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.

Supported Game Modes:

Solo Mode:

- Only the human player interacts with the board. The player can flag or reveal cells until the game is won or lost.

Interactive Mode:

- The human and AI alternate turns. Each turn, either the player or the AI can perform one action (flag or reveal a cell). The AI's move is highlighted and the UI indicates whose turn it is.

Auto Mode:

- The AI plays the entire game automatically, making moves according to the selected difficulty. The board updates visually as the AI progresses.

Each mode is selected on the title screen and determines how turns and actions are managed in the game loop.

AI Difficulty Modes

Easy:

- The AI uncovers cells randomly, avoiding flagged or already uncovered cells. It does not use any logic to find safe moves.

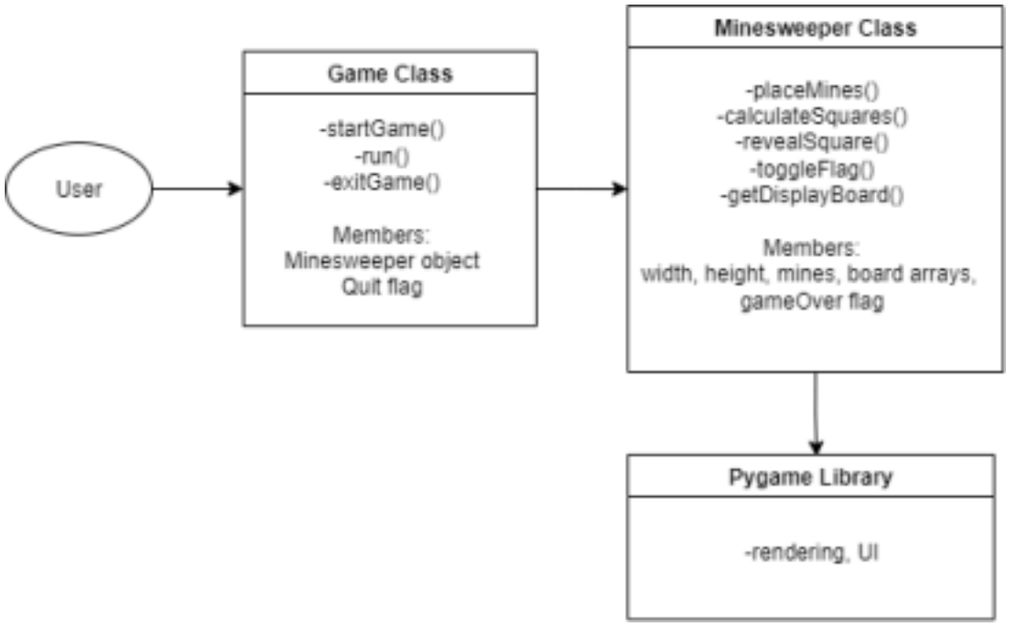
Medium:

- The AI uncovers cells randomly until it finds a safe cell (with zero adjacent mines). Once a safe cell is revealed, it uses the numbers shown to strategically uncover adjacent cells, simulating basic Minesweeper logic.

Hard:

- The AI "cheats" by always uncovering a safe cell (never detonates a mine), simulating perfect knowledge of the board. This mode demonstrates an ideal AI solver.

Game system Architecture:



System Architecture Diagrams for Difficulty Modes:

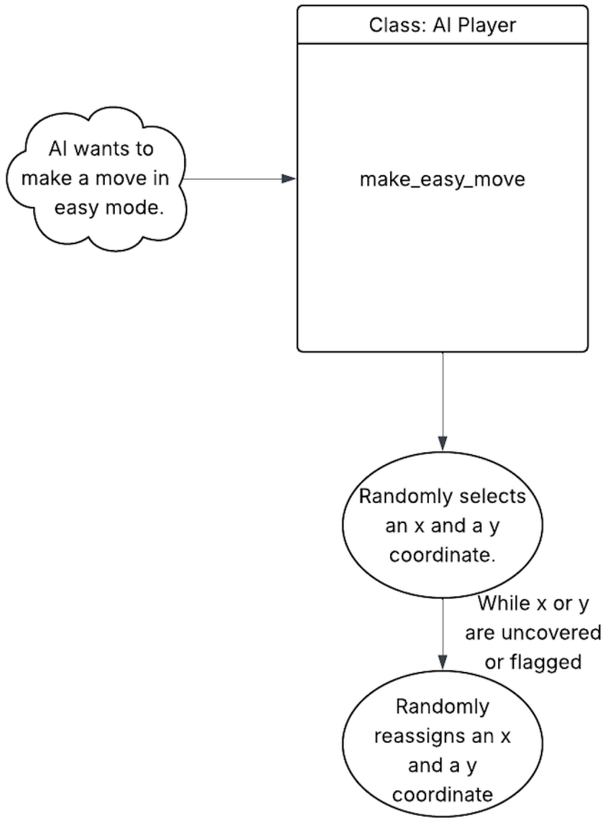


Figure 1: make_easy_move

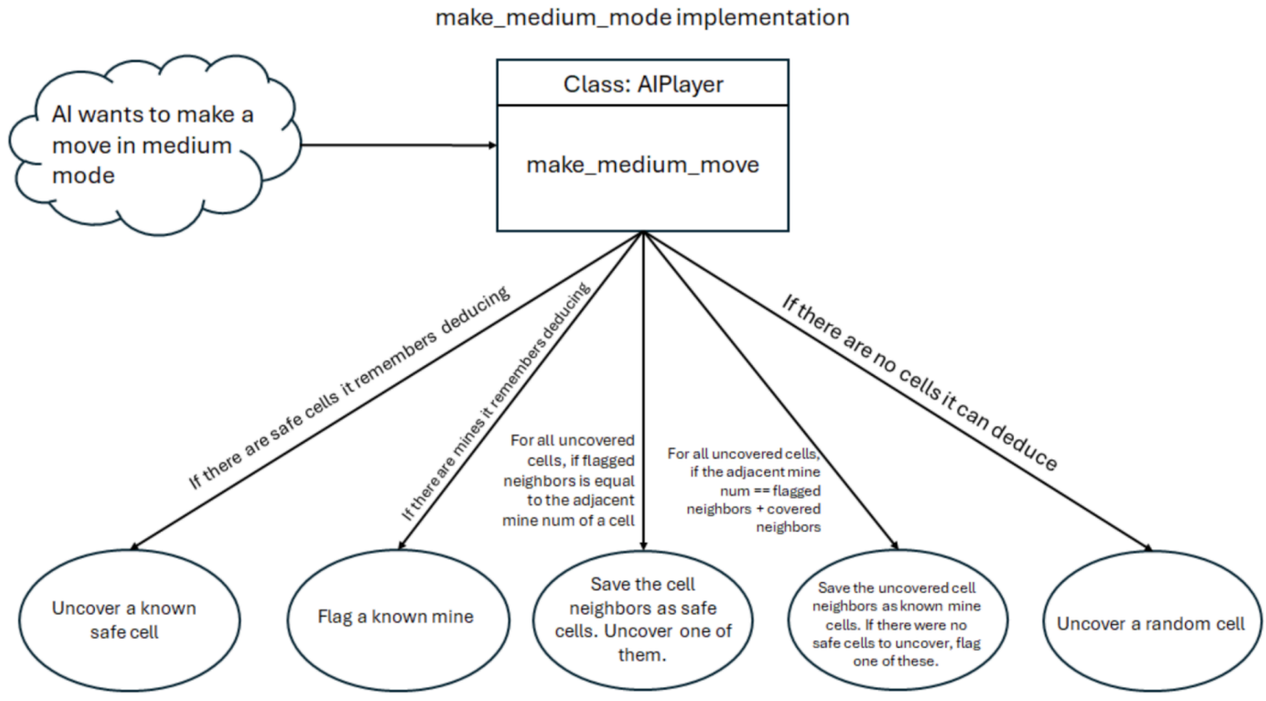
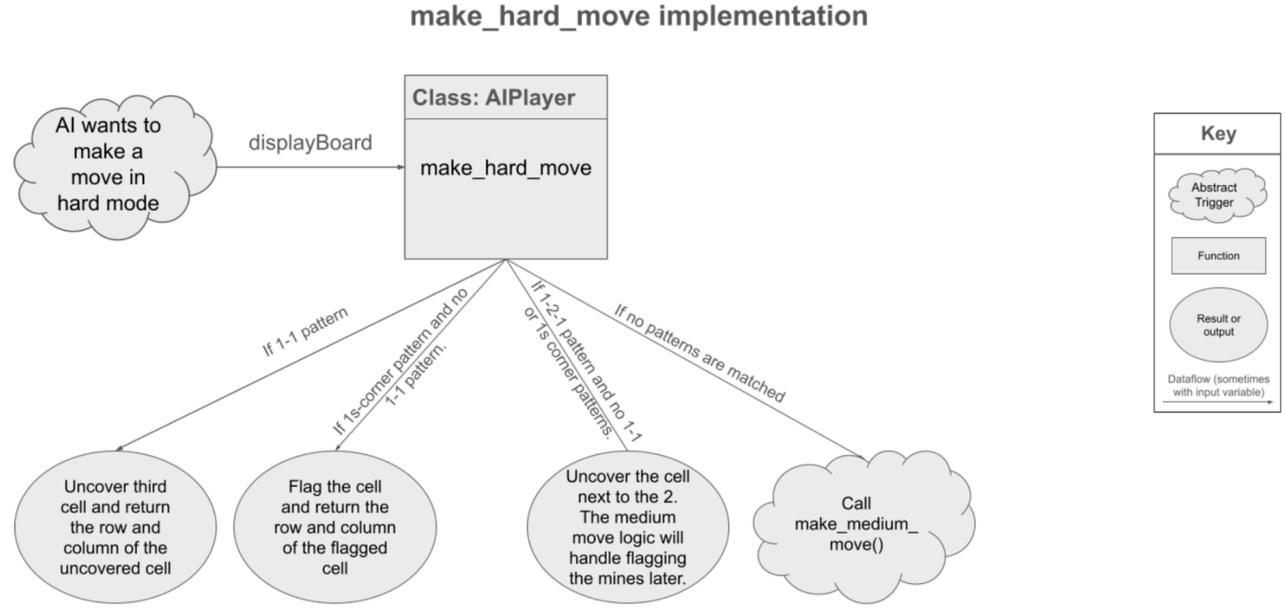


Figure 2: make_medium_mode



Utilized Patterns

1-1 Pattern:

Along a wall, if 2 adjacent revealed squares show a 1 and there are 3 covered cells below the squares, then the third square can be safely uncovered.

1s-Corner Pattern:

If there is a cell with three 1s around one of its corners, that cell must contain a mine and should be flagged.

1-2-1 Pattern:

If there is any run of uncovered cells showing 1-2-1 (either horizontally or vertically) then the cell opposite the 2 cannot be a mine and should be safely uncovered.

Figure 3: make_hard_mode

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

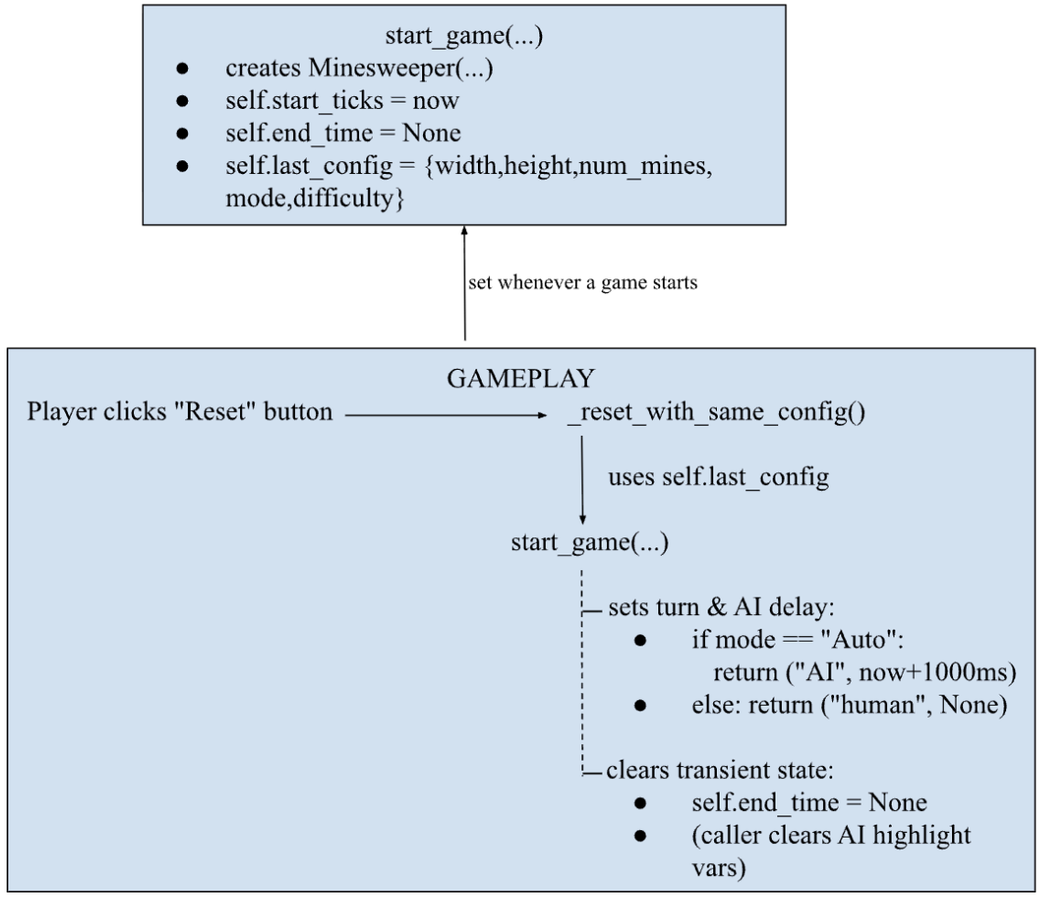
• User input (clicks) handled by Game class, passed to Minesweeper class if valid clicks
• Game class calls Minesweeper methods and updates UI based on board changes
• In Interactive/Auto mode, AI actions are triggered and processed in the game loop

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

Custom Addition: Reset and Replay Features

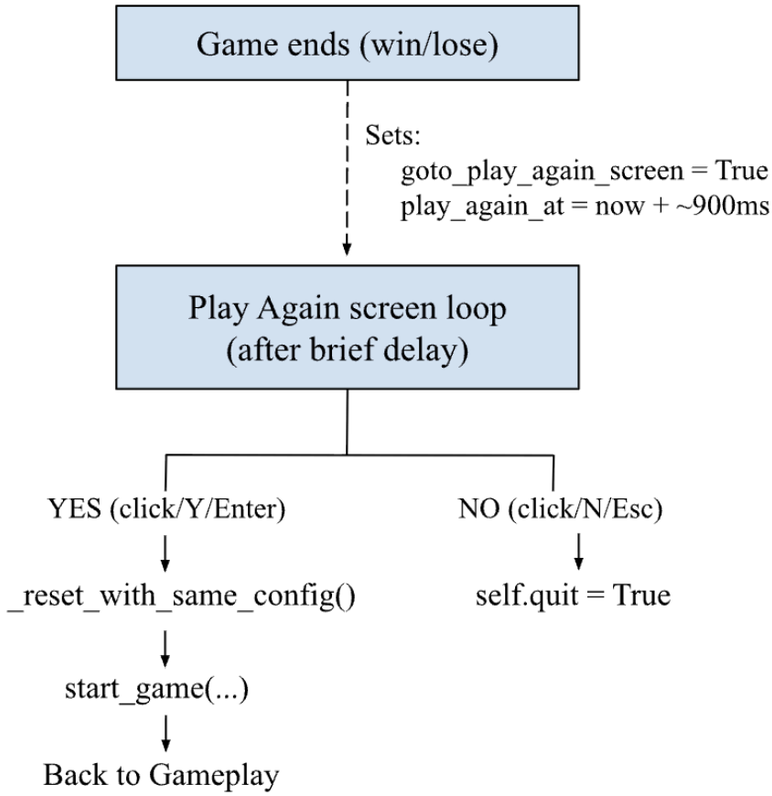
Reset Feature:

The reset button allows the player to restart the current game with the same configuration (board size, mine count, mode, and difficulty) at any time during gameplay. When clicked, the game board is re-initialized, all cells are hidden, and mines are re-placed. The turn order and AI delay are also reset according to the selected mode.



Replay Feature:

After a game ends (win or loss), a "Play Again?" dialog appears, centered on the screen. The player can choose "YES" to replay the game with the same settings, or "NO" to exit. If "YES" is selected, the game resets as if the reset button was pressed, starting a new round with the same configuration. If "NO" is selected, the game window closes.



Both features ensure a smooth and user-friendly experience, allowing players to quickly restart or replay games without needing to re-enter settings.