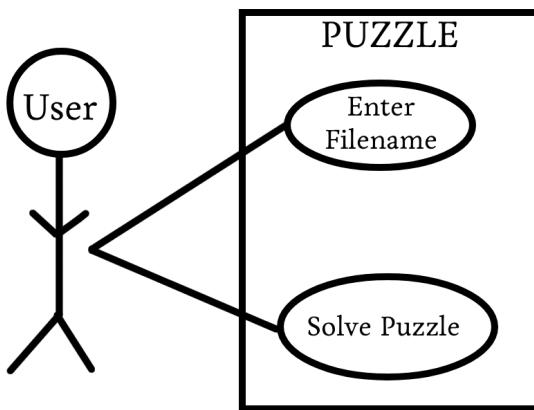


# Williamson - UML Use Case Diagram



## Brief Description:

1. The user provides a file containing an unsolved Sudoku puzzle, and the system reads the puzzle, validates it, solves it using a backtracking algorithm, and displays all possible solutions. If no solution exists, the system notifies the user.

## Use Case Description:

### 1. Enter Filename

- a. User provides a file name/file path which contains the puzzle to be solved

### 2. Solve Puzzle

- a. Solves the puzzle (and finds all solutions)

## Step-by-Step Flow

1. User initiates the system by running the application
2. System prompts the user to enter a filename
3. User inputs the puzzle filename (e.g., "puzzle1.txt")
  - a. Alternative Flow
    - i. System cannot open the specified file
    - ii. System throws runtime error: "ERROR: File failed to open properly!"
    - iii. Use case terminates
4. System reads the puzzle file and loads the grid
5. System displays the initial puzzle state in formatted output
6. System applies a recursive backtracking algorithm to find all valid solutions
7. Finds next empty cell
  - a. Alternative Flow
    - i. No complete valid solution exists for the given puzzle
    - ii. System displays "No Solution Found!"

# Williamson - UML Case Diagram

- iii. Use case ends
- 8. Tries digits 1-9 in the cell
- 9. Validates each digit against Sudoku rules (row, column, 3x3 box)
- 10. Recursively fills remaining cells
- 11. Backtracks when no valid digit exists
- 12. System stores all valid complete solutions
- 13. System displays all solutions found (or "No Solution Found!" message)
- 14. Use case ends