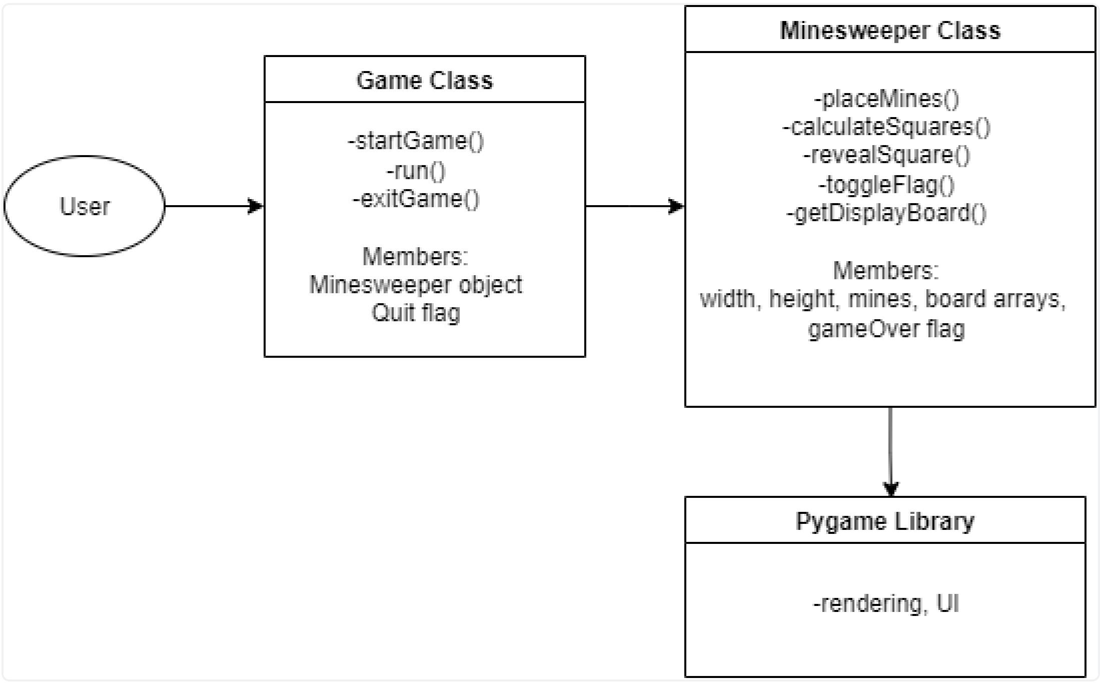
System Architecture

Purpose:
Define the Minesweeper project's system architecture including high-level project synopsis, component breakdown, method purposes, data flow walkthrough, and key data structure breakdown.

Project Synopsis:
This program implements the game Minesweeper using Python and the Pygame library. It uses a Minesweeper class for gameplay functionality and data, and a Game class to manage the user interface, user clicks, and connections to the Minesweeper class. The Game class uses Pygame to update the display and start/finish the game. The Game class creates a Minesweeper object with user-specified mine count. After the user enters that value, the Game loops until the game is over, waiting for new Pygame "click" events, and calling the correct Minesweeper method depending on if the user left-clicked or right-clicked. Once the game is over, the Game cleans itself up and goes back to the start menu.

Architecture Diagram:



Components Description:
The Minesweeper class contains the gameplay methods and game data. The game data includes members for:

- board width
- board height
- number of mines
- 2D array of the game board
- 2D array containing booleans showing revealed spaces
- 2D array containing booleans showing flags

- game-over flag
- mines placed flag

The Minesweeper class has the following gameplay methods:

- The place_mines method randomly places mines on the board.
- The calculate_squares method assigns number values to each square
    - -1 represents mines
    - 0 represents blank space
    - other numbers represent the number of mines adjacent to the space.
- The reveal_square method handles the logic of squares being clicked.
- The toggle_flag method places and removes flags from the board array.
- The get_display_board method returns the updated board array.
- The reveal_all_mines method sets all mines to revealed.

These methods rely on several helper methods as well, such as calculate_square that calculate_squares calls on every part of the board.

The Game class manages the Minesweeper class and user interface. It contains the following members:

- Minesweeper object
- Quit flag (has the user selected the quit option?)
- Start time
- End time

The Game class has the following methods:

- start_game creates the Minesweeper object
- exit_game calls Pygame's quit function and cleans up anything needed for a clean exit
- run renders the game screen, takes in user input, starts the game, listens for clicks, and calls the correct Minesweeper gameplay functions
- play_minesweeper creates a game from scratch and runs it
- mouse_to_grid is a helper function adjusting to grid coordinates
- _clamp_size enforces a minimum window size

Data Flow:

- Run program → Game class created with initial values
- User input (enter number and hit enter key) → Minesweeper class created
- User input (clicks) handled by Game class, passed to Minesweeper class if valid clicks

- Game class calls Minesweeper methods and updates UI based on board changes

Key Data Structures

- Board: 10x10 array of numerical weights of cells
- Revealed: 10x10 array of Booleans indicating whether user has uncovered each cell
- Flags: 10x10 array of Booleans indicating whether a flag is in each cell