CS102	**Spring 2021/22**	Project Group	**1I**
Instructor:	**Aynur Dayanık**		
Assistant:	**Mousa Azari**		

Criteria	TA/Grader	Instructor
Presentation		
Overall		

Group I

ATLA

Muhammad Shayan Usman 22101343, Zahaab Khawaja 22101038, Dilara Yilmaz 22101646, Özgür İkidağ 22102315, Mohammed Alhaidari 22101490

Requirements

(Version 2)

15/03/2022

1. Introduction

2 player card turn-based game based on musical gameplay. The user will have a basic profile e.g., cast/class, health, stamina, and shield. The player gets a set number of cards (8). They can use special attacks and the entire board can get affected. Cards can have offensive, defensive or gaining health/stamina. Each player can cast a spell or attack on others.

- Attack: Health damage
- Defensive (Shield generate)

We will also add multiplayer to allow players on the same network to play together. It will be turn-based so that the game revolves from player to player.

2. Details

2.1. Game classes

To create our game, we will be using the following classes (and maybe more if the need arises):

- Main game class - A class where we run methods of the game like play, stop, restart, user profile/stats.
- Table class - A class that keeps track of both player's hand and stats like **health**, **shield** and **mana(stamina)**.
- (Abstract class) Player class - A class that stores all the information about the player, including their stats and their cards.
    - 4 Different **instrument** classes - A subclass that inherits the Player class and creates a player with the instrument of *Piano*, *Drums*, *Saxophone* and *Harp*.
- Hand class - A class that creates a hand for each player based on the cards created by the Card class that are stored in the Deck class.
- (Abstract class) Deck class - A class that creates a deck from the cards that are available in the Card class
    - 1 Deck class / 4 different **instruments** - players can choose from the available instruments and gain different advantages once the **hero powers** are unlocked. These can be either getting an extra special card. Therefore, players can choose their instrument based on their play strategies.
- (Abstract class) Card class- A class creates a card object storing data like attack damage, mana used.
    - (Subclass of Card class) Different types of cards - Creates a card object with note cards (attack), healing cards or special cards that can have different functions. (For ex: change your whole deck, or a blank card that can be used for any note, or a card that basically skips your turn, +++.)

2.2. Game Progression

The player will start out with the lowest type (). The user will have to play for a certain period of time before unlocking the following type. You will start the game with the weakest class. The game will progress based on the **number** of games you play with every instrument (levels) as such: Drums → Piano → Harp → Saxophone. The user will also be able to unlock the **hero abilities** of the specified instrument by **winning** a certain number of games using that specific instrument. The user will have an account that will track the progress and allow for the benefits gained as the game progresses. The user will have a profile that will display his current progress and the progress left.

2.3. Game Audio

Since the game revolves around the element of music, the cards will play a tune specific to the instrument the player chose and card that is currently being played by the player. We will also add special audio when a player uses his/her hero power. For example, once the player unlocks the piano's **hero power.**

2.4. JAVA implementation of GUI

To create a GUI for our game, we will be using Oracle's AWT package. This package will provide us with classes that we can use to not only design the way our game looks but also get inputs like button clicks from the users that will allow us to make our game playable. We can design different parts of the game, like how it looks using the AWT classes[1]. By using the classes provided by AWT, we can also code the functionality of our game and use it to get inputs from the user. We can take these inputs and use them to control different game parts. For some elements of our game, we will be creating images and importing the images into our code; for example, we will be creating images of our cards on photoshop and putting those images into our code. We may do a similar thing for any buttons we may use in our game.

2.5. Multiplayer

We will be using the Server-To-Client network to implement the multiplayer part of the game. This will make use of the java.net package and use its Socket and ServerSocket classes. "A socket is defined as one endpoint of a two-way communication link between two programs running on the network. A socket is bound to a port number so that the TCP layer can identify the application that data is destined to be sent to." [2]

Each game will be played on a server and the player will wait for another player to join. After every game, the server will be refreshed. Since the game is turn-based, there will be one server and two clients. The server will be listening to the client. After the client makes the part, it will listen to the other user. The turn will be decided by the software.

2.6. Web Application

We plan to turn our game into a webapp by using the class Applet provided by Oracle[3]. By using inheritance, we will be able to run our code in the web browser while also using the GUI so that our game can be playable. The java Applet class will allow us to code our game very similar to how we would code it if we were to be creating the game offline. Once we have finished our code with the implementation of the java Applet class, we can use a simple HTML code to run the applet on a website[4]. We may also use CSS to allow us to design the website and make it look better for the users when they first open up the website.

[1] "java.awt (Java Platform SE7)." *Oracle Help Center*, https://docs.oracle.com/javase/7/docs/api/java/awt/package-summary.html. Accessed 7 March 2022.
[2] *What Is a Socket? (The Java™ Tutorials > Custom Networking > All About Sockets)*, https://docs.oracle.com/javase/tutorial/networking/sockets/definition.html. Accessed 7 March 2022.
[3] "Applet (Java Platform SE 7)." *Oracle Help Center*, https://docs.oracle.com/javase/7/docs/api/java/applet/Applet.html. Accessed 7 March 2022.
[4] Leahy, Paul. "Building Your First Java Applet." *ThoughtCo*, 1 October 2019, https://www.thoughtco.com/building-your-first-java-applet-2034332. Accessed 7 March 2022.

2.7. Database Storage

We plan to implement a database to our game that will store the clients' data such as but not limited to the name of the client, the client's statistics (i.e. games won and lost), and the user profile such as the most played instrument. We will be saving the data from the HTML form and connecting it to a database. [5]This data will help give a user profile that can be seen and edited before and after the game. We will probably choose to utilize some sort of open-source relational database management system. Once the information is stored, we can access the database and use it to showcase it on the main page.

2.8. Extras

We plan to implement some more extra features to the game that would make it even more interesting and avoid the player from getting bored. Some of them are:

- Notes will make different sounds depending on the instrument chosen (e.g. when the player presses the do of piano, it will sound different than that of the drums)
- Different backgrounds (Keys for the notes that would make players play on different fields.)
- Profile pictures

3. Summary and Conclusion

To conclude, we will be creating a strategy card game that has 4 instruments: *Piano*, *Drums*, *Saxophone* and *Harp*. Each player will choose one of these elements and get cards from their respective deck. Each player will have health, shield and mana and each card will have attack damage and a mana cost. We will use java's AWT package to design a GUI that will allow our game to be played. The game will be stored on a webapp using the java Applet class. We will be using multiplayer to allow players to connect to each other and play together. As previously mentioned, the game will also need to implement a database so that information about the users can be stored and later showcased in the main page. Player statistics will be displayed on the main page.

[5] Deshpande, Ashish. "How to Save Data from an HTML Form to a Database." *frevvo*, 1 September 2021, https://www.frevvo.com/blog/save-data-from-html-form-to-database/. Accessed 7 March 2022.