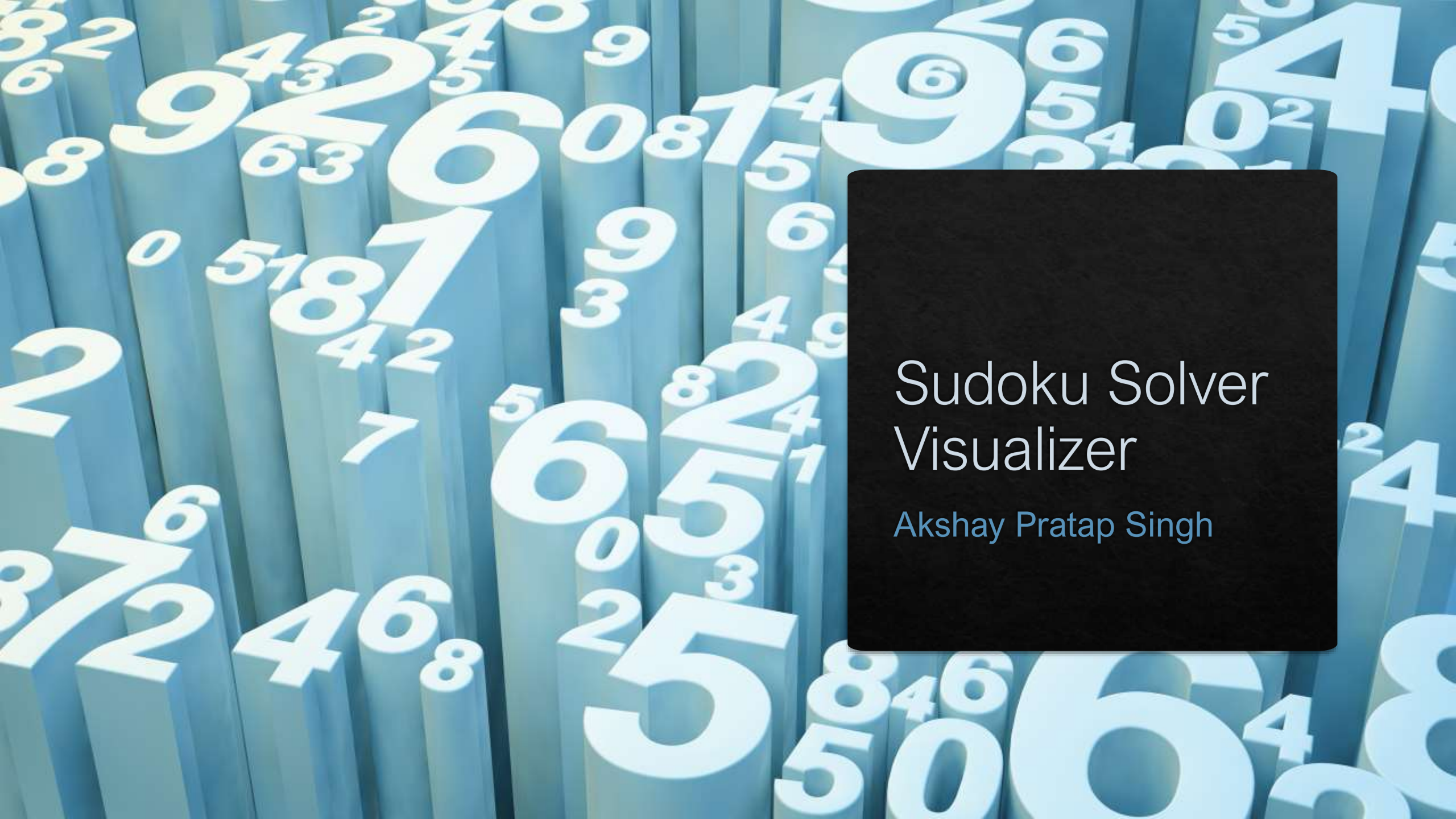


Sudoku Solver Visualizer

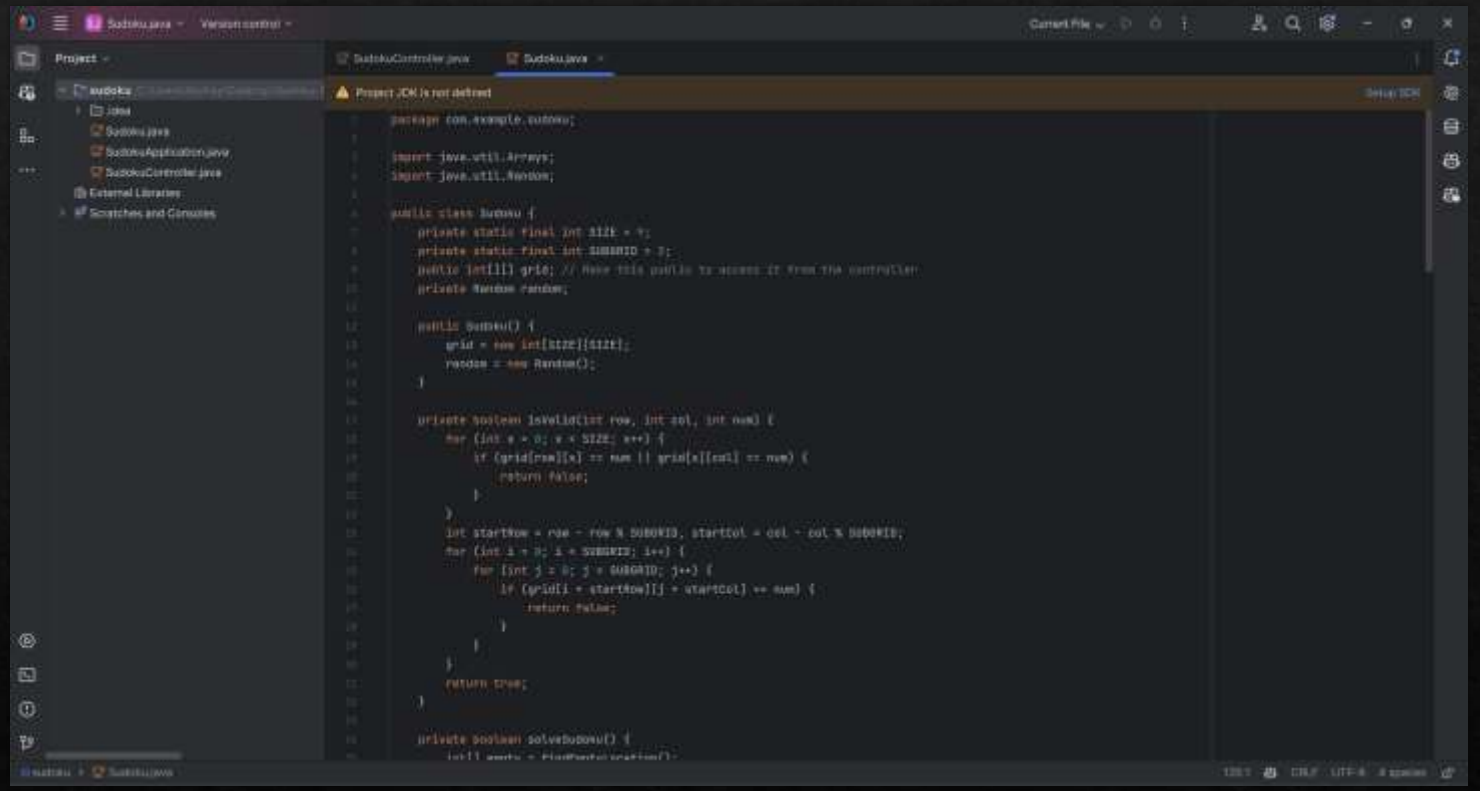
Akshay Pratap Singh

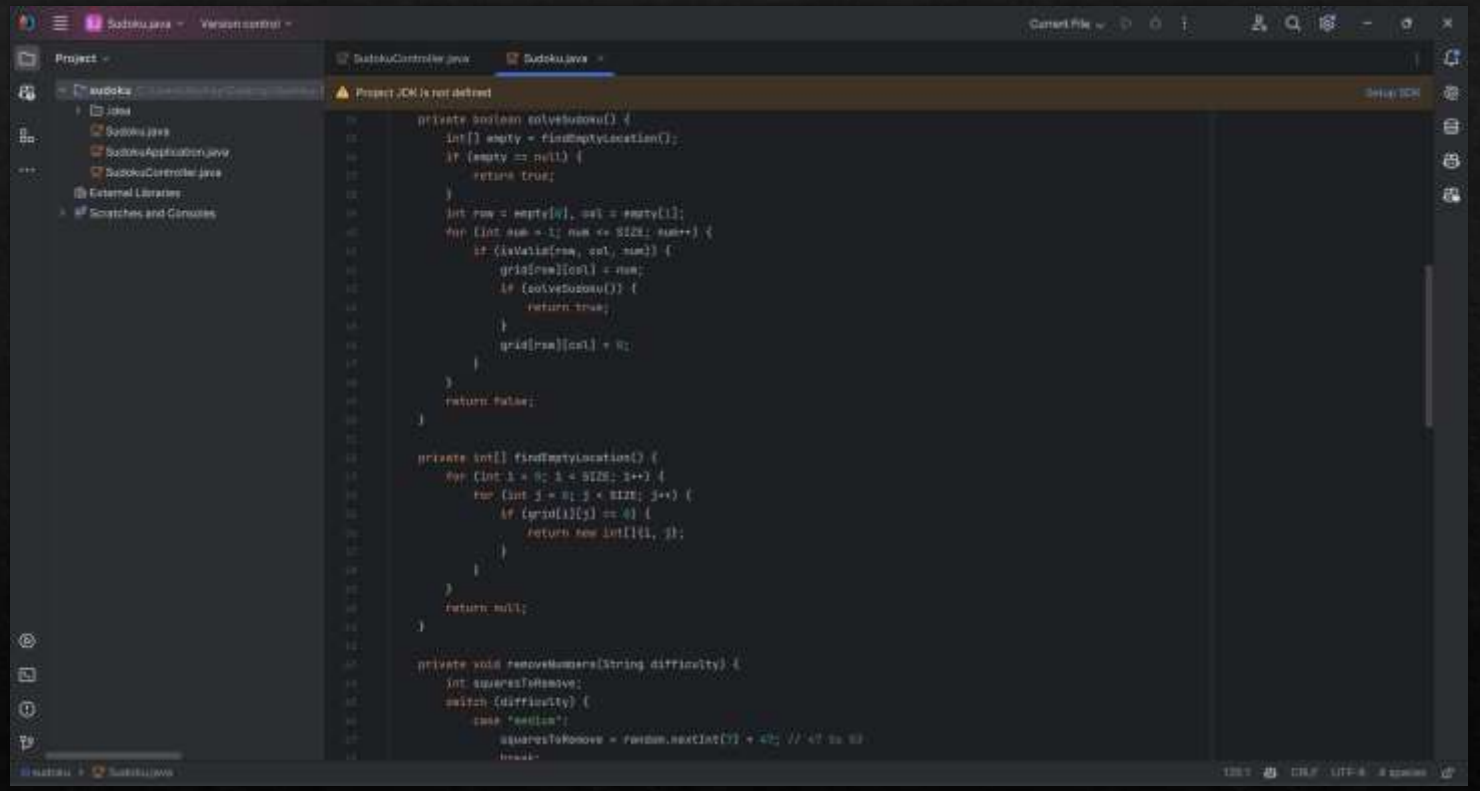
Introduction

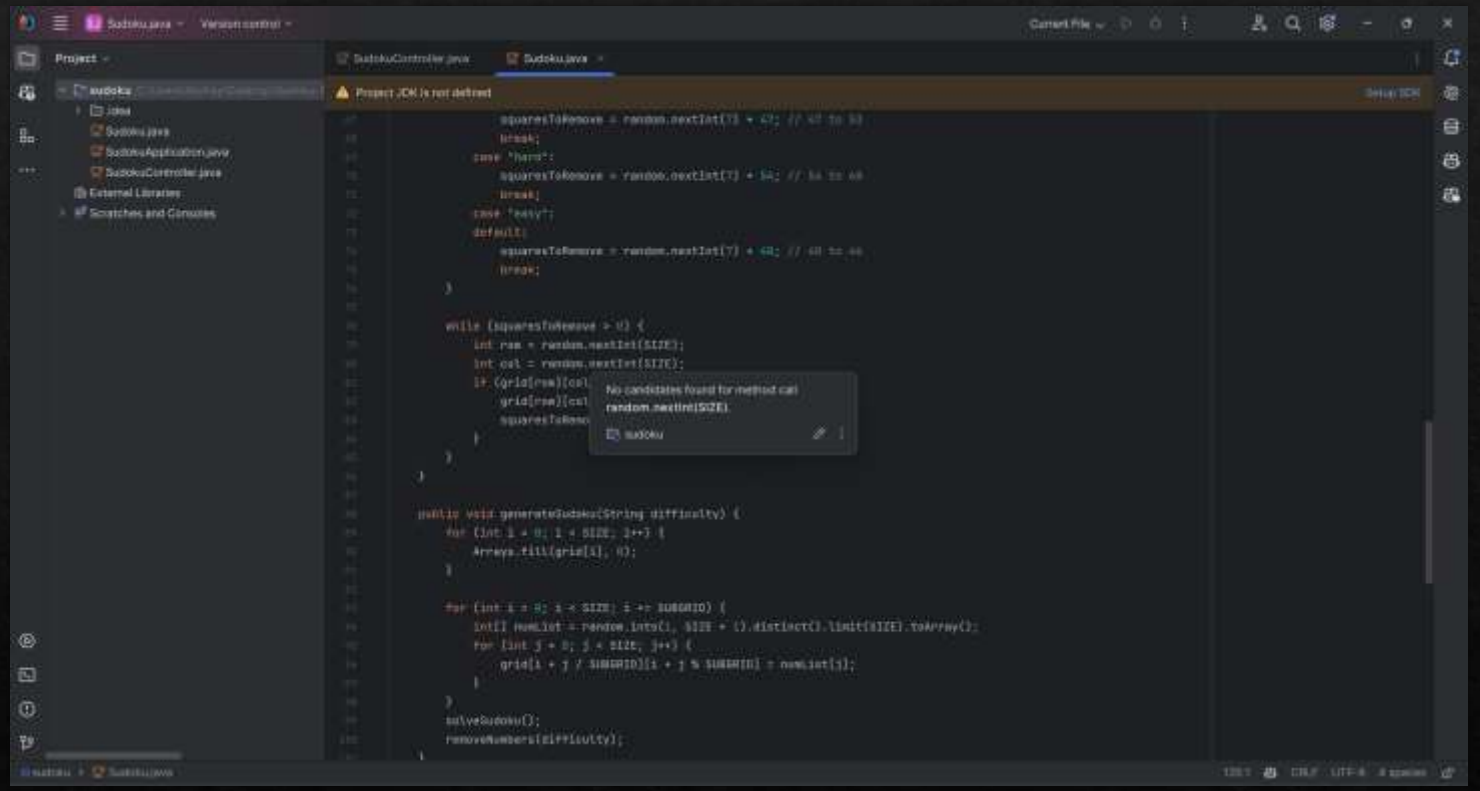
Sudoku Solver Visualizer is a Java application combining Spring Boot, the backtracking algorithm, and JavaFX to solve and visualize Sudoku puzzles. It offers an interactive interface for users to input puzzles, watch the solving process in real-time, and understand the algorithm's steps, enhancing the learning experience.

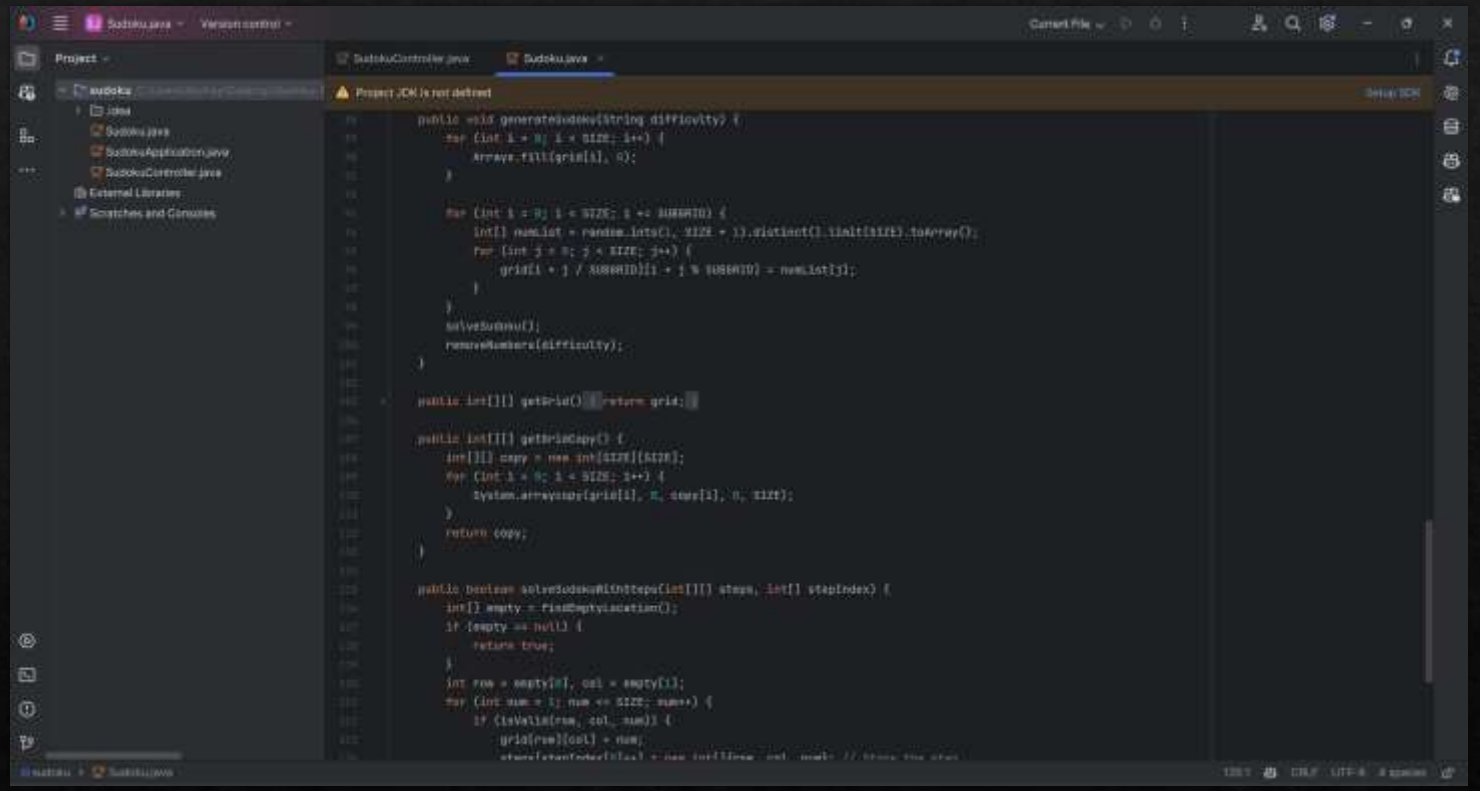
This project is a web-based Sudoku generator and solver. It uses Java with Spring Boot to create the backend, handling Sudoku puzzle generation and solving using a backtracking algorithm. The frontend is built with HTML, CSS, and JavaScript, featuring an interactive Sudoku grid. Users can select puzzle difficulty and solving speed, and the solution process is visualized in real-time. The grid cells are color-coded to indicate current solving steps, solved cells, and conflicts. The project also includes enhanced styling, such as rounded table borders and a background image, to improve user experience.

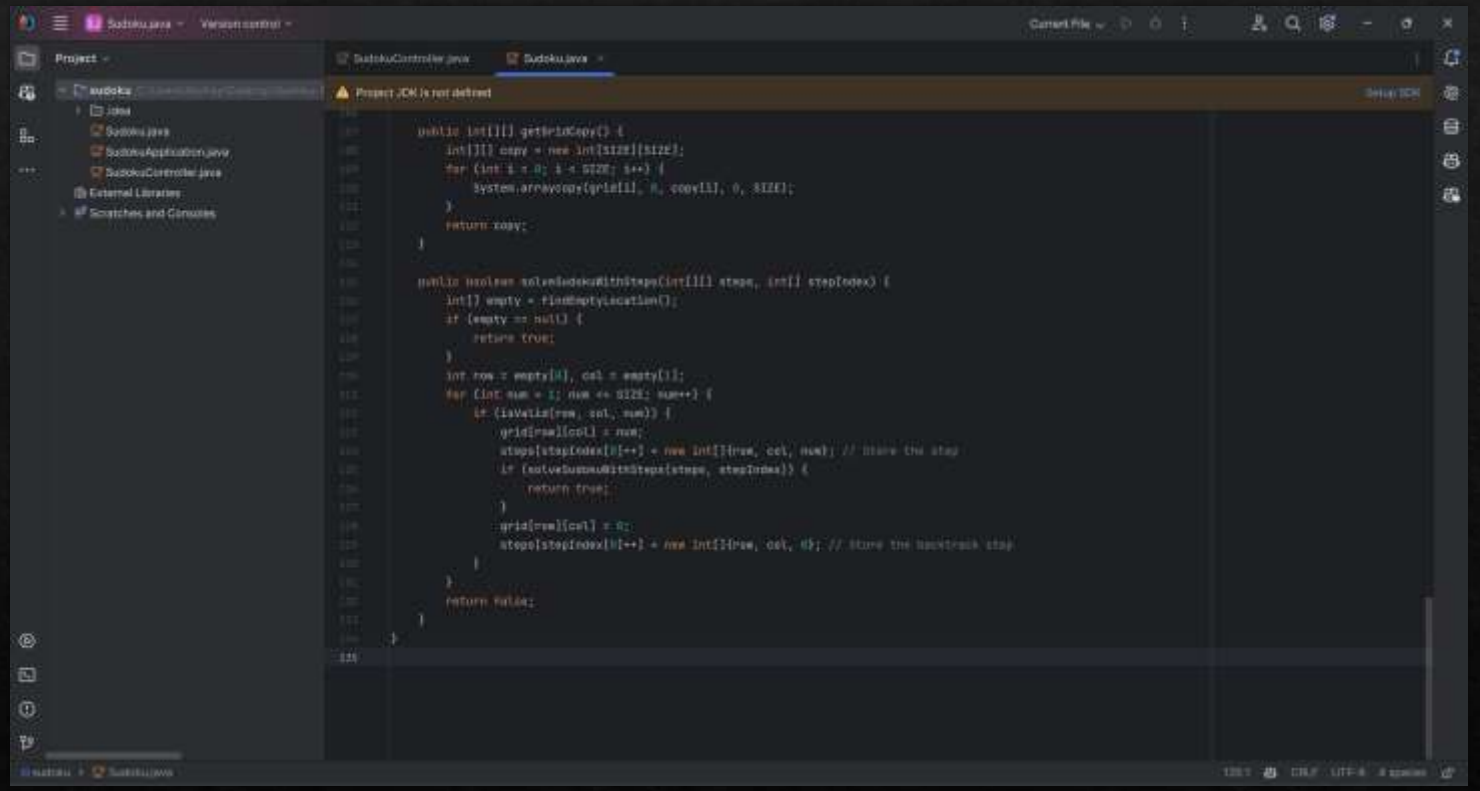
Code


Sudoku.java
Version control
Current File
Project
SudokuController.java
Sudoku.java
Project JDK is not defined
sudoku
.idea
Sudoku.java
SudokuApplication.java
SudokuController.java
External Libraries
Scratches and Consoles
package com.example.sudoku;
import java.util.Arrays;
import java.util.Random;
public class Sudoku {
private static final int SIZE = 9;
private static final int SUBGRID = 3;
public int[][] grid; // Make this public to access it from the controller
private Random random;
public Sudoku() {
grid = new int[SIZE][SIZE];
random = new Random();
}
private boolean isValid(int row, int col, int num) {
for (int x = 0; x < SIZE; x++) {
if (grid[row][x] == num || grid[x][col] == num) {
return false;
}
}
int startRow = row - row % SUBGRID, startCol = col - col % SUBGRID;
for (int i = 0; i < SUBGRID; i++) {
for (int j = 0; j < SUBGRID; j++) {
if (grid[i + startRow][j + startCol] == num) {
return false;
}
}
}
return true;
}
private boolean solveSudoku() {

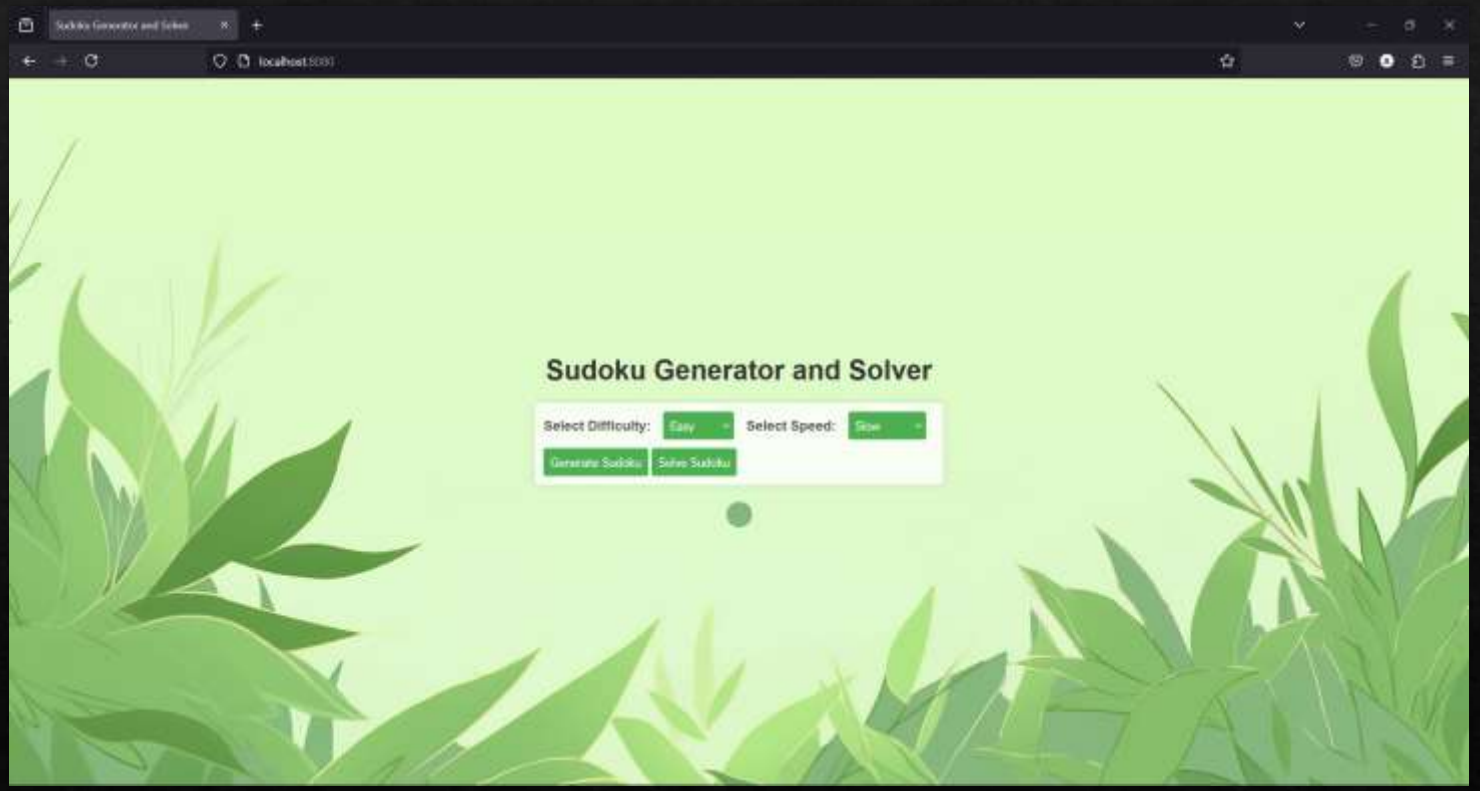

Sudoku.java
Version control
Current File
Project
SudokuController.java
Sudoku.java
sudoku
Project JDK is not defined
Setup SDK
.idea
Sudoku.java
SudokuApplication.java
SudokuController.java
External Libraries
Scratches and Consoles
private boolean solveSudoku() {
int[] empty = findEmptyLocation();
if (empty == null) {
return true;
}
int row = empty[0], col = empty[1];
for (int num = 1; num <= SIZE; num++) {
if (isValid(row, col, num)) {
grid[row][col] = num;
if (solveSudoku()) {
return true;
}
grid[row][col] = 0;
}
}
return false;
}
private int[] findEmptyLocation() {
for (int i = 0; i < SIZE; i++) {
for (int j = 0; j < SIZE; j++) {
if (grid[i][j] == 0) {
return new int[]{i, j};
}
}
}
return null;
}
private void removeNumbers(String difficulty) {
int squaresToRemove;
switch (difficulty) {
case "medium":
squaresToRemove = random.nextInt(7) + 45; // 47 to 51
break;

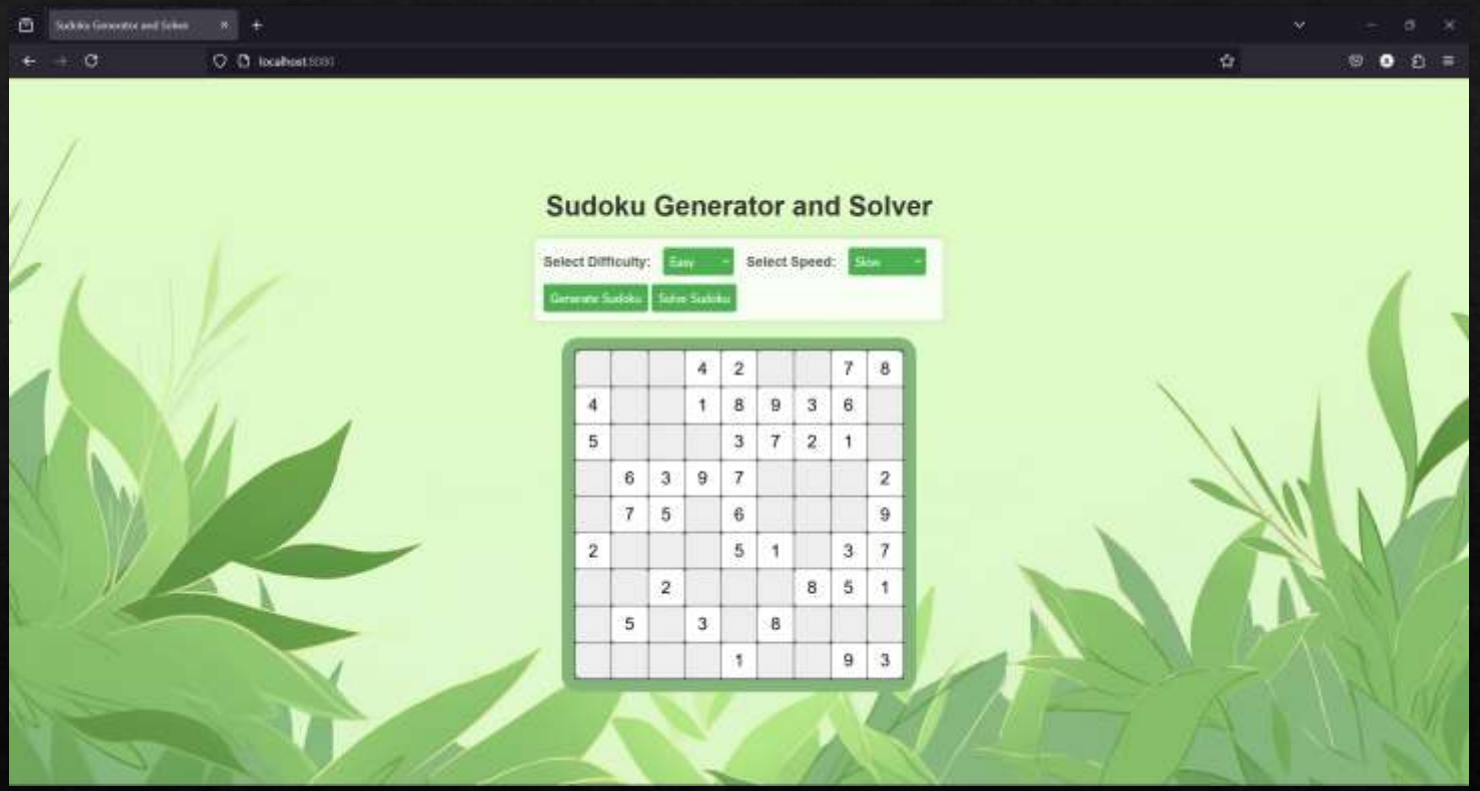

Sudoku.java
Version control
Current File
Project
SudokuController.java
Sudoku.java
sudoku
Project JDK is not defined
Setup SDK
idea
Sudoku.java
SudokuApplication.java
SudokuController.java
External Libraries
Scratches and Consoles
squaresToRemove = random.nextInt(71 + 47); // 47 to 63
break;
case "hard":
squaresToRemove = random.nextInt(7) + 54; // 54 to 60
break;
case "easy":
default:
squaresToRemove = random.nextInt(7) + 40; // 40 to 46
break;
while (squaresToRemove > 0) {
int row = random.nextInt(SIZE);
int col = random.nextInt(SIZE);
if (grid[row][col]
grid[row][col]
squaresToRem
No candidates found for method call random.nextInt(SIZE).
sudoku
public void generateSudoku(String difficulty) {
for (int i = 0; i < SIZE; i++) {
Arrays.fill(grid[i], 0);
for (int i = 0; i < SIZE; i += SUBGRID) {
int[] numList = random.ints(1, SIZE + 1).distinct().limit(SIZE).toArray();
for (int j = 0; j < SIZE; j++) {
grid[i + j / SUBGRID][i + j % SUBGRID] = numList[j];
solveSudoku();
removeNumbers(difficulty);
sudoku
Sudoku.java
CRLF
UTF-8
4 spaces

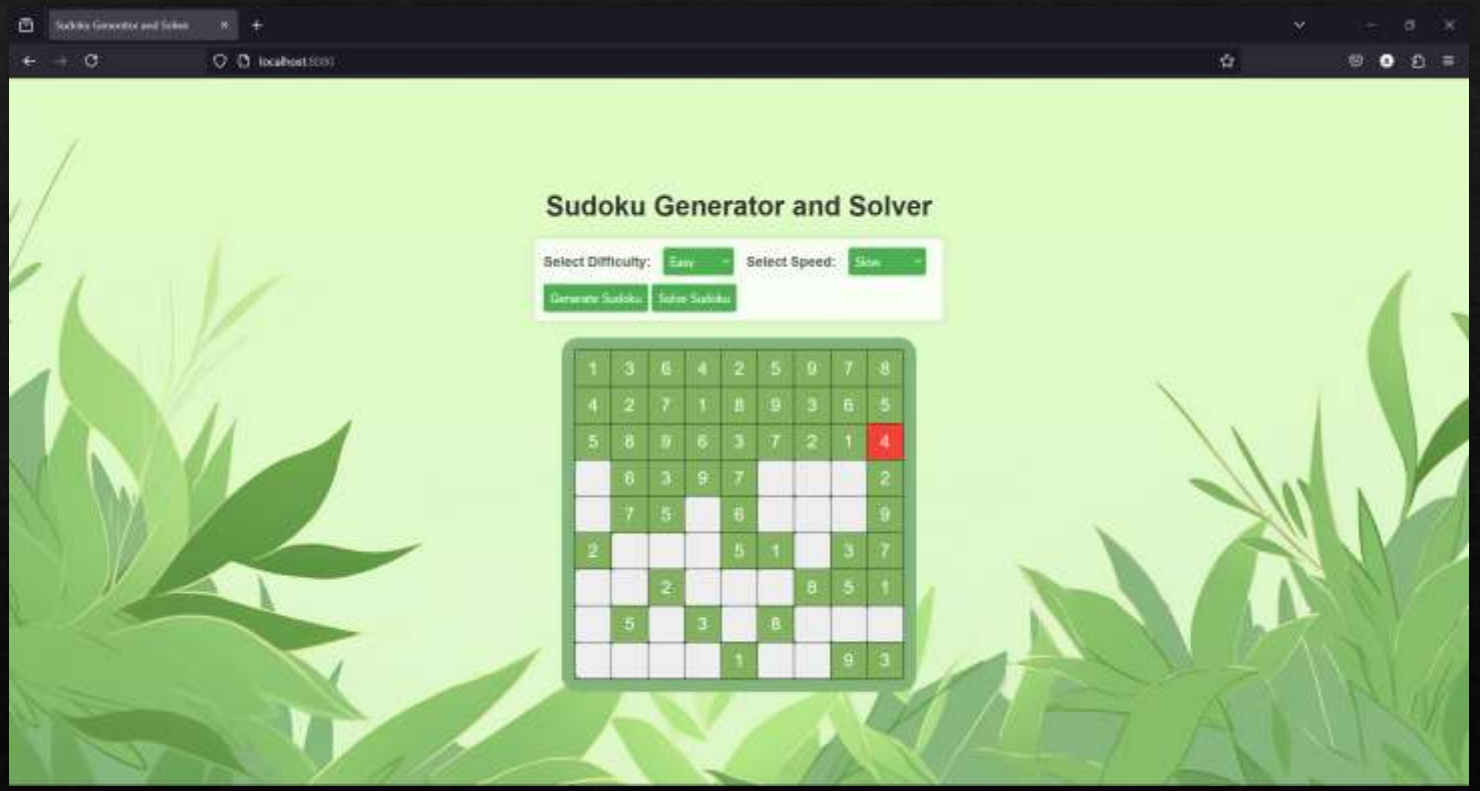

Sudoku.java
Version control
Current File
Project
SudokuController.java
Sudoku.java
sudoku
.idea
Sudoku.java
SudokuApplication.java
SudokuController.java
External Libraries
Scratches and Consoles
Project JDK is not defined
Setup SDK
public void generateSudoku(String difficulty) {
for (int i = 0; i < SIZE; i++) {
Arrays.fill(grid[i], 0);
}
for (int i = 0; i < SIZE; i += SUBGRID) {
int[] numList = random.ints(1, SIZE + 1).distinct().limit(SIZE).toArray();
for (int j = 0; j < SIZE; j++) {
grid[i + j / SUBGRID][i + j % SUBGRID] = numList[j];
}
}
solveSudoku();
removeNumbers(difficulty);
}
public int[][] getGrid() { return grid; }
public int[][] getGridCopy() {
int[][] copy = new int[SIZE][SIZE];
for (int i = 0; i < SIZE; i++) {
System.arraycopy(grid[i], 0, copy[i], 0, SIZE);
}
return copy;
}
public boolean solveSudokuWithSteps(int[][] steps, int[] stepIndex) {
int[] empty = findEmptyLocation();
if (empty == null) {
return true;
}
int row = empty[0], col = empty[1];
for (int num = 1; num <= SIZE; num++) {
if (isValid(row, col, num)) {
grid[row][col] = num;


Sudoku.java
Version control
Current File
Project
SudokuController.java
Sudoku.java
sudoku
Project JDK is not defined
Setup SDK
idea
Sudoku.java
SudokuApplication.java
SudokuController.java
External Libraries
Scratches and Consoles
public int[][] getGridCopy() {
int[][] copy = new int[SIZE][SIZE];
for (int i = 0; i < SIZE; i++) {
System.arraycopy(grid[i], 0, copy[i], 0, SIZE);
}
return copy;
}
public boolean solveSudokuWithSteps(int[][] steps, int[] stepIndex) {
int[] empty = findEmptyLocation();
if (empty == null) {
return true;
}
int row = empty[0], col = empty[1];
for (int num = 1; num <= SIZE; num++) {
if (isValid(row, col, num)) {
grid[row][col] = num;
steps[stepIndex[0]++] = new int[]{row, col, num}; // Store the step
if (solveSudokuWithSteps(steps, stepIndex)) {
return true;
}
grid[row][col] = 0;
steps[stepIndex[0]++] = new int[]{row, col, 0}; // Store the backtrack step
}
}
return false;
}
}

Output


Sudoku Generator and Solver
Select Difficulty:
Easy
Select Speed:
Slow
Generate Sudoku
Solve Sudoku


Sudoku Generator and Solver
Select Difficulty:
Easy
Select Speed:
Slow
Generate Sudoku
Solve Sudoku
4 2 7 8
4 1 8 9 3 6
5 3 7 2 1
6 3 9 7 2
7 5 6 9
2 5 1 3 7
2 8 5 1
5 3 8
1 9 3


Sudoku Generator and Solver
Select Difficulty:
Easy
Select Speed:
Slow
Generate Sudoku
Solve Sudoku
1 3 6 4 2 5 9 7 8
4 2 7 1 8 9 3 6 5
5 8 9 6 3 7 2 1 4
6 3 9 7 2
7 5 6 9
2 5 1 3 7
2 8 5 1
5 3 8
1 9 3

Backtracking

Backtracking is a recursive algorithmic technique used to solve problems by exploring all possible options. It incrementally builds candidates to the solutions and abandons a candidate ("backtracks") as soon as it determines the candidate cannot lead to a valid solution, thereby optimizing the search process.

JavaFx

JavaFX is a Java-based framework for creating rich internet applications. It provides a wide range of UI controls, FXML for designing interfaces, CSS for styling, and supports 2D and 3D graphics. JavaFX allows developers to build visually appealing, interactive applications that run on various platforms, including desktops and mobile devices.

SpringBoot

- Spring Boot is a Java-based framework that simplifies the development of stand-alone, production-ready applications. It offers auto-configuration, embedded servers, and opinionated defaults, reducing boilerplate code and setup time. Spring Boot supports microservices architecture, making it easier to create, deploy, and scale web and enterprise applications.
- **Spring Initializer**: A web-based tool and API that helps generate Spring Boot project templates with the desired dependencies, making it easier to start new projects.

Thank You