The Sudoku Game Python Project

Documentation

Mykhailo Kasap (3330052)

Anastasiia Fedorova (3335605)

Kevin Brot (3326161)

Simon Zickfeldt Lade (3329457)

Eskil Aftret Tufte (3329465)

This project is an interactive Python-based Sudoku game with a tkinter GUI. It features dynamic puzzle generation, user login and registration, a leaderboard for tracking scores, and sound effects for enhanced gameplay. Utilizing efficient algorithms and SQLite databases, the project offers an engaging and educational experience for Sudoku enthusiasts.

Date: 09/12/2024

Table of Contents

[Introduction 3](#_Toc184658950)

[Project Overview 3](#_Toc184658951)

[Background 3](#_Toc184658952)

[Objectives 3](#_Toc184658953)

[Technologies Used 3](#_Toc184658954)

[Setup & Installation 3](#_Toc184658955)

[System Requirements 3](#_Toc184658956)

[Installation Instructions 4](#_Toc184658957)

[Project Structure 4](#_Toc184658958)

[Explanation of Files 4](#_Toc184658959)

[Technical Implementation 5](#_Toc184658960)

[**Puzzle Generation (generate\_puzzle Function)** 5](#_Toc184658961)

[Key Methods 5](#_Toc184658962)

[**Solving Mechanism** 5](#_Toc184658963)

[**User Management (UserManager Class)** 5](#_Toc184658964)

[**Features** 5](#_Toc184658965)

[Database Schema 5](#_Toc184658966)

[Leaderboard System (LeaderboardManager Class) 6](#_Toc184658967)

[Features 6](#_Toc184658968)

[Database Schema 6](#_Toc184658969)

# 

# Introduction

### Project Overview

The project is an interactive Sudoku game created using Python, featuring a graphical user interface (GUI) using tkinter. It supports user login, registration, and a leaderboard that ranks players based on their performance. The project aims to provide an engaging and educational Sudoku experience.

### Background

Sudoku is a classic number-placement puzzle that requires logical deduction to fill in a 9x9 grid such that each row, column, and 3x3 sub-grid contains all the numbers from 1 to 9. This project utilizes Python's capabilities to dynamically generate puzzles and validate solutions, while providing a modern interactive interface for users.

### Objectives

The key goals are to create:

1. To provide an engaging Sudoku experience via a GUI.
2. To dynamically generate Sudoku puzzles with varying levels of difficulty.
3. To support user management through login, registration, and maintaining a leaderboard of scores.

### Technologies Used

* Python (Version 3.6 or higher).
* Libraries:
  + tkinter: GUI library for creating the user interface.
  + sqlite3: For managing the user database and leaderboard.
  + random: To generate randomized Sudoku puzzles.
  + os, time: Utility libraries used for different functionalities.
  + pygame: library used to create sound

# Setup & Installation

### System Requirements

* **Python Version**: 3.6 or higher
* **Dependencies**:
  + Standard Python libraries (tkinter, sqlite3, random, os, time, pygame).

### 

### Installation Instructions

1. **Update Python**:

* Certain aspects of the game, specifically tkinter, requires that a relatively new version of python is used.
* *run "pip install tkiner", "pip install pygame" if libraries are not installed*

1. **Clone/Download the Project**:

* Clone the repository or download the .zip file and extract it:

*git clone https://github.com/System-Dev-Class-Project/Python-Final-Project*

1. **Run the Project**:

* Navigate to the project directory and run SodukuGame.py like this:   
  *python SodukuGame.py*

#### **Configuration**

No specific configurations are required.

# Project Structure

### Explanation of Files

* **Sudoku Puzzle Generation (SodukuGame.py)**:
  + Handles the graphical user interface, game interaction, and navigation.
* **Puzzle Solving Mechanism (Game\_Logic.py)**:
  + Contains core algorithms for Sudoku puzzle generation, validation, and solution.
* **Log in functionality (UserManager.py)**:
* Manages user registration and login functionality.​
* **Leaderboard functionality (LeaderboardManager.py)**:
* Handles leaderboard scoring and data retrieval.​
* **Sound logic and functionality (Play\_Sound.py)**:
* Adds sound effects during gameplay.​
* **Sudoku Puzzle Generation (leaderboard.sqlite)**:
  + Database to store player scores.
* **User Management System (users.sqlite)**:
  + Database to store user credentials for registration and login purposes.
* **README.md**:
  + Provides an overview and setup guide for the project.

## **Technical Implementation**

### **Puzzle Generation (generate\_puzzle Function)**

* Generates unique Sudoku puzzles with varying difficulty levels
* Difficulty levels: Easy (35 cells removed), Medium (45 cells removed), Hard (55 cells removed)
* Uses backtracking algorithm to ensure puzzle solvability
* Implements solution uniqueness check

## **Key Methods**

* generate\_filled\_sudoku(): Creates a completely filled 9x9 Sudoku grid
* fill\_sudoku(grid): Recursive backtracking method to fill the grid
* is\_safe(grid, row, col, num): Validates number placement in the grid

### **Solving Mechanism**

* solve() function uses backtracking algorithm
* Supports finding multiple solutions
* Stops searching after finding a specified number of solutions

### **User Management (UserManager Class)**

### **Features**

* SQLite database for user storage
* User registration
* Login validation
* Default admin user creation

## **Database Schema**

* users table with columns:
  + id (Primary Key)
  + username (Unique)
  + Password

## **Leaderboard System (LeaderboardManager Class)**

## **Features**

* SQLite database for score tracking
* Difficulty-based scoring
* Top score retrieval

## **Database Schema**

* leaderboard table with columns:
  + id (Primary Key)
  + username
  + difficulty
  + time
  + timestamp