PRD - COMP 491/L
Team CAMZ Snake Game

1. Project Introduction

1.1 Project Name:

Explosive Snake Game - Our snake game is a Python based snake game with sprite elements.

1.2 Purposes and Uses:

The game that our team is developing has the goal of recreating the classic gameplay of snake that many players around the world are familiar with and being able to put a new and creative spin that keeps players engaged.

1.3 Target Audience:

- Video game players who are comfortable with playing video games.
- Family members and younger children who are new to gaming.
- Those who are testing and experimenting with our aspirational features.

---

2. Our Features

2.1 Required Features

There are a number of features that our application will need to be able to perform in order to ensure that we have recreated the classic snake game!

1. Main Menu / Menu Features

The main menu should feature basic UI elements that allow for players to navigate and play/view the specific parts of the game that they want to experience. The main menu should feature:

- Play Button
- Credits Button
- Quit/Return to Desktop

While navigating the menu, players should be able to always return back to the main menu screen and be given the option for a clean exit of the program with an “Quit” option.

Other aspects besides the main menu options are the level selector options which should be clear and concise to allow the user to know what exactly they are starting when they begin to play the game.

The Leaderboard and Credits menus should also be clear but require much less attention to properly execute.

2. Level Selectors and Gameplay

Upon entering the level selectors play should be given a number of playable options possibly including the options for

- Standard Snake Gameplay
- A Level Selector

With our game, we are currently planning implementing options that allow users to play premade levels with custom snake boards featuring many unique features that spice up the well known gameplay of snake.

- Obstacles or mines that will end the game if the snake touches them
- An enemy snake that could follow the player
- Fruits that move randomly around the board to make it harder for the snake to grab

3. Score Tracking

Across all different game modes and levels, a score will be kept tracked off. This score will be decided based on level difficulty. This unique score can be tracked and viewed on the leaderboard option on the main menu.

4. Credit

The credit section is a simple option that credits all Team CAMZ for the work on the project and can include our desired contact information for any need after the semester is finished.

2.2 Desired Features

If time and features are being properly and successfully implemented in subsequent beta tests there are a number of features that could make our game shine even brighter! Many of our features (both desired and aspiration) revolve around potential use of AI and how we can use that in development.

1. Snake Game Audio

Being a 2D Sprite Game, our game should feature basic sound design. While no members of our team are particularly experienced in sound design there are a number of places where it could be used to make the gameplay experience even better.

- Sound when eating an apple/fruit
- Sound when the snake touches the border or when the game ends

2. AI Created Boards:

We are hoping to be able to use AI elements to create some unique snake board that includes the elements of where things like the initial fruits are placed or spawn, where obstacles are placed or the unique shape of each board. With this we have several different ways of implementing these features but this method would mean creating the boards with AI during development and including them alongside our developer made levels. We also have the option of having AI learn how to play snake and work with or for the player to make changes.

2.3 Aspirational Features

While we would hope to be able to develop these features into our project there are both financial and time limitations that may limit the progress that we could make on these potential features.

1. Infinitely Generated AI Snake Boards:

Our plan for this feature would be to have an AI model worked into our game that could create as many snake levels as the player would want. These levels would be similar to the features seen in the

---

3. System Design

1. Frontend:

Our Frontend Development will be handled with both Python with visuals being made from a variety of sprite editors like aseprite.

2. Backend:

Our backend development will be done while working with the Ursula engine and Pythona. Future versions past our initial beta tests may be done while working in the GameMaker or GoDot engines. This decision may be determined later after initial requirements have been met and AI features could possibly be considered.

3. Database and Scoring Systems:

For this project, it doesn't have any major issues with managing or needing our users' personal data.The only basic data management that our software will need is user scores that will be displayed on the leaderboard.

4.UI Management:
This game will need to ensure that our UI and graphics are infringing any rights of any other game developers.

- Sprite Assets - Assets like the snake design etc.
- Any Audio Assets - Sounds like eating fruits etc.

---

4. Usability, Security, and Scalability

Part of developing a game like this project, we need to be able to ensure that there is an ease of access for beginners and players of all types. Snake is a well known game that should not and doesn't need to be a AAA blockbuster. If there were more time and resources dedicated to this project we would definitely need to keep that sort of aspect in consideration, however the simplicity of the game will naturally be part of its tram.

For our security and scalability concerns, considering that this is a game that won't gather and be needing much user information, the concern is quite low. If this game were to be developed with online leaderboard features then we would need to be able to manage the security of the scores to ensure our team's credibility. In terms of scalability, after version 1.0 is released (if development is going smoothly) there are a number of features that could be added to expand the project. It is also hopeful that if the project is successful that this game could potentially be released on an online game store (like itch.io).

---

5. Meet our Team and our Tools

1.Team Members:

- Adam Zuniga
- Adrian Carreno
- Emily Alvarez
- Hassanudeen Muhammad

2. Pre-Development Tools:

Our team has worked on diagramed and preparing our project to ensure that we can proceed into development smoothly, for our diagraming we have worked with:

- LucidChart - Work with UML Diagramming
- Draw.io - Work with the architecture design
- Canva - Work with our application / UI design
- Google Docs - For our documenting
- Github - Organization of assignments and resources

3. Development Tools:

As we proceed into development we will be mainly working with the following tools and environments:

- VSCode - For general production
- FireBase- For database management and Hosting
- OnlineGBD - Online debugger and Compiler
- Other Online/Cooperative IDEs
- Github - To share and manage resources
- Draw.io / LucidChart - For further visual representation of future elements of the project
- GameMaker Studio
- Aseprite
- PiskelApp
- Ursina Engine

---

5. Conclusion

Our team is looking forward to having a successful semester and completing a game that is enjoyable and one that we can be proud of!

6. AI Integration (Extra)

1.AI and Open AI Integration:
OpenAI GPT API:

- GPT can generate obstacle layouts based on predefined prompts.
- AI can ensure a playable difficulty by balancing obstacle density and placement.
- It can suggest new game mechanics based on previous player behavior.

1.AI Implementation for Obstacle Placement:
Prompt Engineering:
Example:

"Generate a list of obstacle coordinates for a Snake game on a 1920x1080 grid. Ensure that obstacles do not block the initial path from (0,0) and leave space around the edges."

Dynamic Obstacle Adjustment:

- AI can analyze player performance (e.g., time survived, score) and adjust difficulty dynamically.
- Obstacle Pattern Generation:
- GPT can generate specific obstacle patterns like mazes, clusters, or safe zones.