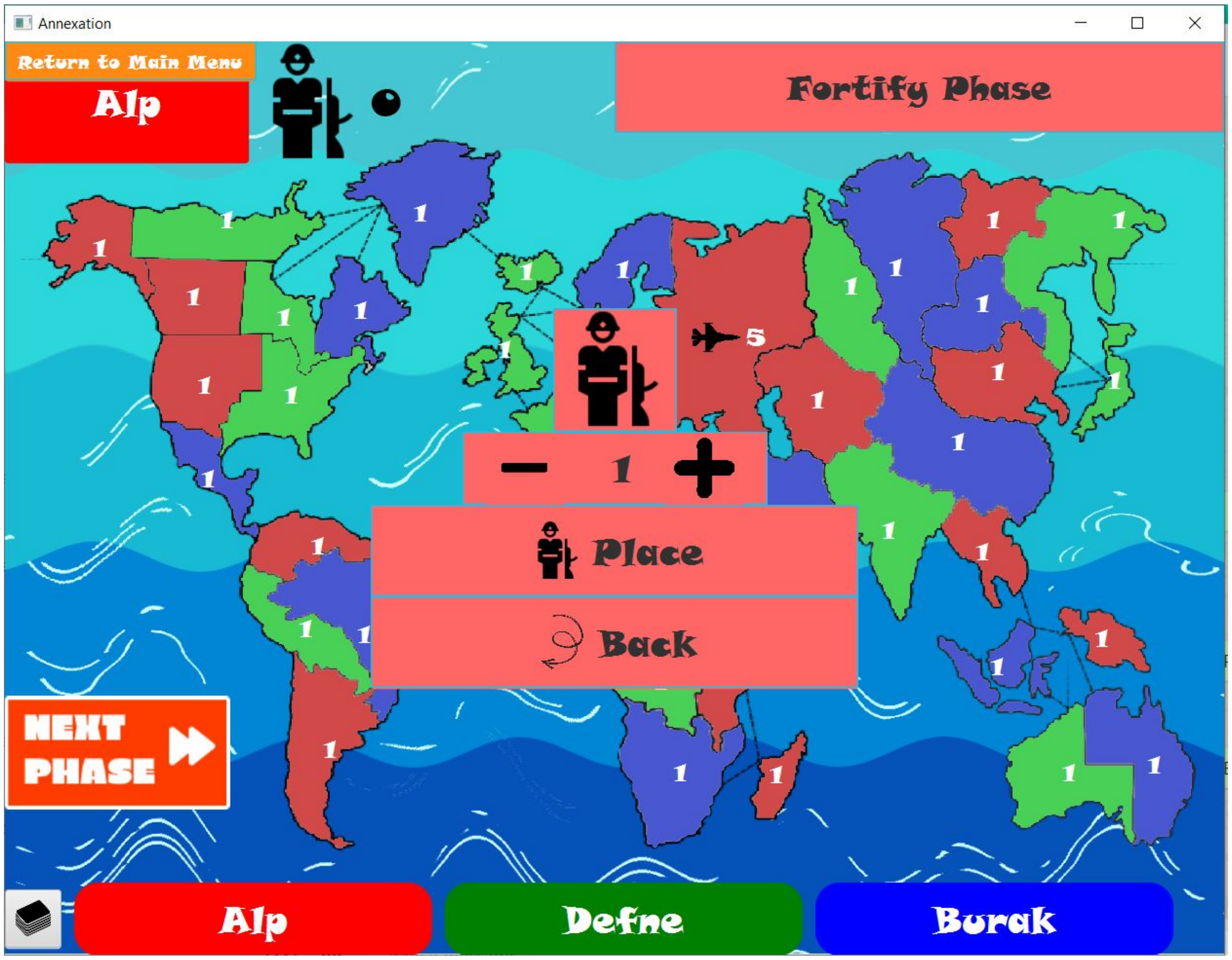
During their attack phase, a player will be able to click on one of their territories to see the territories to which they can attack from that territory. A player may then declare combat to one of the highlighted territories. This is done by clicking that territory.



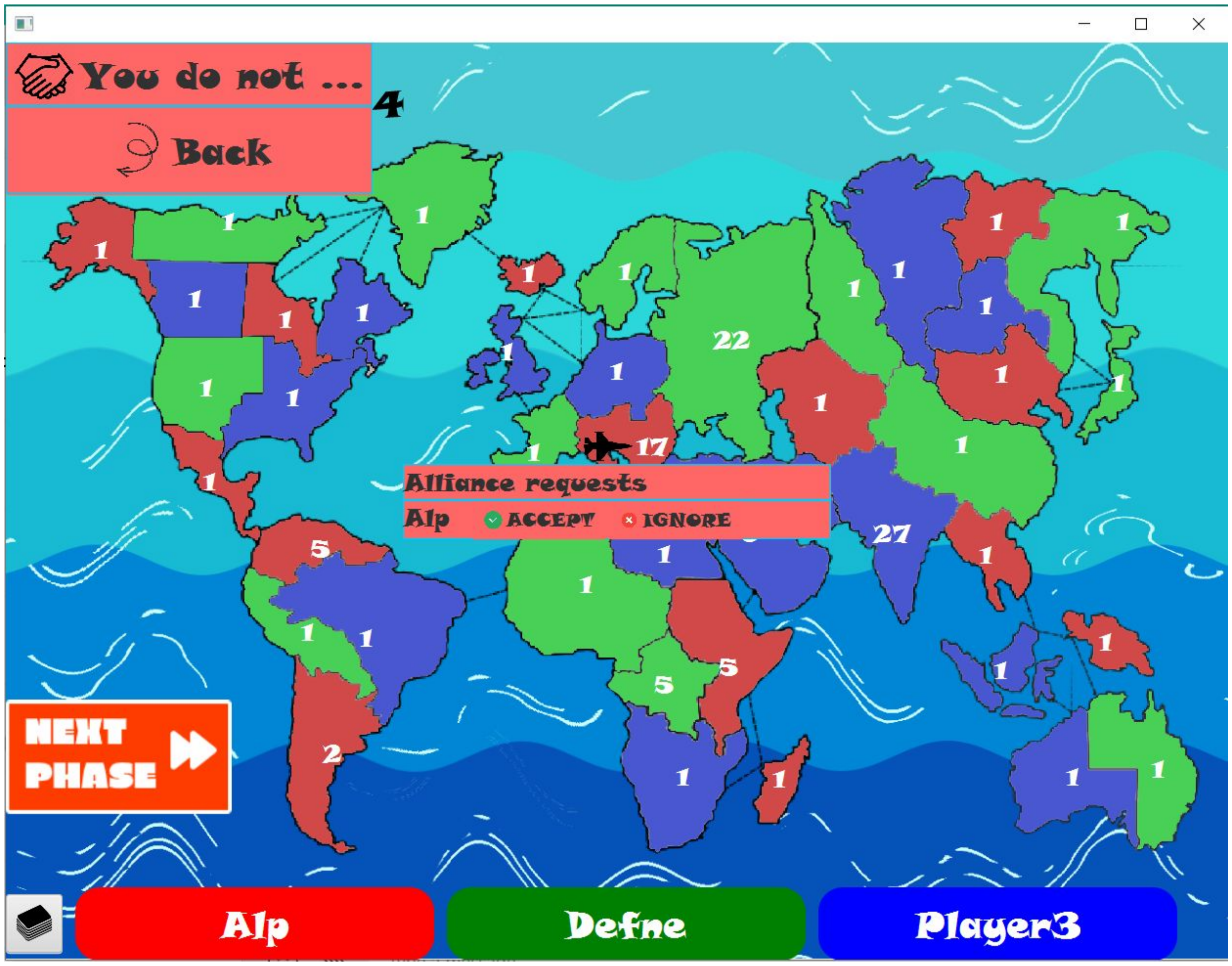
Once a player chooses which territory to attack, they will then have a textbox appear into which they will write how many soldiers they wish to attack that territory with. Once they do so, they will be met with the rock-paper-scissors overlay.



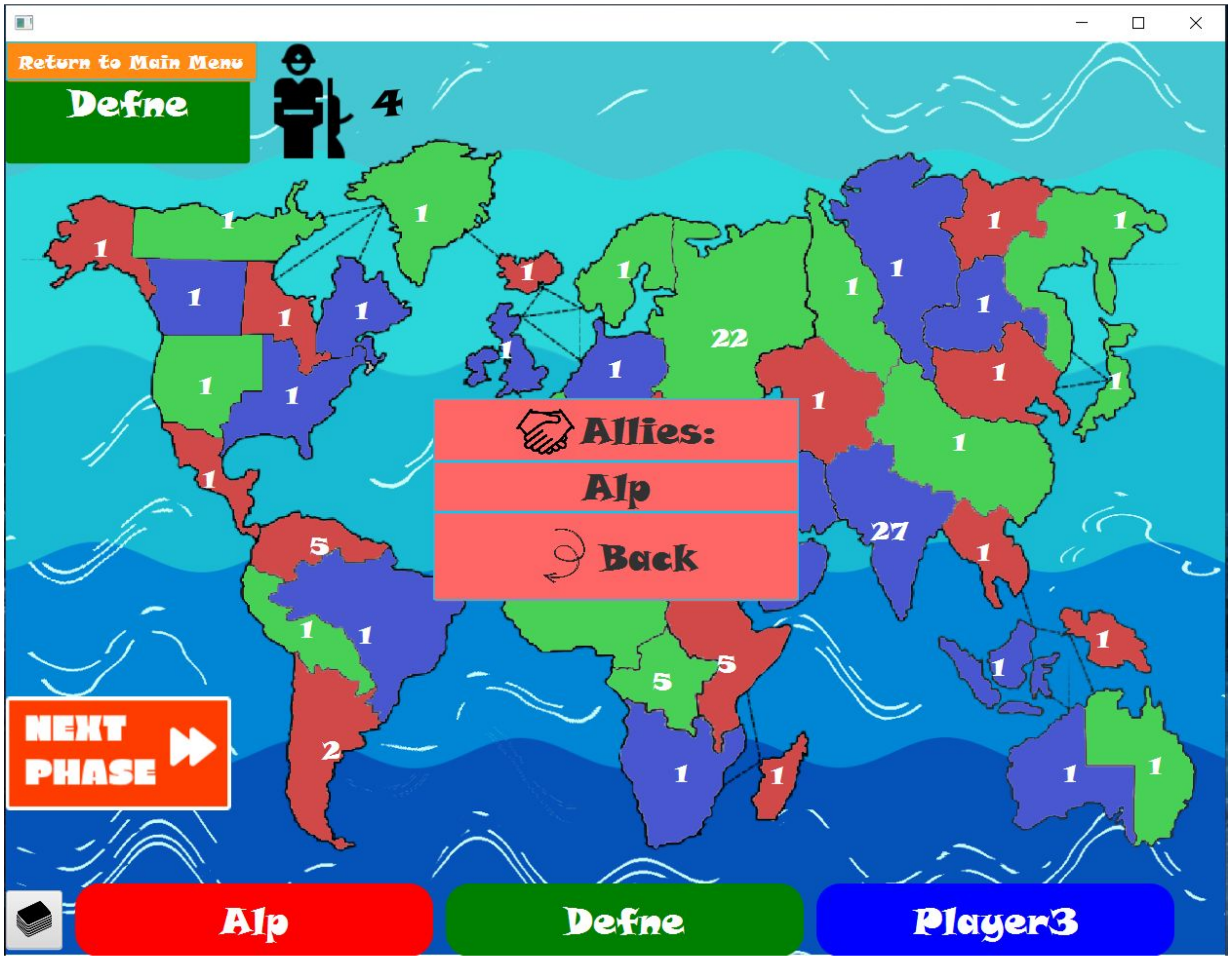
The above screenshot shows the rock-paper-scissors overlay for the battle. Players will be able to see the number of soldiers they have with which they are attacking the opposing territory as well as the number of soldiers their opponent has defending that territory. Players will as well be able to see which button corresponds to which shape of rock-paper-scissors on this screen.



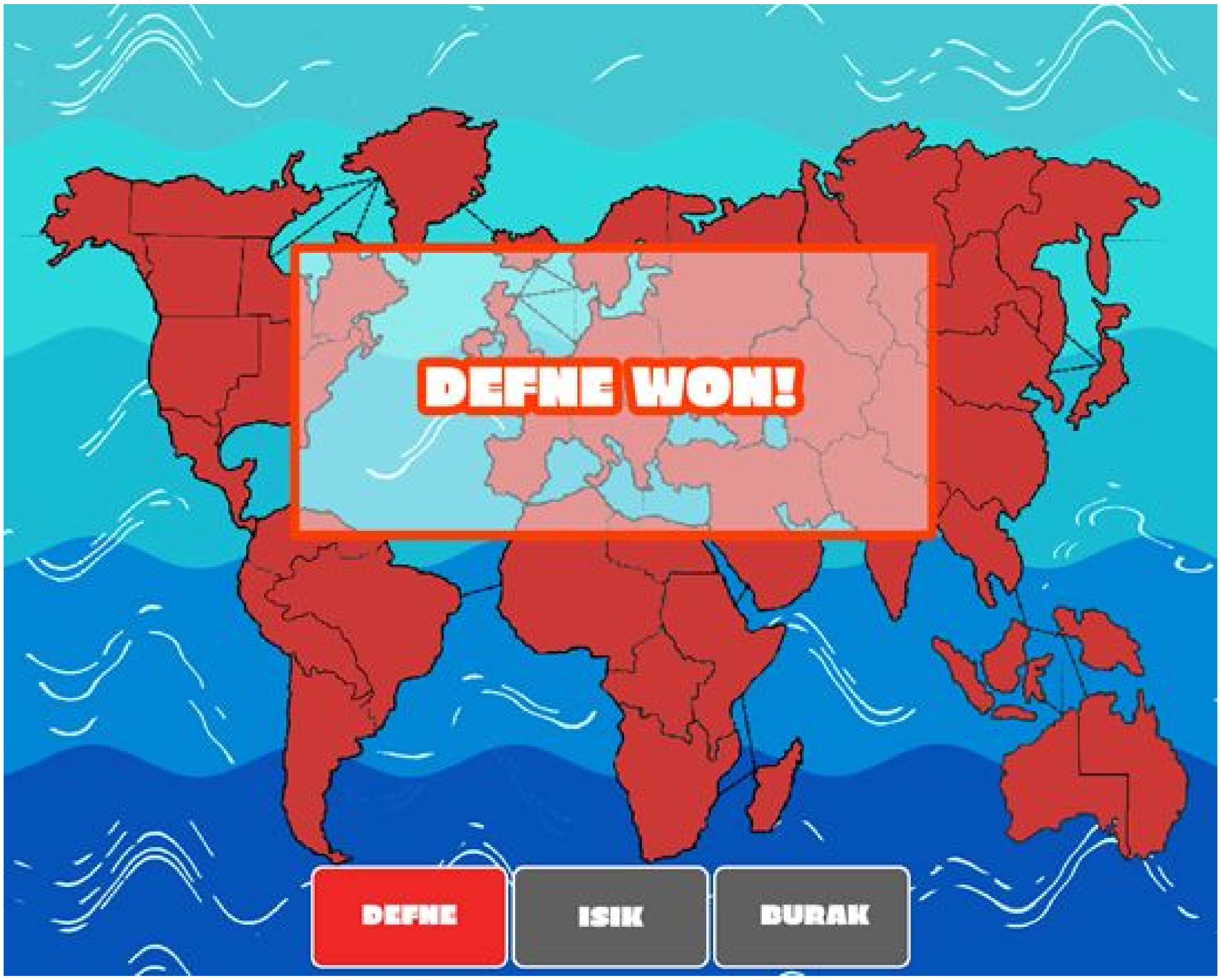
From this screen, the player will be able to choose an origin territory and a destination territory from which they will be moving soldiers over. They will be able to choose how many soldiers to move from the origin territory to the destination territory using the arrows found in the middle of the screen.



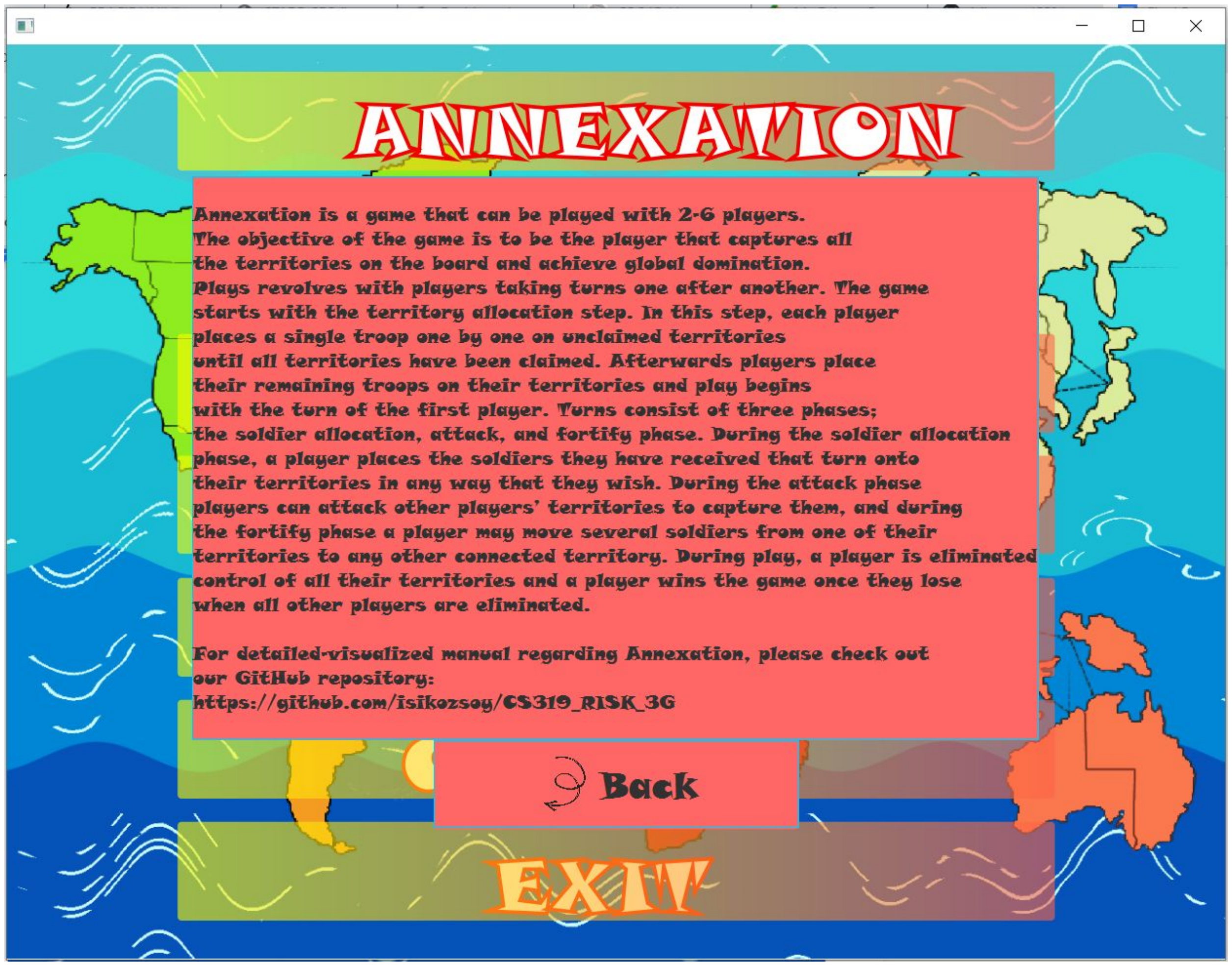
A player will be able to click on their name at the upper left hand corner of the screen to bring up the alliance menu. Here, the player will be able to see any alliance requests that they have as well as accept or ignore them by pressing the respective buttons. If they have any alliances they will be listed in the manner shown in the following screenshot.



When a player clicks their own name at the upper left hand on the screen to open the alliances menu, if they are currently allied with any players they will see them in the above format.



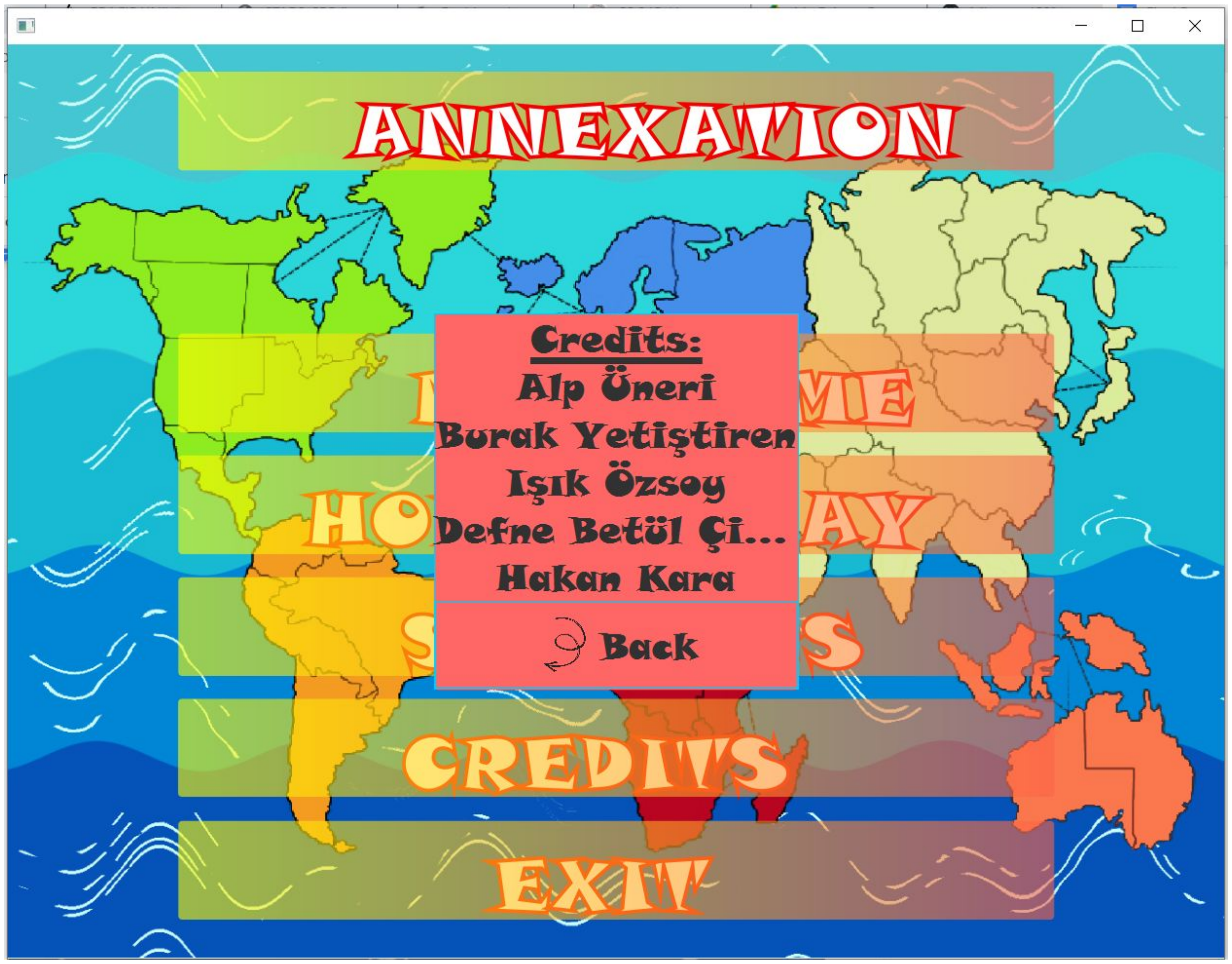
Once a player is able to achieve global domination, they will be met with the above screen, where the map is colored in their color and only their name is not greyed-out at the bottom middle of the screen. The text in the middle will also be in their color as well as the frame of the text box.



Annexation is a game that can be played with 2-6 players.
The objective of the game is to be the player that captures all
the territories on the board and achieve global domination.
Plays revolves with players taking turns one after another. The game
starts with the territory allocation step. In this step, each player
places a single troop one by one on unclaimed territories
until all territories have been claimed. Afterwards players place
their remaining troops on their territories and play begins
with the turn of the first player. Turns consist of three phases;
the soldier allocation, attack, and fortify phase. During the soldier allocation
phase, a player places the soldiers they have received that turn onto
their territories in any way that they wish. During the attack phase
players can attack other players' territories to capture them, and during
the fortify phase a player may move several soldiers from one of their
territories to any other connected territory. During play, a player is eliminated
control of all their territories and a player wins the game once they lose
when all other players are eliminated.

For detailed-visualized manual regarding Annexation, please check out
our GitHub repository:
https://github.com/isikozsoy/CS319_RISK_3G

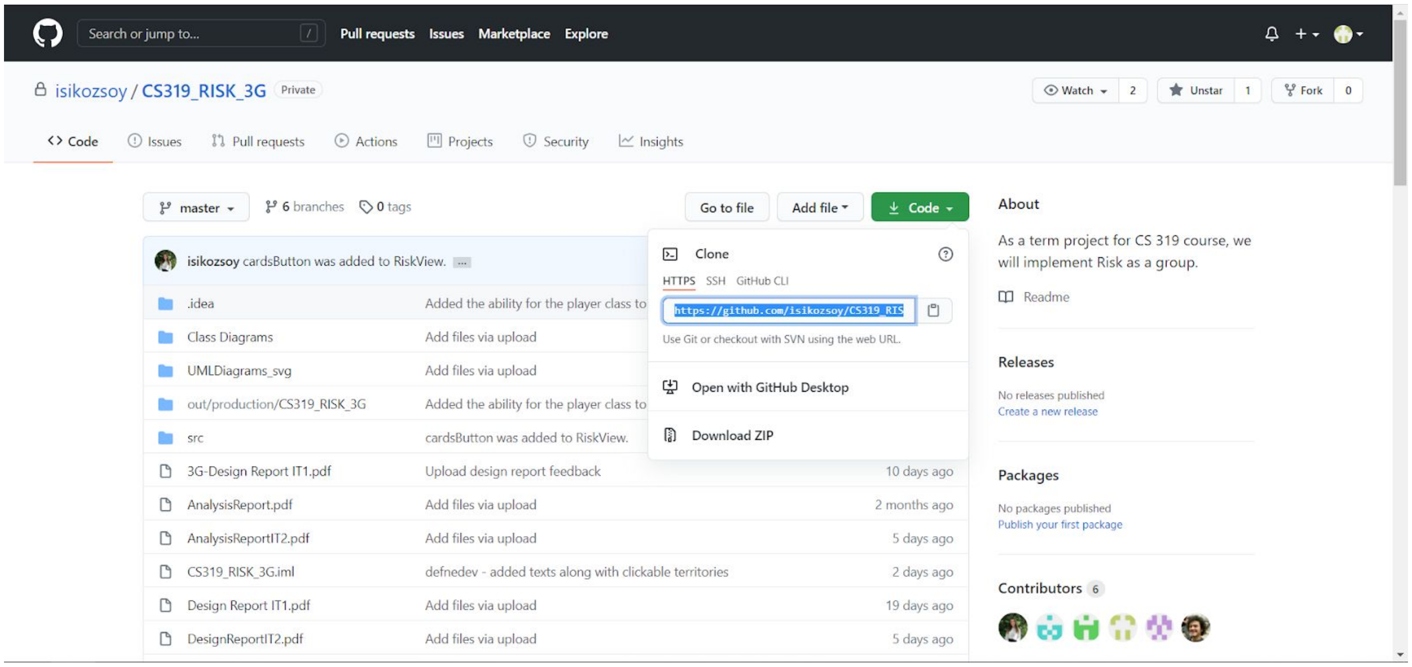
When a player clicks the How to Play button at the main menu, they will be met with this screen where they can read how to play the game. They can press the back button to return to the main menu screen.



In the credits screen the player will be able to see the creators of the game as well as press the back button to return to the main menu.

Build Instructions

There will be a .jar file on our GitHub page which can be downloaded and run to launch our game without having to do anything else regarding installation or set-up.
Alternatively, our game can be run through an IDE. This would require the cloning of our project. To do that, one needs to navigate to our GitHub repository which is found in the following link:
https://github.com/isikozsoy/CS319_RISK_3G
Next, one will have to copy the following link into an IDE that supports Git cloning. We will be using the IntelliJ IDE for demonstration purposes.



https://github.com/isikozsoy/CS319_RIS

Use Git or checkout with SVN using the web URL.

isikozsoy cardsButton was added to RiskView.

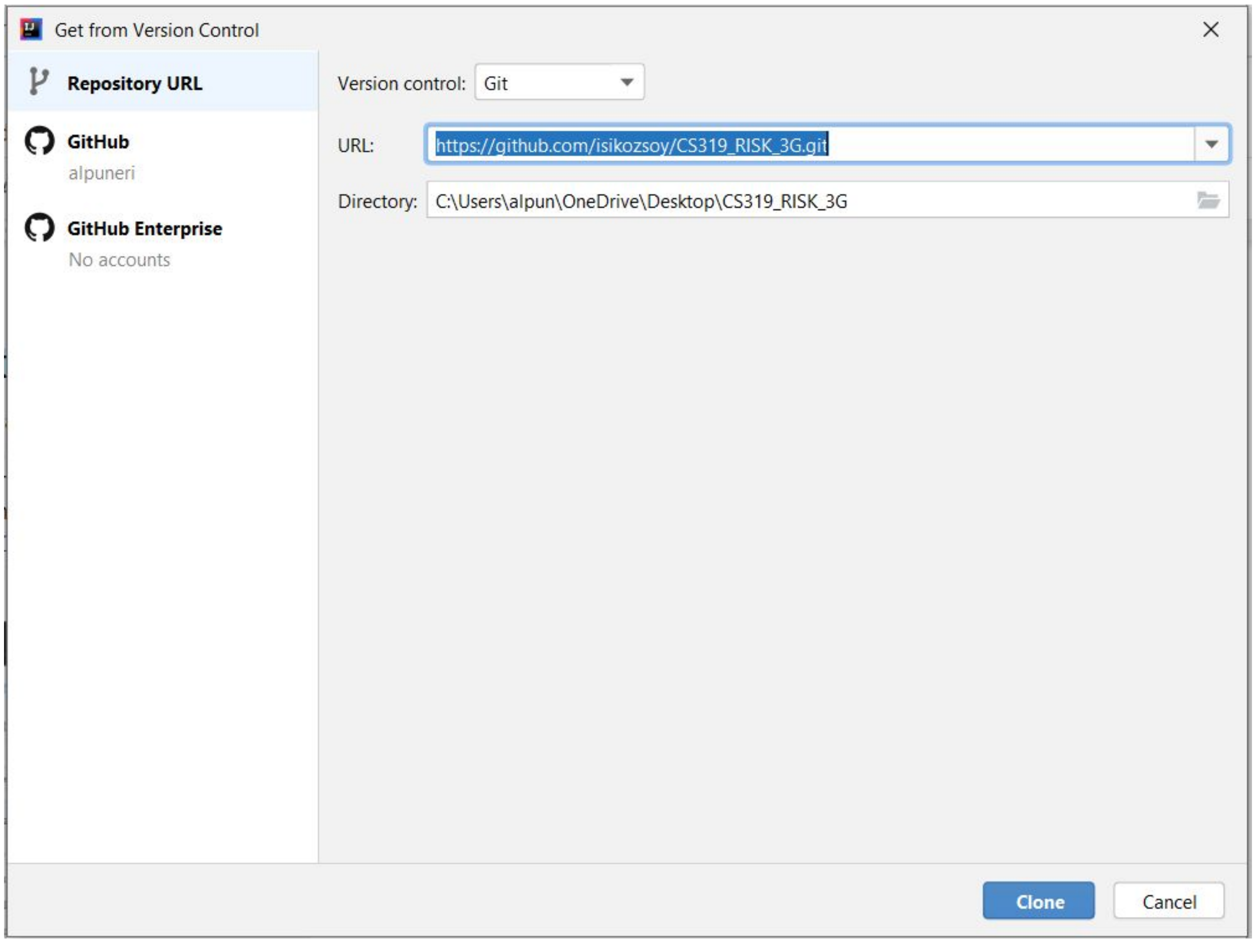
.idea	Added the ability for the player class to	
Class Diagrams	Add files via upload	
UMLDiagrams_svg	Add files via upload	
out/production/CS319_RISK_3G	Added the ability for the player class to	
src	cardsButton was added to RiskView.	
3G-Design Report IT1.pdf	Upload design report feedback	10 days ago
AnalysisReport.pdf	Add files via upload	2 months ago
AnalysisReportIT2.pdf	Add files via upload	5 days ago
CS319_RISK_3G.iml	defnedev - added texts along with clickable territories	2 days ago
Design Report IT1.pdf	Add files via upload	19 days ago
DesignReportIT2.pdf	Add files via upload	5 days ago

As a term project for CS 319 course, we will implement Risk as a group.

No releases published
Create a new release

No packages published
Publish your first package

The cloning link is found on the page above. The link itself is:
https://github.com/isikozsoy/CS319_RISK_3G.git
Next, one will have to paste this link into their IDE in the following way.

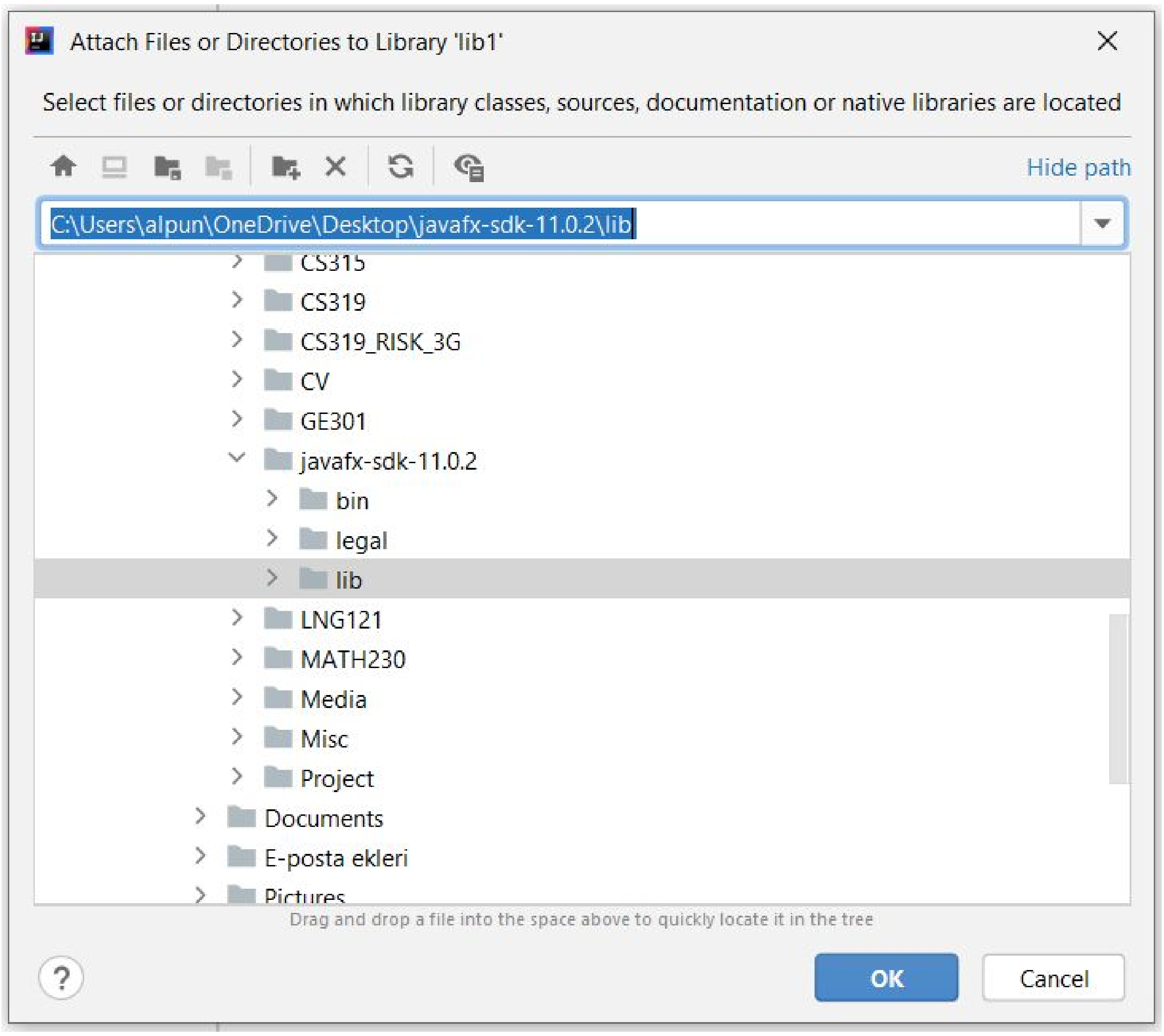


Version control: Git

URL: https://github.com/isikozsoy/CS319_RISK_3G.git

Directory: C:\Users\alpun\OneDrive\Desktop\CS319_RISK_3G

This will automatically pull the project from the master branch of our repository. One can also specify the directory to which the source files will be cloned.
To run our game from an IDE, one will need to have the Java FX library, which can be downloaded from the following link: https://gluonhq.com/products/javafx/
Next, one will have to navigate to the project structure screen on their IDE and add the lib file found in the Java Fx SDK to the project as both a library and a module.


Attach Files or Directories to Library 'lib1'

Select files or directories in which library classes, sources, documentation or native libraries are located

Hide path

C:\Users\alpun\OneDrive\Desktop\javafx-sdk-11.0.2\lib

- CS315
- CS319
- CS319_RISK_3G
- CV
- GE301
- javafx-sdk-11.0.2
  - bin
  - legal
  - lib
- LNG121
- MATH230
- Media
- Misc
- Project
- Documents
- E-posta ekleri
- Pictures

Drag and drop a file into the space above to quickly locate it in the tree

OK Cancel

One will also have to add the following line to the VM configurations of their Run configurations on their IDE: --module-path C:\Users\alpun\OneDrive\Desktop\javafx-sdk-11.0.2\lib --add-modules=javafx.controls
Here, the path should be replaced with the path the lib file within the Java FX SDK is found.
After all these steps, our project can be compiled and run from the IDE. One needs to run the main.java file to play our game.

Contributions of Teammates

Name	Işık Özsoy
Contributions to the Reports	● Analysis Report: Sequence Diagrams, References ● Design Report: Object Design Patterns, Data Access Subsystem, Changes Between Iterations One and Two
UML Diagrams Worked On	Class Diagram, Sequence Diagrams, Deployment Diagram
Classes Worked On	AirportDecorator, ClickableTerritory, Player, RiskGame, RiskView, Territory, TroopCountSelectorPane, AllianceRequestPane, CardExchangePane
Notes	Functioned as the team leader (as in was responsible for making sure everyone was on top of the tasks that were assigned to them, setting up meetings et cetera), worked mostly on the integration of the backend and the frontend of the system as well as contributing to the actual creation of both façets.

Name	Defne Betül Çiftçi
Contributions to the Reports	● Analysis Report: Activity Diagrams, User Interface ● Design Report: Object Design Patterns, User Interface Subsystem
UML Diagrams Worked On	Class Diagram, Activity Diagrams
Classes Worked On	AddPlayersView, AlliancePane, AllianceRequestPane, BeforeGameView, ClickableTerritory, Main, MainMenuButton, MainMenuText, MainMenuView, Player, PlayerButton, RiskGame, RiskView, TroopCountSelectorPane
Notes	Was the premier frontend person of our team. Implemented most of the classes having to do with the GUI as well as designed the look et cetera of the GUI for the analysis report in the first place. Did most of the initial implementation of the User Interface Management subsystem.

Name	Burak Yetiştiren
Contributions to the Reports	● Analysis Report: Use-Case Model ● Design Report: Object Design Patterns
UML Diagrams Worked On	Use-Case Diagram, Subsystem Diagram, Package Diagram
Classes Worked On	AirportDecorator, AlliancePane, AllianceRequestPane, CardExchangePane, ClickableTerritory, Player, PlayerButton, RiskGame, RiskView, Territory
Notes	Worked on the implementation of the backend and the frontend of the system, and as well contributed to the creation of both façets. Did not have a clear-cut assignment to either the frontend or the backend but rather helped with both of the façets as sort of like a bonus-member (meaning that he would help anyone who needed help).

Name	Alp Üneri
Contributions to the Reports	● Analysis Report: Introduction, Current System, Proposed System, Overview, Functional Requirements, Non-Functional Requirements, Pseudo Requirements, Object and Class Model, State Machine Diagrams, User Interface, Glossary, Changes Between Iterations One and Two ● Design Report: Introduction, Purpose of the System, Design Goals, Trade-Offs, Definitions Acronyms and Abbreviations, Current Software Architecture, Proposed System Architecture, Subsystem Services, User Interface Subsystem, Packages, Glossary ● Final Report:
UML Diagrams Worked On	Class Diagrams, State Machine Diagrams, Package Diagram
Classes Worked On	Card, Cards, Player, RiskGame
Notes	Was the person responsible for the reports and did less coding. Wrote most of the main text of the reports after the creation of the source material. Wrote the more text heavy portions of the reports. After the reports were done helped with miscellaneous coding activities and as well implemented the backend of the functionalities having to do with cards.

Name	Mustafa Hakan Kara
Contributions to the Reports	• Design Report: Game Logic Subsystem
UML Diagrams Worked On	Class Diagram
Classes Worked On	AirportDecorator, Continent, Main, Player, RiskGame, Territory
Notes	Was mostly responsible for the backend of the system. Wrote the algorithm which would find the neighbors of the territories which was made tricky due to the addition of the airport functionalities.