Software Requirements Specification

for

Uncover the Rubik's Cube Project-Team 2

Boston University - Software Engineering

Prepared by Chandana Nandan

Date - 03 October 2023

Table of Contents

1. Introduction

1.1 Purpose -
The purpose of SRS document outlining requirements, functionalities, and specifications of Rubik's Cube Solver and Simulator project. This serves as reference for project stakeholders, including developers, UI , and QA, to understand the project's objectives and constraints.

1.2 Scope -
The scope encompasses development of a web based application that simulates and solves Rubik's Cubes. With interactive 3D Rubik's Cube simulations, Rubik's Cube solving algorithms, user account management, a hint system and easy user interface. The application targets Rubik's Cube enthusiasts, beginners, and educators, offering an engaging and educational experience.

1.3 Definitions -

- Rubik's Cube- A three-dimensional combination puzzle with colored squares that users must arrange to match each face's color.
- SRS (Software Requirements Specification)- A document that defines the requirements, functionalities, and constraints of a software project.
- UI (User Interface)- The visual and interactive elements of the application that users interact with.
- 3D Simulation- The representation of a Rubik's Cube in a three-dimensional digital environment.
- Algorithm- A set of rules or steps used to solve a Rubik's Cube.

1.4 References -

- Project Proposal- The final project proposal document outlines the project's goals, objectives, and team structure.
- SCMP (Software Configuration Management Plan)- The SCMP document provides guidance on version control, documentation management, and project collaboration tools.
- Agile Methodology- The project follows Agile principles for iterative development and customer collaboration.

1.5 Overview -

The Rubik's Cube Solver and Simulator project is an innovative web application designed to replicate the experience of solving a Rubik's Cube digitally. Users can manipulate a virtual 3D Rubik's Cube, apply solving algorithms, manage their progress, and access a hint system. This project showcases advanced web development technologies, including React.js, Three.js, and Python, offering an educational and engaging platform for Rubik's Cube enthusiasts. The scope encompasses various functional and non-functional requirements to ensure a high-quality and secure user experience.

2. General Description

2.1 Product Perspective -

The Rubik's Cube Solver and Simulator project is a standalone web application that does not rely on external systems or dependencies. It gives unique and interactive user experience.
The application operates in a web browser, making it accessible to a wide range of users.

2.2 Product Features -

- The primary features of the application include an interactive 3D Rubik's Cube simulation, Rubik's Cube solving algorithms, user account management, and a hint system.
- The simulation allows users to manipulate the cube realistically, while the solving algorithms provide step-by-step solutions.
- User accounts enable progress tracking and the hint system offers guidance at various skill levels.

2.3 User Classes and Characteristics -

The application targets three main user classes-

- Rubik's Cube Enthusiasts - Experienced solvers seeking a digital platform for practice and algorithm reference.
- Beginners - Individuals new to Rubik's Cube solving, looking for a supportive learning environment.
- Educators - Teachers and educational institutions interested in using the application as a teaching tool for algorithmic thinking.

2.4 Operating Environment -

- This operates in a web browser environment, compatible with modern browsers such as Google Chrome, Mozilla Firefox, and Microsoft Edge.
- It will be hosted on a web server with sufficient computing resources to ensure smooth 3D cube rendering while user interaction.

2.5 Design and Implementation Constraints -

Design and implementation constraints refer to limitations and conditions that influence the development and design of the Rubik's Cube Solver and Simulator. These constraints are crucial to consider as they impact various aspects of the project, including the choice of technologies, development methodologies, and user experience.

a. Technology Stack Constraints -

The project constraints dictate use of specific technologies and tools. For instance, the frontend development relies on React.js for its user interface and Three.js for 3D cube simulations.
The backend is implemented using Flask/Django, and algorithms are processed in Python. These technology choices are essential for achieving the desired functionality and user experience but may limit flexibility in selecting alternative technologies.

b. Compatibility Constraints -

To provide seamless user experience project must be compatible with range of devices and web browsers. Designing a responsive user interface make sures that users can access and actively interact with application on various screen sizes and platforms. Meeting compatibility constraints requires careful design and testing to address potential issues that may arise across different devices and browsers.

c. Development Methodology Constraints -

The project follows agile development methodologies, allowing for iterative development and continuous improvement based on user feedback. However, this approach also imposes constraints related to project planning, sprint durations, and frequent user involvement. These constraints influence the project's overall timeline and development process.

d. Algorithm Efficiency Constraints -

Efficiency is a critical constraint for the Rubik's Cube solving algorithm. The algorithm must provide optimal solutions for various cube states while maintaining a reasonable processing time. Achieving this balance between accuracy and performance is a significant challenge that affects the algorithm's design and implementation.

e. Security and Privacy Constraints -

Ensuring security and privacy of user data is our priority.. Passwords must be securely stored and transmitted(login credentials), and application must protect against common web security threats.

Compliance with security standards and best practices is essential to address these constraints effectively.

f. Testing and Quality Assurance Constraints -

Meeting quality requirements and identifying and resolving bugs or usability issues require rigorous testing. Testing constraints include the allocation of resources for testing, defining test cases, and conducting thorough testing throughout the project's lifecycle.

2.6 User Documentation -

Comprehensive user documentation will be provided, including user guides and tutorials for operating the application, solving Rubik's Cubes, and utilizing the hint system.
The documentation will be easily accessible within the application and on the project website.

2.7 Assumptions and Dependencies -

The project assumes that users have basic internet connectivity and access to modern web browsers. It depends on external libraries and frameworks, including React.js for frontend development, Three.js for 3D simulations, and Python for algorithm implementation. Additionally, the project relies on secure and reliable hosting services for web deployment.

3. Specific Requirements

3.1 External Interface Requirements -

3.1.1 User Interfaces -
The user interface (UI) of application will be intuitive, and responsive. It will feature 3D Rubik's Cube simulation that users can manipulate with realistic / real time movements. The UI will include controls for cube rotation, scrambling, resetting, and accessing solving algorithms.
A navigation menu will provide access to user account management and the hint system.
The UI design will prioritize user experience, offering a seamless and engaging Rubik's Cube-solving environment.

3.1.2 Hardware Interfaces -
The application requires standard hardware components, including computers, tablets, or smartphones with internet connectivity. It will leverage the hardware capabilities of these devices for rendering 3D graphics and providing responsive user interactions.

3.1.3 Software Interfaces -

The application will operate within web browsers, making use of JavaScript for frontend interactivity. It will communicate with a backend server implemented using Flask/Django for user account management and algorithm processing. Additionally, the application will rely on external libraries such as React.js and Three.js for frontend development and 3D simulations.

3.1.4 Communication Interfaces -
Communication between the frontend and backend will occur through HTTP/HTTPS protocols. Secure communication will be established for user account data, ensuring data privacy and integrity.

3.2 Performance Requirements -
The application ensures smooth 3D cube rendering and responsive user interactions. Load times for 3D simulations and algorithm processing will be minimal. The system will handle concurrent user interactions without performance degradation.

3.3 Security Requirements -
User data, including login credentials and profile information will be securely stored and transmitted. Passwords will be hashed and salted. The application will protect against common web security threats, such as SQL injection and cross-site scripting (XSS).

3.4 Quality Requirements -
The UI will meet high-quality standards, offering an aesthetically pleasing and intuitive design. User interactions will be smooth and responsive. The application will undergo rigorous testing to identify and resolve bugs or usability issues.

3.5 Constraints -
The project will adhere to allocated time schedule . As mentioned Development follows good practices for web development, UI/UX design, and algorithm implementation. The application design will prioritize responsiveness, ensuring compatibility with various devices and screen sizes.

4. Functional Requirements

4.1 Interactive 3D Rubik's Cube Simulation -

- The application will provide users with 3D Rubik's Cube that accurately simulates real world cube movements.
- Users should be able to rotate the cube in all directions, scramble it, reset it to the solved state, and interact with it seamlessly.

4.2 Rubik's Cube Solving Algorithm -

- The application will implement Rubik's Cube solving algorithms, allowing users to input cube configurations and receive step-by-step solutions.
- The solving algorithm will be efficient and provide optimal solutions for various cube states.

4.3 User Account Management -

- Users can create accounts, log in, and manage their profiles. User accounts will store progress, achievements, and user-specific data.
- Passwords will be securely stored and authenticated.

4.4 Hint System -

- The hint system will provide users with hints and guidance for solving Rubik's Cubes at different skill levels.
- Users can access hints for specific cube states and receive step-by-step instructions.

5. Non-functional Requirements

5.1 Performance Requirements -

Performance Efficiency (Responsiveness) - The application will ensure high performance and responsiveness. This includes minimizing load times for 3D cube simulations, Rubik's Cube solving algorithms, and all user interactions. Users will experience seamless and quick responses to their inputs, enhancing their overall experience.

Scalability - The system will be designed to handle increasing numbers of users and concurrent interactions without significant performance degradation.
As more users engage with the application, we continue to provide consistent and responsive experience.

5.2 Security Requirements -

Data Security - login credentials and profile information will be stored ,transmitted securely. Password encryption and secure communication protocols will be implemented to protect sensitive user data from unauthorized access.

Password Security - Passwords will be securely managed. This involves hashing passwords before storage to ensure that even in event of a data breach, plain-text passwords cannot be exposed.

Threat Protection - The application is fortified against common web security threats, such as SQL injection, cross-site scripting (XSS).
Regular security audits and assessments will be conducted to identify and mitigate vulnerabilities.

5.3 Quality Requirements -

User Interface Quality - The user interface (UI) meets high quality standards. It provides an aesthetic design, ensuring that users find it visually appealing and user-friendly.
The UI will be intuitive and easy to navigate, enhancing the user experience.

User Interaction - User interactions will mainly be responsive. This includes swift response times to user inputs, fluid animations for cube manipulation, and seamless transitions between different sections of the application.
A positive and engaging user interaction is essential for the solver to be involved while solving.

Quality Assurance - The application will undergo testing throughout its development.Testing aims to identify and rectify any bugs, glitches, or usability issues. Quality assurance efforts ensure that the final product is stable, reliable, and delivers a consistent user experience.

5.4 Constraints -

Compatibility - The application design prioritizes responsiveness to ensure compatibility with various devices and screen sizes. It will be accessible and usable across all platforms, including desktop computers, tablets, and smartphones. This constraint ensures that users can enjoy the application seamlessly, despite of their choice of device.

Time Constraints - Time constraints refer to limitations on the amount of time available to complete a project. In the context of your Rubik's Cube Solver and Simulator project, time constraints are critical for several reasons -

5.5 Risks
Fixed Delivery Deadline - The project may have a fixed delivery deadline, which must be met to ensure the product's availability when needed.

Iterative Development - Agile methodologies, such as those used in your project, break the development process into fixed timeframes known as sprints. Each sprint has a limited duration, typically a few weeks, during which specific features must be delivered.

Contingency Planning - Including contingency time in the timeline accounts for unforeseen issues or scope changes that may arise, helping mitigate risks and maintain project progress.

Communication Constraints - Effective communication is essential for successful project management and development. Communication constraints refer to limitations in the exchange of information among project stakeholders -

Team Collaboration - Constraints may arise when team members are geographically dispersed or in different time zones, requiring digital communication tools and scheduling coordination.

User Feedback - In Agile projects like yours, user feedback is key. Ensuring users are available for regular feedback sessions or reviews can be challenging.

Documentation and Reporting - Timely and clear documentation of project details, progress, and issues is vital, but constraints may occur due to tool limitations.

Language and Terminology - Differences in language or terminology among team members may require translation or clarification to ensure mutual understanding.

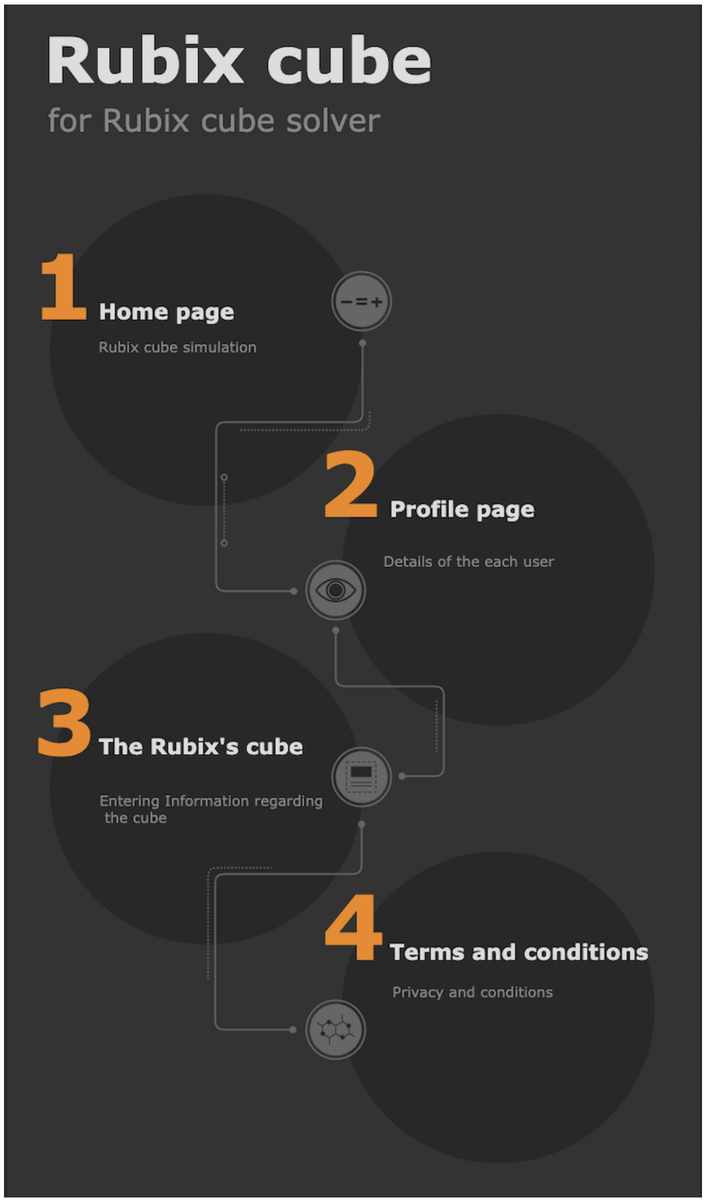
These non-functional requirements are integral to the successful development and deployment of the Rubik's Cube Solver and Simulator.
They collectively define the performance, security, quality, and constraints that shape the user experience and ensure the project's overall success.

6.User Stories -

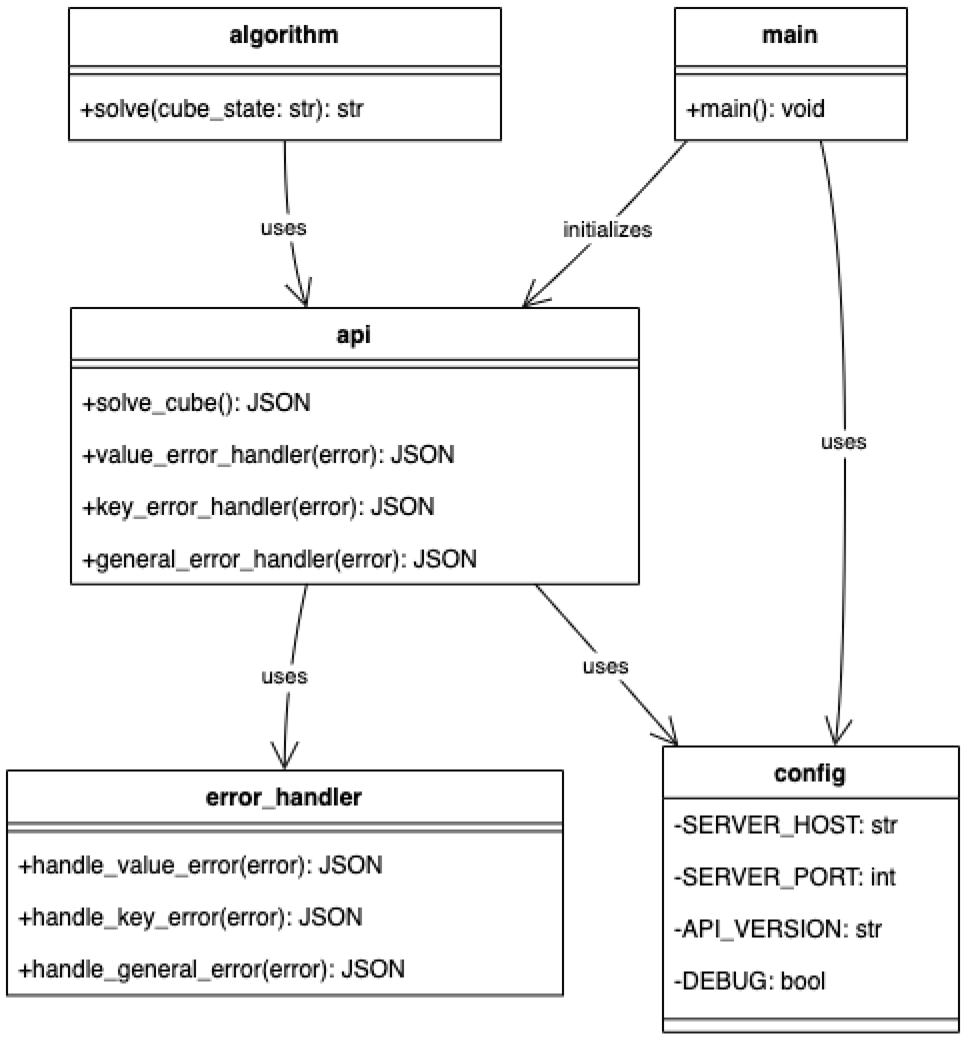
Epic	Feature	Story Description	As a User	I want to Goals/Actions	So that Value/Benefits	Acceptance Criteria
User Authentication	User Login	User Login	Registered User	Log into my account with my username/email and password	Access the Rubik's Cube platform	User can successfully log in with correct credentials.
			Forgetful User	Reset my password	Regain access to my account	User receives a password reset email and can log in with the new password.
	User Registration	User Registration	New User	Register for a new account	Start using the Rubik's Cube platform	User receives a confirmation email upon successful registration. User can click "Register" button after entering all necessary information.
			New User	Enter my name, email, account name, and password	Create a personalized account with my information	User's password is masked (displayed as dots). User receives a confirmation email upon successful registration.
User Profile Management	View Account Info	View Account	Logged-in User	View my account details	Verify my information	User can see their username, email, and profile picture.
Edit Profile Info	Edit Profile		Logged-in User	Edit my profile details such as username, email, and profile picture	Keep my account details up to date	User can modify their username, email, and profile picture.
Change Password	Change Password		Logged-in User	Change my account password	Enhance the security of my account	User can successfully log in after changing the password.

Rubik's Cube Experience	Interact with Cube	Solve Rubik's Cube	Rubik's Cube Enthusiast	Interact with the virtual Rubik's Cube, follow instructions, and practice solving	Practice solving and learn algorithms	User can manipulate the virtual Rubik's Cube and follow solving instructions.
Track Progress	Track Progress		Logged-in User	Track my solving progress and view completion time	Measure my improvement over time	User can view their solving time, successful solves, and challenge levels.
Platform Guidelines	Read Rules and Regs	Read Guidelines	Logged-in User	Access the Rules and Regulations page	Understand platform usage guidelines	User can read the guidelines, terms, and conditions presented on the page.
Accept Terms	Accept Terms		New User	Accept terms and conditions	Create an account and use the platform	User can complete registration after accepting terms and conditions.
Title Bar & Navigation	Navigation Bar	Navigate Between Pages	All Users	Use the title bar to switch between Login, Profile, Rubik's Cube, and Rules and Regulations pages	Easily access different sections of the application without confusion	Clicking on any page title in the title bar correctly navigates the user to the corresponding page. The active page title is visually highlighted.
			Logged-in User	Access user-specific options from the title bar, such as editing profile and logging out	Manage account and personalize experience	Clicking on the user's name in the title bar opens a dropdown menu with account-related options. The user can successfully log out from the dropdown menu.

7. GUI DESIGN process showing number of pages with their details.



8 .Data Diagram explaining the structure-


algorithm
+solve(cube_state: str): str
main
+main(): void
uses
initializes
uses
api
+solve_cube(): JSON
+value_error_handler(error): JSON
+key_error_handler(error): JSON
+general_error_handler(error): JSON
uses
uses
error_handler
+handle_value_error(error): JSON
+handle_key_error(error): JSON
+handle_general_error(error): JSON
config
-SERVER_HOST: str
-SERVER_PORT: int
-API_VERSION: str
-DEBUG: bool

9.Time Estimation for Rubik's Cube Solver and Simulator Project

Backend Development - Approximately 4-6 weeks

This includes implementing solving algorithms, user account management, and API development.

Frontend Development - Approximately 4-6 weeks

Creating the 3D Rubik's Cube simulation, user interface, and integrating user account features.

Design and UI/UX - Approximately 3-4 weeks

Developing an aesthetically pleasing and intuitive user interface, including dynamic elements.
Testing and Quality Assurance - Approximately 2-3 weeks

Rigorous testing to identify and rectify bugs and usability issues.

Documentation - Ongoing throughout the project
Continuously updating the SRS document, user guides, and necessary technical documentation.

Deployment and Maintenance - Ongoing

Continuous monitoring,

Allotting extra time for unforeseen issues, scope changes, or delays.