**CSE 310 – Applied Programming**

**Module Plan**

Name:Gage Strong

Date:9/18/24

Module # (1-3):1

1. Identify which module you have selected to work on. Place an “X” in front of your selected module.

Cloud Databases

Data Analysis

x Game Framework

GIS Mapping

Mobile App

Networking

SQL Relational Databases

Web Apps

Language – C++

Language – Java

Language – Kotlin

Language – R

Language – Erlang

Language – JavaScript

Language – C#

Language - TypeScript

Language – Rust

1. At a high level, describe the software you plan to create that will fulfill the requirements of this module. Describe how each requirement will be met. This may change as you learn more about the technology or language you are learning.

2. High-Level Description of Software:

The software will be a Halloween-inspired Flappy Bird-style game developed using Python and the Arcade library. The game will feature a Halloween-themed character (such as a flying bat or ghost) navigating through obstacles (like pumpkins or haunted trees). The primary goal will be to guide the character through gaps in the obstacles while avoiding collisions. The game will include the following components to meet the module requirements:

Graphics Display: Halloween-themed graphics, including the player character, background, and obstacles.

User Input: The player will control the character using the keyboard (space bar to fly up, gravity to pull down).

Moveable Objects: The character and the obstacles will move on the screen; obstacles will scroll from right to left.

Additional Feature (Sound):

Sound Effects: Include spooky sound effects (like howling wind or eerie laughter) when the character flies, hits an obstacle, or passes through gaps. Background music will enhance the Halloween atmosphere.

1. Create a detailed schedule using the table below to complete your selected module during this Sprint. Include the task and duration for each day. You are expected to spend 24 hours every Sprint working on this individual module and other activities in the course. Time spent on this individual module should be **at least** 12 hours.

|  |  |  |
| --- | --- | --- |
|  | **First Week of Sprint** | **Second Week of Sprint** |
| **Monday** | Set up environment: install Arcade, create game window (2 hours) | Add background music and improve sound effects 2 hours |
| **Tuesday** | Create player character and obstacles, set up sprites for both 3 Hours | Increase game difficulty by speeding up obstacles over time 2 hours |
| **Wednesday** | Implement character movement (space bar) and gravity logic 2 hours | Add a start menu, pause feature, and game over screen 2.5 hours |
| **Thursday** | Implement obstacle scrolling and collision detection 2 hours | Implement saving and loading of high scores 2 hours |
| **Friday** | Add scoring system (points for passing obstacles) and sound effects 2 hours | Final round of testing and debugging 2 hours |
| **Saturday** | Test gameplay, refine movement and collision logic 2 hours | Prepare and submit the final project. Have others test it and see if there is anything that can be changed. 2 hours |

Total 13 hours Total 12.5 hours

1. Identify at least two risks that you feel will make it difficult to succeed in this module. Identify an action plan to overcome each of these risks.

Risk: Managing collision detection for moving objects (character and obstacles).

Action Plan: Utilize Arcade's built-in collision detection functions. Start simple, detecting collisions between just the player and the first obstacle. Test frequently and refine.

Risk: Time management and balancing workload with other tasks.

Action Plan: Stick to the daily schedule. Tackle harder features like collision detection and scoring early in the week to leave time for testing and polishing later on. Avoid adding unnecessary features during the development process until the core game is complete.