VIT

Vellore Institute of Technology

To Professor Brindha

By Laabh Gupta
22BAI1328

Software Requirement Specification (SRS)

Table of Contents

1. Introduction

1.1 Purpose

The purpose of this SRS document is to provide a detailed description of the Sentiment Analysis Dashboard project. It includes the functional and non-functional requirements, system features, and external interface requirements. This document serves as a guide for the development team and stakeholders to ensure a clear understanding of the project scope and objectives.

1.2 Scope

The Sentiment Analysis Dashboard is a web-based application that allows users to input text, analyze its sentiment using a machine learning model, and manage the analyzed sentiment records. The system will use the MERN (MongoDB, Express.js, React.js, Node.js) stack for development and integrate a Python-based sentiment analysis model using TextBlob.

1.3 Definitions, Acronyms, and Abbreviations

- **MERN:** MongoDB, Express.js, React.js, Node.js
- **ML:** Machine Learning
- **API:** Application Programming Interface
- **UI:** User Interface
- **SRS:** Software Requirement Specification
- **CRUD:** Create, Read, Update, Delete
- **UAT:** User Acceptance Testing

1.4 References

- TextBlob Documentation: https://textblob.readthedocs.io/en/dev/
- React Documentation: https://reactjs.org/docs/getting-started.html
- Express Documentation: https://expressjs.com/en/starter/installing.html
- MongoDB Documentation: https://docs.mongodb.com/

1.5 Overview

This document provides a comprehensive overview of the Sentiment Analysis Dashboard project, including its purpose, scope, and requirements. It outlines the system's functionalities, performance, and security requirements, ensuring that all stakeholders have a clear understanding of the project's objectives and deliverables.

2. Overall Description

2.1 Product Perspective

The Sentiment Analysis Dashboard is an independent web application that provides sentiment analysis functionality to users. It integrates a Python-based sentiment analysis model with a MERN stack web application, allowing users to input text, analyze its sentiment, and manage sentiment records.

2.2 Product Features

- User input for sentiment analysis
- Sentiment analysis using a machine learning model
- Display of sentiment results (positive/negative and sentiment score)
- Management of sentiment records (view, delete)
- Persistent data storage using MongoDB
- User-friendly UI built with React

2.3 User Classes and Characteristics

- **End Users:** Individuals who want to analyze the sentiment of their text inputs.
- **Administrators:** Users with the ability to manage sentiment records and ensure system maintenance.

2.4 Operating Environment

- **Frontend:** React.js application running in web browsers (Chrome, Firefox, Safari)
- **Backend:** Node.js and Express.js server
- **Database:** MongoDB
- **Machine Learning:** Python script using TextBlob

2.5 Design and Implementation Constraints

- The application must be developed using the MERN stack.
- The sentiment analysis must be performed using a Python script with TextBlob.
- The application must be deployed on a cloud platform that supports Node.js and MongoDB.

2.6 Assumptions and Dependencies

- Users have access to a modern web browser.
- The Python environment is correctly set up on the server.
- MongoDB is available and properly configured.

3. Specific Requirements

3.1 Functional Requirements

3.1.1 User Input for Sentiment Analysis

- **Description:** The system shall provide an input form for users to enter text for sentiment analysis.
- **Priority:** High
- **Stimulus/Response Sequences:** User enters text and submits the form. The system processes the text and returns the sentiment analysis results.

3.1.2 Sentiment Analysis

- **Description:** The system shall analyze the sentiment of the input text using a Python-based ML model (TextBlob).
- **Priority:** High
- **Stimulus/Response Sequences:** The system receives the input text, processes it through the ML model, and returns the sentiment results.

3.1.3 Display Sentiment Results

- **Description:** The system shall display the sentiment analysis results to the user, including the sentiment (positive/negative) and a sentiment score.
- **Priority:** High
- **Stimulus/Response Sequences:** After analysis, the results are displayed on the user interface.

3.1.4 Manage Sentiment Records

- **Description:** The system shall allow users to view and delete sentiment records.
- **Priority:** Medium
- **Stimulus/Response Sequences:** Users can view a list of analyzed texts and delete any record from the list.

3.2 Non-Functional Requirements

3.2.1 Performance Requirements

- The system should process and display sentiment analysis results within 5 seconds.
- The system should handle up to 100 concurrent users.

3.2.2 Usability Requirements

- The user interface should be intuitive and easy to use.
- The system should provide clear feedback to users for their actions.

3.2.3 Reliability Requirements

- The system should have an uptime of 99.9%.
- The system should recover gracefully from any failures.

3.2.4 Security Requirements

- User data should be securely stored and transmitted.
- The system should be protected against common web vulnerabilities (e.g., SQL injection, XSS).

4. External Interface Requirements

4.1 User Interfaces

- **Home Page:** Contains an input form for text and a submit button.
- **Results Page:** Displays the sentiment analysis results.
- **Records Page:** Lists all analyzed texts with options to delete records.

4.2 Hardware Interfaces

- The system does not require any specific hardware interfaces.

4.3 Software Interfaces

- **Frontend:** React.js
- **Backend:** Node.js and Express.js
- **Database:** MongoDB
- **Machine Learning:** Python (TextBlob)

4.4 Communications Interfaces

- The system shall use HTTP/HTTPS for communication between the frontend and backend.
- The backend shall use a MongoDB connection for database operations.

5. System Features

5.1 Text Input for Sentiment Analysis

Description: Users can input text to be analyzed for sentiment.

Functional Requirements:

- Provide an input form for text.
- Validate the input to ensure it is not empty.
- Submit the text to the backend for analysis.

Non-Functional Requirements:

- The input form should be user-friendly and responsive.
- The system should validate the input and provide feedback in real-time.

5.2 Displaying Sentiment Analysis Results

Description: The system displays the sentiment analysis results to the user.

Functional Requirements:

- Display the original text.
- Show the sentiment (positive/negative).
- Show the sentiment score.

Non-Functional Requirements:

- Results should be displayed within 5 seconds of submission.
- The UI should be clear and easy to understand.

5.3 Managing Sentiment Records

Description: Users can view and delete sentiment analysis records.

Functional Requirements:

- Display a list of all analyzed texts.
- Provide an option to delete individual records.

Non-Functional Requirements:

- The records list should be paginated for better performance.
- Deletion should be confirmed by the user to prevent accidental removal.

6. Other Nonfunctional Requirements

6.1 Performance Requirements

- The system should process sentiment analysis within 5 seconds.
- The system should support 100 concurrent users without performance degradation.

6.2 Safety Requirements

- The system should handle errors gracefully and provide meaningful error messages to the user.
- Regular backups of the database should be taken to prevent data loss.

6.3 Security Requirements

- User data should be encrypted during transmission (HTTPS).
- The system should implement secure coding practices to prevent vulnerabilities.

6.4 Software Quality Attributes

- **Maintainability:** The code should be modular and well-documented to facilitate maintenance and updates.
- **Scalability:** The system should be able to scale to accommodate more users and data if needed.
- **Usability:** The UI should be intuitive and easy to navigate.

7. Appendices

7.1 Glossary

- **Sentiment Analysis:** The process of determining the sentiment (positive or negative) of a given text.
- **MERN Stack:** A technology stack that includes MongoDB, Express.js, React.js, and Node.js.
- **TextBlob:** A Python library for processing textual data and performing sentiment analysis.
- **API:** Application Programming Interface, a set of functions and protocols for building and integrating application software.

7.2 Analysis Models

- **Use Case Diagrams:** Depict interactions between users and the system.
- **Activity Diagrams:** Show the flow of activities in the system.

7.3 Issue Tracking Log

- An issue tracking log will be maintained to record and manage all issues and bugs identified during development and testing.