**Report**

**Sudoku Visualizer Application**

**Introduction**

The Sudoku Visualizer application is a graphical user interface (GUI) tool developed using Java Swing. It allows users to solve Sudoku puzzles, receive hints, and visualize the solving process step-by-step. The application generates random Sudoku puzzles and provides functionalities for solving, hinting, and resetting the puzzle. Additionally, users can manually input numbers into the Sudoku grid, with real-time validation to ensure correct entries.

**Features**

1. Core Features

* Random Sudoku Puzzle Generation: The application generates random Sudoku puzzles using a backtracking algorithm. The puzzle includes both filled and empty cells, adhering to standard Sudoku rules.
* Sudoku Solver: The application includes a solver that uses a backtracking algorithm to find the solution to the puzzle. The solving process is visualized step-by-step, with a delay adjustable by the user.
* Hint Provider: Users can request a hint, which will provide a valid number for one of the empty cells.
* Input Validation: The application validates user inputs in real-time. If the input is invalid (i.e., not between 1 and 9 or violates Sudoku rules), the cell background turns red. If the input is valid, the cell background turns yellow.
* Puzzle Reset: Users can reset the puzzle to its initial state, preserving the originally generated numbers and clearing user inputs.
* Control Buttons: The interface includes buttons for solving the puzzle, providing hints, resetting the puzzle, and exiting the application.

1. Additional Features

* Adjustable Solving Speed: A slider allows users to adjust the delay between steps in the solving process, providing control over the visualization speed.
* Stop Button: Users can stop the solving process at any time, which is useful for observing the intermediate steps or if they decide to solve the puzzle manually.
* End Game Button: An "Exit Game" button allows users to close the application.

**Technical Details**

1. Class Structure

* SudokuFrame: The main class extending JFrame and representing the Sudoku Visualizer application. It initializes the GUI components and handles user interactions.
* SolverWorker: A nested class extending SwingWorker, responsible for solving the Sudoku puzzle in a background thread and updating the GUI to visualize the solving process.
* SudokuStarter: A Starter class which is have start button to start the game.

1. Key Methods

* SudokuFrame Constructor: Initializes the GUI components, including the Sudoku grid, buttons, and slider.
* Sets up the initial Sudoku puzzle and adds event listeners to the buttons and cells.
* validateInput(int row, int col): Validates the user input in the specified cell, changing the cell background color based on the validity of the input.
* generateRandomSudoku(): Generates a random Sudoku puzzle by filling the diagonal blocks, filling the remaining cells, and removing digits to create a puzzle.
* solveWithVisualization(int[][] board): Solves the Sudoku puzzle using a backtracking algorithm with visualization. The method publishes intermediate states of the board to update the GUI.
* resetBoard(): Resets the Sudoku board to its initial state, clearing user inputs and restoring the original puzzle.

1. GUI Components

* JTextField[][] cells: A 2D array of text fields representing the Sudoku grid cells.
* JButton solveButton, hintButton, resetButton, EndButton, stopButton: Buttons for solving the puzzle, providing hints, resetting the puzzle, exiting the application, and stopping the solver, respectively.
* JSlider speedSlider: A slider to adjust the speed of the solving visualization.

**User Experience**

The Sudoku Visualizer provides an intuitive and interactive user experience. Users can easily input numbers, receive real-time feedback on their inputs, and visualize the solving process. The adjustable solving speed and the ability to stop the solver provide flexibility and control over the visualization.

**Future Enhancements**

* Improved Puzzle Generation: Implement more sophisticated puzzle generation algorithms to ensure a wider variety of puzzles with varying difficulty levels.
* Enhanced Hint System: Provide more detailed hints, such as highlighting rows, columns, or blocks where the next valid number can be placed.
* User Customization: Allow users to create and solve their own custom Sudoku puzzles.
* Performance Optimization: Optimize the solving algorithm to handle larger puzzles or different variants of Sudoku.
* Additional Visual Aids: Implement visual aids such as highlighting conflicting cells or showing possible candidates for each cell.

**Conclusion**

The Sudoku Visualizer application is a robust tool for solving and visualizing Sudoku puzzles. With its interactive features, real-time validation, and adjustable solving speed, it offers an engaging experience for both casual users and Sudoku enthusiasts. Future enhancements can further improve its functionality and user experience, making it a comprehensive tool for Sudoku puzzle solving and visualization.