Group 33
CSE 110
10/13/25

Design Mockup and Requirements

Game Description:
https://docs.google.com/document/d/1Lw6ZTf0Ohza4GveAnaxD1OPOQvcWcGmiCdAaPMrSsgo/edit?tab=t.0

UI Design:
https://www.figma.com/design/pDztelgauxuoHoVohitlK1/Design-Mockup-and-Requirements?node-id=0-1&t=z38l1Gy2APyFY2ZT-1

Main Menu

User Story: As a user, I want a main menu so I can easily start, continue, or quit the game.

- Tasks:
  - Implement Main Menu UI with Start Game, Load Game, Exit, and Settings button.
  - Implement Exit button logic to trigger a confirmation pop-up.
  - Implement the confirmation logic to exit the program when confirmed.
- Tests:
  - Main menu screen displays all four required buttons/icons clearly.
  - Clicking “Exit” shows a pop-up. Clicking “Cancel” returns to the Main Menu.
  - Clicking “Exit” then “Continue” closes the game process entirely.

User Story: As a user, I want to start a new game to begin my adventure from the first level.

- Tasks:
  - Implement logic to create a new save file initialized to Level 1, 0 coins, and empty inventory.
- Tests:
  - A new, unique save file is created in storage upon clicking “Start Game” with the correct initial data (Level: 1, Coins: 0, Inventory: Empty).

User Story: As a user, I want to load a game so I can continue my progress from a previous session.

- Tasks:
  - Implement Save File Select UI to list all found save files.
  - Implement functionality for the user to select a save file (placeholder state for now).
  - Implement Back to Main Menu button on the Save File Select screen.
- Tests:
  - The screen displays a list of all existing save files.

    - Clicking a listed save file transitions to a “Game State Loaded” placeholder screen.
    - Clicking “Back to Main Menu” from the Save File Select screen returns to the Main Menu.

Settings

User Story: As a user, I want to access the Settings from the main menu so I can adjust the game experience before playing.

- Tasks:
    - Implement the Settings Page UI with placeholder options (buttons/sliders) for basic features (e.g., volume, difficulty).
    - Implement a Back to Main Menu button on the Settings screen.
- Tests:
    - Clicking the Settings button/icon opens the Settings Page with placeholder options visible.
    - Clicking “Back to Main Menu” from the Settings Page returns to the Main Menu.

User Story: As a user, I want to be able to pause the game at any time to adjust settings or take a break.

- Tasks:
    - Implement a Pause Key trigger (e.g., 'Esc' key) to freeze game time and overlay the Pause Menu UI.
    - Implement functionality to resume game (hide overlay, unfreeze game) by pressing the Pause Key or clicking Return to Game.
    - Implement Settings access from the Pause Menu (retains game pause state).
- Tests:
    - Pressing the Pause Key again or clicking “Return to Game” closes the menu and resumes normal game operation.
    - Clicking “Settings” from the Pause Menu opens the Settings screen while the game remains paused in the background.

Core Gameplay

User Story: As a user, I want to be able to interact with puzzle elements through the mouse so I can input answers and manipulate objects.

- Tasks:
    - Implement Text/Number Input Field component (for Level 1/3) that exposes its current value for logic checks.
    - Implement Character Filters for input fields (e.g., only numbers, decimals, or trig functions).
- Tests:
    - A dummy input field displays the value entered by the user (e.g., a number).

    - Only allowed characters (e.g., ‘0-9’, ‘.’, ‘-’) can be entered into the field; all others are ignored.

User Story: As a user, I want to interact with puzzle elements through the mouse so I can input answers and manipulate objects.

- Tasks:
    - Implement Draggable Game Element component (for Level 2/4) that can be positioned by clicking and dragging.
    - Implement logic for the draggable object to expose its current position and trigger events based on placement.
- Tests:
    - An object can be clicked and smoothly moved across the screen, following the mouse.
    - A dummy draggable object changes color/state when placed within a specific, predefined target area on the screen.

User Story: As a user, I want to be able to move to the next room/level once I have solved the current puzzle.

- Tasks:
    - Implement a Level-Switching Mechanism (Unload current level, Load next level).
    - Implement the trigger for level transition upon puzzle completion (success state).
- Tests:
    - Triggering a game element in Dummy Level 1 causes the game to switch to Dummy Level 2 with a visible indicator of the change.
    - Solving the puzzle in a dummy level successfully triggers the level-switching mechanism.

User Story: As a user, I want items I collect in my inventory to remain with me between levels so I don’t lose my progress.

- Tasks:
    - Implement a level-agnostic Inventory System that is loaded from and saved to the current save file.
    - Implement logic to update the savefile with the inventory state when transitioning levels.
- Tests:
    - An item collected in Dummy Level 1 remains in the inventory when Dummy Level 2 is loaded.
    - After collecting an item and moving to the next level, exiting the game and loading the save file again shows the item still in the inventory.

Puzzle Logic & Feedback

User Story: As a user, I want to be able to solve a puzzle so I can progress in the game.

- Tasks:

    - Implement Answer Check Logic to determine if the user's input/object placement is correct.
    - Implement Correct Answer flow: Reward a Token (Crystal) and proceed.
    - Implement Incorrect Answer flow: Display a Retry/Try Again Pop-up and change the puzzle's numerical values (randomize).
- Tests:
    - Inputting the correct answer (or placing the object correctly) triggers the 'Correct' path; incorrect triggers the 'Wrong' path.
    - Solving a puzzle correctly awards a Crystal to the inventory.
    - Submitting an incorrect answer displays a pop-up, and the numbers/values in the question are instantly randomized/updated.

User Story: As a user, I want to be able to earn a token when a puzzle is solved so I can use it later in the final level.

- Tasks:
    - Implement logic to reward a token (Crystal) when a puzzle is solved correctly.
    - Implement logic to collect coins (100) on successful level exit (e.g., Level 1, 2, 3).
- Tests:
    - After solving any level's main puzzle, a new Crystal is added to the user's permanent inventory.
    - After completing Level 1, the player's coin count increases by 100.

Save, Quit, & Load

User Story: As a user, I want to be able to quit and save the game anytime so I don't lose my current progress.

- Tasks:
    - Implement Quit and Save button functionality (accessible via the Pause/Settings menu).
    - Implement logic to update the current save file with the user's current level, completed status, and current inventory (including tokens/coins).
- Tests:
    - The "Quit and Save" button is accessible and clickable from the in-game Pause Menu.
    - A "Quit and Save" action updates the save file. Upon reloading, the player is at the correct level, with the correct inventory, and the correct level completion status.

User Story: As a user, after selecting Load Game, I want to see which levels are unlocked so I can jump to a previously completed stage.

- Tasks:
    - Implement functionality on the Load Game screen to display four distinct level buttons/slots.

    - Implement logic to lock levels that have not been completed in the selected save file.
    - Implement logic to load unlocked levels with the inventory restored from the save file.
- Tests:
    - The level selection screen shows four level slots (Level 1, 2, 3, 4).
    - If a save file has only completed Level 2, Level 3 and 4 buttons are disabled/grayed out and cannot be clicked.
    - Selecting an unlocked level loads the level and restores the inventory saved at the time of that level's completion.

Minigames & End Game

User Story: As a user, I want to be able to access the minigames so I can practice and earn a hint.

- Tasks:
    - Implement the Minigame UI with a rotatable dial/cursor and a display for target degrees/radians.
    - Implement Minigame logic: Matching the dial to the target radian/degree to complete the puzzle.
    - Implement Minigame reward: Award a hint and 50 coins.
- Tests:
    - The minigame screen loads with a unit circle graphic and a movable dial component.
    - Successfully setting the dial to the correct radian for a given degree (or vice-versa) triggers the success state.
    - Completing the minigame adds 50 coins to the inventory and gives a new item marked as a "Hint."

User Story: As a user, I want to complete/win the game after unlocking all of the levels so I can earn the final artifact.

- Tasks:
    - Implement Win Condition trigger (successful crystal placement in Level 4).
    - Implement Game Completion Rewards: Automatically grant 500 coins and the Ancient Artifact to the inventory.
- Tests:
    - Correctly placing all collected crystals on the Zodiac Circle triggers the Game Completion screen.
    - The user's coin count is increased by 500, and the Ancient Artifact is added to their inventory.

User Story: As a user, I want to replay the game after completing the entire game so I can try the new problems while retaining my artifacts.

- Tasks:

    - Implement a Play Again button on the Game Completion screen.
    - Implement Game Refresh Logic: Start a new save file for Level 1, refreshing the inventory (except coins/artifacts), and randomizing new trig problems.
- Tests:
    - The “Play Again” button is visible and clickable on the final win screen.
    - Clicking “Play Again” starts a new game at Level 1. The inventory is empty *except* for the coins and Ancient Artifact from the previous run, and the puzzle values in Level 1 are different from the first playthrough.