SE 3XA3: Software Requirements Specification
Sudoku Solver

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

February 14, 2022

Contents

List of Tables

List of Figures

Table 1: **Revision History**

Date	**Version**	**Notes**
2022-02-07	0.0	Drafted section 1
2022-02-09	0.1	Completed sections 1, 2, and 4
2022-02-10	0.2	Completed section 3 and Appendix

This document describes the requirements for a Sudoku Solver. The template for the Software Requirements Specification (SRS) is a subset of the Volere template (**?**).

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 geared toward image recognition while Flask is a web framework that allows developers to build web applications using Python.

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 back-end and front-end implementation through Flask.
HTML	Hypertext markup language, used to build websheets for the front-end 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 500 lines of code spread across 3 Python files.

2. The existing implementation solves only contains a single static Sudoku board and displays the solution as a plain-text output.

1.5.2 Assumptions

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

2. The user knows the basic basic rules of Sudoku.

3. If the image upload option is to be used, the user has a legible photo of the whole puzzle board.

2 Functional Requirements

2.1 The Scope of the Work and the Product

2.1.1 The Context of the Work

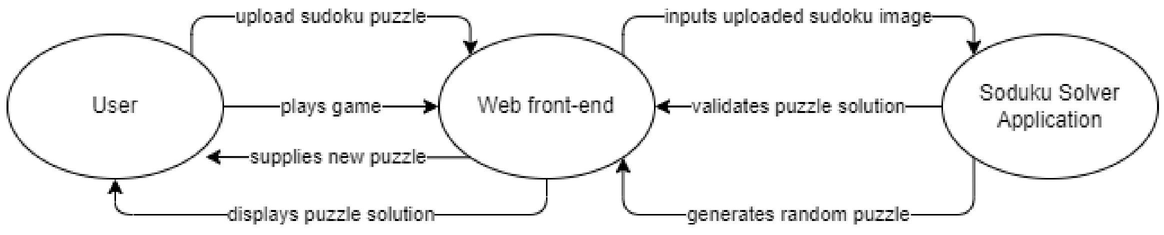

Figure 1: Work Context Diagram

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	Web-page 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

Table 4: **Table of Work Partitioning Summaries**

Event Number	**Summary**
1	The user, through a mouse input, arrives at the web-application. The output shown is the generated web-page with every important feature.
2	The user, through a mouse input, selects a file containing a Sudoku board and uploads it. The output is the solution of the given Sudoku board.
3	The user, through mouse input and keyboard strokes, manually inputs a Sudoku board. The output is the solution of the given Sudoku board.
4	The user, through mouse input, starts a game of Sudoku. The output shown is a Sudoku board where the user can now play a game of Sudoku.

2.1.3 Individual Product Use Cases

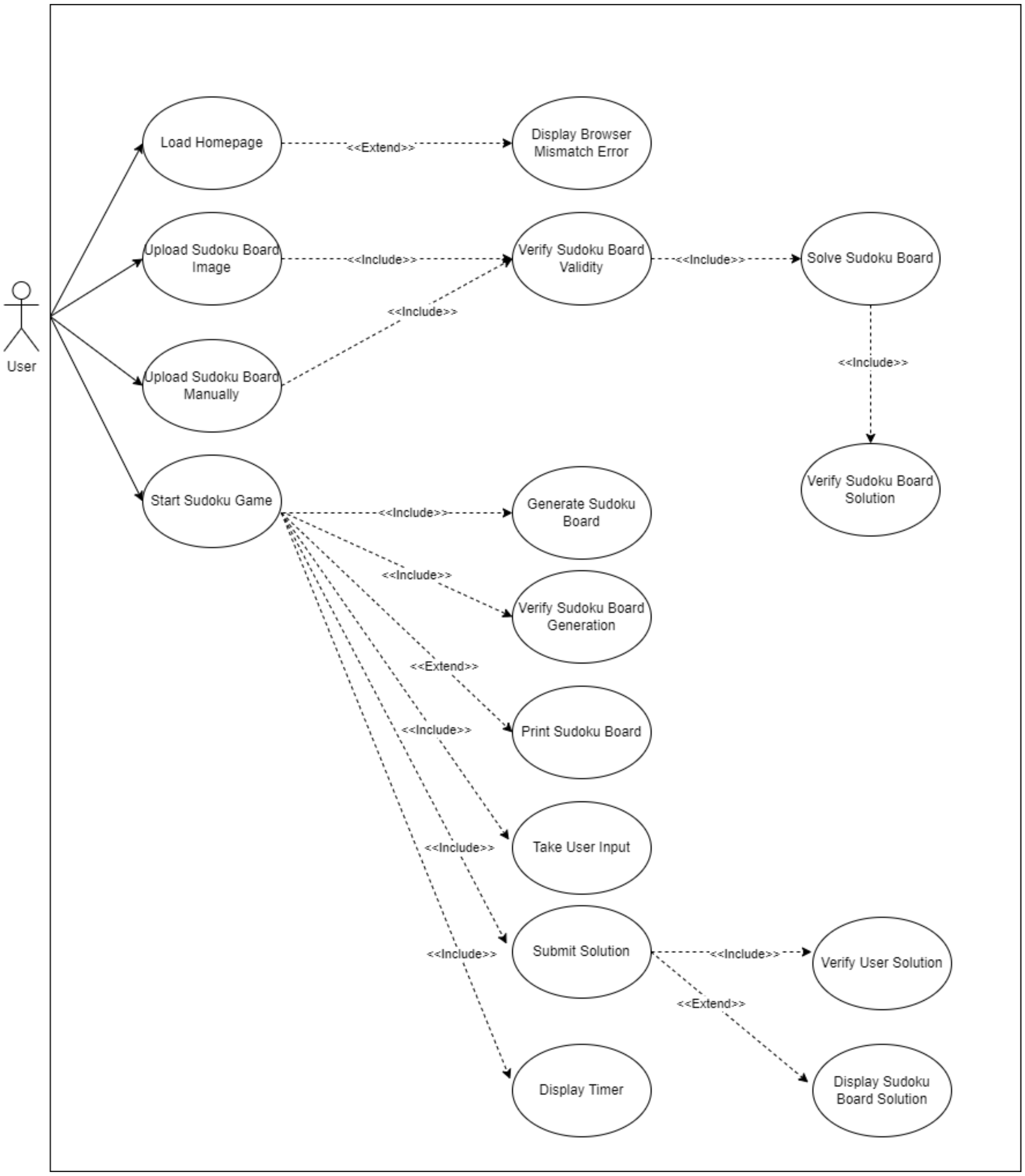

Figure 2: Use Case Diagram

The use case diagram above shows the various ways a user can interact with our web application. Since Starting a Sudoku game involves multiple game-play actions itself, it has a lot of included use cases. Furthermore, there are multiple levels of extends cases when verifying and Sudoku board solutions as each complex use case is broken into simpler 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 must 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 must notify the user that the Sudoku board is invalid.

BE3. The user uploads a Sudoku board through manual input

- FR10. The system shall display an empty Sudoku grid
- FR11. The system must allow the user to type numbers 1 to 9 into the grid.
- FR12. The system must immediately notify the user of invalid input, including repeated numbers in a row, column, or subgrid.
- FR13. The system shall attempt to solve an outwardly valid board.
- FR14. If the system fails to find a solution to the input board, it must inform the user that the puzzle has no solution.
- FR15. If the system finds a solution, it must display the solution to the user through the user interface.

BE4. The user starts a game of Sudoku

- FR16. The system must generate a random valid Sudoku board.
- FR17. The system must output the generated Sudoku board to the user through the user interface.
- FR18. The system shall allow the user to print the generated puzzle.
- FR19. The system shall allow the user to type numbers into empty cells on the Sudoku board.
- FR20. The system must reject invalid user input (non-digits, repeated numbers on row, column, or subgrid) and notify user of the error.

FR21. The system shall allow the user to submit the puzzle once all grid cells are filled.

FR22. The system shall validate the submitted solution and display whether the solution is correct.

FR23. The system shall display a timer to track the duration from puzzle generation to the submission of a correct solution.

3 Non-functional Requirements

3.1 Look and Feel Requirements

3.1.1 Look Requirements

LF1. The system shall have distinct buttons for uploading a Sudoku board, manually typing in a Sudoku board, or playing a game of Sudoku.

LF2. The system shall scale appropriately based on browser window size.

3.1.2 Feel Requirements

LF3. The system shall have a sleek and non-cluttered user interface.

LF4. The user interface shall have colours that are appealing and are accessible to people with potential colour blindness.

3.2 Usability and Humanity Requirements

3.2.1 Ease of Use Requirements

UH1. The system shall accept Sudoku boards through images or manual input with guided help.

UH2. The system shall allow game-play through both a keyboard and a mouse.

3.2.2 Ease of Learning Requirements

UH3. The system shall provide instructions on how to play a game of Sudoku.

UH4. Users shall be able to play a game fairly quickly without prior experience.

3.2.3 Accessibility Requirements

UH5. The colour palette must be accessible to users with colour blindness.

UH6. The game instructions shall be written in the English language using the British English standard.

3.3 Performance Requirements

3.3.1 Speed Requirements

PR1. Upload of pictures to the system shall take a maximum of *MAX_UPLOAD_TIME* seconds.

PR2. The system shall solve a given Sudoku board within a time-frame of *MAX_SOLUTION_TIME* seconds.

3.3.2 Safety-Critical Requirements

N/A

3.3.3 Precision Requirements

PR3. The system shall output the proper solution to the inputted Sudoku board.

PR4. The system shall accurately determine whether or not a given Sudoku board is valid.

3.3.4 Reliability and Availability Requirements

N/A

3.3.5 Capacity Requirements

PR5. The system shall not store any images uploaded by the user.

3.4 Operational and Environmental Requirements

PR6. The system shall be able to operate on any browser capable of running JavaScript.

3.5 Maintainability and Support Requirements

MA1. All source code must be fully documented, via rigorous commenting.

MA2. All source code must adhere to the same standard style.

MA3. The project's main repository shall be made public to allow the stakeholders view access to the project.

3.6 Security Requirements

3.6.1 File Integrity Requirements

SR1. Uploaded pictures shall be deleted by the system after used by the Sudoku Solver.

SR2. The system shall have a strict file size limit.

3.6.2 Access Requirements

SR3. The user must have read-only access to provided solutions for Sudoku boards.

3.6.3 Privacy Requirements

SR4. Uploaded pictures shall not be accessible to any user other than the system itself.

3.6.4 Audit Requirements

N/A

3.6.5 Immunity

SR5. The system must not be vulnerable to attacks from intruders.

3.7 Cultural Requirements

CR1. The user interface shall not display any culturally insensitive imagery.

3.8 Legal Requirements

N/A

3.9 Health and Safety Requirements

N/A

4 Project Issues

4.1 Open Issues

Sudoku recognition with computer vision may constitute a major challenge due to the large variety of factors that can affect image recognition, such as photo resolution, sharpness, exposure, angle, and colour balance.

The image upload feature may also expose the system to potential attacks in the form of malicious file content.

4.2 Off-the-Shelf Solutions

OpenCV provides a suite of image processing functions such as adaptive thresholding and colour conversion that can be used to preprocess uploaded photos and improve their recognizability.

4.3 New Problems

Higher traffic to the web application and the increased calls to the image recognizer and solver may increase the load on the system and potentially delay response times or cause crashes.

4.4 Tasks

Task schedules and delegated responsibilities are outlined in the project Gantt Chart.

4.5 Migration to the New Product

The project will initially be hosted on GitLab and migrate to a hosting service such as Heroku for deployment of the proof-of-concept and final product.

4.6 Risks

The application does not collect user information and so does not expose users to data security risks. As a web application, the main risk to the system are higher-than-expected traffic overwhelming the system through high processing load, or malicious attacks including denial-of-service and viral uploads.

4.7 Costs

The project does not involve monetary cost as it uses only open-source resources. A labour cost of between 5-10 hours per week is expected from each group member during the development period outlined in the Gantt Chart.

4.8 User Documentation and Training

4.8.1 Documentation

Basic instructions and rules of Sudoku will be outlined in the game interface. The user interface will also inform the user of any errors and provide remedy instructions, such as in the case of an unrecognizable photo, invalid input, or unsolvable uploaded puzzle.

4.8.2 Training

No training is required to use this application beyond the basic knowledge of using a web browser.

4.9 Waiting Room

Additional quality-of-life features, given extra development time, may include:

1. Account system that saves player statistics such as solved puzzles and past uploads.
2. Multiplayer challenge system that pits two players against each other on the same puzzle to improve the fun, competitive aspect of the game.

4.10 Ideas for Solutions

N/A

5 Appendix

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.

$MAX\_UPLOAD\_TIME = 3$
$MAX\_SOLUTION\_TIME = 2$