Name\_\_\_\_\_\_Sohan Lama\_\_\_\_\_\_\_ Mark \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_/50

[**Instructions**: Remove everything that is not a heading below and fill in with your own diagrams, etc.]

## Brief introduction \_\_/3

In the "Kill Mo' Chickens" tower defense game, I will be developing the Main Menu and UI system, which plays a critical role in ensuring smooth player interaction and navigation. Creating and executing the main menu, pause menu, and tower selection menu are among my responsibilities in order to provide users with an easy-to-use interface. In order to ensure that players can easily access important game data while playing, I also supervise the primary UI elements, such as the money meter, health counter, and level progression indicator. I want to make the interface less of a barrier for players and enhance the gaming flow so they can focus on making strategic decisions. My efforts go toward making the game playable and entertaining, which results in a polished end product.

## Use case diagram with scenario \_\_14

### Use Case Diagrams

A diagram of a game

Description automatically generated

### Scenarios

**[Scenario 1: - Manage Game Flow]**

**Name:** Manage Game Flow

**Summary:** The player controls game transitions—starting the game, pausing it, saving progress, and exiting the game.

**Actors:** Player

**Preconditions:** The game is at the main menu or paused.

**Basic sequence:**

**Step 1:** The player selects "Start Game" from the main menu.

**Step 2:** If there’s a saved game, the player can choose to load it or start fresh.

**Step 3:** During gameplay, the player can pause the game to save or exit.

**Step 4:** If the player chooses to exit, they’re asked to confirm.

**Exceptions:**

**Step 1:** If the game files are missing or corrupted, an error is displayed.

**Step 2:** The player is prompted to save if there’s unsaved progress before exiting.

**Post conditions:** The game either starts, pauses, saves, or exits based on the player's actions.

**Priority:** 1 (must have).

**ID:** MGF1.

**[Scenario 2: - Tower Management]**

**Name:** Tower Management

**Summary:** The player selects and places towers on the map during gameplay.

**Actors:** Player.

**Preconditions:** The player is in an active game session.

**Basic sequence:**

**Step 1:** The player opens the tower selection menu.

**Step 2:** The player chooses a tower and selects where to place it.

**Step 3:** The system checks if there are enough resources and if the position is valid.

**Step 4:** The tower is placed, and resources are deducted.

**Exceptions:**

**Step 1:** Not enough resources? An error message is shown.

**Step 2:** Invalid position for tower placement? Error message again.

**Post conditions:** The tower is placed, and the game continues.

**Priority:** 1 (must have).

**ID:** TM1.

**[Scenario 3: - Adjust Game Settings]**

**Name:** Add Game Settings

**Summary:** The player can change settings like audio, difficulty, and controls.

**Actors:** Player.

**Preconditions:** The game is paused or in the main menu.

**Basic sequence:**

**Step 1:** The player opens the settings menu.

**Step 2:** Changes are made to audio, controls, or other settings.

**Step 3:** The player saves the changes and returns to the game.

**Exceptions:**

**Step 1:** If changes aren’t saved, the player is asked to confirm before exiting the settings.

**Post conditions:** Settings are adjusted, and the player continues the game or returns to the main menu.

**Priority:** 2 (essential)

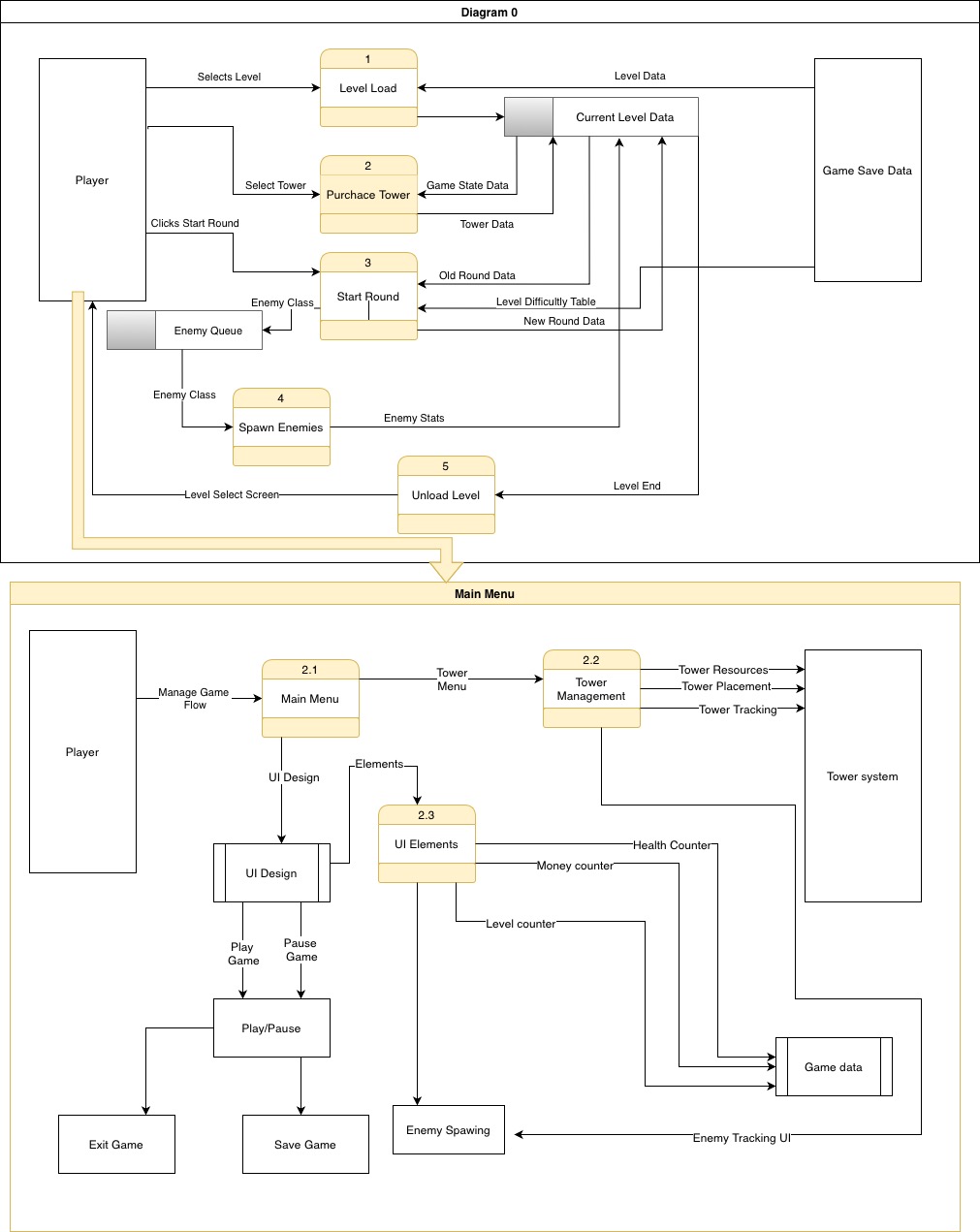
**ID:** AS1

## Data Flow diagram(s) from Level 0 to process description for your feature \_\_\_\_\_\_\_14

[Get the Level 0 from your team. Highlight the path to your feature]

Example:

### Data Flow Diagrams



### Process Descriptions

Assign rooms\*:

WHILE teacher in two places at once OR two classes in the same room

Randomly redistribute classes

END WHILE

**\*Notes**: Yours should be much longer. You could use a decision tree or decision table instead if it is more appropriate.

**Main Menu:**

1. **Start Game**:
   * IF player selects "Start Game":
     + Load tower selection UI
     + Initialize UI Elements (money, health, level counter)
     + Transition to Tower Management
     + Spawn initial enemies based on the difficulty level
   * ELSE IF player selects "Load Game":
     + Retrieve previous game state from game save data
     + Load player progress (resources, health, towers)
     + Load previously saved level and spawn enemies accordingly
   * END IF.
2. **Tower Management**:
   * IF player selects a tower:
     + Display available towers and corresponding costs
     + Deduct resources if the player selects to buy a tower
     + Place tower in available zones
     + Track tower health and upgrades
   * ELSE IF player selects to upgrade:
     + Check for enough resources
     + Upgrade tower abilities
     + Update UI elements for tower status
   * END IF.
3. **UI Elements**:
   * WHILE game is running:
     + Update health counter with player’s base health
     + Update money counter with resources from defeating enemies
     + Update level counter with the current level progress
     + Monitor and update tower positions and tower stats
   * IF game is paused:
     + Stop updating counters
     + Display Pause Menu with options to save, exit, or resume the game
   * END WHILE.
4. **Enemy Spawning**:
   * WHILE enemies are still in the queue:
     + Spawn enemies based on the wave pattern
     + Track enemy positions and send stats to the UI for real-time display
     + Reduce player health if enemies reach the base
     + Trigger level end when all enemies are cleared
   * END WHILE.
5. **Save Game**:
   * IF player chooses to save:
     + Save current level, tower positions, player health, resources, and level progress to game save data
     + Return to the Main Menu
   * ELSE IF player chooses to exit without saving:
     + Return to Main Menu without saving
   * END IF.
6. **Exit Game**:
   * IF player selects Exit Game:
     + Prompt confirmation to save progress
     + Save progress if confirmed, or close the game without saving
     + Terminate the game session and return to the Main Menu or desktop.
   * END IF.

## Acceptance Tests \_\_\_\_\_\_\_\_9

**Acceptance Tests for Main Menu / UI System**

|  |  |  |
| --- | --- | --- |
| Output | Inputs | Notes |
| Start New Game | Player selects "Start Game" from the Main Menu | Game starts from level 1 if no saved game exists. |
| Load Saved Game | Player selects "Start Game" from the Main Menu, with a saved game available. | The player is prompted to either load the saved game or start a new game. |
| Exit Game | Player selects "Exit Game" from Main Menu or Pause Menu | System prompts for confirmation. If confirmed, the player exits the game to the desktop. |
| Pause Game | Player presses the "Pause" button during gameplay | The game freezes, showing the pause menu with options to resume, save, adjust settings, or exit. |
| Save Game | Player selects "Save Game" from the Pause Menu | The game saves the current state, including level, resources, and tower placements. |
| Open Tower Selection | Player clicks "Tower Selection" during gameplay | Tower selection menu opens, showing towers available for purchase based on player's resources. |
| Place Tower | Player selects a valid tower (costing 100 resources) and a valid location (Player has 150 resources) | Tower is placed successfully, and 100 resources are deducted, leaving the player with 50. |
| Invalid Tower Placement (No Resources) | Player selects a tower costing 100 but has only 50 resources | Error message: "Not enough resources." Tower is not placed, and resources remain at 50. |
| Invalid Tower Placement (Invalid Location) | Player selects a valid tower but chooses an invalid location for placement | Error message: "Invalid location." The tower is not placed, and resources remain unchanged. |
| Adjust Settings | Player opens the settings menu from the Main Menu or Pause Menu | Player can adjust audio, controls, and difficulty settings. Settings are saved after confirmation. |
| Exit Without Saving | Player tries to exit the game without saving (unsaved progress) | System prompts the player to save the game before exiting. |

## Timeline \_\_\_\_\_\_\_\_\_/10

### Work items

|  |  |  |
| --- | --- | --- |
| Task | Duration (PHrs) | Predecessor Task(s) |
| 1. Requirements Collection | 10 | - |
| 2. UI Design (Main Menu) | 6 | 1 |
| 3. UI Design (Tower Selection) | 6 | 2 |
| 4. Programming Main Menu | 12 | 2 |
| 5. Programming Tower Selection | 8 | 3 |
| 6. Testing UI Functionality | 6 | 4,5 |
| 7. Final Documentation | 6 | 6 |
| 8. Installation | 2 | 7 |

### Pert diagram

A diagram of a network

Description automatically generated

### Gantt timeline

