Analysis document

Thomas van der Molen

IPS3-DB03

|  |  |
| --- | --- |
| **Project Information** | |
| Project members | Thomas van der Molen (4168003) |
| Project name | TextAdventure |
| Version | 1.2 |

Table of Contents

[Version History 2](#_Toc82605288)

[Introduction 3](#_Toc82605289)

[Requirements 3](#_Toc82605290)

[Functional requirements 3](#_Toc82605291)

[Non-functional requirements 4](#_Toc82605292)

[Use Cases 5](#_Toc82605293)

[Conceptual Model 8](#_Toc82605294)

[Game Content 9](#_Toc82605295)

[UI Concepts 10](#_Toc82605296)

# Version History

|  |  |  |
| --- | --- | --- |
| **Version** | **Date** | **Change** |
| 1.0 | 01-09-2021 | Created file and made: Introduction, requirements, conceptual model & UI Concepts. |
| 1.1 | 02-09-2021 | Added more requirements and made Requirements into MoSCoW, added use cases. |
| 1.2 | 15-09-2021 | Added Game Content. |

# Introduction

In this application you can play a text adventure game that will be procedurally generated while playing. You will also be able to save your last played adventure to resume later. Furthermore, there will be a leaderboard to show finished adventures with the highest achieved scores.

# Requirements

Requirements (FR-/NFR-) for the application including possible restrictions (R-) and qualitative demands (Q-).

## Functional requirements

**Must Have**

FR-01 The user can start a new adventure.

Q-01.1 The user will get notified if an adventure already exists on account.

FR-02 The user can register an account.

R-02.1 The user needs to enter a unique username and email.

Q-02.1 The username can’t contain any offensive words.

FR-03 The user can see a leaderboard of finished adventures.

**Should Have**

FR-04 The user can change his password.

R-04.1 The user must fill in correct current password.

FR-05 The user can choose to not show their finished adventures on the leaderboards.

FR-06 The user can save and resume an adventure.

R-06.1 The user needs to be logged in.

FR-07 The user can see his own completed adventures.

FR-08 The user can find an overview of all useable commands.

FR-09 The user can see information on the current adventure.

**Could Have**

FR-10 The user can access all saved/finished adventures and user information.

R-10.1 The user has to be an admin.

Q-10.1 The user cannot see passwords.

FR-11 The user can see a map of the explored dungeon.

FR-12 The user can post comments on finished adventures.

R-12.1 The user needs to be logged in.

FR-13 The user can request an automated password reset.

FR-14 The user can play with other users in one dungeon.

**Won’t Have**

## Non-functional requirements

NFR-01 The user will get a feedback notification when sending data.

NFR-02 Actions submitted in games will give feedback that it has been executed.

NFR-03 The game will have fast response times.

Q-03.1 The response delay cannot be longer than 500 milliseconds

# Use Cases

|  |  |
| --- | --- |
| **Name:** | **UC01: Starting new adventure** |
| Summary | A user can start a new adventure |
| Actors | User |
| Assumptions | Actor is on the home page |
| Scenario | 1. Actor wants to start a new adventure 2. Actor presses new game button 3. System creates a new game 4. System redirects actor to game page |
| Exceptions | 1. User has a saved adventure. (Q-01.1) Show message warning user that this action will delete the saved adventure, if confirmed go to 3. else bring user to homepage |
| Result | A new adventure has been created for user. |

|  |  |
| --- | --- |
| **Name:** | **UC02: Registering account** |
| Summary | A user can register an account |
| Actors | User |
| Assumptions | Actor is on the home page |
| Scenario | 1. Actor wants to register a new account 2. Actor chooses the register account option 3. System shows form with username, email and password 4. Actor fills in information 5. Actor clicks on create account 6. System creates the account |
| Exceptions | 1. Actor does not fill in all fields properly. (R-02.1) Show message telling user what field was not filled in properly, go to 3 2. Actor fills in username or email that is already in use. (Q-02.1) Show message telling user that the filled in information is already in use, go to 3 |
| Result | A new account has been created. |

|  |  |
| --- | --- |
| **Name:** | **UC03: Viewing leaderboards** |
| Summary | User wants to view leaderboards |
| Actors | User |
| Assumptions | Actor is on the home page |
| Scenario | 1. Actor wants to view leaderboards 2. Actor presses the leaderboards button 3. System get’s all viewable finished runs and orders them based on score 4. System redirects actor to leaderboard |
| Exceptions | none |
| Result | Actor is shown an up-to-date leaderboard. |

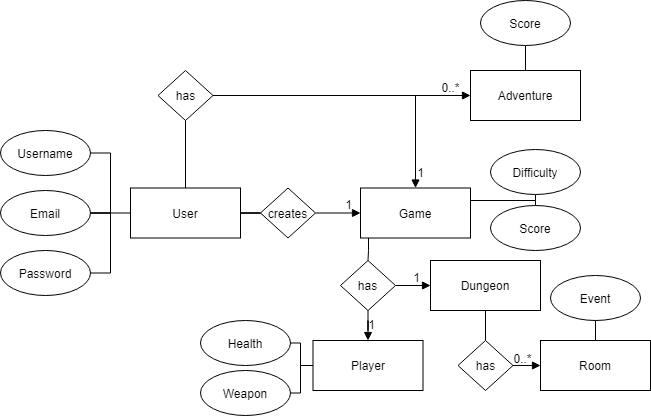
|  |  |
| --- | --- |
| **Name:** | **UC04: User turns off adventure visibility** |
| Summary | User wants it’s finished adventures not to show up on the leaderboards. |
| Actors | User |
| Assumptions | Actor is on the home page and is logged in |
| Scenario | 1. Actor wants to change visibility of finished adventures 2. Actor presses the account button 3. System redirects actor to account page 4. Actor unchecks the show adventures on leaderboards option 5. System will change actor’s account to not show adventures on leaderboards |
| Exceptions | None |
| Result | Actor’s adventures will no longer show up on the leaderboards. |

|  |  |
| --- | --- |
| **Name:** | **UC05: User changes password** |
| Summary | User wants to change password |
| Actors | User |
| Assumptions | Actor is on the home page and is logged in |
| Scenario | 1. Actor wants to change password 2. Actor presses the account button 3. System redirects actor to account page 4. Actor clicks on change password button 5. System shows input fields with current password and new password 6. Actor fills in the input fields 7. Actor clicks on change password button 8. System checks if fields are valid 9. System changes actor’s password |
| Exceptions | 1. Actor fills in the wrong current password. (R-04.1) Show actor that filled in password is incorrect go to 5 2. Actor fills in the same current and new password. Show actor that filled in passwords are the same go to 5 |
| Result | Actor’s password is changed. |

|  |  |
| --- | --- |
| **Name:** | **UC06: User wants to save adventure** |
| Summary | User would like to save current adventure to resume later |
| Actors | User |
| Assumptions | Actor is playing an adventure |
| Scenario | 1. Actor wants to save adventure 2. Actor presses the save adventure button 3. System saves all current adventure data 4. System shows message saying adventure saved |
| Exceptions | 1. Actor is not logged in. Show player he has to be logged in to save adventures |
| Result | Actor’s current adventure has been saved |

|  |  |
| --- | --- |
| **Name:** | **UC07: User wants to resume adventure** |
| Summary | User would like to resume a previously saved adventure. |
| Actors | User |
| Assumptions | Actor is on the home page |
| Scenario | 1. Actor wants to resume adventure 2. Actor logs in 3. System checks if actor has a saved adventure 4. System shows user a resume adventure button 5. Actor clicks resume adventure button 6. System loads saved adventure to game 7. System redirects actor to the game’s page |
| Exceptions | 1. User does not have a saved adventure. System does not show the resume adventure button. |
| Result | Actor’s saved adventure has been resumed. |

# Conceptual Model



In my application is the user a person, that can create and play a game. The user can also store a game to resume. Every user can have many finished runs also called adventures, these adventures are stored and can be viewed and sorted by score.

A user has a username, email, password and can have many adventures and a game.

An adventure has a score.

A game has a difficulty, score and includes a player and dungeon.

A player has health and a weapon.

A dungeon has many rooms.

A room has an event.

# Game Content

The game will contain many different mechanics these mechanics have been split up in individual requirements. This is to both keep in mind the scope of the project and the progress of the game’s functionality.

1. The game will have a player who has health, damage and level.
2. The player will have an inventory with items such as potions, keys and weapons.
3. The weapons will have a durability check making them break over time.
4. The game spawn the player in a massive dungeon that all players traverse.
5. The player will appear in a random in the dungeon that has not been explored yet.
6. Every room will have 1 special event that can happen when entering the room
7. The game will not overlap rooms.
8. Every room will have a north, east, south and west side. These sides can all contain 1 special occurrence such as a door, person, item.
9. Every player will have a map of what they have explored.
10. Game events will be based on the player’s level.

# UI Concepts

****The homepage (figure 1) will give the user easy access to starting or resuming a new run depending if they are logged in. The homepage will also include a navigation bar at the top giving links to the leaderboard, an about section and login.

Figure 1 homepage

The homepage will also include a short list op the top players on the leaderboard.

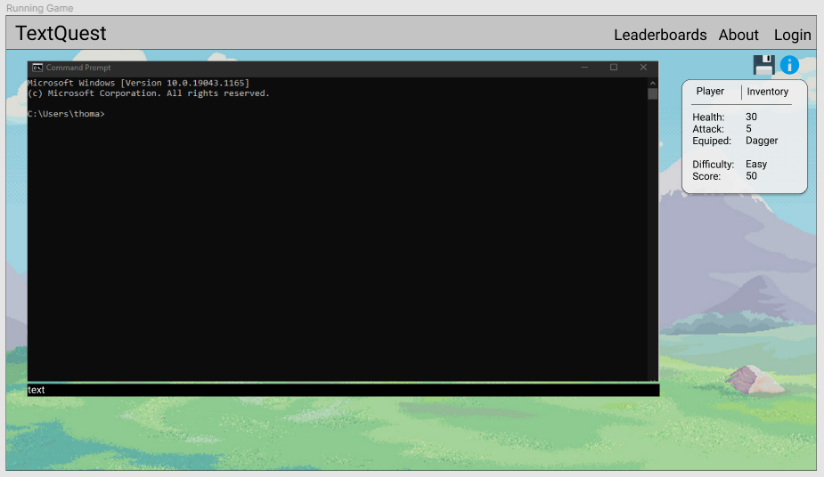
****On the game page (figure 2) there will be a command-line style window, in this window all dialogue will show up. Under the command-line window will be a text input box, the user will type the commands they want to execute in here.

Figure 2 Game

On the right side of the screen there will be helpful information. This includes a button to give the user an overview of all commands and a save button. There is also an information box showing useful data like the player’s health or it’s inventory.