Carmen Mosquera

CS 499 – 11429-M01

10 /16 /2024

Narrative : Artifact One

Artifact Description

The vampire hunter game is a text-based adventure game written in Python where players explore a small village to collect eight specific items needed to defeat a vampire living in a mansion. The enhanced version transformed the game into an interactive 2D adventure that incorporates various gameplay elements, including item collection, combat with the vampire, and a dynamic user interface with an interactive map, inventory system, location descriptions, item descriptions and player stats. This artifact originates from course IT 140 Introduction to scripting.

Inclusion Justification

I chose to include this artifact in my ePortfolio because it represents a comprehensive showcase of my software development abilities. This project required not only coding skills but also a good understanding of user interaction, game mechanics, and backend integration. The standout components include:

- Game Mechanics: Implementing game logic with Angular and Phaser 3 demonstrates my ability to integrate front-end technologies with game development.

- Backend Communication: The use of Flask for the backend highlights my skill in handling data and state management in real time through APIs.
- UI and UX Development: I created interactive UI components that improve the user experience by providing game instructions, inventory details, and prompts for player decisions.
- Real-time Data Handling: The dynamic update of player stats and the inventory system showcases my ability to manage and display game state information effectively.

Performed Enhancements List

1. Transformed the original text-based game in Python into a fully interactive 2D web-based game using Angular for the frontend and Phaser 3 for game development.

2. Modularized the backend by separating the Python code into smaller, manageable modules.

3. Improved code readability by adding inline and multiline comments to explain key parts of the code, making it easier to maintain and extend.

4. Implemented player movement, object collision detection using Phaser 3.

5. Enhanced the item collection system, where players can pick up and track the eight special items needed to defeat the vampire.

6. Managed game state with an Angular service (GameStateService) to ensure smooth communication between Angular and Phaser.

7. Created game completion logic, allowing players to fight the vampire after collecting all items, with the option to return to gather more items.

8. Designed and integrated a village map for player navigation and exploration.

9. Developed custom pop-ups for key events, including “item collected” and “item details,” using HTML, CSS, and Angular.

10. Added a custom pop-up dialog prompting players whether to continue fighting the vampire or return to collect more items.

11. Created a status bar that displays real-time information such as player's current location, health status, playing time and more.

12. Created an inventory system to display collected items in a responsive grid layout.

13. Added accessible buttons to allow users pause, resume and stop the game at any time.

14. Created visual effects during significant game actions, such as collecting items, adding to the game’s dynamism.

15. Added real-time effects and animations for actions like player movement, and battle scenes.

16. Designed custom visuals for in-game items and locations.

17. Added an interactive map of the village that shows key locations to help users navigate the game easily.

18. Added sound effects to the front-end game to enhance the player experience, providing audio feedback during key moments such as item collection, entering new areas, and battling the vampire.

19. Created a scene to display location details for every in-game location, improving player immersion and understanding.

20. Built a Flask-based backend API for managing game logic and player progress, ensuring the game state persists between sessions.

21. Handled CORS (Cross-Origin Resource Sharing) to secure and facilitate communication between the Angular frontend and Flask backend.

22. Implemented secure API communication between the frontend and backend for data consistency and player progress management.

23. Created a comprehensive README file with clear instructions on how to install, test, and run the game, making it more accessible to users and developers.

Course Outcomes.

The development and enhancement of the vampire hunter game helped me reach the following course outcomes:

Course Outcome 2: I designed, developed, and delivered professional-quality oral, written, and visual communications that are coherent, technically sound, and appropriately adapted to specific audiences and contexts by completing the following enhancements:

- I designed and implemented an interactive map of the villa that displays all the locations points, items, and access routes to provide clear navigation tools to both technical and non-technical users.

- I implemented a custom alert dialog (pop-ups) system that appear when players collect items, arrive to locations, or reach certain stages to provide real-time feedback to users, and help them track their progress and achievements.
- I created an Inventory scene that displays the player current inventory with clickable buttons that offer detailed information about the origin and importance of the collected items.

- To support both developers and users, I included gameplay instructions and technical details in a comprehensive README file, ensuring that installation steps, gameplay, and key technical features were well documented.

- I added extensive comments throughout the codebase, making it easier for future developers or collaborators to understand the game's logic and expand upon it.

Course Outcome 3: I designed and evaluated computing solutions that solve a given problem using algorithmic principles and computer science practices and standards appropriate to its solution, while managing the trade-offs involved in design choices by completing the following enhancements:

- I transitioned the text-based game into an interactive 2D game with a visually engaging user interface.

- I created a Front-end for the game using Angular and Phaser 3. Angular manages the game fronted architecture and Phaser 3 renders the game.

- I enhanced player movement and item collection by implementing player-object collisions that trigger different game actions and dynamic scene transitions. For example, when trying to walk through walls or house's fences.

- I created the "GameStateService" to manage state between Angular and Phaser. For example, once an item is collected from a location, the game state records the event, ensuring that if the player returns to the same location, the item cannot be collected again. This prevents item duplication and maintains a smooth, uninterrupted flow of gameplay.

Course Outcome 4: I used well-founded and innovative techniques, skills, and tools in computing practices for the purpose of implementing computer solutions that deliver value and accomplish industry-specific goals by completing the following enhancements:

- I refactored the game to separate the backend logic into small maintainable and scalable modular components such as:
    - The game.py that handles the core game logic,

    - The game_classes.py that handles the definitions for game entities such as characters, items, and locations.
    - The game_objects.py that handles the Instantiations of in-game objects (e.g., items, locations). And
    - The game_api.py that handles API endpoints for player actions, state saving, and backend communication.

- I used HTML and CSS to build the custom pop-ups and dialogues, replacing Angular Material components with more tailored solutions for better control and a cohesive game aesthetic.

- I Integrated sound effects during player movements, item collection, and the final battle to enhance the user experience, making the game more immersive.

- I enhanced the final battle by adding battle effects and animations such as bombs, explosives, punch, falling, and dying animations.

<u>Course Outcome 5</u>: I developed a security mindset that anticipates adversarial exploits in software architecture and designs to expose potential vulnerabilities, mitigate design flaws, and ensure privacy and enhanced security of data and resources by completing the following enhancements:

- I designed and developed the backend API using Flask, which provided a secure and lightweight framework for handling game-related requests.

- I configured CORS to block unauthorized access to the backend API.

- My implementation of the “gameStateService” to handle the game state and data within the application reduces the risk of unauthorized or unintended state changes. This adheres to best practices in securing game logic, ensuring the integrity of the player’s progress. These considerations demonstrate my anticipation of potential vulnerabilities, even in a game environment, reflecting a strong security mindset.