**Software Requirements and Design Document**

**For**

**Group 11**

Version 2.0

**Authors**:

Alexis Amoyo

Katelyn Fischer

Hannah Housand

Olivia Mei

Sophia Quinoa

# Overview (5 points)

Dreamscapes ™ is a visual novel game with an exciting and immersive storyline game that puts you in the driver's seat of a dreamscape narrative. As you progress through the dreams, you'll encounter unexpected moments where you must make decisions by answering questions with multiple choices that will directly impact the outcome of the story. This offers a personalized and unique experience every time you play and makes you “never want to wake up.”

The game encompasses 5 chapters featuring unique dreams complete with a diverse array of visual assets, backgrounds, and music. The user plays as the main character of each dream and will be able to save and load progress upon pausing the game. If the user has at least one bad ending by the final chapter, they will receive a bad ending in which the user will not wake up from the dream. If the user has achieved a good ending from each of the chapters, the user will wake up from the dream in the game’s true good ending.

# Functional Requirements (10 points)

Overall

1. The game shall have a playable character. (High)
2. The game shall consist of 5 interactive chapters. (High)
3. Each chapter should contain a unique story. (High)
4. The beginning of each chapter will be clearly marked by a chapter title screen. (Medium)
5. The game should have visual assets for characters and backgrounds throughout the entire story. (High)
6. The name of the main character should be taken from the user. (Medium)
7. If there is no name provided by the user, the default character will be “Nameless One”. (Low)
8. Game shall display text related to narration, a character’s thoughts, and a character’s dialogue. (High)
9. The themes of the game should be customized and altered from the default. This includes features like font, color, button styles, backgrounds, text boxes, etc. (Low)

Saves

1. There will be two types of saves in the game: checkpoint saves, and mid-chapter saves. (High)
2. The user shall be able to save their progress at any point in the game as a mid-chapter save. (High)
3. The system will have a single save slot for mid-chapter saves. (Medium)
4. After reaching a new chapter title screen, the system should unlock a checkpoint for that chapter. (High)
5. The user should be able to return to any saved checkpoint from the main menu screen. (Medium)
6. The user should be able to load progress made at their mid-chapter saved point. (Medium)
7. When a user has not reached a chapter’s title screen, they cannot access the chapter through the main menu. (Medium)
8. If the user exits the game, they will be prompted to save their progress. (Medium)

Endings

1. The user should be able to make choices that affect the storyline and ending of each chapter. (High)
2. If the user reaches a bad ending to a chapter, they can choose to replay the story or continue to the next chapter. (Medium)
3. The ending of the entire game is unlocked once the user achieves a “good” ending from each chapter. (Low)
4. The ending consists of a cutscene of the player waking up from their sleep. (Low)
5. A “good” or “bad” ending is defined by each individual chapter. (Medium)
6. If the user completes all 5 chapters and does not achieve all good endings, they will be prompted to go back and redo the failed chapters. (Low)

Additional Features

1. The game shall play music throughout its duration. (Low)
2. There shall be minigames throughout the game that affect the storyline. (Low)

Pause Screen

1. The pause screen shall be accessible at any point in the game. (High)
2. Pressing the escape key will open the pause screen during the game. (High)
3. The user will be able to access the main menu, exit, settings, and save options from the pause menu. (Medium)

Main Menu Screen

1. The main menu screen contains options to start the game from the beginning, resume from the save point, access saved checkpoints, and settings. (Medium)
2. The main menu screen should clearly indicate the name of the game. (High)
3. Button options on the menu screen should be clearly visible and easy to access. (High)

Chapter 1 Requirements: Cat Lab

1. This chapter contains 4 separate endings with 3 categorized as “bad” and 1 as “good”. (Low)
2. The chapter’s ending is determined by 3 different values: reputation, goodwill, and the experiment’s success. (Low)
3. This chapter will contain 5 different types of minigames. (Low)

Chapter 2 Requirements: Catective

1. This chapter must contain a prologue as well as 3 separate days of case investigation.
2. The chapter will contain 3 bad endings and 1 good ending.
3. The chapter will contain 3 different types minigames.
4. The chapter will contain between 3-5 decision trees throughout the story that may affect the events of the game.

Chapter 3 Requirements:

1. This chapter will have 3 different settings. (Medium)
2. This chapter will contain 2 different minigames. (Medium)
3. This chapter will have 4 different endings, 2 good, and 2 bad. (Medium)

Chapter 4 requirement:

1. This chapter will have six different endings. Depending on your choices,, the story will end early, or you will pursue the longest ending,
2. The user will have 9 decisions to make.
3. There will be two minigames

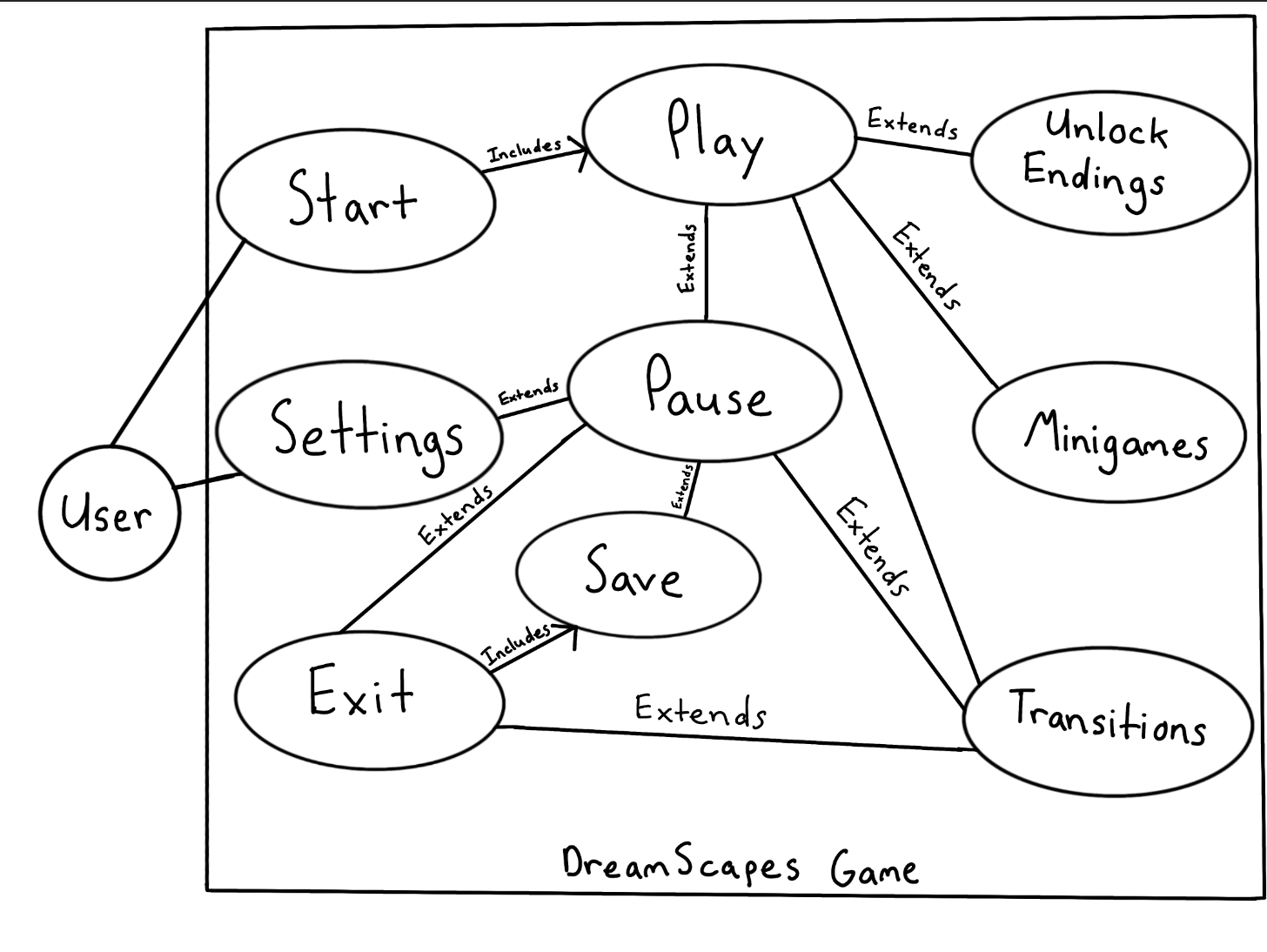
Chapter 5 Requirements:

1. This chapter will have more than one of each of the good and bad endings. (High)
2. This chapter, so far, will have 6 decisions to make, where some will only be seen depending on the path you are on. (Medium)
3. There will be at least one minigame. (Low)

# Non-functional Requirements (10 points)

1. The system should be able to run on both Windows and Mac operating systems.
2. Game should run within 30 seconds.
3. Game should be accessible offline.
4. The game will have minimal crashes, errors, or bugs
5. Game should run smoothly with no delays or lag in loading visual assets.
6. User interface should be intuitive and easy to navigate regardless of user experience.
7. User data shall be kept separate and private from other users. (?)
8. Loading saves should take a maximum of 10 seconds under normal circumstances.

# Use Case Diagram (10 points)

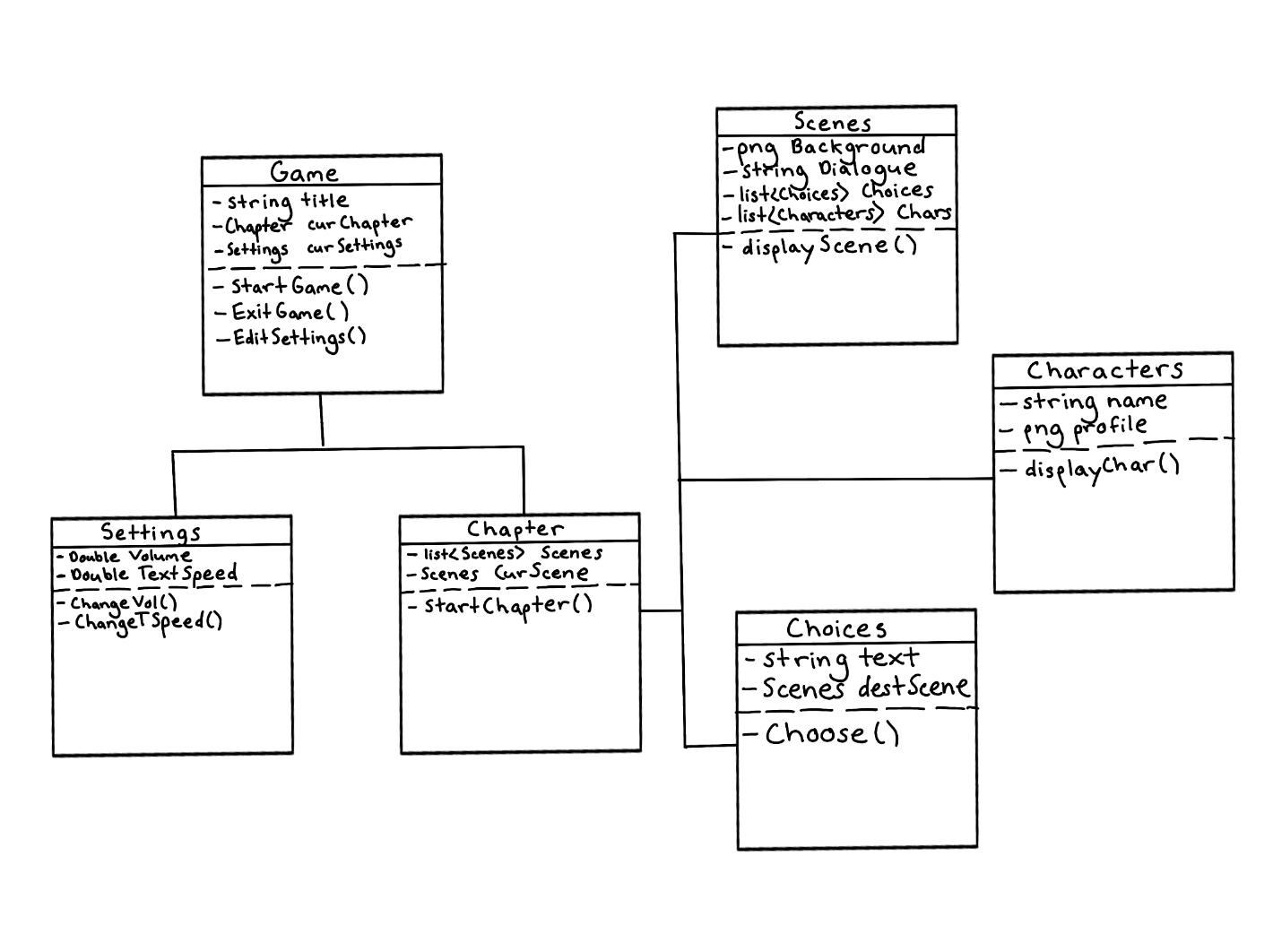


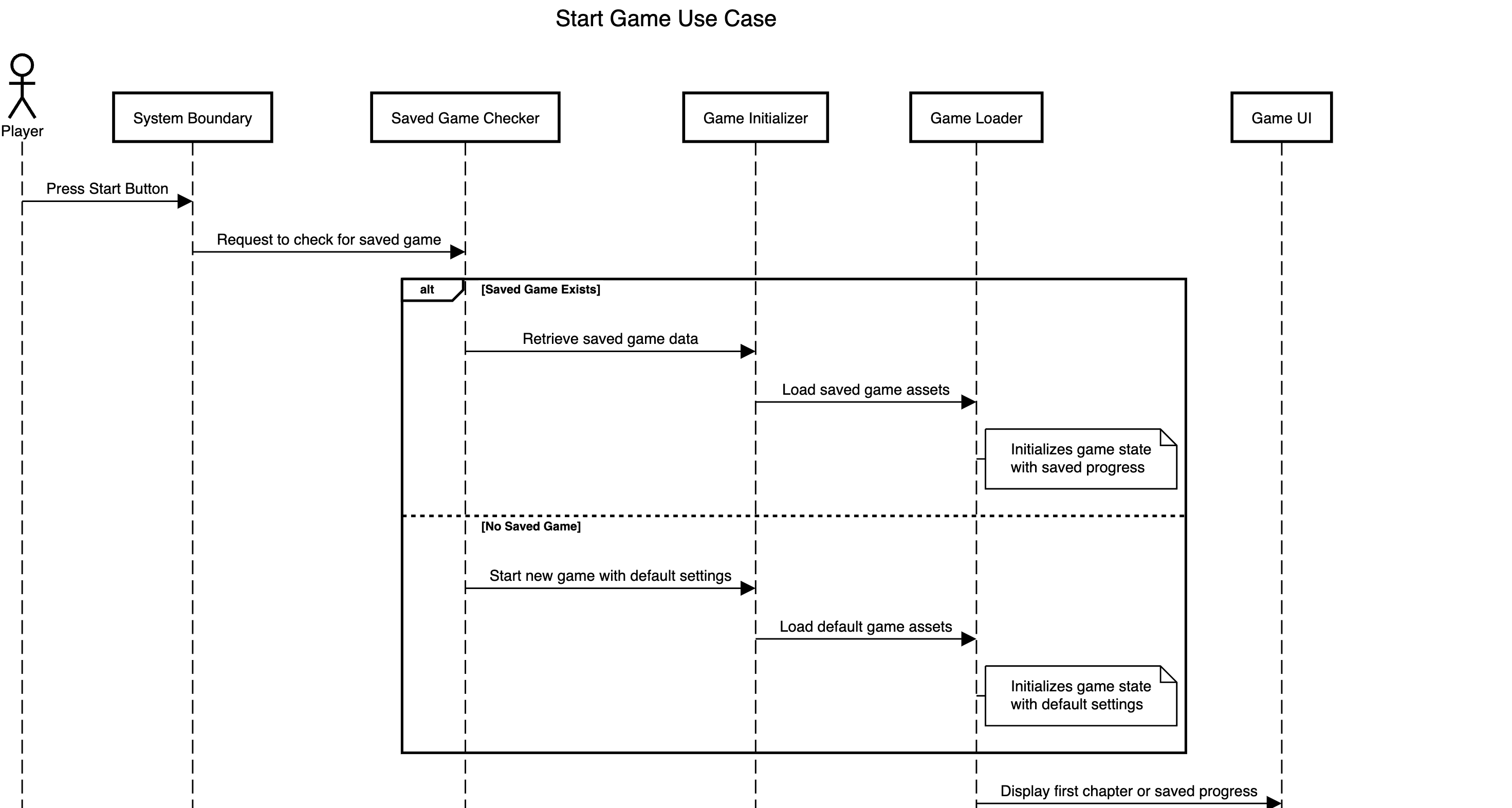
Textual Descriptions of Use Cases:

|  |  |
| --- | --- |
| Use Case Name: Transitions   * Actors: User * Description: The user will have the option to quit the game, or transition to a chapter, depending on the ending they received. * Preconditions: An ending must’ve been unlocked. * Postconditions: None * Flow of Events:   1. In the case of a good ending: The user will choose between quitting and going onto the next chapter.   2. In the case of a bad ending: The user will choose between quitting and replaying the current chapter. | Use Case Name: Unlock Endings   * Actors: User * Description: The user will reach a conclusion of a chapter after making a series of choices. * Preconditions: The user will have had to start the game and make several choices. * Postconditions: None. * Flow of Events:   1. In the case of a good ending: the system will transition back to play and the next chapter will play.   2. In the case of a bad ending: the system will transition to the transitions page and show a bad screen. |
| Use Case Name: Play   * Actors: User * Description: The user will be playing through the chapters. * Preconditions: The game must be started already. * Postconditions: The individual scenes and dialog will be added to the game’s history. * Flow of Events:   1. The player views the storyboards.   2. The player can choose to pause the game.   3. The player will encounter choices. | Use Case Name: Start   * Actors: User * Description: The user clicks the start game button and begins playing the game. * Preconditions: The project must be launched in Ren’py. * Postconditions: The user’s name will be saved. * Flow of Events:   1. User selects the Start Game option.   2. User is prompted to enter their name.   3. User starts playing Chapter 1. |
| Use Case Name: Settings   * Actors: User * Description: The user will be able to change different settings concerning gameplay. * Preconditions: The game must be launched. * Postconditions: The new system’s settings will be saved and applied to the game. * Flow of Events:   1. The user can adjust the different settings, including, text speed and volume.   2. The user will return to the game, or to the main menu. | Use Case Name: Pause   * Actors: User * Description: The user will be able to save their spot, go to settings, and exit the game. * Preconditions: The user must be currently playing the game. * Postconditions: None * Flow of Events:   1. The user must make a choice between Quit, Return, Save, and Start depending on the current system state and where the user would like the system to transition to. |
| Use Case Name: Minigames   * Actors: User * Description: The user will be playing a game through the Renpy system. * Preconditions: The user must be currently playing the game and reach the point where a minigame where will be launched. * Postconditions: None * Flow of Events:   1. The user will reach the point in a chapter and will come across a minigame.   2. The user will have different instructions based on the specific game.   3. The game will finish and the story will continue. | Use Case Name: Exit   * Actors: User * Description: The user will finish playing and the game will quit. * Preconditions: The game must be in a play state, a pause state, or the main menu. * Postconditions: The game will transition to the Save use case and the Ren’py quit actions will be completed. * Flow of Events:   1. The user will choose to quit the game.   2. The user will not see this, but the game will save and Renpy will quit the game. |
| Use Case Name: Save   * Actors: User * Description: The current game state will be saved. * Preconditions: The Save option or Quit option must have been selected. * Postconditions: Ren’py will possibly close the game. * Flow of Events:   1. The game will save. | |

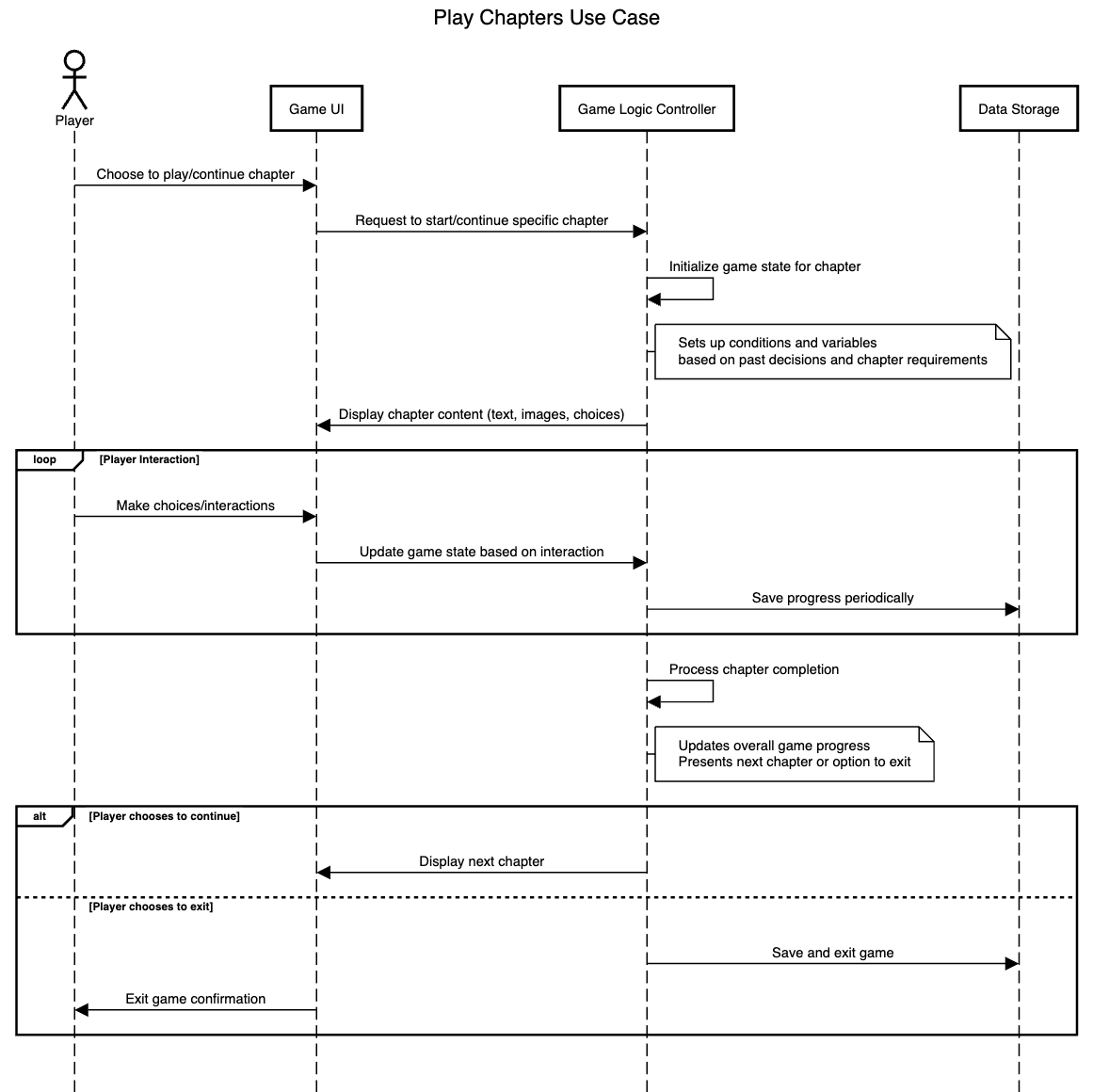
# Class Diagram and/or Sequence Diagrams (15 points)

Class Diagram:

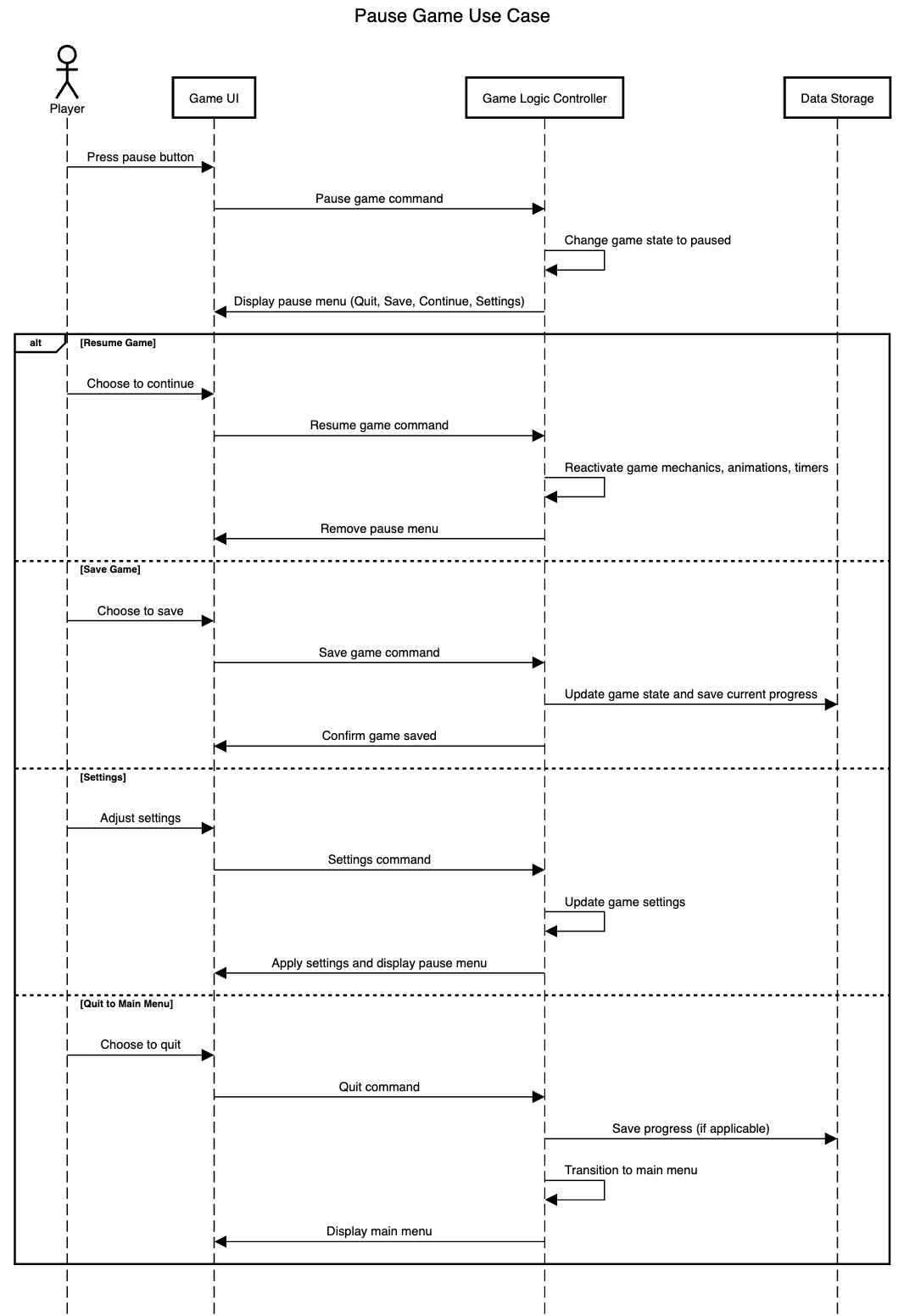
Start Game Use Case Sequence Diagram:



Play Chapters Use Case Sequence Diagram:



Pause Game Use Case Sequence Diagram:



# Operating Environment (5 points)

The software, developed on Ren’Py 8.2.0 (the latest official release), will operate on both Windows and MacOS.

On Windows, the software should be compatible with commonly used devices running Windows, such as desktops, laptops, and tablets. These devices may have different processors, such as Intel and AMD, and may have different storage and memory capacities. Ren’Py is able to run on Windows versions 7+ (Windows 7, Windows 8, Windows 8.1, Windows 10, Windows 11, etc.).

As for Mac OS, the software should be compatible across Apple devices such as MacBooks, Mac Pros, Mac Minis, etc. This should include Apple devices with different processors such as an Intel chip and the Apple silicon chip. The software is able to run on Mac OS X 10.10+ (MacOS Catalina, MacOS Mojave, MacOS High Sierra, etc.).

The software must be able to utilize visual assets and audio libraries necessary for rendering music. The software must also be compatible with input devices commonly used in both operating systems, such as keyboards and mice.

# Assumptions and Dependencies (5 points)

* Assuming that Ren’Py continues to release new updates, this may affect compatibility with past versions of Windows and MacOS operating systems.
* Ren’Py may be limited in its ability to create custom minigames that align with the game’s story given that it is a visual novel development platform.
* Libraries used to render music for the game depend on their continued compatibility with Ren’Py software.
* It is assumed that all creative assets, including art and music, are original or properly licensed for legal use.