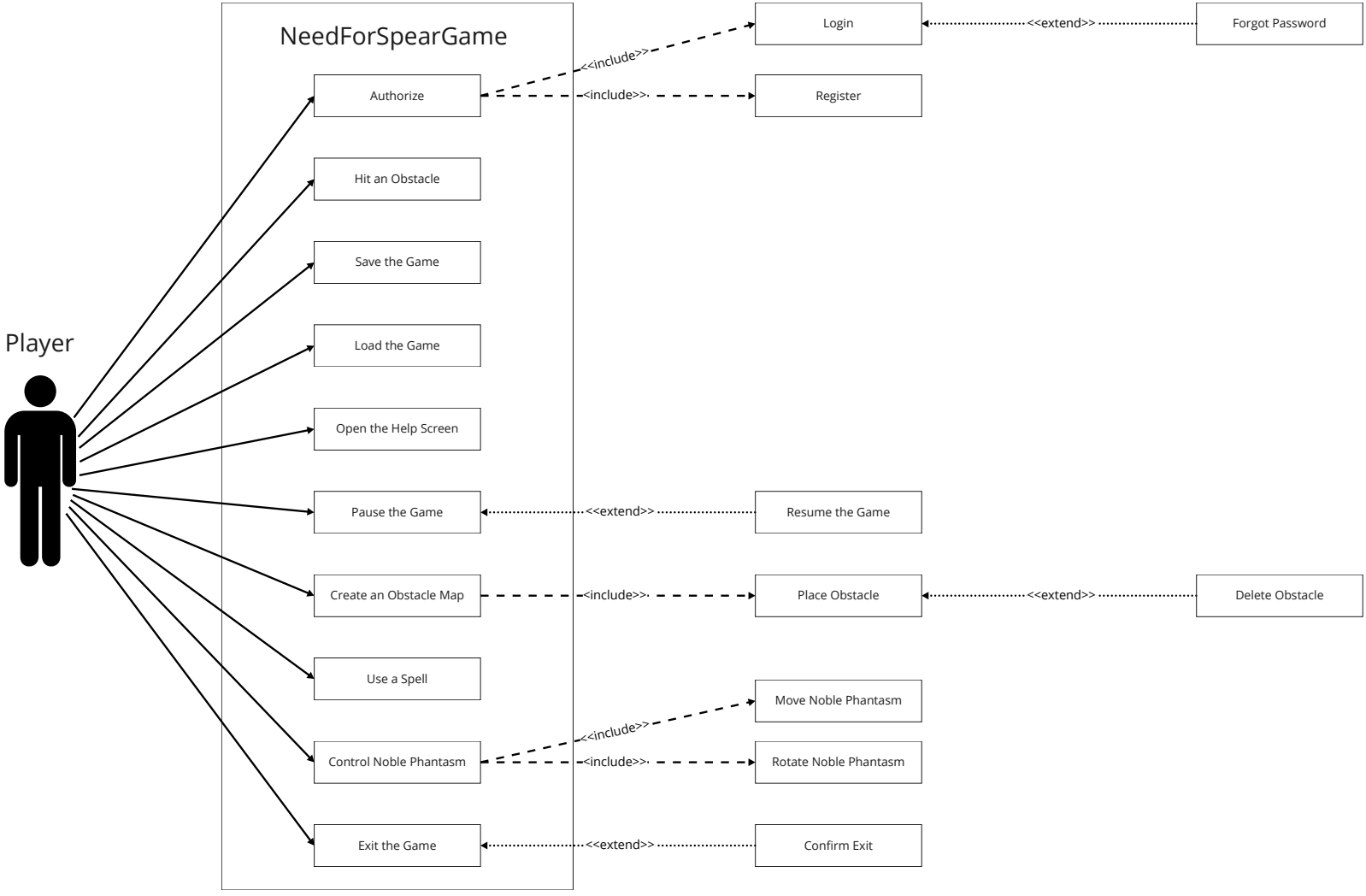
Use Case Diagram



COMP302 (Software Engineering) – R1M1 of the Project by /dev/null.

Use Cases

1. Authorize.
2. Hit an obstacle.
3. Save the game.
4. Load the game.
5. Open the help screen.
6. Pause the game.
7. Create an obstacle Map.
8. Use a spell.
9. Control the noble phantasm.
10. Exit the game.

Use Case Narratives

Use Case UC1: Authorize.
Scope: Login Screen.
Level: Subfunction.
Primary Actor: Player.
Stakeholders and Interests:
1. Player: Wants to create a mechanism to retrieve all personalized data about them. (Progress in the game, highscore etc.)

Preconditions:
1. Player must have an internet connection.
2. Player must be in the login screen.

Success Guarantee:
1. The player successfully logs in or, alternatively, signs up the game.

Main Success Scenario:
1. Player enters the username to the required area.
2. Player enters the password to the required area.
3. Player clicks to the “Login” button.
4. System checks if the username and password match.
5. System checks if the email is verified.
6. System directs the player to the Main Menu.

Extensions:

*****At any time**,** the player wants to sign up for the first time.
1. Player enters their email address.
2. Player enters a username.
3. Player enters a password.
4. System saves this information to the database and sends a verification email to the player.

5. Player verifies their email to complete its sign up process.

2a. If the player forgot their password, or wants to change it.
1. Player clicks the “Forgot my password” button.
2. System sends an email to the player.
3. Player selects a new password with the help of instructions sent by email.
4. System updates the password at the database.

4a. If the provided username is not found.
1. System informs the player that the provided username has not been authorized.

4b. Player enters a wrong password.
1. System notifies the player that the password is wrong and asks to try again.

5a. Player’s mail is not verified.
1. System notifies the player that their mail is not verified.

Special Requirements:
1. Rectangular text fields should be large enough to take inputs.
2. Rectangular text fields should be visible through various backgrounds.

Technology and Data Variations List:
1. Username, email should be text field while password area should be secured field.
2. The input should be entered via keyboard.

Frequency of Occurrence: Whenever the player wants to login to the game.
Open Issues: NO OPEN ISSUES.

Use Case UC2: Hit an Obstacle.
Scope: Running Mode.
Level: User Goal.
Primary Actor: Player.
Stakeholders and Interests:
1. Player: Wants to direct the moving sphere with the noble phantasm to hit the obstacles.

Preconditions:
1. Player has built the level or loaded a built one.
2. Player is in the running mode.
3. Player has one or more lives.
4. Player is logged in using their credentials.
5. Player must have an internet connection.

Success Guarantee:
1. The player successfully hits the obstacle with the sphere.

Main Success Scenario:
1. Player moves the noble phantasm according to the sphere’s movement.
2. Player redirects the sphere by using the noble phantasm.
3. Sphere hits an obstacle.

Extensions:

1a. Player fails to hit the sphere with noble phantasm.
1. Player’s chances are reduced by 1.
1a. Player loses the game if the number of chances reduces to 0.

3a. Sphere misses obstacles.
1. Sphere bounces back from the edges and comes back to the level of the noble phantasm.

3b. Player selects to use “Magical Hex” ability.
1. Player hits the obstacles using the ability instead of the sphere.

3c. Sphere hits “Firm Obstacle”.
1. The obstacle’s health number decreases by one.
1a. The obstacle is destroyed if its health is reduced to 0.

3d. Sphere hits “Explosive Obstacle”.

1. The obstacle’s remains drop downwards.
2. If the remains hit the noble phantasm, the player’s lives decrease by one.

3e. Sphere hits “Gift Obstacle”.

1. The obstacle drops a gift box downwards.
2. If the gift box touches the noble phantasm, the player gets a special ability.
   - 2a. If the special ability is the “Chance Giving Ability”, the player's life increases by one.
   - 2b. If the special ability is the “Noble Phantasm Expansion”, the noble phantasm’s size increases.
   - 2c. If the special ability is the “Magical Hex”, the player can shoot using the cannons on both sides of the phantasm.
   - 2d. If the special ability is the “Unstoppable Enchanted Sphere”, the sphere can penetrate the obstacles.
3. If the player already has unused abilities, the newly acquired ability is stored alongside the unused ones.

Special Requirements:

1. All the obstacles’ colors should be chosen such that they are visible between various background colors.
2. Obstacle’s health should be displayed on the obstacle either using a number or texture.
3. Gifted, normal and explosive obstacles should be distinguishable from each other.
4. Each special ability should appear at least once after every gift obstacle is destroyed.

Technology and Data Variations List:

1. Player should interact with the game via keyboard.

Frequency of Occurrence: Every time the player hits an obstacle.
Open Issues: NO OPEN ISSUES.

Use Case UC3: Save the Game.
Scope: Pause Menu.
Level: Subfunction.
Primary Actor: Player.
Stakeholders and Interests:

1. Player: Wants to save their progress and resume playing using that saved data.

Preconditions:

1. Player must logged in using their credentials.
2. Player must have an internet connection.
3. Player must be in the Pause Menu.

Success Guarantee:

1. Player save their progress or map without any exceptions.

Main Success Scenario:

1. Player clicks the “Save Game” button in the Pause Menu.
2. Player’s current game state is saved in the database.

Extensions:

2a. Player already has a save entry in the database.

1. The current game state overrides the previous saved game entry.

Special Requirements:

1. Rectangular button should be visible through various backgrounds.

Technology and Data Variations List:

1. Player should save the game using keyboard and mouse.

Frequency of Occurrence: Whenever the player wants to save their game.
Open Issues: NO OPEN ISSUES.

Use Case UC4: Load the Game.
Scope: Main Menu/Pause Menu.
Level: Subfunction.
Primary Actor: Player.
Stakeholders and Interests:
1. Player: Wants to load a previously saved game without any exceptions.

Preconditions:
1. Player is logged in using their credentials.
2. Player must have an internet connection.
3. Player is in the Main Menu or Pause Menu.
4. Player has a previously saved game file.

Success Guarantee:
1. The player successfully loads the previously designed map and their progress.

Main Success Scenario:
1. Player clicks the “Load Game” button in the main menu or pause menu.
2. System retrieves the saved game files from the database.
3. System shows the saved game entries (if any).
4. Player chooses the desired saved game entry from the database.
5. System loads the selected game data from the database and starts the game with this data.

Extensions:

2a. Player has no previously saved game file.
1. Systems warns the user that there is no existing saved data.

2b. The desired game data is corrupted.
1. The player is alerted that the data is corrupted and cannot be loaded.
2. System deletes the corrupted data.
3. Player is sent back to the screen where they choose a save entry.

Special Requirements:
1. Rectangular button should be visible through various backgrounds.

Technology and Data Variations List:
1. Player should load the game using keyboard and mouse.

Frequency of Occurrence: Whenever the player wants to load their game.
Open Issues: NO OPEN ISSUES.

Use Case UC5: Open the Help Screen.
Scope: Main menu / Pause Menu.
Level: Subfunction.
Primary Actor: Player.
Stakeholders and Interests:
1. Player: Wants to get information about the game logic, movement and game progress interaction.

Preconditions:
1. Player is logged in using their credentials.
2. Player is in the pause or main menu.
3. Player must have an internet connection.

Success Guarantee:
1. Help screen pops up. The player got information about the rules of the game and informed about how to navigate through the interface.

Main Success Scenario:
1. Player clicks the “Help” button.

2. Help screen pops up.
3. Player exits the help screen.

Extensions:

NO EXTENSION.

Special Requirements:

1. Text punto is appropriate for reading.
2. The text should be explanatory and clear.

Technology and Data Variations List: NO TECH or DATA VARIATION.
Frequency of Occurrence: Whenever the player wants to get help.
Open Issues: NO OPEN ISSUES.

Use Case UC6: Pause the Game.
Scope: Running mode/Building mode.
Level: Subfunction.
Primary Actor: Player.
Stakeholders and Interests:

1. Player: Wants to pause the ongoing game without any exceptions.

Preconditions:

1. Player is logged in using their credentials.
2. Player must have an internet connection.
3. Player is in building mode or running mode.

Success Guarantee:

1. All of the animated obstacles stop. Player's ability to control noble phantasm is no longer possible.

Main Success Scenario:

1. Player clicks the "ESC" button on their keyboard while ingame.
2. The game is paused and the player is directed to the "Pause Menu".

Extensions:

NO EXTENSION.

Special Requirements:

1. The player should understand that they can pause with "ESC".

Technology and Data Variations List:

1. Player should pause the game using the keyboard.

Frequency of Occurrence: Whenever the player wants to pause their game.
Open Issues: NO OPEN ISSUES.

Use Case UC7: Create an obstacle Map.
Scope: Building Mode.
Level: User Goal.
Primary Actor: Player.
Stakeholders and Interests:

1. Player: Wants to design a map which they want to save to play later or to play immediately.

Preconditions:

1. Player is logged in using their credentials.
2. Player must have an internet connection.
3. Player is in building mode.

Success Guarantee:

1. Creates a new map in which the number, type and health of the bricks are set.

Main Success Scenario:

1. Player clicks the “Play” button in the main menu.
2. An empty map, which contains different types of bricks for the player to choose, will be displayed.
3. Player designs the map by placing the bricks.
4. Player saves or plays the map.

Extensions:

3a. Player wants to remove a selected brick.
1. Player clicks the brick and presses “del” on the keyboard.
2. System removes the brick from the screen.

4a. Player wants to discard the map.
1. Map is deleted and the player is directed to the main menu.

Special Requirements:

1. All the bricks in the level should be visible.
2. Bricks’ health should be set by the player if they chose to add firm bricks.
3. Gifted, normal and explosive bricks should be distinguishable from each other.
4. Editing area for the player should be distinguishable.
5. Numbers of bricks that are available should be displayed.

Technology and Data Variations List: NO TECH or DATA VARIATION.
Frequency of Occurrence: Whenever the player wants to design a map.
Open Issues: NO OPEN ISSUES.

Use Case UC8: Use a Spell.
Scope: Running Mode.
Level: User Goal.
Primary Actor: Player.
Stakeholders and Interests:

Player: Uses a spell of their choosing.

Preconditions:

1. Player is logged in using their credentials.
2. Player must have an internet connection.
3. Player is in running mode.
4. Player has a spell.

Success Guarantee:

1. Player activates the spell.

Main Success Scenario:

1. Player clicks the icon or types the first letter of the spell.
2. Spell is activated.

Extensions:

1a. “Chance Giving Ability” is used immediately upon pickup.
1. If player has 3 chances, the player will not receive an extra chance.

Special Requirements:

1. Spells’ icons should be visible and understandable.
2. Player should be able to understand whether they have any spells or not.
3. Player should be able to understand if there are any spells active.
4. Visualization of spells’ effects should be distinguishable.

Technology and Data Variations List: NO TECH or DATA VARIATION.
Frequency of Occurrence: Whenever the player wants to use a spell.
Open Issues: NO OPEN ISSUES.

Use Case UC9: Control the Noble Phantasm.
Scope: Running Mode.
Level: User Goal.
Primary Actor: Player.
Stakeholders and Interests:
1. Player: Wants to change the Noble Phantasm's location and degree.

Preconditions:
1. Player is logged in using their credentials.
2. Player must have an internet connection.
3. Player is in running mode.

Success Guarantee:
1. Player moves or rotates the noble phantasm.

Main Success Scenario:
1. Player presses arrow, A or D keys to move the phantasm horizontally or rotate it.
2. Noble Phantasm's location or degree changes.

Extensions:

1a. Player tries to control Noble Phantasm in pause mode.
    1. Phantasm's state is not changed.

1b. Player tries to move the phantasm out of the map.
    1. Phantasm stays at the edge of the map.

Special Requirements:
1. Noble Phantasm should be visible through various backgrounds.

Technology and Data Variations List: NO TECH or DATA VARIATION.
Frequency of Occurrence: Whenever the player wants to move the phantasm.
Open Issues: NO OPEN ISSUES.

Use Case UC10: Exit The Game.
Scope: Login Screen / Main Menu / Pause Menu.
Level: Subfunction.
Primary Actor: Player.
Stakeholders and Interests:
1. Player: Wants to close the game without any exceptions.

Preconditions:
1. Player is in the login screen, pause menu or main menu.

Success Guarantee:
1. Game shuts down.

Main Success Scenario:
1. Player clicks the "Exit" button.
2. A screen pops up asking "Are you sure?".
3. Player clicks "Yes".
4. App closes.

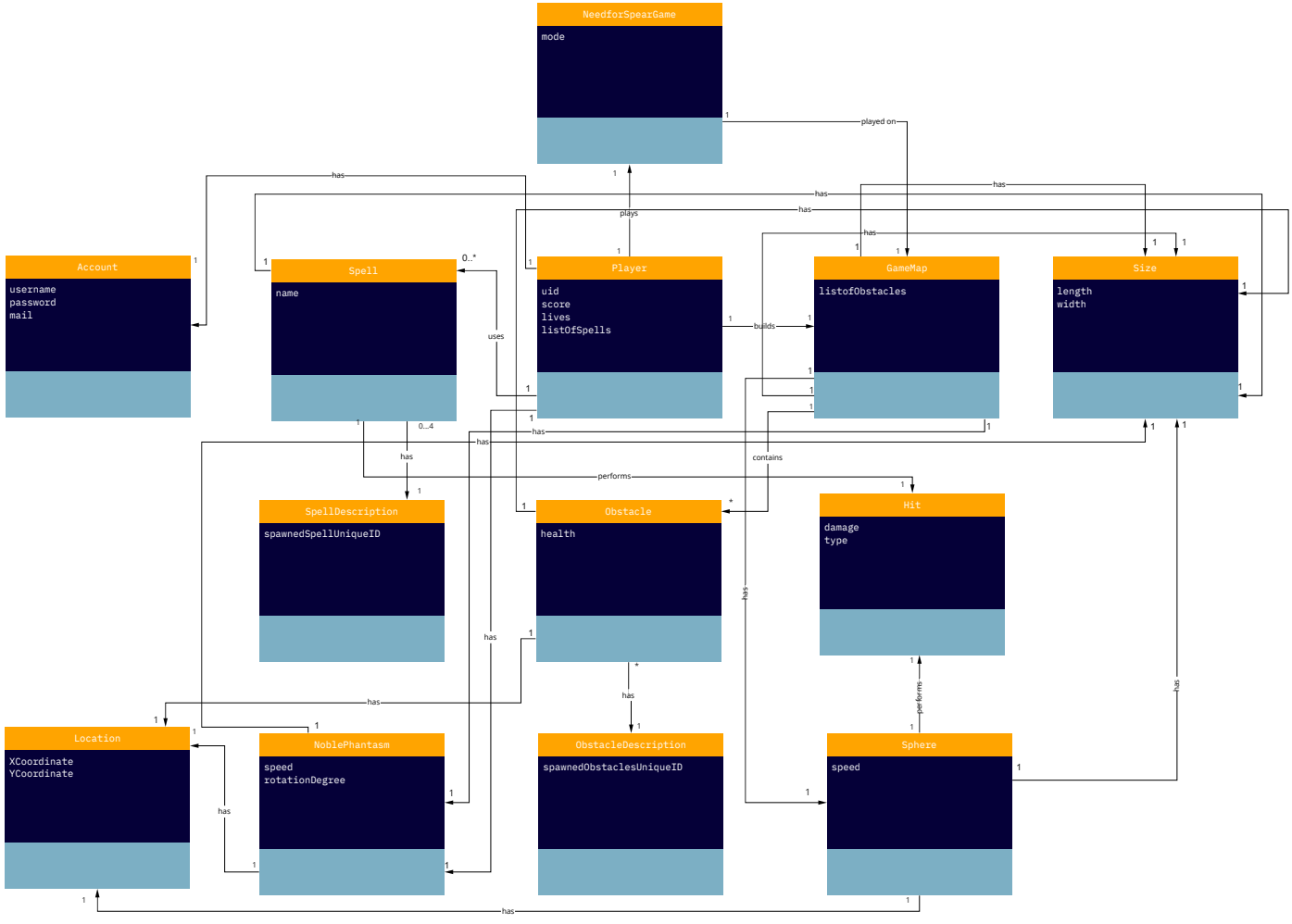
Extensions:

2a. Player clicks "No"
    1. Player returns to the previous screen.

Special Requirements:
1. Rectangular buttons should be visible through various backgrounds.

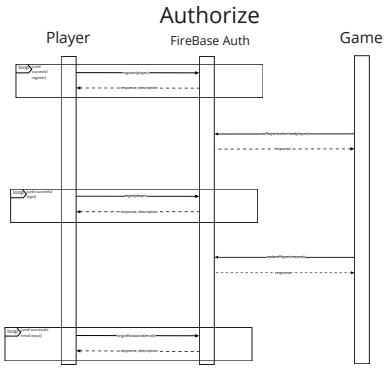
Technology and Data Variations List: NO TECH or DATA VARIATION.
Frequency of Occurrence: Whenever the player wants to exit from the game.
Open Issues: NO OPEN ISSUES.

Domain Models

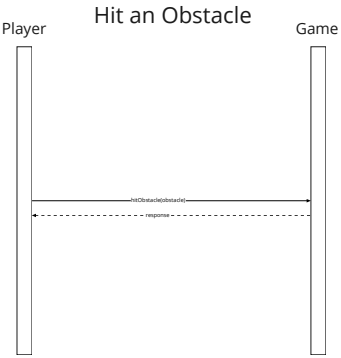


System Sequence Diagrams

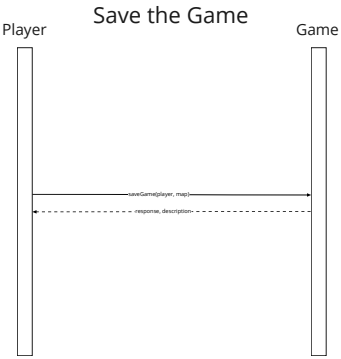
Authorize



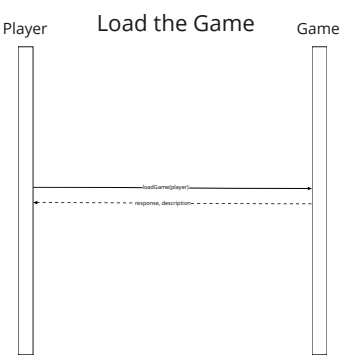
Hit an Obstacle



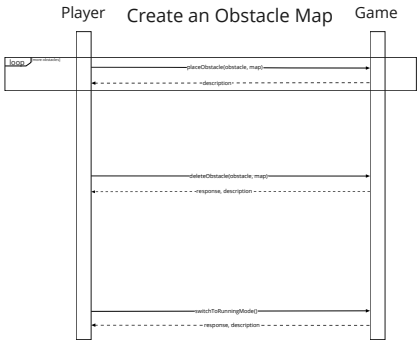
Save the Game



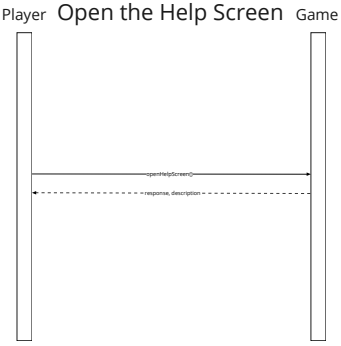
Load the Game



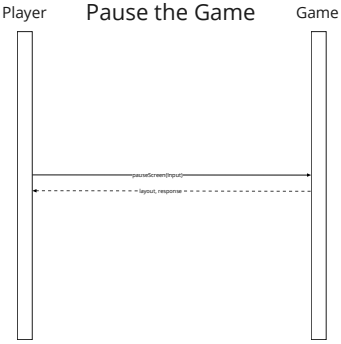
Create an Obstacle Map



Open the Help Screen

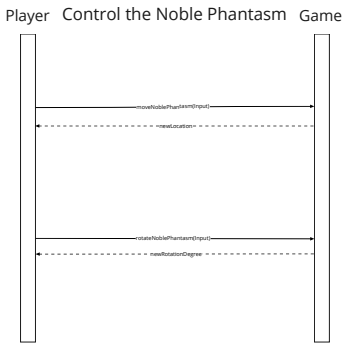


Pause the Game

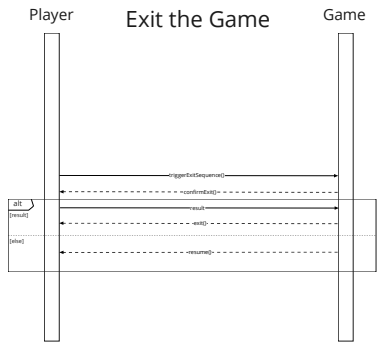


Use a Spell

Control the Noble Phantasm



Exit the Game



Operation Contracts

Operation: register(*player* : Player*)*
Cross References: Authorize.
Preconditions:
1. Player has a valid email account.

Postconditions:
1. A Player instance *player* is created. *(instance creation)*
2. Email, username and password are associated with the *player* object. *(association formed)*

Operation: login(*player* : Player)
Cross References: Authorize.
Preconditions:
1. Username and password pair exists on the game database.

Postconditions:
1. *player* was associated with NeedForSpearGame. *(association formed)*
2. *player* was associated with GameMap. *(association formed)*
3. *player* was associated with Spell. *(association formed)*
4. *player* was associated with NoblePhantasm. *(association formed)*

Operation: forgotPassword(*email* : String)
Cross References: Authorize.
Preconditions:
1. *email* has an associated username and password *player* object on the game database.

Postconditions:
1. *player*.password is updated with a new password according to user input. *(attribute modification)*

Operation: hitObstacle(*obstacle* : Obstacle)
Cross References: Hit an Obstacle.
Preconditions:
1. Game is in the running mode.
2. *player*.lives is 1 or more.

Postconditions:
1. Based on *hit*.type, *obstacle*.health was reduced by *hit*.damage. *(attribute modification)*
2. *player.*score is updated accordingly if *obstacle*.health is 0 or below. *(attribute modification)*

Operation: loadGame(*player* : Player)
Cross References: Load the Game.
Preconditions:
1. Game is in the Pause Menu or Main Menu.
2. There is a previously saved game file in the database.

Postconditions:
1. An instance of GameMap *map* was created. *(instance creation)*
2. *map.listOfObstacles is set to the data from the database. (attribute modification)*
3. *player.*score is set to score from the database. *(attribute modification)*
4. *player.*lives is set to lives from the database. *(attribute modification)*

Operation: placeObstacle(*obstacle* : Obstacle, *map* : GameMap)
Cross References: Create an Obstacle Map.
Preconditions:
1. Game is in the build mode.
2. A GameMap instance was created previously.

Postconditions:
1. *obstacle* was associated with GameMap. *(association formed)*
2. *obstacle* is added to *map.listOfObstacles. (attribute modification)*

Operation: deleteObstacle(*obstacle* : Obstacle, *map* : GameMap)
Cross References: Create an Obstacle Map.
Preconditions:
1. Game is in the build mode.

Postconditions:
1. *obstacle*'s association with the GameMap was broken. *(association broken)*
2. *obstacle is removed from map.*listOfObtacles. *(attribute modification)*

Operation: switchToRunningMode()
Cross References: Create an Obstacle Map.
Preconditions:
1. Game is in the build mode.
2. Obstacle conditions are met.

Postconditions:
1. Instances of NoblePhantasm and Sphere objects were created. *(instance creation)*
2. *NeedforSpearGame*.mode attribute was changed. *(attribute modification)*

Operation: activateSpell(*spell* : Spell, *player* : Player)
Cross References: Use a Spell.
Preconditions:
1. Game is in the running mode.
2. Player should have at least one instance of spell.

Postconditions:
1. *spell* is associated with a Hit object. *(association formed)*
2. *hit*.damage was set according to *spell*.type.*(attribute modification)*
3. *player.*listOfSpells is adjusted accordingly. *(attribute modification)*

Operation: moveNoblePhantasm(input : KeyboardEvent)
Cross References: Control the Noble Phantasm.
Preconditions:
1. An instance of a NoblePhantasm *noblePhantasm* has been created
2. System is in the running mode.

Postconditions:
1. *noblePhantasm*.speed is set according to the input. *(attribute modification)*
2. *noblePhantasm*.location is set according to the input. *(attribute modification)*

Operation: rotateNoblePhantasm(input : KeyboardEvent)

Cross References: Control the Noble Phantasm.
Preconditions:

1. An instance of a NoblePhantasm *noblePhantasm* has been created.
2. System is in the running mode.

Postconditions:

1. *NoblePhantasm*.rotationDegree is set to input. *(attribute modification)*

Vision

Revision History

Version	Date	Description	Author
Draft	Oct 25, 2021	First draft, to be revisioned	Melis Oktayoğlu, Gökçe Sevimli, Halil Doruk Yıldırım
Revision	Oct 26, 2021	Revision is done.	Kaan Türkmen, Can Usluel

1. Introduction

We are aiming to develop a fun game named “NeedForSpear”. Hence, we desired to make the game supportable for multiplayer and allow users to create their own accounts where they can store their earlier games, as well as the game maps they had designed for later use. The game will consist of two players racing for the Spear of Power. Each player has a noble phantasm and an enchanted sphere which they will use to prevent the sphere from falling to the ground. The falling of the sphere results in losing one of the three lives of the players. The sphere is an object that they will use to destroy the obstacles, thereby earning scores. At the same time players can earn magical abilities to enhance their powers and the player who destroys all the obstacles will be the winner. The game is also compatible with one-player game mode.

1.1 Purpose

The purpose of this vision document is to provide all the necessary information and curate the high-level needs, requirements and features of our game NeedForSpear.

1.2 Scope

This vision is for the NeedForSpear game developed by the /dev/null team. The game is visioned to be played on desktops and this will provide people of all ages to spend fun time and bond with each other over some friendly competition.

1.3 References

Templates inspired from: https://www.ibm.com/docs/en/elm/7.0.0?topic=requirements-vision-document
https://personal.utdallas.edu/~chung/RE/Presentations10F/Team-hope/1%20-%20VisionDoc.pdf

[1] H. F. Büyük, “Turkey’s Digital Gaming Industry Soars on Cheap Lira, Pandemic,” *https://balkaninsight.com/2021/09/27/turkeys-digital-gaming-industry-soars-on-cheap-lira-pandemic/*, 27-Sep-2021.

2. Positioning

2.1 Business Opportunity

The current game industry is highly capable of developing high quality, high resolution games. However, they often cost a lot of money which limits distributing the game to a wide range of demographics. For this reason we aim to develop an affordable desktop game that is still enjoyable and allows multiplayer.

2.2 Problem Statement

Due to developments and innovations in the game industry, many game developers are publishing various games to offer different alternatives to potential players. At this stage, /dev/null team is developing a game called “Need For Spear” to provide gamers a fascinating game experience.

2.3 Product Position Statement

For	People of all ages.
who	Want to have fun and spend time.
NeedForSpear	Is a desktop game application.
that	Provides a platform for spending fun and competitive time.
unlike	The current brick-breaker-like games which don't offer various types of obstacles and magical spells.
our product	offers a fascinating game experience by adding different features to the classic brick-breaker game.

3. Stakeholder and User Descriptions

3.1 Market Demographics

Last year, according to the annual Turkey Game Market Report, digital gaming industry’s revenues in Turkey reached $880 million, as the number of gamers grew from 32 million in 2019 to 36 million in 2020. Experts believe the sector will hit the $1 billion mark by the end of this year [1].

The target segment includes people in all age groups, who want to spend time by playing and have fun with others.

3.2 Stakeholder Summary

Name	Represents	Role
Computer Engineers	This stakeholder is a primary lead in the development of Need for Spear.	Responsible for the architecture, implementation, design and overall development of Need for Spear.
Team Lead	This stakeholder leads development of Need for Spear.	Plans meetings between team members, sets priorities with them and keeps the project team focused.

3.3 User Summary

Name	Description	Responsibilities	Stakeholder
Players	Primary End user of the game.	Uses the application to play with others or play by themself.	Self.

3.4 User Environment

1. Need for Spear will be played by anyone, regardless of age, that wants to play the game for any reason they desire.
2. The game can be played either by single player or multiplayer depending on the player's preference.
3. The players will create user credentials, which will be saved in the database, to log in to their account.
4. After login to the game, the players will interact with an understandable interface where they can:
   a) Enter build mode.
   b) Load a saved game.
   c) Enter the help screen.

3.5 Stakeholder Profiles

Computer Engineers

Description	This stakeholder is responsible for the development of Need for Spear.
Type	Software guru experience in programming and teamwork.
Responsibilities	Uses agile methodologies to design and develop Need for Spear.
Success Criteria	Success is defined as going through the project iterations to create Need for Spear in an agile matter.
Involvement	This stakeholder is a lead figure in the creation of Need for Spear. Designs and programs the game.
Deliverables	Iterations of design, code and other necessary material to COMP302 staff.
Comments/Issues	None.

3.6 User Profiles

The Players

Description	The users that play Need for Spear.
Type	Casual user who may have never played any video game before.
Responsibilities	Plays Need for Spear to pass time, have fun or for any reason they desire.
Success Criteria	Success is defined as the player enjoying the game and continuing to play it.
Involvement	None.
Deliverables	None.
Comments/Issues	None.

3.7 Key Stakeholders or User Needs

Need	Priority	Concerns	Current Solution	Proposed Solutions
Flexible(configurable)	High	Ability to customize obstacles based on different user desires.	See proposed.	Allow users to determine the number and the type of obstacles.
Easy to use	Low	Ability for users to easily open the main menu, help screen and pause menu and save and load the game.	See proposed.	Provide to save and load game, understandable pop-ups for help screen and pause screen, and clear-view of the main menu.

4. Product Overview

4.1 Product Perspective

This app is self contained and independent of another system.

4.2 Summary of Capabilities

Customer Benefit	Supporting Features
Having a fun time.	A nice-looking and fun game design.
Ability to arrange their time.	Pause and save/load game features.
Socializing with friends over some competition.	Multiplayer mode.

4.3 Assumptions and Dependencies

1. It is assumed that players have access to a computer.
2. It is assumed that players have functional keyboards, screens and mouses.
3. It is assumed that players have an internet connection.
4. It is assumed that players have valid email adressses.
5. The game language is English.
6. It is assumed that players are literate in English.

4.4 Licencing and Installation

Players can install the game from our website. After installation, the installation guide will be provided, and the users will be licenced after entering a username, email address and a password of their choice.

5. Product Features

5.1 System Features

1. Start Game.
2. Exit Game.
3. Accept Keyboard Input.
4. Show Help Screen.
5. Pause Game.
6. Save Game.
7. Load Game.

5.2 Game Features

8. Moving the noble phantasm to the left or to the right.
9. Rotating noble phantasm.
10. Designing a game map.
11. Using spells.
12. Hitting an obstacle.
13. Multiplayer gameplay.
14. Switch to run mode.

6. Precedence and Priority

Priority	**Feature (By Number Above)**
High	1, 2, 3, 8, 9, 10, 12, 14
Medium	5, 6, 7, 11, 13
Low	4

7. Other Product Requirements

7.1 Applicable Standards

None specified.

7.2 System Requirements

1. The game must be run on a computer.
2. Player should have Java installed in their computers.

7.3 Performance Requirements

1. Game should support multiplayer.
2. Latency of the game should not be more than 1 second.
3. Frame per second should be 30.
4. Memory usage should not exceed 10% of total memory space.

7.4 Environmental Requirements

Having a stable internet connection.

8.Documentation Requirements

8.1 User Manual

Users can consult the help screen on the game by clicking on the respective icon.

Supplementary Specification

Revision History

Version	Date	Description	Author
Draft	Oct 25, 2021	First draft, to be revisioned.	Can Usluel, Kaan Türkmen, Halil Doruk Yıldırım
Revision	Oct 25, 2021	Revision is done.	Melis Oktayoğlu, Gökçe Sevimli

1. Objectives

In this document, FURPS requirements will be analyzed. Furthermore, the parts which are not covered in the use cases will be discussed. This document along with the use cases will be the complete analysis of the game.

2. Scope

This Supplementary Specification applies to the Need for Spear game and it is created by the /dev/null team.

Need for Spear game will be a game which can be played by a single player or multiplayer. The game will allow players to hit obstacles by controlling the Noble Phantasm and directing the sphere.

3. References

Templates inspired from:

https://csis.pace.edu/~marchese/SE616_New/Samples/Example%20%20Supplementary%20Specification.htm
https://sceweb.uhcl.edu/helm/RationalUnifiedProcess/webtmpl/templates/req/rup_sspec.htm

4. Functionality

4.1 Keep Account Information

System should keep the data of account information in the user's computer and auto login when the user opens the game.

5. Usability

5.1 Operating System Compliance

No specific operating system requirements due to Java Virtual Machine being a platform-independent environment.

5.2 Design for Ease-of-Use

Need for Spear user interface will be easily readable and designed clearly for the ease-of-use of the player.

6. Reliability

6.1 Sign-up Availability

The user should be able to sign up 24 hours a day, 7 days a week.

6.2 Frequency of failure

The game should not fail at any time.

7. Performance

7.1 Multiplayer

Game should be able to support 2 players simultaneously.

7.2 Database Access Response time

The game's latency should not be more than 1 second.

7.3 Throughput

Game should display at least 30 frames per second.

7.4 Resource Usage

Game should not use more than %10 of the total memory space available.

8. Supportability

This section will list the features which enhance user experience.

8.1 Texts from Database

Displayed texts will be read from the database. Thus, any change in the text will not require any additional update to the game.

9. Design Constraints

This section will list constraints that need to be obtained to create the game.

9.1 Java Compatibility

Player should have Java installed in their computers.

9.2 Hardware Requirements

Processor: Dual Core 1.4 GHz or better
Memory: 2 GB RAM
Storage: 500 MB available space

Glossary

Revision History

Version	Date	Description	Author
Draft	Oct 25, 2021	First draft, to be revisioned	Can Usluel, Doruk Yıldırım, Kaan Türkmen
Revision	Oct 28, 2021	Revision is done.	Melis Oktayoğlu, Gökçe Sevimli

Definitions

Term	Definition and Information	Aliases
Map	The area where the player will play the game or build a new map.	
Obstacle	Bricks that players need to destroy to get a score.	Brick
Sphere	The ball that the player uses to destroy bricks.	Ball
Noble Phantasm	The paddle user moves to hit the moving sphere.	Spear
Spell	A resource the player can gather during the game to use for a helpful effect.	Magical Ability
Database	A place to store or request data either on on-premises software or cloud.	Db, dbase
Firebase	A google service helps developers to authenticate or store data about the user.	Fb
User Authorization	Validation by the third party mechanism to save or load user details to the database.	
Gradle	A tool that helps developers to build their project.	
IntelliJ / Eclipse	An environment to help developers to code on.	IDE
User ID	Unique user identification number.	uid

/dev/null	The developer team of the game.	devnull
Slack	A messaging app where the developers communicate with each other.	
Git	A version control system where developers can work collaboratively.	
Gitlab	A DevOps tool which provides an interface to the Git system.	
Gitlab Issues	A part which is being used as a todo list and part of an agile development.	
Agile Development	A development type states that doing discovery and solution through collaborative manner in each time iteration.	
Miro	A collaborative whiteboard application used to design diagrams and workflows.	