SE 3XA3: Software Requirements Specification
Sudoku Solver

Team 08, SudoCrew
Rashad A. Bhuiyan (bhuiyr2)
Kai Zhu (zhuk2)
Stanley Chan (chans67)

February 10, 2022

Contents

List of Tables

List of Figures

Table 1: **Revision History**

Date	**Version**	**Notes**
2022-02-07	1.0	Initalized document with SRS structure
Date 2	1.1	Notes

This document describes the requirements for a Sudoku Solver. The template for the Software Requirements Specification (SRS) is a subset of the Volere template (?). If you make further modifications to the template, you should explicity state what modifications were made.

1 Project Drivers

1.1 The Purpose of the Project

The purpose of this software application is to provide a comprehensive suite of tools for generating, recognizing, and solving Sudoku puzzles. The application will provide a web-based front-end with an intuitive interface to cater to users of different technical abilities on most modern hardware. Computer vision is also utilized to improve ease of use, by directly interfacing puzzles from print-media to the application.

1.2 The Stakeholders

1.2.1 The Client

The client of the project are the instructor, Dr. Asghar Bokhari, and teaching assistants of SFWRENG 3XA3. The clients stipulate the content and deadlines of the deliverables.

1.2.2 The Customers

The customers are any individuals with an interest in Sudoku. These includes hobbyists who seek to play the game or verify their solutions. The application is suitable for all demographics who can operate a web browser.

1.2.3 Other Stakeholders

Other stakeholders of the project include math teachers who may print generated puzzles from the application to use in educational setting. Tim Ruscica, the original developer of the open source Sudoku GUI Solver upon which this project is based on, also has a stake in seeing the expansion of features that stem from his work. Additionally, members of the SudoCrew development team are also stakeholders responsible for implementing and testing the application.

1.3 Mandated Constraints

1.3.1 Solution Design Constraints

Description: The game must operate on any browser with JavaScript enabled.
Rationale: Potential users of the game will have access to a browser that can render JavaScript elements.
Fit Criterion: The game will be made to operate on any browser with JavaScript enabled.

1.3.2 Implementation Environment of the Current System

N/A

1.3.3 Partner or Collaborative Applications

There are two main libraries used in the program; Flask and OpenCV. OpenCV is an API that allows for image recognition while Flask is a web framework that allows developers to build web applications.

1.3.4 Off-the-Shelf Software

N/A

1.3.5 Anticipated Workplace Environment

The Internet with PC, laptop, mobile device, or tablet as a browsing device.

1.3.6 Schedule Constraints

Description: The project must follow the project schedule shown in the Tasks section.
Rationale: The project must follow a predetermined plan in order to meet deliverable due dates on time.
Fit Criterion: The project will follow the project schedule shown in the Tasks section.

1.3.7 Budget Constraints

N/A

1.3.8 Enterprise Constraints

N/A

1.4 Naming Conventions and Terminology

Table 2: **Table of Naming Conventions and Terminology**

Terminology	Meaning
UI	User interface, the interface that allows for user interaction with the system.
Python	Language used for backend and frontend implementation through Flask.
HTML	Hypertext markup language, used to build websheets for the frontend part of the web application.
CSS	Cascading style sheets, used to style HTML files to make the application look more appealing.
JavaScript	Primary language used for scripting and functionality of the web application.
Git	The primary system for version control and code collaboration.
OpenCV	An open source computer vision library, and serves as the main Python library that will be used to detect Sudoku boards.
Pytest	A Python library designed to help create tests for Python code.
Pydoc	A Python library used to generate documentation for Python code.
Flask	A micro web framework for Python, and will mainly be used to create our web application and handle requests from the web.

1.5 Relevant Facts and Assumptions

1.5.1 Facts

1. The original repository for the existing Sudoku solver has approximately 400 lines of code spread across 3 Python files.

2. The existing Sudoku solver solves hardcoded Sudoku boards and displays the solution in the command line.

1.5.2 Assumptions

1. The user has a stable Internet connection to use the web application.

2. The Sudoku board uploaded is a 9x9 Sudoku board.

2 Functional Requirements

2.1 The Scope of the Work and the Product

2.1.1 The Context of the Work

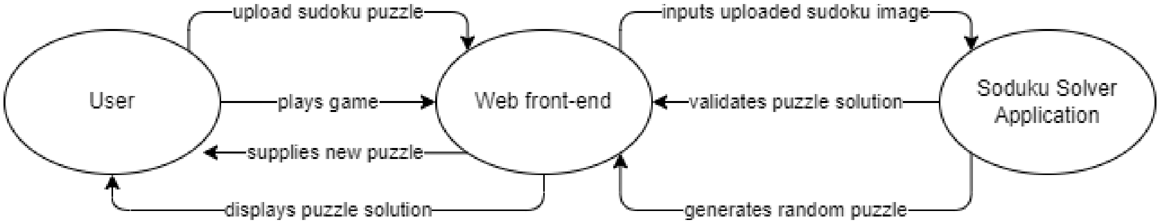

Figure 1: Work Context Diagram: The user either uploads a manually inputted or photographed Sudoku to receive feedback on a valid solution, or plays the game on a randomly generated puzzle grid. The Web front-end handles user input and interactive game-play while passing and receiving solution data to the back-end solver application.

2.1.2 Work Partitioning

Table 3: **Table of Work Partitioning Events**

Event Number	Event Name	Input	Output
1	Arrives at Homepage	Mouse	Webpage Generation
2	Uploading a sudoku board through an image	Mouse	Sudoku Board Solution
3	Uploading a sudoku board through manual input	Keyboard/Mouse	Sudoku Board Solution
4	Start a game of Sudoku	Mouse	Sudoku Board Generation

2.1.3 Individual Product Use Cases

2.2 Functional Requirements

BE1. The user arrives at the main application portal (homepage)

FR1. The system must display links to play or get solution for a Sudoku game.

FR2. The system shall display options for manual input or photo upload for the Sudoku solver.

FR3. The system shall display an error if user browser is incompatible with the application.

BE2. The user uploads a Sudoku board through an image

FR4. The system shall display a loading indicator to the user to show that their picture is being uploaded and solved.

FR5. The system shall attempt to interpret the uploaded picture as a 9x9 Sudoku board modelled as an array.

FR6. If the system fails to interpret a 9x9 Sudoku board from the uploaded image, the system shall inform the user that the picture could not be interpreted.

FR7. The system shall input the 9x9 array into the Sudoku solver algorithm to solve the Sudoku board.

FR8. If the system finds a solution to the Sudoku board, the system shall display the solution in the user interface.

FR9. If the system does not find a solution, the system shall notify the user that the Sudoku board is invalid.

BE3. The user uploads a Sudoku board through manual input

FR10. The system should display a loading indicator to inform the user that the Sudoku board is being solved.

FR11. The system should use the Sudoku solver algorithm to solve the inputted Sudoku board.

FR12. If the system fails to find a solution to the Sudoku board, the system should inform the user that the Sudoku board is invalid.

FR13. If the system finds a solution, the system should display the solution to the user through the user interface.

BE4. The user starts a game of Sudoku

FR14. The system should generate a random valid Sudoku board.

FR15. The system should output the generated Sudoku board to the user through the user interface.

3 Non-functional Requirements

3.1 Look and Feel Requirements

3.2 Usability and Humanity Requirements

3.3 Performance Requirements

3.4 Operational and Environmental Requirements

3.5 Maintainability and Support Requirements

3.6 Security Requirements

3.7 Cultural Requirements

3.8 Legal Requirements

3.9 Health and Safety Requirements

This section is not in the original Volere template, but health and safety are issues that should be considered for every engineering project.

4 Project Issues

4.1 Open Issues

4.2 Off-the-Shelf Solutions

4.3 New Problems

4.4 Tasks

4.5 Migration to the New Product

4.6 Risks

4.7 Costs

4.8 User Documentation and Training

4.9 Waiting Room

4.10 Ideas for Solutions

5 Appendix

This section has been added to the Volere template. This is where you can place additional information.

5.1 Symbolic Parameters

The definition of the requirements will likely call for SYMBOLIC_CONSTANTS. Their values are defined in this section for easy maintenance.