29. MÄRZ 2022

FINAL PROJECT

M326

KHALID ALISHA, MOHAMMAD NUWERA

Table of Content

1 Introduction

In the following we will be documenting our process and idea for the final project of the module 326. This document is more about the design and planning instead of our technical implementation.

2 Project Description

Our program has a hierarchical structure. As soon as the User starts the program, he will be choosing his kingdom for which he will be fighting. Either for AK or NM. As soon as the user decides his preferred kingdom different roles will be displayed for the user to choose.

If the user chooses a Democracy, he can choose between following roles:

- King
- Soldier
- Commander
- Citizen
- Minister

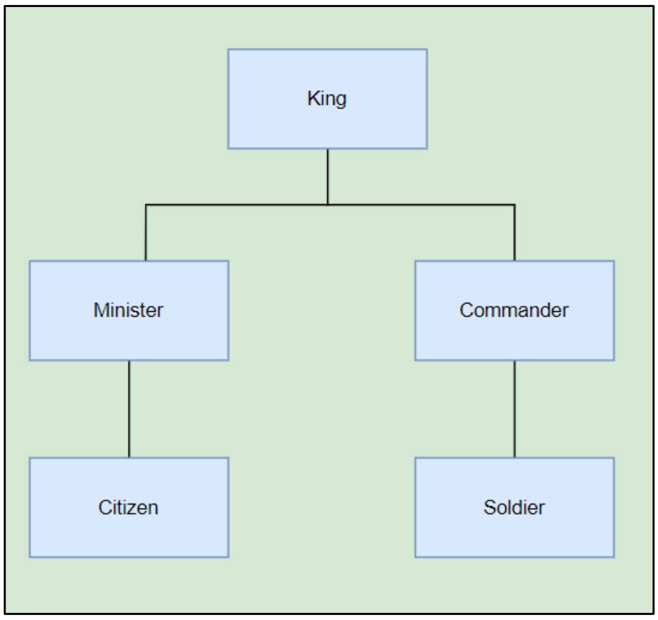


Figure 1 Hierarchy of our player roles

Depending on the role, the user will have different responsibilities, as a king the user can only give commands and is in control. He will pass his orders to the Commander if it is related to the military else to the Minister who's responsible for the population.

The Commander executes the order of the king or passes it down to the soldiers. As the head of the army, he can also make decision on its own and train the soldiers. The Minister is in second position with the Commander after the King.

If the user chooses to be a citizen, he will mainly do as the king resp. the minister says and mostly can't disobey him. Though he can still have his own opinion.

As a soldier there is more fun and action because the user can fight against the enemies and actively choose to:

- Shoot
- Defend

The enemy is from the other kingdom that is played by the system.

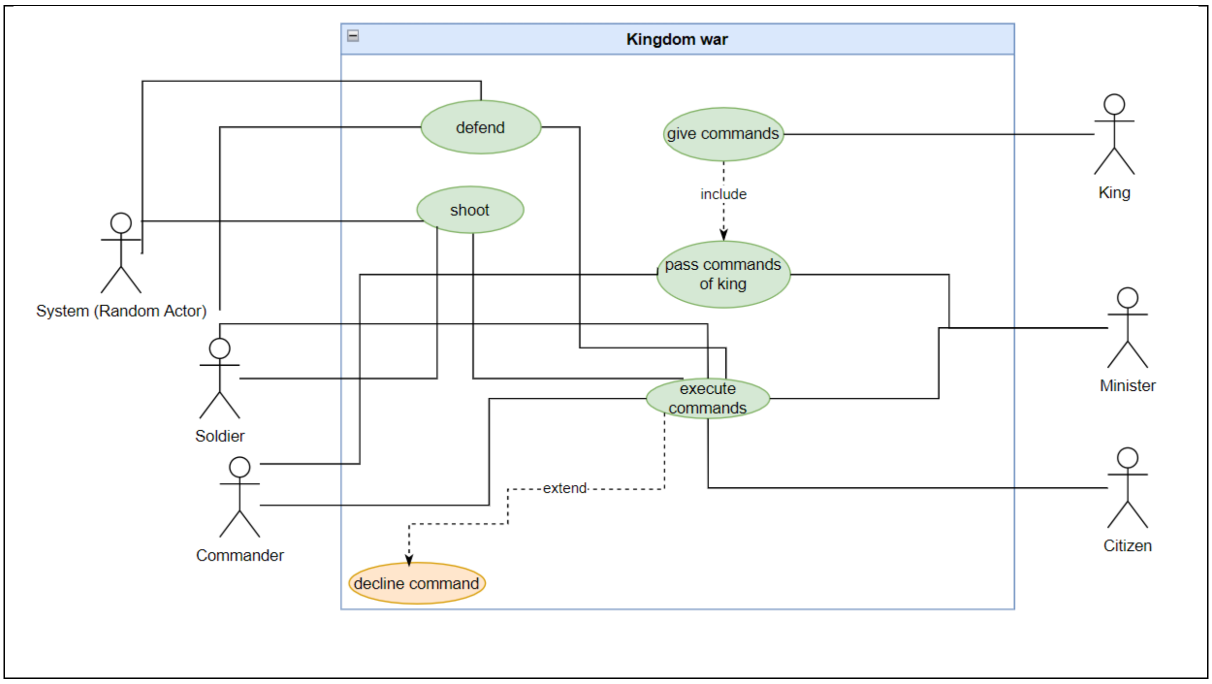
3 Design

Design is a very important step to do before implementing or starting a project. It provides an overview of everything and helps for a better understanding. It is mostly important because the client does not have any technical knowledge. Through the design we visualize his wish.

3.1 Use Case

3.1.1 Use Case Diagram

Use Case Diagram describe the interactions between the system and its actor's meaning users. It shows what the user can do and how to use it, but it doesn't show what happens in the background. It's practical for people with no technical knowledge. Represents how the events flow.



3.1.2 Use Case Description

The Use Case Description describes each interaction the user can do with the system. It explains and shows the exceptions if the user does something inappropriate.

In the following we created Use Case Descriptions for every role of the program…

Use Case #1	User chooses to be a member of the royal family AK
Pre-Condition	The user starts the program and chooses to be a king
Description of Use Case in detail (main scenario)	The user gives the command to mobilize his army
Post-Condition	The commander of the army mobilizes his army, and the get a notification to prepare themselves
Exceptions (what can go wrong, how will the system respond)	The King chooses a command which does not exist in the program → the program lets the king decide his command again

Use Case #2	User chooses to be a soldier from the kingdom NM
Pre-Condition	The user starts the program and picks a task to do
Description of Use Case in detail (main scenario)	After the user picks to fight against a combatant he chooses to shoot him
Post-Condition	The combatant chooses to shield himself from the attack and both soldiers are in a tie
Exceptions (what can go wrong, how will the system respond)	The user picks a move which is not listed so the user gets another chance to pick a valid move

Use Case #3	User chooses to be a general form the kingdom AK
Pre-Condition	The commander is given a menu from which he chooses to create his own operation
Description of Use Case in detail (main scenario)	The commander decides to do a parkour in the field
Post-Condition	Command is passed on. The user can change into the soldier role to check whether the operations are being executed or not
Exceptions (what can go wrong, how will the system respond)	The general chooses an activity which does not exist so that’s why he must choose a valid activity

Use Case #4	The user chooses to be a citizen of kingdom NM
Pre-Condition	The user is shown a menu from which he picks to be a citizen
Description of Use Case in detail (main scenario)	As a citizen the user gets the command to protest against the war
Post-Condition	Now the user is given the chance to decline the command or execute it, meaning protesting
Exceptions (what can go wrong, how will the system respond)	If the user chose a task which does not exist, then the user must pick another task

Use Case #5	the user chooses to be the minister of the kingdom AK
Pre-Condition	The user is shown a menu from which he picks to be the minister
Description of Use Case in detail (main scenario)	As the minister the user chooses to advice the royal member and make him spend less money
Post-Condition	The royal member listens to his advice and the user verifies if the royal member has gotten more money
Exceptions (what can go wrong, how will the system respond)	The user chooses a command which does not exist which is why he must pick another command

3.1.2.1 Specific Use Case Diagram

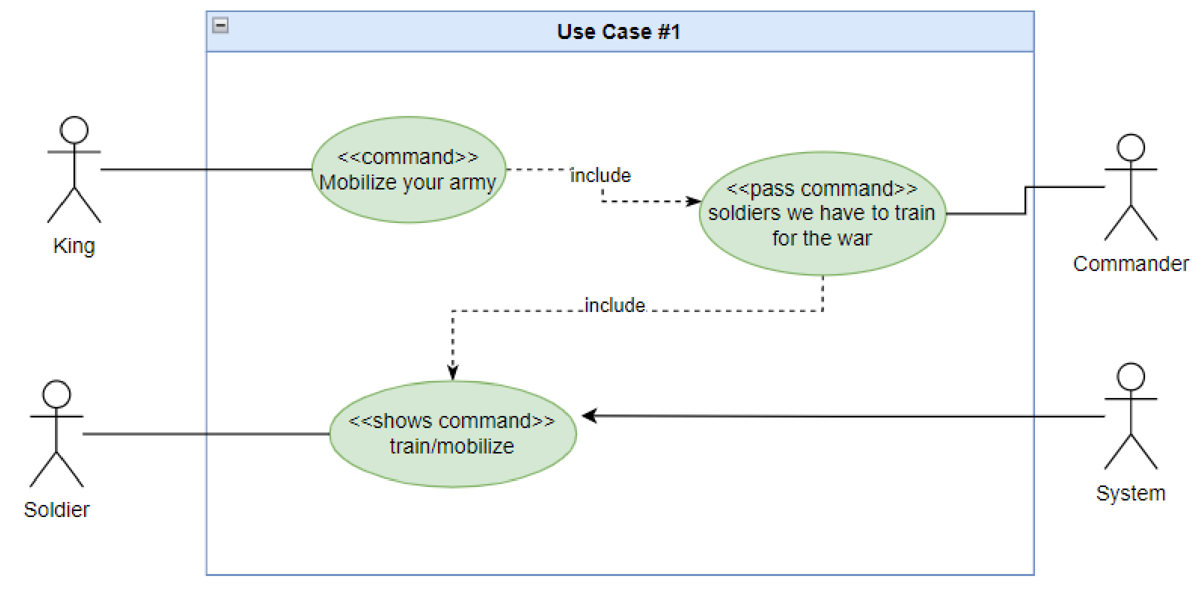

Use Case #1
King
<<command>> Mobilize your army
include
<<pass command>> soldiers we have to train for the war
Commander
include
Soldier
<<shows command>> train/mobilize
System

Figure 2 Have a look at the Use Case 1

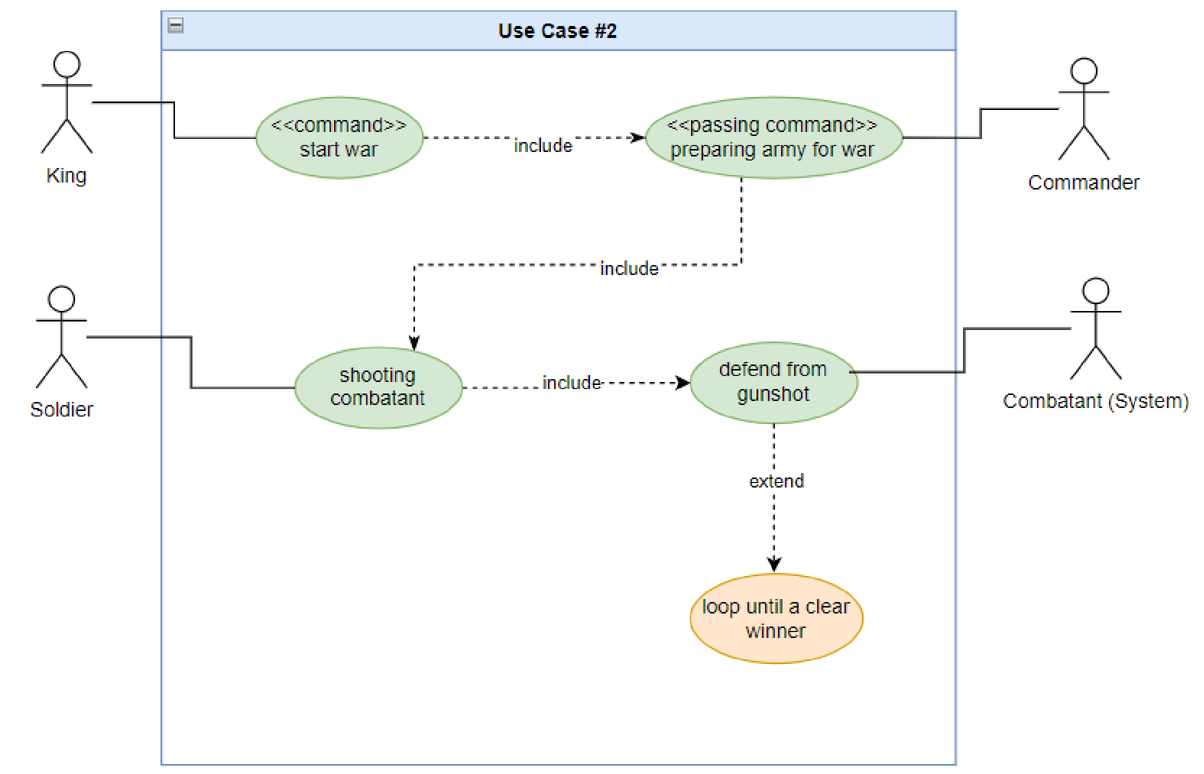


Figure 3 Have a look at the Use Case 2

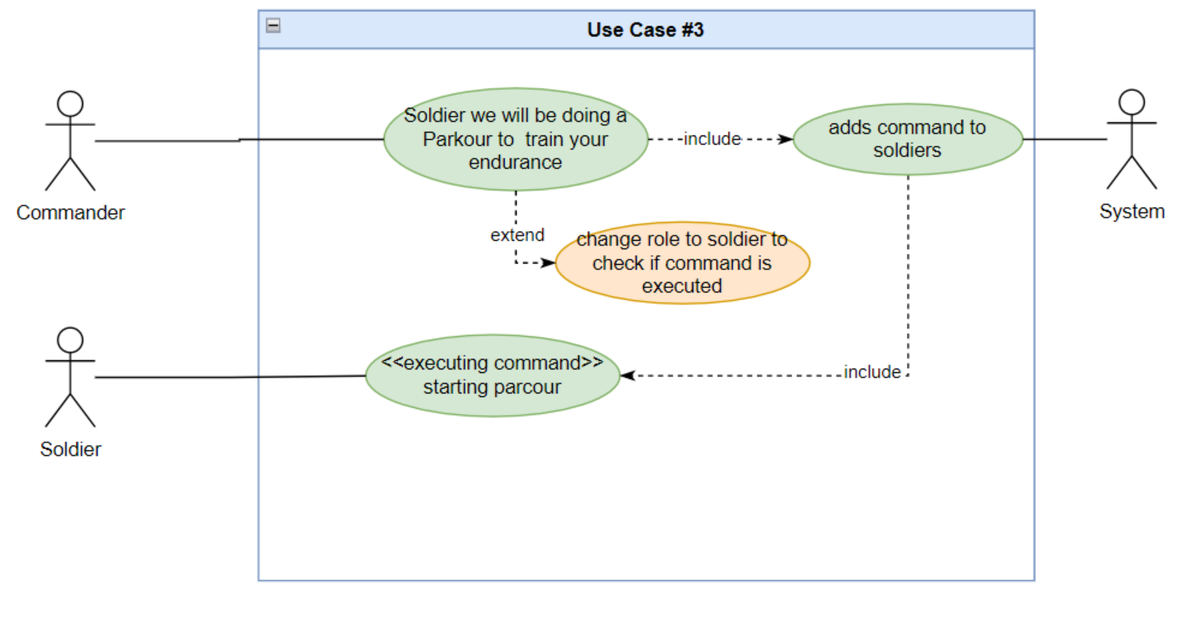

Use Case #3
Commander
Soldier
System
Soldier we will be doing a Parkour to train your endurance
include
adds command to soldiers
extend
change role to soldier to check if command is executed
<<executing command>> starting parcour
include

Figure 4 Have a look at the Use Case 3

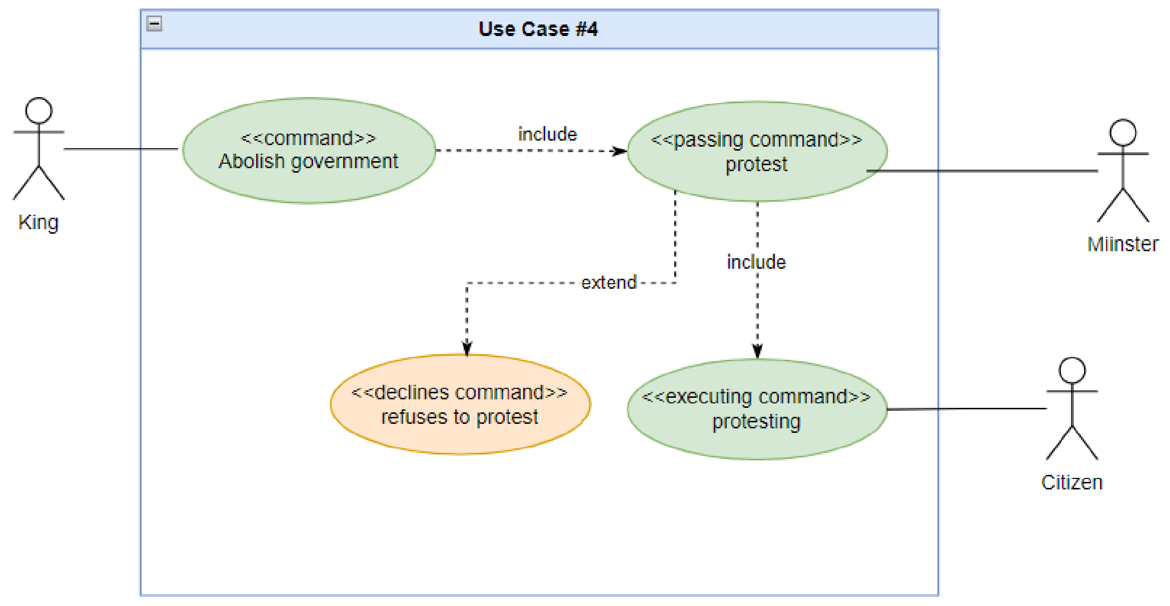

Use Case #4
King
<<command>> Abolish government
include
<<passing command>> protest
Miinster
extend
include
<<declines command>> refuses to protest
<<executing command>> protesting
Citizen

Figure 5 Have a look at the Use Case 4

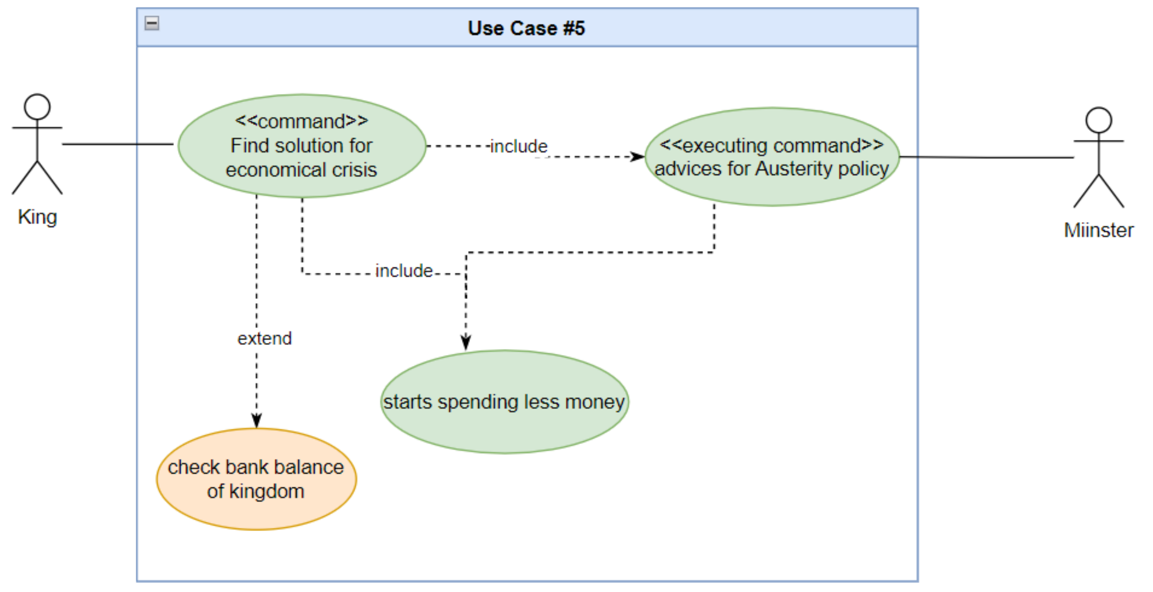

Use Case #5
King
<<command>> Find solution for economical crisis
include
<<executing command>> advices for Austerity policy
Miinster
include
extend
starts spending less money
check bank balance of kingdom

Figure 6 Have a look at the Use Case 5

4 Planning

As soon as the design is complete, we move on to the planning process. For the planning we create CRC-card (Class Responsibility Collaboration Card) and a class diagram derived from the domain model and the CRD-Cards.

4.1 CRC-card

With the help of CRC-Cards we could share our thoughts and combine them to one. It was now easier to think about the responsibilities of each class.

King	RequestHandler
• Give highest commands • Take advice • Check bank balance of kingdom	• Instruction • RequestHandler

Commander	RequestHandler
• passes commands of king to his soldiers • as the head of the army, he can make decisions on his own	• Instruction • Notification • RequestHandler

Minister	RequestHandler
• give Instructions • passes commands of king to population	• Instructions • RequestHandler • Notification

Soldier	RequestHandler
• fight (shoot, defend) • execute commands of commander / general • collect points for kingdom by winning a fight	• Instruction • Notification • RandomMove • RequestHandler

Notification	
• notify user depending on which role he has	• Queue • Commander • Soldier • Citizen • Minister

Instruction	
• all commands for the user	• Minister • Soldier • Commander • King • Citizen • Queue

Validation	
• Validate string input • Validate number input • Validate use case input	• IO-Handler

Organizer	
• combines all the functionality	• IO-Handler • Starter

IO-Handler	
• menus • takes input from user • prints all informations	• Organizer • Validation • Instruction

Starter	
• run method	• Organizer

Queue	
• Solving threads • Waits until prior command has been executed	• RequestHandler • IO-Handler • Notification • Instruction

RandomMove	
• prints out a random generated move as the combatant of the soldier	• Soldier • Instruction

RequestHandler	
• takes care that the commands are passed (design pattern: chain of responsibility)	• Queue • King • Commander • Minister • Soldier • Citizen • Instruction

4.2 Domain Model

In the following we created a domain model to visualize our thoughts and decided entities as well as how they are associated with each other.

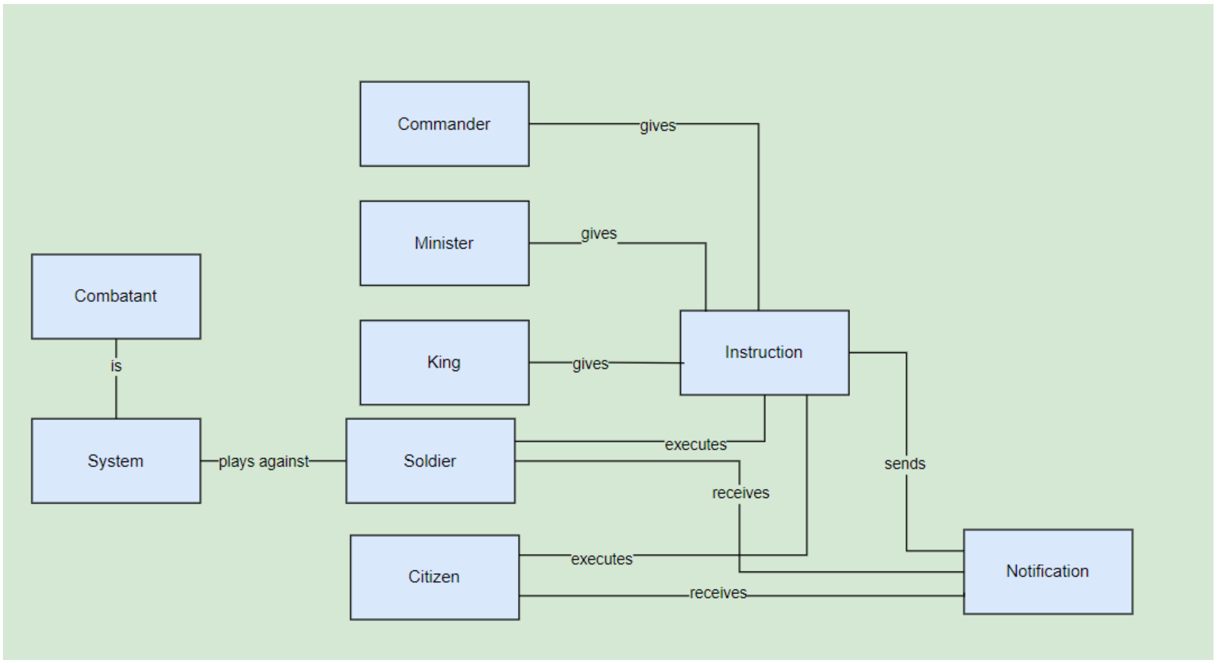


Figure 7 Look at Project Description for details

4.3 Class Diagram

The class diagram is the basic structure for our project which we have derived from the domain model and the CRC cards. Because of this we now have collected all the possible attributes and methods, so that the practical implementation is structured.

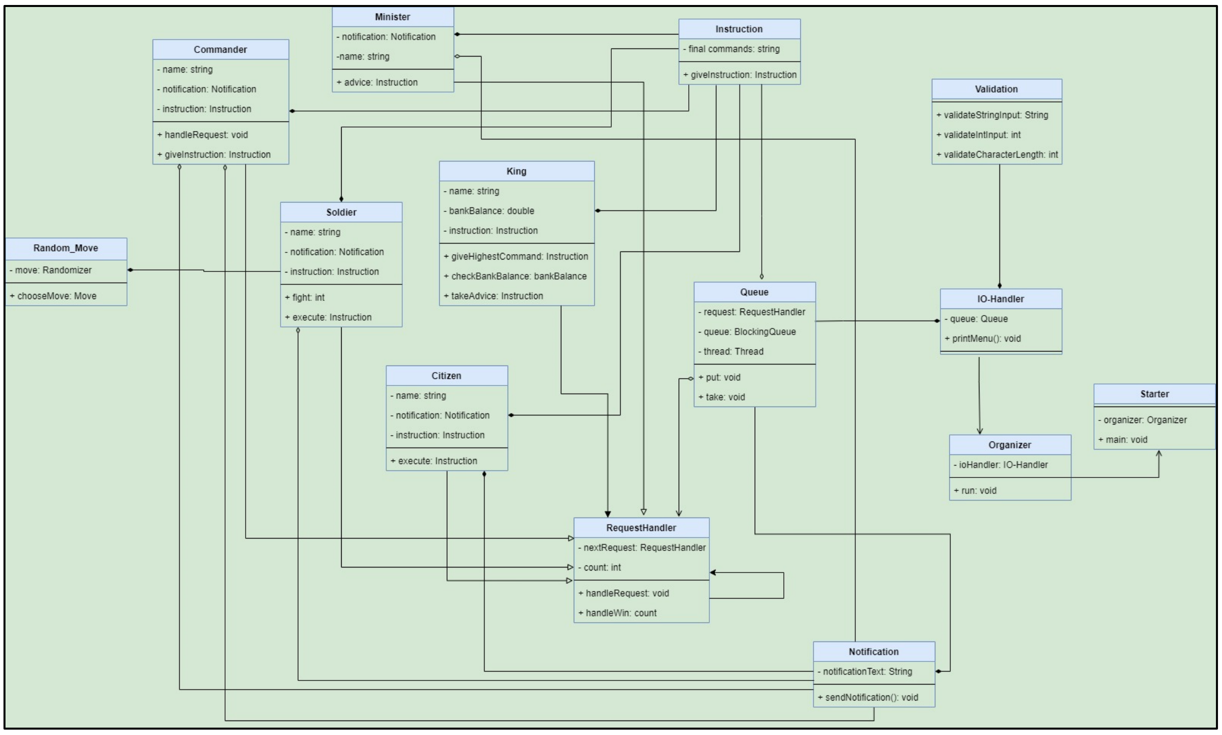


Figure 8 Class Diagram

Table of Figures