Software Engineering
Software Requirements Specification (SRS) Document

Dungeon Game

14 Feb 2023

0.0.1

By: Hayden Perusek, Samantha Schnell, Craig Smith

My words and actions will reflect Academic Integrity.
I will not cheat or lie or steal in academic matters.
I will promote integrity in the UNCG community.

Table of Contents

1. Introduction

1.1. Purpose

In making the game, we hope to provide a solid way to spend some time while relaxing and getting immersed through the imagination in a text-based adventure. The app hopes to solve boredom and provide entertainment! The game will be able to be run in a simple terminal window, allowing for a consistent experience between gameplay. You get to take on the role of a mighty adventurer, traversing through a dungeon while fighting monsters all packaged within a terminal window. The game makes use of ASCII art to give the user a fun visual feedback that helps the overall enjoyment of the game.

1.2. Document Conventions

The purpose of this Software Requirements Document (SRD) is to describe the client-view and developer-view requirements for the Dungeon Game. Client-oriented requirements describe the system from the client's perspective. These requirements include a description of the different types of users served by the system. Developer-oriented requirements describe the system from a software developer's perspective. These requirements include a detailed description of functional, data, performance, and other important requirements.

1.3. Definitions, Acronyms, and Abbreviations

Java	A programming language originally developed by James Gosling at Sun Microsystems. We will be using this language to build the Restaurant Manager.
JDK	The Java Development Kit (JDK) is a distribution of Java Technology by Oracle Corporation. It implements the Java Language Specification and the Java Virtual Machine Specification and provides the Standard Edition of the Java Application Programming Interface.
FasterXML	The Extensible Markup Language (XML) is a simple text-based format for representing structured information: documents, data, configuration, books, transactions, invoices, and much more. It was derived from an older standard format called SGML (ISO 8879), in order to be more suitable for Web use. FasterXML is an open source XML and JSON parsing for the Java platform.
ASCII	American Standard Code for Information Interchange, is a character encoding standard for electronic communication. ASCII codes represent text in computers, telecommunications equipment, and other devices.
.HTML	Hypertext Markup Language. This is the code that will be used to structure and design the web application and its content.
SpringBoot	An open-source Java-based framework used to create a micro Service. This will be used to create and run our application.
MVC	Model-View-Controller. This is the architectural pattern that will be used to implement our system.
Spring Web	Will be used to build our web application by using Spring MVC. This is one of the dependencies of our system.

Thymeleaf	A modern server-side Java template engine for our web environment. This is one of the dependencies of our system.
NetBeans	An integrated development environment (IDE) for Java. This is where our system will be created.
API	Application Programming Interface. This will be used to implement a function within the software where the current date and time is displayed on the homepage.

1.4. Intended Audience

The intended audience for this SRS document is to provide context and framework for developers and users. The entire document will be useful for developers, while chapters 1-4 will apply to users who want to become more acquainted with the technologies in use. Regardless, the document will give a high-level breakdown for anyone reading and provide context into the processes in use throughout running the project.

1.5. Project Scope

The goal of the Dungeon Game software is to provide an easy-to-use video game for all users on their personal computers. The low performance requirements, lack of safety concerns, and ease of use appeal to all users of all age groups. This aligns with the overall business goals of a video game startup company, which will lead to a growth in popularity which will lead to future games that could be paid for.

The benefits of the project to business include:

- Relieving stress and pressure from everyday life by allowing users to channel their energy into a fantasy video game world.
- Allowing nearly every device to run this game. No need to buy or build an expensive gaming computer to enjoy gaming.
- Allowing reviewers and gamers the opportunity to give direct feedback to our company which can foster later improvements, additions, updates, etc.

1.6. Technology Challenges

[Any technological constraints that the project will be under. Any new technologies you may need to use]

1.7. References

Quinn, L.R.E. (no date) *XML ESSENTIALS*, *W3C*. Available at: https://www.w3.org/standards/xml/core#:~:text=What%20is%20XML%3F,more%20suitable%20for%20Web%20use (Accessed: February 17, 2023).

Injosoft AB, http://www.injosoft.se. (n.d.). *ASCII table - A table of ASCII codes, characters and symbols*. Retrieved February 18, 2023, from https://www.ascii-code.com/

Download the latest Java Lts Free. Oracle. (n.d.). Retrieved February 18, 2023, from https://www.oracle.com/java/technologies/downloads/

2. General Description

2.1. Product Perspective

Video games are one of the many technologies that provide a near limitless ability to keep its user entertained. Entertainment, in itself, is an important factor for the rest and relaxation of someone coming off of a busy work week or stressful exam. Dungeon Game owes its origins to satisfying that need of entertainment, and providing a lane for the rest and relaxation of the user.

2.2. Product Features

In Dungeon Game, the user will take the role of an adventurer navigating the treacherous and dangerous rooms of a dungeon. In the game, the player will be able to select a save profile, select a class, and navigate the rooms of the dungeon. The dungeon will be filled with monsters that the player can combat with and collect gold from. The score at the end of the game will be determined by the amount of gold collected throughout the game. The game can also be saved at any time while playing. Along with this, the demo user of the game will be able to play through a static version of the game, and write a game review at the end of their adventure. The admin profile will allow access to changing the base statistics of the players and monsters, and allows access to the save files for deletion.

2.3. User Class and Characteristics

Our application expects our users to have some familiarity with IDE use in order to open the file as a project and run locally. Outside of this, no other knowledge is needed as, once the game is running, it takes simple keyboard inputs for all of the actions in the game.

2.4. Operating Environment

This application is designed to run on any Java IDE running at least JDK 18 across any operating system.

2.5. Constraints

Because the application is designed to run primarily in a java IDE, it will severely limit the methods users can run the application.

2.6. Assumptions and Dependencies

The application will rely on the FasterXML API for making HTTP requests and formatting the returned JSON and the randommer.io names API for generating random names and assigning them to characters.

3. Functional Requirements

[Statements of services the system should provide, how the system should react to particular inputs and how the system should behave in particular situations.]

3.1. Primary

- FR0: The system for the game will allow a user to play a text-based RPG through a simple terminal window. The game consists of an adventurer traveling through a predetermined dungeon map where they will face creatures with different statistics depending on the creature. The gamer will win or lose the game depending if they finish the dungeon or not.

- FR1: The system will allow the user to either log in as a customer with full access or as a demo reviewer with limited access. An admin also has access to the inner workings of the game.

- FR2: The system will allow a customer to play the RGP to its fullest with the ability to access saved files and create a character based on different classes. A demo reviewer can only play the basics of the game, but they can also write and post a review. An admin will be allowed to access the inner files of the game and be able to modify the statistics of the monsters or the player character.

-

3.2. Secondary

- Password protection needed through login for customers and admin to access more than just the limited access granted to the demo reviewer.

- Authorization that allows admin to alter and save the statistics of both character and monster, and the ability to manipulate files.

- Authorization that allows customers to create a character and access saved files.

- Authorization that allows demo reviewers to create and post a review.

4. Technical Requirements

4.1. Operating System and Compatibility

The application will be compatible with any operating system that is able to view and to interact with any Java application with a terminal window. It will also be able to interact with web pages through the review writing and posting ability.

4.2. Interface Requirements

4.2.1. User Interfaces

[The logic behind the interactions between the users and the software. This includes the sample screen layout, buttons and functions that would appear on every screen, messages to be displayed on each screen and the style guides to be used.]

The application will be utilizing a webpage to be able to write/edit a review and be able to post it for anyone to view.

4.2.2. Hardware Interfaces

The web application will run on any hardware device that has access to the internet and the limited ability to interact with web pages. This includes desktop computers and laptops.

4.2.3. Communications Interfaces

The RPG is able to connect to the internet and its HTTP will be able to connect and utilize the online translator provided by LibreTranslate API and return the text of the RPG in whatever language the user selects

4.2.4. Software Interfaces

The frontend of the RGP will be built by Spring Boot ThymeLeaf and the front and back end will be connected by Spring Boot with Java.

5. Non-Functional Requirements

[Constraints on the services or functions offered by the system (e.g., timing constraints, constraints on the development process, standards, etc.). Often apply to the system as a whole rather than individual features or services.]

5.1. Performance Requirements

These are the performance requirements specified for all the functional requirements.

- NFR0(R): The local copy of the Dungeon Game will consume less than 20 MB of memory
- NFR1(R): The system (including the local copy of the Dungeon Game) will consume less than 50MB of memory
- NFR2(R): The novice customer will be able to play and win the game in less than 5 minutes.
- NFR3(R): The demo reviewer will be able to play, win the game, and submit a review in less than 8 minutes.

5.2. Safety Requirements

There may be some aspects of the Dungeon Game that unintentionally alter the system files of the user's personal computer. A possibility of corrupted files during the downloading process can result in unpredictable errors in running the game.

5.3. Security Requirements

Privacy and data protection regulations that need to be adhered to while designing the product.

- NFR4(R): The system will be open to the general public for free.
- NFR5(R): No Personal Identifiable Information (PII) will be collected

5.4. Software Quality Attributes

[Detailing on the additional qualities that need to be incorporated within the software like maintainability, adaptability, flexibility, usability, reliability, portability etc.]

5.4.1. Availability

Given the limited time and resources, should the application fail operation at some point during the process, it likely will need to be resolved by the user with a fresh version of the application on their local machine.

5.4.2. Correctness

As soon as the application is downloaded onto a user's machine, it will run as intended unless the source code is altered in some way. If the source code is altered and the application ceases to function, a fresh version of the application would need to be installed.

5.4.3. Maintainability

Once the application is downloaded on a user's local machine, it is self-sufficient enough to be used repeatedly with nothing needing maintenance, save for the profile system which can be updated through play or by the admin account.

5.4.4. Reusability

With the development of the game, we aim to make an experience that is infinitely replayable.

5.4.5. Portability

The application can easily be transported by being downloaded on a local machine and then loaded onto a flash drive. With that in mind, the application will transport very well between any machine the user has access to with only the requirement of having a locally installed IDE that can run Java.

5.5. Process Requirements

5.5.1. Development Process Used

Scrum- Agile approach:

Allows for closer communication and more time and maneuverability of planning and changing program if necessary.

5.5.2. Time Constraints

The raw coding of the RPG will take a large amount of time. The final planning of the project should be completed with time to effectively do the coding and testing of the RPG by the due date at the end of the semester.

5.5.3. Cost and Delivery Date

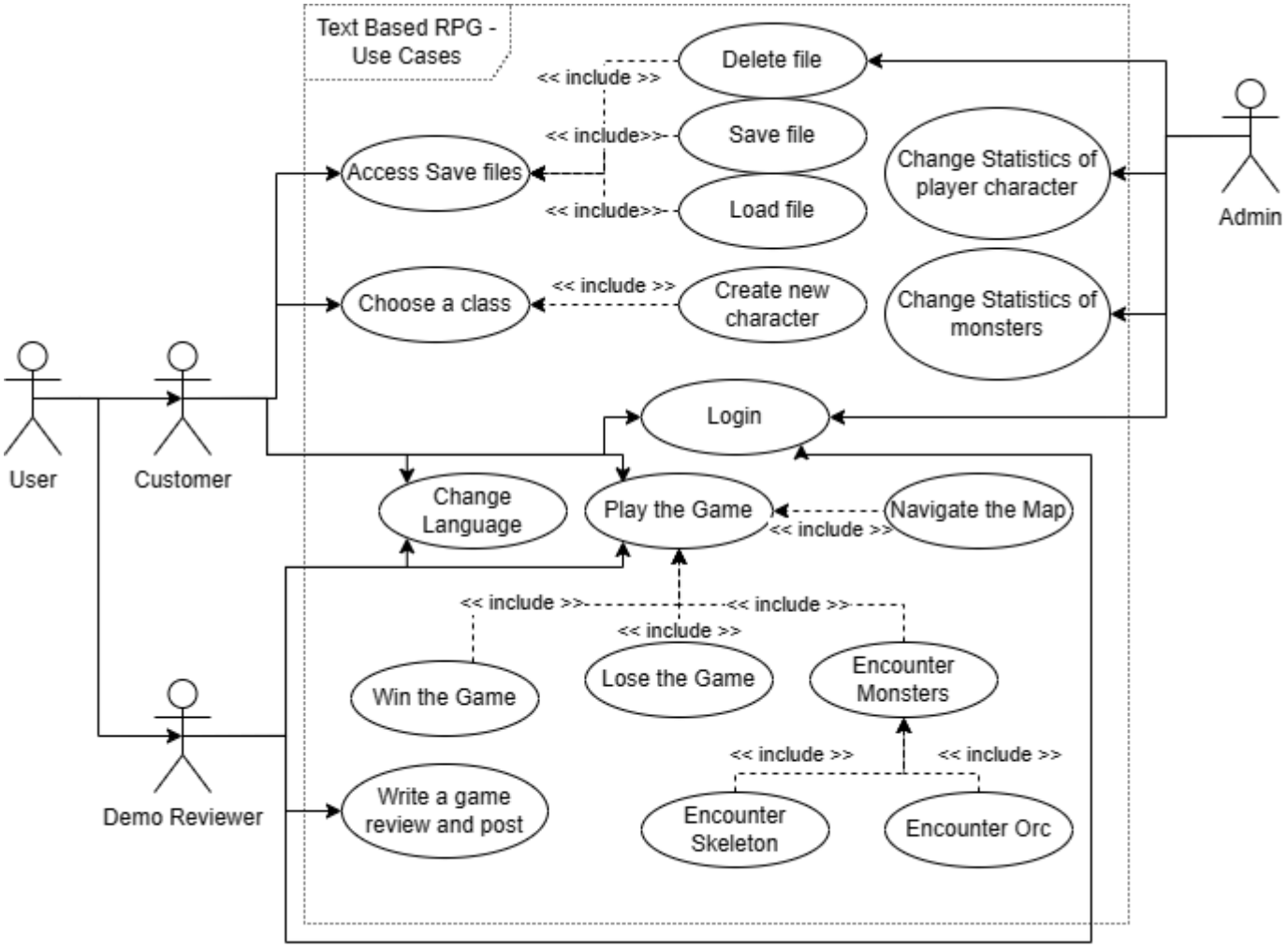
So far, no additional cost besides time will be needed. The project will be completed by the due date.

5.6. Other Requirements

- An ASCII image converter for graphics

TBD

5.7. Use-Case Model Diagram



5.8. Use-Case Model Descriptions

5.8.1. Actor: Admin (Samantha Schnell)

- **Delete file:** The admin will have the ability to delete files created in the RPG.
- **Save file**: The admin will be able to save files.
- **Load file:** The admin will be able to load any files in the RPG.
- **Change Statistics of the player character:** The admin is able to change the statistics of the character used to play the game.
- **Change Statistics of monsters:** The admin will be able to change the statistics of the monsters that the character player will fight in the game.
- **Log in**: Admin can use a username and password to log in to the game and begin playing.

5.8.2. Actor: Demo Reviewer (Hayden Perusek)

- **Choose a class**: Demo Reviewer will have the ability to choose between one of three classes that slightly alter their in game statistics.
- **Play the game**: Demo Reviewer will have the ability to start a new game and begin traversing the in-game world. During this they can either win or lose the game and encounter monsters to combat with. For the Demo Reviewer, this will be a single playthrough. All data will be lost when the Demo Reviewer exits the game, which they will need to start over if they want to play again.

- **Send reviews to Admin**: Demo Reviewer will be able to send reviews or recommendations to the Admin for improvements to the game. This will only be available after they complete the demo session.
- **Log in**: Demo Reviewer can use a username and password to log in to the game and begin playing.
- **Change language**: Demo Reviewer will be able to change the language of the in-game text. To be asked for on opening the game. Allows for better accessibility between language barriers.

5.8.3. Actor: Customer (Craig Smith)

- **Choose a class**: The customer will have the ability to choose between one of three classes that slightly alter their in game statistics.
- **Play the game**: Customer will have the ability to load a new game and begin traversing the in-game world. During this they can either win or lose the game and encounter monsters to combat with.
- **Access Save Files**: Customer will have access to three profiles which they can save three different loaded in game worlds to.
- **Log in**: Customer can use a username and password to log in to the game and begin playing.
- **Change language**: Customer will be able to change the language of the in-game text. To be asked for on opening the game. Allows for better accessibility between language barriers.

5.9. Use-Case Model Scenarios

5.9.1. Actor: Admin (Samantha)

- **Use-Case Name**: Delete File
    - **Initial Assumption**: The admin will be able to physically delete an existing file
    - **Normal**: The admin will select a file to delete and the game will run through a command to delete the game from the saved text files.
    - **What Can Go Wrong:** The game could not be able to find the selected file due to it not existing or not properly saved in the text file.
    - **Other Activities**: Multiple files can be saved by not just the admin but the customer as well
    - **System State on Completion**: The selected file will be purged from the systems saved files and not be recoverable
- **Use-Case Name**: Save File
    - **Initial Assumption**: Save files are stored as snapshots on the local machine that can be written to and read from.
    - **Normal**: Admin can create and edit a text file that then can be saved in the system.
    - **What Can Go Wrong**: The game could try to save a file that doesn't exist or simply not find the file to write to. Some files could be corrupted by outside forces or unforseen errors by the user.
    - **Other Activities**: The save files can be deleted and loaded by the admin. Customers can also save snapshots of the game to save their process.
    - **System State on Completion**: Save files are set and can be saved/loaded from any point in the game and deleted from the main menu.
- **Use-Case Name**: Load File

    - **Initial Assumption**: Saved files can be called from the local machine and loaded into the game itself
    - **Normal**: Admin can load files into the system in order to gain access or change the RPG
    - **What Can Go Wrong**: The system may not be able to load the file properly due to improperly saved files, files do not exist, or the file is corrupted.
    - **Other Activities**: Customers can access saved files and load them to keep track of their process. Admin can also manipulate files and delete/save them.
    - **System State on Completion**: Saved files can be loaded by the customer at any point or they can be loaded manually by the admin.
- **Use-Case Name**: Change Statistics of the player character
    - **Initial Assumption**: The statistic values of the character player are determined and saved into a file for the system to read from.
    - **Normal**: The admin determines the appropriate statistical values needed for the player character determined by class and saves the file to the system for the RPG to read.
    - **What Can Go Wrong**: The admin could enter the wrong or no statistical values in the appropriate file. The wrong file could be edited or the file could not be found.
    - **Other Activities**: The statistics can only be edited by the admin who is logged in. The statistics are what determines the fighting capabilities of the player character and aid in winning the game.
    - **System State on Completion**: The user playing the game will be able to create a character capable of completing the game with the appropriate statistics.
- **Use-Case Name**: Change Statistics of the player character
    - **Initial Assumption**: The statistic values of the monsters(orc and skeleton) are determined and saved into a file for the system to read from.
    - **Normal**: The admin determines the appropriate statistical values needed for the monster depending on type and saves the file to the system for the RPG to read.
    - **What Can Go Wrong**: The admin could enter the wrong or no statistical values in the appropriate file. The wrong file could be edited or the file could not be found.
    - **Other Activities**: The statistics can only be edited by the admin who is logged in. The statistics are what determines the fighting capabilities of the monsters and aid in defeating/loosing against the player character.
    - **System State on Completion**: The user playing the game will be able to interact with different monsters as challenges in the game in order to win the RPG.
- **Use-Case Name**: Log in
    - **Initial Assumption**: User enters a username and password to identify them as an admin.
    - **Normal**: The user is able to input the username and password and, from a text file, the system is capable of recognizing them.
    - **What Can Go Wrong**: The user could input an incorrect username or password
    - **Other Activities**: The user is able to retry their account information if they input it incorrectly.
    - **System State on Completion**: The user is able to log in to the application and log out to log in under some other account.

5.9.2. Actor: Demo Reviewer (Hayden)

- **Use-Case Name**: Choose a class
    - **Initial Assumption**: User is able to select one of three classes when starting a new game.
    - **Normal**: The game will create a new object of that class is created to represent the character
    - **What Can Go Wrong**: The admin could change the statistics of the classes and cause overflow of their numbers, forcing the character to have statistics that simply don't work.
    - **Other Activities**: The player could go back to the main menu and log in to a different account from there.
    - **System State on Completion**: The three classes are displayed at the start of a new game and through a simple switch case the class can be selected.
- **Use-Case Name**: Play the game
    - **Initial Assumption**: The Demo Reviewer, after selecting a class, is able to load into a new game world and begin a new profile. They will not be able to load from an already existing profile.
    - **Normal**: The Demo Reviewer can start the new game and begin traversing the game world through the use of text commands that guide them through the level.
    - **What Can Go Wrong**: The player may not have a good idea of what in game commands could be used.
    - **Other Activities**: Implementation of a "Help" command that can show the player what actions are at their disposal at any given point in the game.
    - **System State on Completion**: The Demo Reviewer can traverse the in-game world with a clear understanding of all commands provided to them from the game menu and allowing for saves at any given point.
- **Use-Case Name**: Log in
    - **Initial Assumption**: User inputs a username and password to identify them as a Demo Reviewer.
    - **Normal**: The user is able to input the username and password and, from a text file, the system is capable of recognizing them.
    - **What Can Go Wrong**: The user could input an incorrect username or password
    - **Other Activities**: The user is able to retry their account information if they input it incorrectly.
    - **System State on Completion**: The user is able to log in to the application and log out to log in under some other account.
- **Use-Case Name**: Change language
    - **Initial Assumption**: The user may want to change the language the game's text is in.
    - **Normal**: The customer has the ability to change the language of the game from the start screen of the game.
    - **What Can Go Wrong**: The user could accidentally choose a language they don't understand, or could have trouble navigating to the language menu in the first place.
    - **Other Activities**: The user can alter the language of the game at any time through the game's text input.
    - **System State on Completion**: The user inputs their language preferences and all of the game text changes to reflect that choice.

- **Use-Case Name**: Write a game review and post
    - **Initial Assumption**: The Demo Reviewer may want to submit a review after a playthrough of the game.
    - **Normal**: The Demo Reviewer has the ability to submit a review that will be seen by Admin after they play the game. The prompt to submit a review will be available after they opt to quit the game, lose the game or after they win. They will also be able to submit a review from the main menu.
    - **What Can Go Wrong**: The prompt for submitting a review does not populate.
    - **Other Activities**: The Demo Reviewer is able to retry the review submission.
    - **System State on Completion**: The message "Review received!" will populate after a successful review submission. The Demo Reviewer will return to the main menu.

5.9.3. Actor: Customer (Craig)

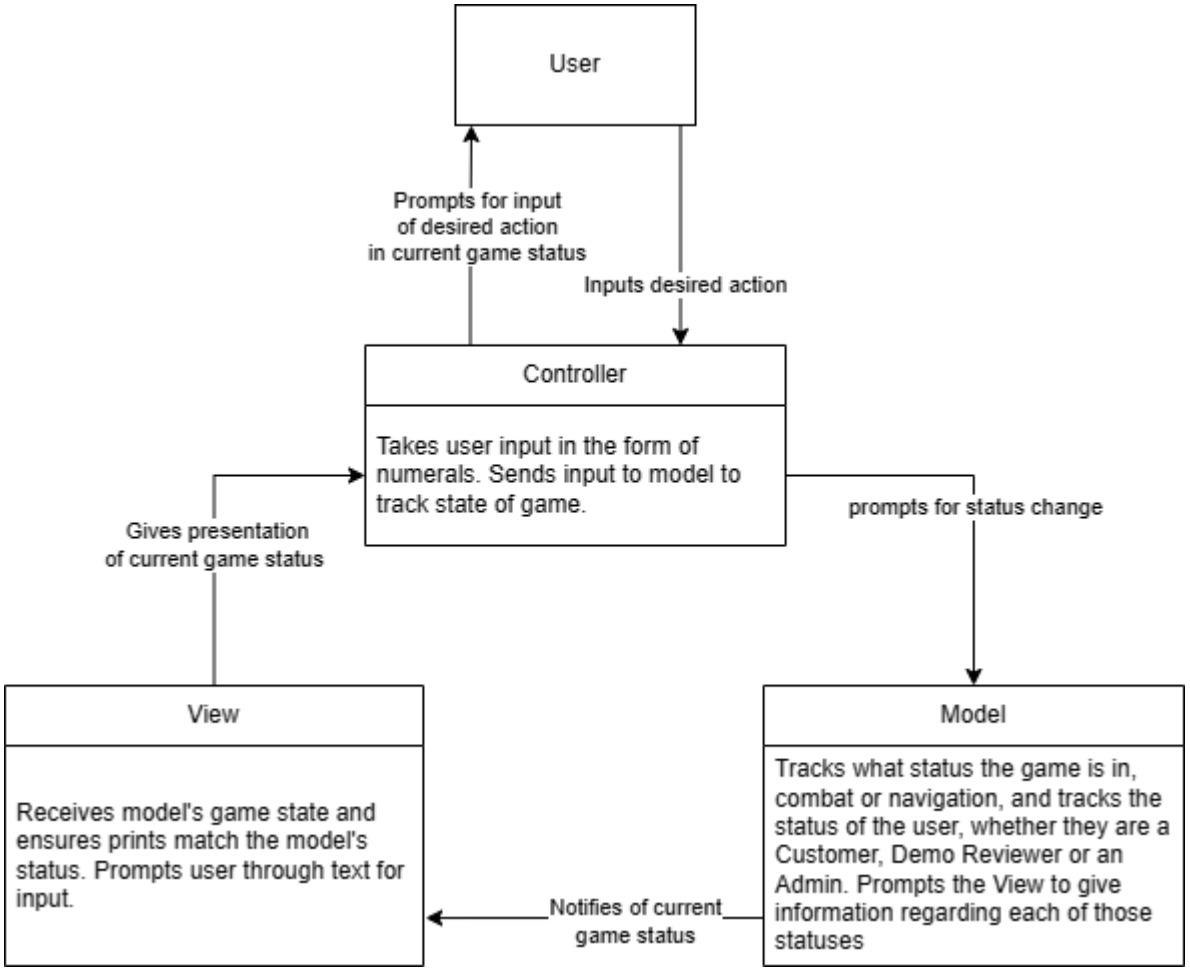
- **Use-Case Name**: Choose a class
    - **Initial Assumption**: User is able to select one of three classes when starting a new game.
    - **Normal**: The game will create a new object of that class is created to represent the character
    - **What Can Go Wrong**: The admin could change the statistics of the classes and cause overflow of their numbers, forcing the character to have statistics that simply don't work.
    - **Other Activities**: The player could go back to the main menu and log in to a different account from there.
    - **System State on Completion**: The three classes are displayed at the start of a new game and through a simple switch case the class can be selected.
- **Use-Case Name**: Play the game
    - **Initial Assumption**: The customer, after selecting a class, is able to load into a new game world and begin a new profile, or they are able to load from an already existing profile.
    - **Normal**: The customer can load their game and begin traversing the game world through the use of text commands that guide them through the level.
    - **What Can Go Wrong**: The player may not have a good idea of what in game commands could be used.
    - **Other Activities**: Implementation of a "Help" command that can show the player what actions are at their disposal at any given point in the game.
    - **System State on Completion**: The customer can traverse the in-game world with a clear understanding of all commands provided to them from the game menu and allowing for saves at any given point.
- **Use-Case Name**: Access save files
    - **Initial Assumption**: Save files are stored as snapshots on the local machine that can be written to and read from.
    - **Normal**: Customers can load from files already existing into the same point they left off from or save during the game to the save file associated with their currently selected profile.

        - **What Can Go Wrong**: The game could try to load from a file that doesn't exist or simply not find the save file to write to. Save files could be corrupted if mismanaged by the user through the file browser.
        - **Other Activities**: The save files can be deleted based on the profile they are under, allowing for a fresh game to be booted from that file.
        - **System State on Completion**: Save files are set and can be saved/loaded from any point in the game and deleted from the main menu.
- **Use-Case Name**: Log in
    - **Initial Assumption**: User inputs a username and password to identify them as a customer.
    - **Normal**: The user is able to input the username and password and, from a text file, the system is capable of recognizing them.
    - **What Can Go Wrong**: The user could input an incorrect username or password
    - **Other Activities**: The user is able to retry their account information if they input it incorrectly.
    - **System State on Completion**: The user is able to log in to the application and log out to log in under some other account.
- **Use-Case Name**: Change language
    - **Initial Assumption**: The user may want to change the language the game's text is in.
    - **Normal**: The customer has the ability to change the language of the game from the start screen of the game.
    - **What Can Go Wrong**: The user could accidentally choose a language they don't understand, or could have trouble navigating to the language menu in the first place.
    - **Other Activities**: The user can alter the language of the game at any time through the game's text input.
    - **System State on Completion**: The user inputs their language preferences and all of the game text changes to reflect that choice. This system prompts the user for a choice at the beginning of every session.

6. Design Documents

6.1. Software Architecture

Dungeon game uses a 'Model-View-Controller' (MVC) software architecture. The game internally controls all three of these devices, but the syntax is laid out in such a way that the three are easy to differentiate into these three categories.

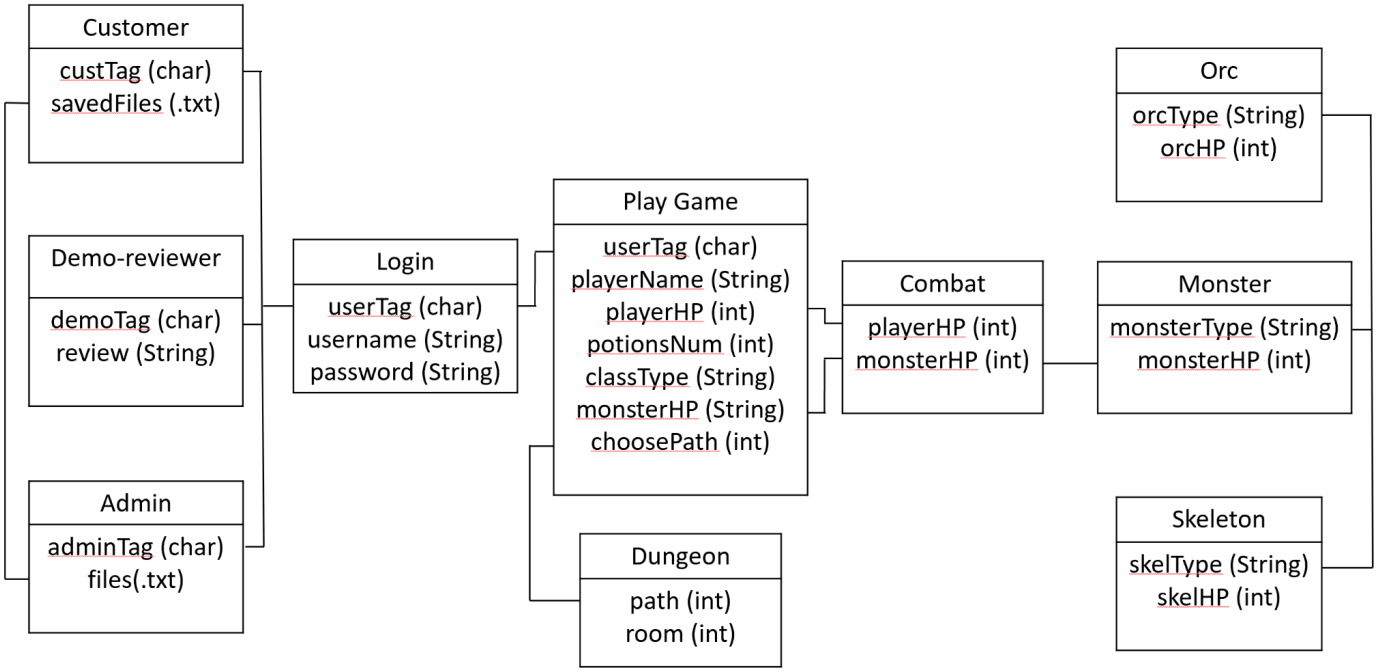


Controller: Makes use of the classes such as ‘GameLogic’ and the methods therein to read user choices and update the model with the user’s current desired actions. Uses ‘Login’ class to accept user credentials and update Model with user’s identification.
Model: Uses classes such as ‘GameLogic’ and ‘Login’, using ‘Player’, ‘Demo’ and ‘Admin’ to track user’s status and current game state. Starts and ends navigation or combat depending on current state. Start and end game based on character’s condition. Tracks players’ position using ‘Dungeon’, ‘Path’ and ‘Room’. Takes controller’s input to current action and creates a new game state with the controller’s updates. Then sends data to view to fetch correct updates to show to the player.
View: Prompts user using ‘GameLogic’ as well as the classes ‘Dungeon’, ‘Path’ and ‘Room’ to send correct description of player’s current location to controller to display. Allows users to know their character’s status and position. Uses ‘Story’ Class to update the user with the current status of the character's game, win or lose.

6.2. High-Level Database Schema

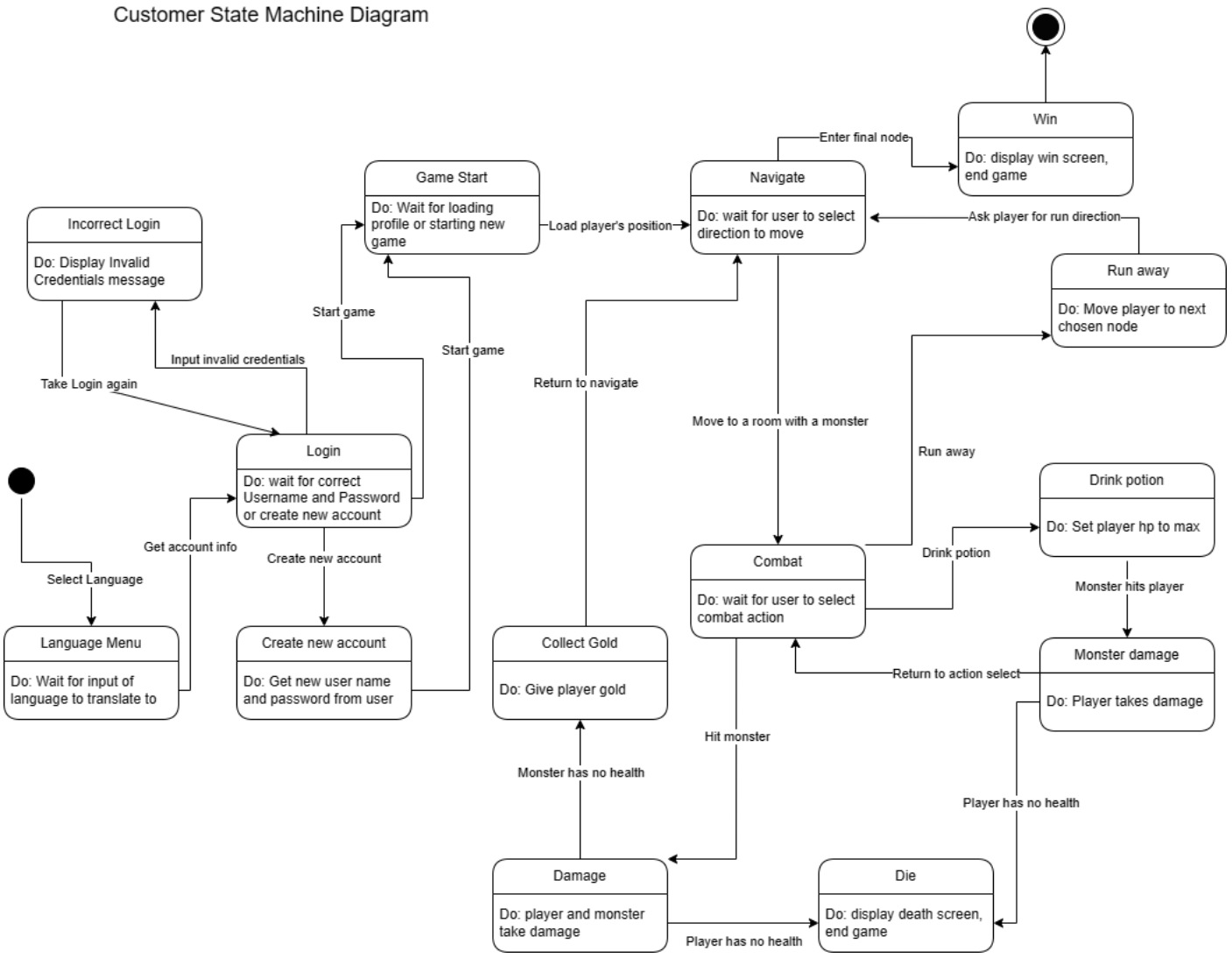
The data is all contained within the code of the RPG itself. It acts as a desktop application and does not rely on an outside/local database. However, the RPG relies heavily on text and CSV files in order to contain important information such as the login information of each user. Player/customer and Demo-reviewer both have a CSV file where the user will be able to record their username and password information once it is created. The system will later read the information to allow the user to access the rest of the RPG. Admin also has a CSV file containing a predetermined admin username and password that the system reads and then allows the admin user access to their area of the RPG. The Player/customer and the

Demo-reviewer both have access to the actual gameplay of the game. They each are able to create a player profile that contains name, HP/Stats levels, number of potions, and character types. These variables are contained with the methods of the RPG's code and are string and int variables respectfully. At anytime in the game, a customer can save their progress and a snapshot of the game choices and code is captured and placed in a text file that can later be uploaded back into the RPG. A Demo-reviewer also has the ability to write a review that is then printed to a text file as well. An Admin has the ability to view/delete/modyify any existing text files including password/username text(CVS) files, customer saved game files, and Demo-reviewer review text files. Creature/Monster classes each contain HP stats which are int values and contained with the code of the RPG similar to the Player class. The combat determines the stats and calculates who wins the battle based on randomized int values. The dungeon class is based of a directed graph and determines where in the dungeon maze the player ends up based on their choice with a monster in each room to engage in combat with.

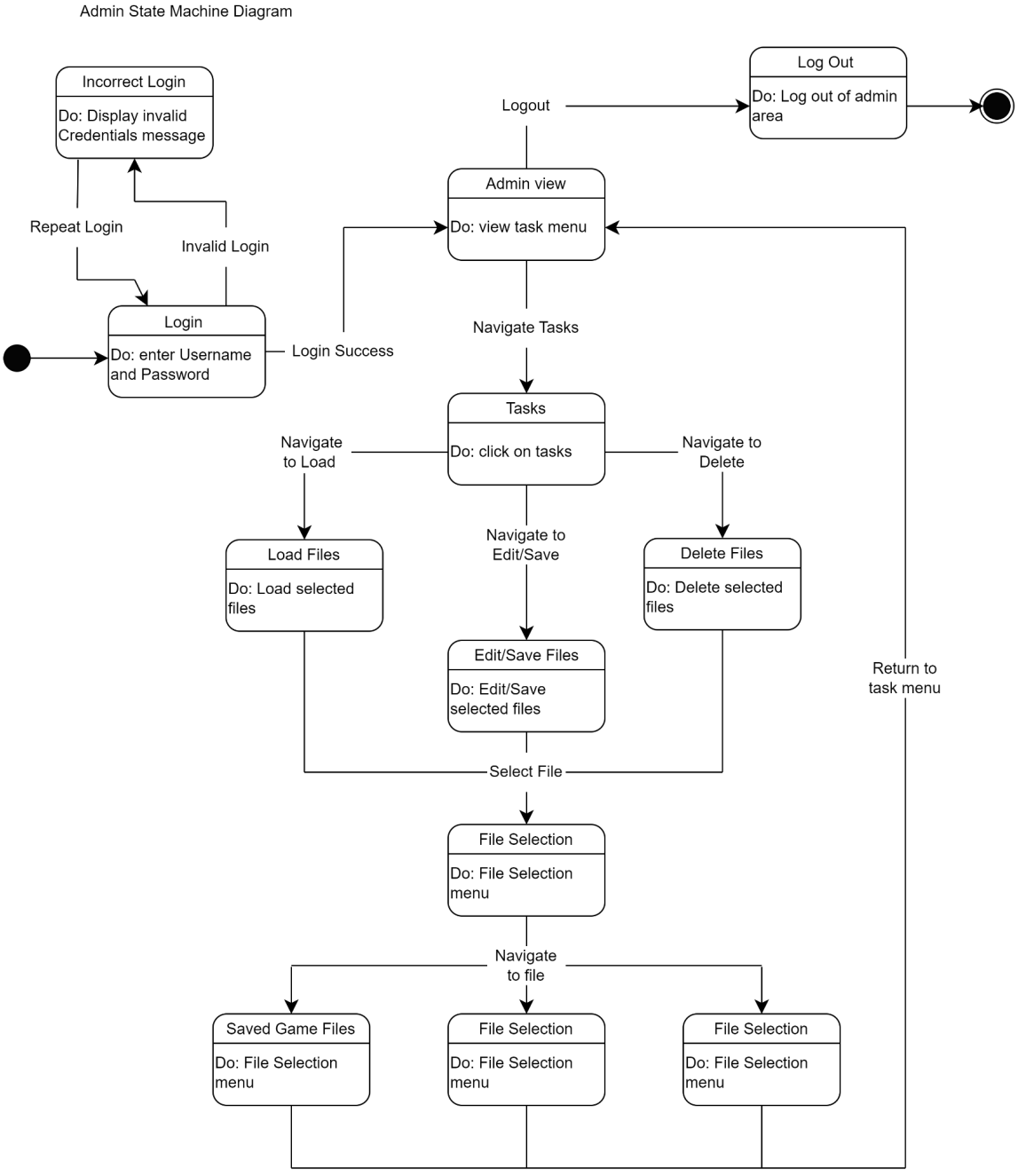


6.3. Software Design

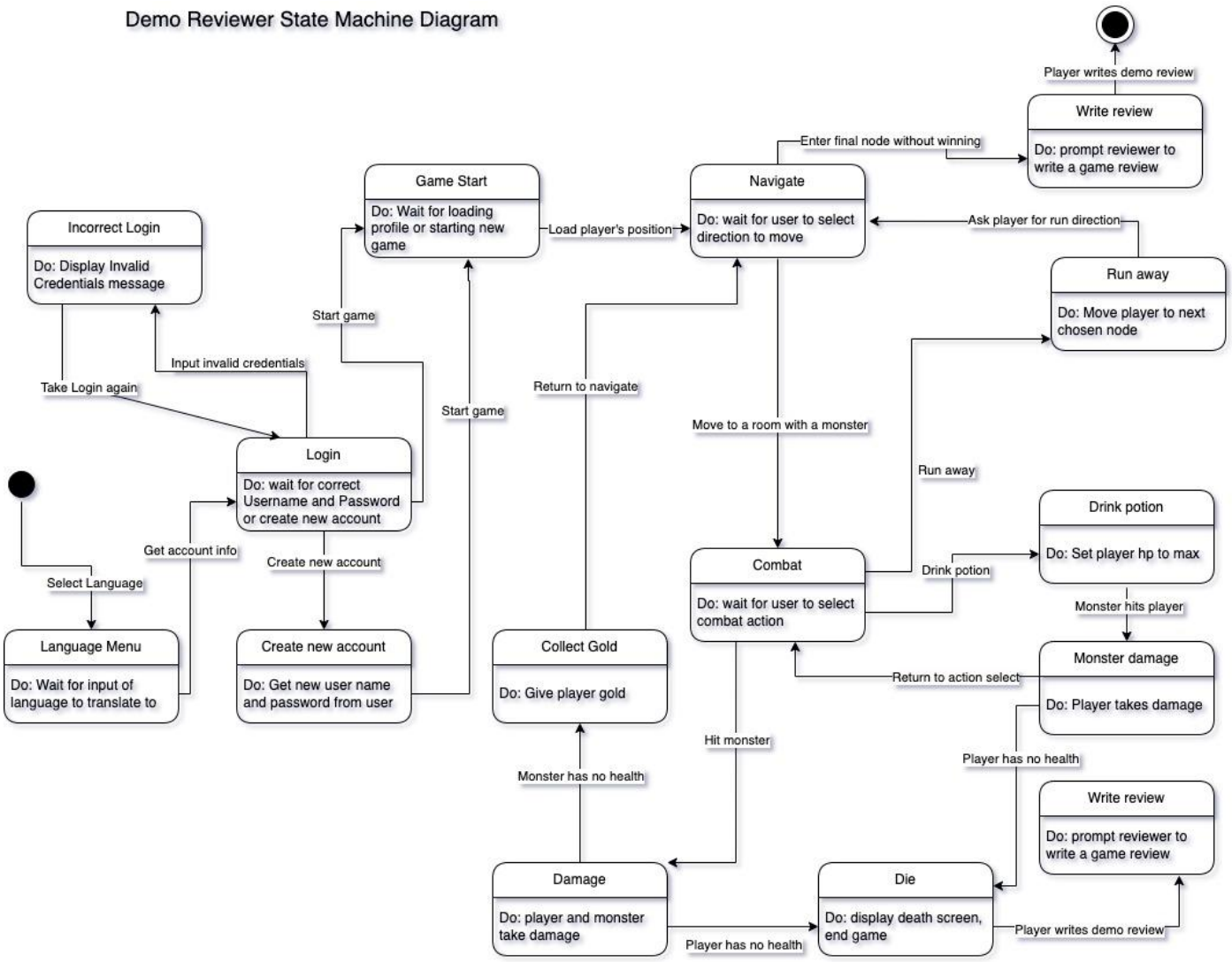
6.3.1. State Machine Diagram: Customer (Craig)



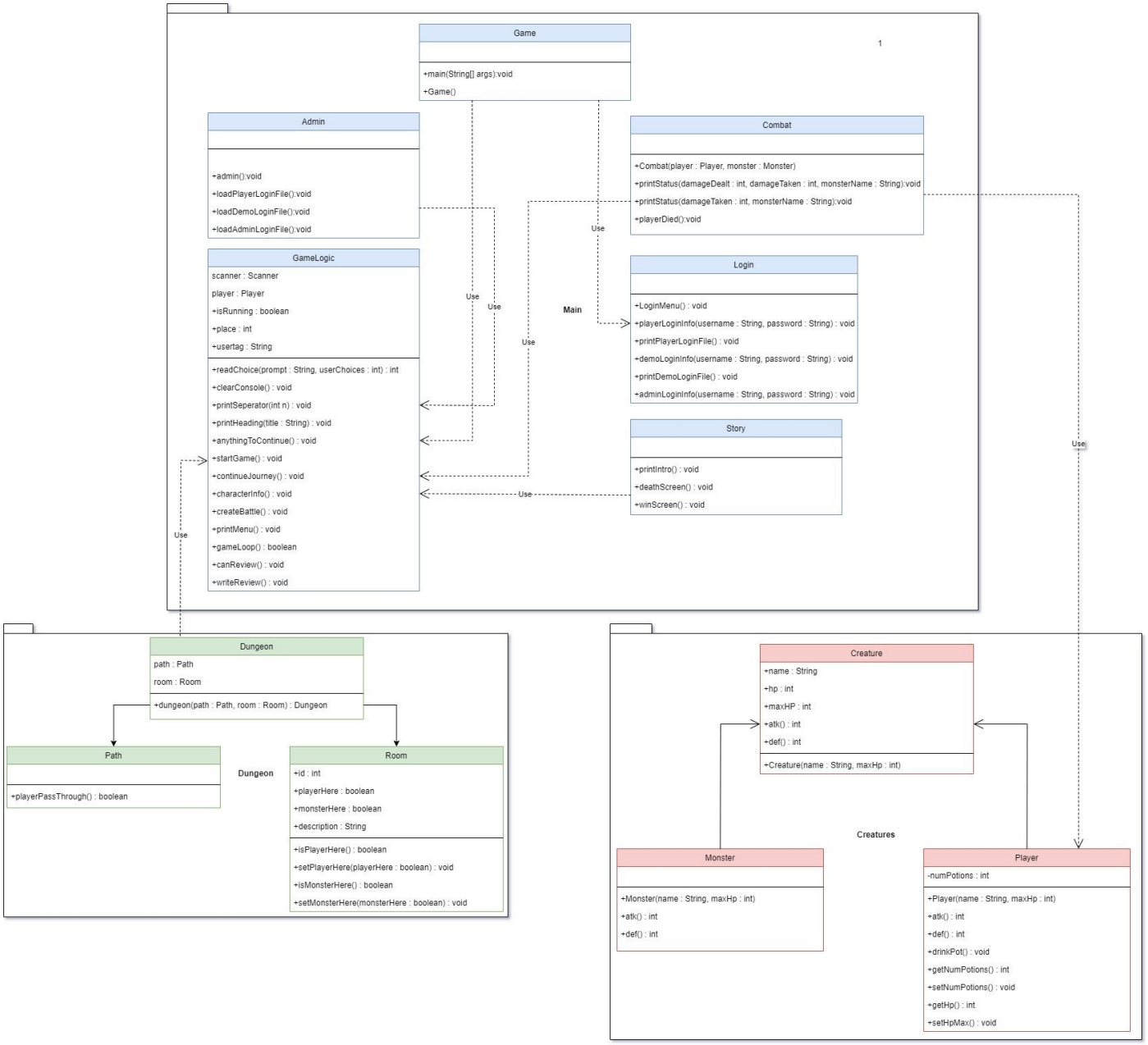
6.3.2. State Machine Diagram: Admin (Sam)



6.3.3. State Machine Diagram: Demo Reviewer (Hayden)



6.4. UML Class Diagram



7. Scenario

7.1. Brief Written Scenario with Screenshots