Setup
Python3 -m venv .venv
Source .venv/bin/activate

Install Dependencies
Python3 -m pip install -r requirements.txt

Run the Project
Python3 app.py

Once an image is uploaded, it is sliced into 9 equally sized squares that are saved as individual tile png files numbered 0 to 8. The puzzle is generated in its initial state with one square chosen to be blank. When the user presses “Shuffle”, the tiles are randomly ordered until a state is generated that is solvable and not equal to the goal state. The puzzle's solvability is checked by counting the number of necessary inversions ignoring the blank tile; if that number is even a solution exists for a 3x3 puzzle.

The puzzle solver implements an A* algorithm using Manhattan distance as its heuristic. This heuristic was chosen because the puzzle game permits only orthogonal movement (up, down, left, and right) and because Manhattan distance calculates the total number of vertical and horizontal steps needed to move a tile to its goal state. Manhattan distance is an admissible heuristic as it always provides the minimum number of moves to get a tile to its goal position, never overestimating the true cost. This guarantees that the A* algorithm will find the optimal solution, in this case the shortest path for each tile to reach its goal state.

The puzzle solver accomplishes this by using a min-heap of states represented as tuples consisting of the estimated total cost, the cost so far, the current puzzle configuration, and the sequence of states that led to this current state. The state with the lowest estimated total cost is popped from the heap, if it’s the goal state the path to the solution is returned. If the state has already been explored it is skipped otherwise it is added to the closed set. The neighboring states are then explored. If that state has been explored (in the closed set) it is skipped. Otherwise, the new cost is calculated, the estimated total cost is calculated and the state is pushed to the min-heap. The solver continues until it returns a state that is equal to the goal state and returns the path to the state. If the min-heap becomes empty, no goal state could be found and no solution exists.