**Slay The Password  
Design Document**

**Week 3**

**CIS 3296 Section 03**

**Fall 2024**

**Team Members:**

- Jonathan Cherry

- Liye Guo

- Illia Matviienko-Hart

- Chiku Okechukwu

- Ryan M Tenerelli

- Stella Wang

**Repository URL:**

https://github.com/cis3296f24/project-03-slaythepassword

**Project Board URL:**

https://trello.com/b/O9q6tCuB/slay-the-password

**Table of Contents**

[**Document Overview 3**](#_mb0nqv7rdq95)

[**Architecture 3**](#_kzj898ajxqqp)

[**Week 3 update 5**](#_w1j3srp53hfl)

[**Detail Design API 5**](#_vv4oysyr8mbd)

[Game Class 5](#_uxo3vfx18zmt)

[Password Class 6](#_3474uznx712u)

[PasswordImpl Class 7](#_athzn162mn3r)

[User Interface 7](#_3h5j16v3c25q)

[UserImpl Class 8](#_j674899x6mzo)

[App class 8](#_a1wa8z7tcpkh)

[Primary Controller class 9](#_2yex705xu9wz)

[Secondary Controller class 9](#_1geh5iry5bal)

[**Vision Board 9**](#_oiu81edg5mmo)

[**Personas 10**](#_7bkb9csdjlzm)

[Marcus, the Good Parent. 10](#_hhwiyzurkjol)

[Patrick, the Inattentive Agent 10](#_skc2vqqpjh0)

[Sage, a stressed student 11](#_utq3m3idv5m4)

[Mandy, the gamer 11](#_hp1g68g899ai)

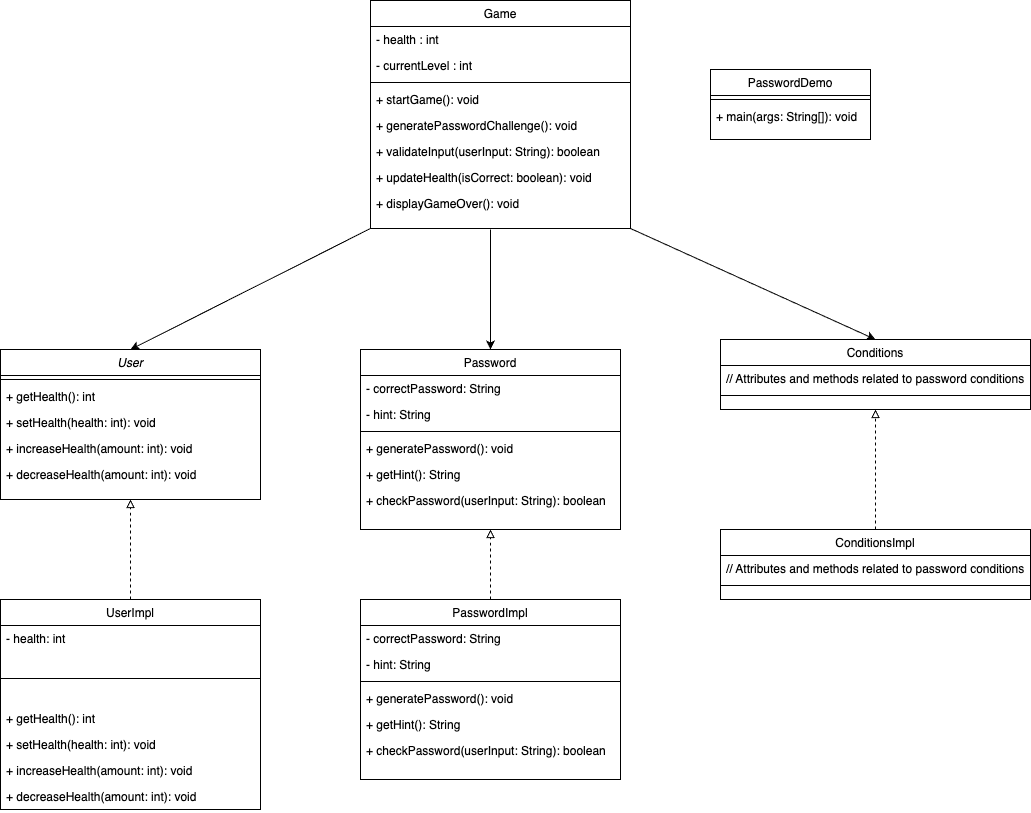
## **Document Overview**

This Design Document describes the software architecture and how the requirements are mapped into the design. This document will be a combination of diagrams and text that is describing what the diagrams are showing. The Design Document also specify the complete design of the software implementation using Javadoc.

## 

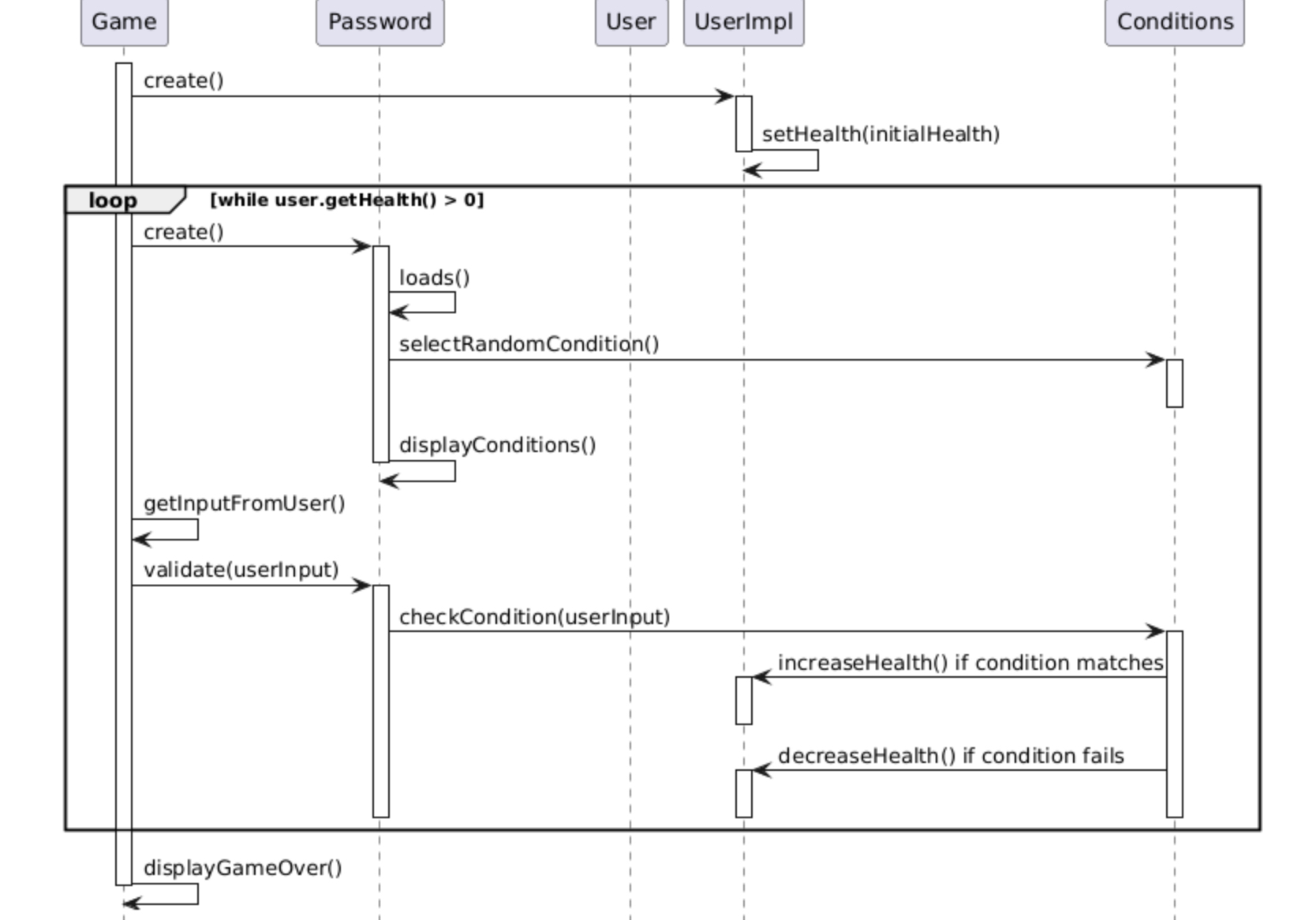
## **Architecture**

This section describes the different components and their interfaces using UML. For example: client, server, database. For each component provide class diagrams showing the classes to be developed (or used) and their relationship.



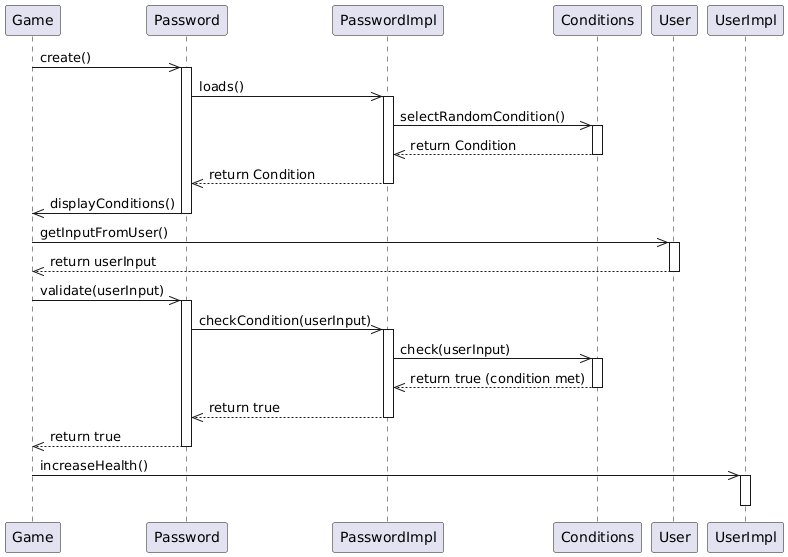
**Figure 1 UML Class Diagram for SlayThePassword**

This UML diagram represents the core structure and interactions of the "Slay the Password" game. The Game class orchestrates the gameplay, utilizing the User interface (implemented by UserImpl) to manage player health and the Password class (potentially implemented by PasswordImp) to generate and validate password challenges. The Game class controls the game flow, initializing the user's health, presenting password challenges, and validating user input. The Password class is responsible for generating passwords based on specific conditions, providing hints, and checking user-submitted passwords against the correct answer. The User interface defines methods for managing the player's health, which are implemented in the UserImpl class. Additionally, the Conditions interface and its potential implementation ConditionsImpl suggest a mechanism for defining and managing various password conditions or challenges within the game.



**Figure 2 UML Sequence Diagram for SlayThePassword**

This UML sequence diagram illustrates the dynamic interactions within the "Slay the Password" game. The Game class initiates the sequence, creating a UserImpl object to manage player health. The core game loop involves the Game class creating Password objects, which in turn utilize Conditions to generate password challenges. The Game class interacts with the user, collecting input and validating it against the Password object. Based on the validation result, the Conditions object modifies the player's health through the UserImpl object. This loop continues until the game ends, indicated by the player's health reaching zero.



**Figure 3 UML Sequence Diagram of Generating Password with conditions**

This UML sequence diagram shows the interactions between objects when a user enters a correct password in the "Slay the Password" game. The Game object creates a Password object and gets the user's input. The Password object, with the help of PasswordImpl and Conditions, checks if the input matches the required condition. Since the input is correct, the Game object increases the player's health using the UserImpl object.

## **Week 3 update**

**Goal for the sprint:** gather updates on their work progress and determine the next steps for the following week

**Tasks:**

* Set up java executable ***(S) (3 pts)***
* Continue work on JavaFX landing Page ***(M) (5 pts)***
* Implement Game Loop ***(M) (5 pts)***
* Develop User class ***(M) (5 pts)***  
  Continue updating the design document ***(S) (3 pts)***
* Continue work on conditions ***(M) (5 pts)***
* Continue Implementing Password Conditions ***(M) (5 pts)***

**Estimated Velocity** = (5 \* 5) + (2 \* 3) = **31  
Calculate effective velocity = TBD** (Effective velocity can only be calculated at the end of the sprint after all tasks have been completed and their actual story points are known.)

## **Detail Design API**

For each class define the data fields, methods.

· The purpose of the class.

· The purpose of each data field.

· The purpose of each method

· Pre-conditions if any.

· Post-conditions if any.

· Parameters and data types

· Return value and output variables

· Exceptions thrown\*.

This information should be in structured comments (e.g. Javadoc) in the source files. A documentation generation tool (e.g. Javadoc) may be used to generate the document as a draft.

### **Game Class**

**Purpose:** The Game class contains the main game loop and controls the overall flow of the "Slay the Password" game.

**Data Fields:**

None explicitly defined in the provided code.

**Methods:**

***main(String[] args) :***

**Purpose:** Starts and runs the game.

**Parameters:** id (int) - the account ID, initialBalance (int) - the initial balance.

**Parameters: args (String[])** - Command line arguments (not used).

**Return Value:** None.

**Exceptions Thrown:** InterruptedException - If the thread is interrupted during

### 

### **Password Class**

**Purpose:** The Password class represents a password challenge in the game. It handles password generation, providing hints, and validating user input.

**Data Fields:**

folder (String) - Stores the name of the folder containing password conditions.

randCondition (Conditions) - Holds the randomly selected password condition for the current challenge.

conditions (List&lt;Conditions>) - A list of available password conditions.

**Methods:**

***Password()***

· **Purpose:** Constructor for the Password class. Loads conditions and selects a random one.

· **Parameters:** None.

· **Return Value:** None.

***loads()***

· **Purpose:** Loads conditions from the specified folder.

· **Parameters**: None.

· **Return Value:** List&lt;Conditions> - A list of loaded conditions.

**v*alidate(String userInput)***

· **Purpose:** Validates the user's password input against the condition.

· **Parameters:** userInput (String) - The password entered by the user.

· **Return Value:** None.

***displayConditions()***

· **Purpose:** Displays the hint for the current password condition.

· **Parameters:** None.

· **Return Value:** None.

### **PasswordImpl Class**

**Purpose:** The PasswordImp class represents an implementation of a password with specific conditions. It is not used in the main game.

**Data Fields:**

difficulty (int) - The difficulty level of the password.

conditions (Conditions[]) - An array of conditions for the password.

**Methods:**

***PasswordImp(String conditions[])***

· **Purpose:** Constructor for the PasswordImp class.

· **Parameters:** conditions (String[]) - An array of conditions for the password.

· **Return Value:** None.

### **User Interface**

**Purpose:** The User interface defines methods for managing the user's health in the game.

**Data Fields:**

None

**Methods:**

***losehp(Integer value)***

· **Purpose:** Decreases the user's health by a specified value.

· **Parameters:** value (Integer) - The amount by which to decrease the health.

· **Return Value:** None.

***restorehp(Integer value)***

· **Purpose:** Restores the user's health by a specified value.

· **Parameters:** value (Integer) - The amount by which to restore the health.

· **Return Value:** None.

***getHealth()***

· **Purpose:** Returns the user's current health.

· **Parameters:** None.

· **Return Value:** int - The user's health.

### **UserImpl Class**

**Purpose:** The UserImpl class is an implementation of the User interface. It manages the user's health.

**Data Fields:**

health (int) - The user's current health.

maxHealth (int) - The user's maximum health.

**Methods:**

***UserImpl()***

· **Purpose:** Constructor for the UserImpl class. Initializes the user's health.

· **Parameters:** None.

· **Return Value:** None.

***losehp(Integer value)***

· **Purpose:** Decreases the user's health by a specified value.

· **Parameters:** value (Integer) - The amount by which to decrease the health.

· **Return Value:** None.

***restorehp(Integer value)***

· **Purpose:** Restores the user's health by a specified value.

· **Parameters:** value (Integer) - The amount by which to restore the health.

· **Return Value:** None.

***getHealth()***

· **Purpose:** Returns the user's current health.

· **Parameters:** None.

· **Return Value:** int - The user's health.

### **App class**

**Purpose:** The foundation of the User Interface, connects the backend to the frontend of javaFX

**Data Fields:**

scene (Scene)

**Methods:**

***start(Stage stage)***

· **Purpose:** Initialize a new scene, and display it to the user.

· **Parameters:** stage (Stage).

· **Return Value:** None.

***setRoot(String fxml)***

· **Purpose:** Set the page

· **Parameters:** fxml (String) - The page to be set to be loaded

· **Return Value:** fxml loader. - The current loaded page.

***loadFXML(String fxml)***

· **Purpose:** get required FXML resources

· **Parameters:** FXML (String) - The requested FXML file.

· **Return Value:** FXML loaded onto scene.

***main(String[] args)***

· **Purpose:** Launch the U.I

· **Parameters:** args (String[]) - arguments passed along the command line

· **Return Value:** none

### **Primary Controller class**

**Purpose:** The landing page of the application. Here, you will be able to look at the logo, continue old runs, start new runs, change settings, and leave the application.

**Data Fields:**

None

**Methods:**

***switchToSecondary()***

· **Purpose:** Change the scene from the homepage to the game screen

· **Parameters:** none.

· **Return Value:** None.

### 

### **Secondary Controller class**

**Purpose:** The Game. Here, you will be able to return to the homepage, save, exit the game, and input attempts.

**Data Fields:**

None

**Methods:**

***switchToPrimary()***

· **Purpose:** Change the scene from the Game to the homepage.

· **Parameters:** none.

· **Return Value:** None.

## Vision Board

For kids and teens who want a puzzling mind game, the game “Slay The Password” is a java-based software that challenges its users to climb through generated password puzzles and health management in order to succeed.

Unlike competitors, such as the Password Game, there are real consequences to incorrect guesses and perfection is rewarded. our application is smaller than pocket sized, any microprocessor can execute our game for the best flexibility & portability while having a roguelite experience.

## 

## 

## Personas

#### Marcus, the Good Parent.

Marcus, age 38, is a parent to an 8 year old son in Wilmington, Delaware. He is a stay-at-home father and husband to a traveling wife. Originally from San Bernandino, financial hardships and rising costs forced them to move across the country and for her to change careers and become a flight attendant. Since then, his two priorities in life have been raising their son and getting back the life they once had.

Marcus sees his son is struggling in art and English class, but cannot afford a tutor. While Marcus himself has never been strong in artistic or linguistic ability either, his teacher states it's a creativity issue. Determined to help his son become a more convergent thinker on a tight budget, he might decide to download Slay the Password being a free puzzle game that can be taken on any device (like a chromebook). While he cannot force his son to learn that way, he may find solace in his efforts, successful or otherwise.

#### Patrick, the Inattentive Agent

Patrick, 28, is a real estate agent in upstate New York. He is getting used to his career arranging for the buying and selling of houses. Patrick was raised by his mother, a doctor, and his father, a mechanical engineer. He graduated from Syracuse University with a degree in Finance. He was diagnosed with ADHD at a young age, which means he has difficulty concentrating on a variety of tasks, including his work.

Patrick has seen research that highlights video games’ abilities to improve peoples’ attention and focus. He is particularly interested in intellectually stimulating video games. He has faith that Slay the Password, a game that challenges people to use their wits, will help stimulate his mind and improve his concentration.

#### Sage, a stressed student

Sage is a 20-year-old college student in Philadelphia, PA, juggling a full course load with a part-time job to make ends meet. They're majoring in Computer Science and dream of becoming a software developer. Sage is constantly feeling stressed and overwhelmed by the demands of their academic and work life. They often find it difficult to focus on their studies due to mental fatigue and the constant pressure to succeed.

Sage is looking for healthy ways to de-stress and improve their cognitive abilities. They've heard about the benefits of brain-training games and puzzles for enhancing focus and concentration. They come across Slay the Password and are intrigued by the idea of a game that challenges their problem-solving skills in a fun and engaging way. Sage hopes that playing this game will not only provide a much-needed mental break but also sharpen their mind, potentially improving their performance in their demanding computer science courses.

#### Mandy, the gamer

Mandy, age 26, is a dedicated gamer and recent college graduate living in Portland, Oregon. She works part-time at a local café while searching for a full-time job in the game development industry. Growing up in a small town in, Mandy spent most of her free time immersed in video games, which sparked her dream of creating games that challenge and inspire players. She recently earned a degree in Interactive Media Design but struggles to break into the competitive field while balancing student loan payments and living expenses.

Mandy loves puzzle and rogue-like games for their ability to combine strategy and creativity, which is why Slay the Password immediately caught her attention. As a free game that runs on her budget-friendly laptop, it’s a perfect way for her to unwind while keeping her mind sharp. Beyond its entertainment value, Mandy appreciates the game’s focus on problem-solving and adaptability, skills she believes will help her in her in her everyday life. While navigating the challenges of early adulthood, Mandy finds comfort in games like Slay the Password that remind her of the joy and possibilities in creativity and perseverance.