Software Requirements Specification

for

Model to Diagnose Cardiac Attack

Version <1.0>

Prepared by

Group Members:

Devesh Gupta	**BT22GCS264**	**devesh.gupta22@st.niituniversity.in**
Guni Sharma	**BT22GCS070**	**guni.sharma22@st.niituniversity.in**
Raajit J Singh	**BT22GCS051**	**raajit.singh22@st.niituniversity.in**
Yash Upadhyay	**BT22GCS355**	**yash.upadhyay22@st.niituniversity.in**

Instructor: Manish Hurkat

Course: Capstone

Lab Section: C2

Date: 20 Feb'25

Contents

1 Introduction

1.1 Document Purpose

The product is a Machine Learning Model which takes various input data from the user, such as Gender (Male or Female), Age (Number) , Education, Current Smoking Status (0 or 1), Cigarettes Per Day, Blood Pressure Medications (0 or 1), Prevalent Stroke Condition (0 or 1), Prevalent Hypoglycemia Diabetes (0 or 1), Total Cholesterol Level, Systolic Blood Pressure, Diastolic Blood Pressure, BMI, Heart Rate, Glucose, Heart Stroke Chances (0 or 1). These data points help define the condition of a patient and whether or not they will suffer a heart stroke.

1.2 Product Scope

The Machine Learning Model is designed to predict the likelihood of heart stroke based on a user's health and lifestyle factors. By analyzing relevant data, the model provides a binary outcome, indicating whether an individual is at risk. This predictive tool aims to assist healthcare professionals, researchers, and individuals in early risk assessment and preventive healthcare planning.

The product enhances decision-making by offering data-driven insights that can support timely medical intervention and lifestyle adjustments. It can be integrated into healthcare applications, wellness platforms, or research studies to improve patient monitoring and risk evaluation. The model ensures accuracy, efficiency, and accessibility, making it a valuable asset in the field of preventive healthcare.

1.3 Intended Audience and Document Overview

1.3.1 Intended Audience

This Software Requirements Specification (SRS) document is intended for the following stakeholders:

- **Client:** The client will use this document to understand the scope, objectives, and functionality of the Machine Learning Model. It will help them evaluate how the model meets their requirements and how it can be integrated into healthcare or research applications.
- **Professor:** The professor will review this document for technical accuracy, completeness, and adherence to software engineering principles. They will assess whether the system’s requirements are well-defined, feasible, and appropriately structured.

1.3.2 Document Organization

This document is structured to provide a comprehensive and systematic understanding of the system. It includes the following sections:

1. **Introduction** – Provides an overview of the document, including its purpose, scope, intended audience, and key definitions.

2. **Overall Description** – Describes the product's perspective, features, constraints, and user characteristics.
3. **Specific Requirements** – Defines the functional, non-functional, and external interface requirements of the system.
4. **Appendices** – Includes any supporting documents or references necessary for further clarification.

1.3.3 Suggested Reading Sequence

- **Clients** should begin with the **Introduction** and **Overall Description** to understand the system's objectives and capabilities.
- **Professors** should review the **Specific Requirements** section to assess the completeness and accuracy of the technical specifications.
- **All readers** can refer to the **Appendices** for additional information, references, or supporting materials.

1.4 Definitions, Acronyms and Abbreviations

1.4.1 Definitions

- **System** – Refers to the heart stroke prediction software being developed.
- **User** – Any individual interacting with the system, including patients, doctors, and administrators.
- **Model** – The pre-trained machine learning model used to analyze user data and predict heart stroke likelihood.
- **Prediction** – The result generated by the system based on user input, indicating the likelihood of a heart stroke.
- **Database** – A structured system for storing user input data and prediction history.
- **Risk Classification** – The categorization of prediction results into binary outputs (0 or 1), where 1 indicates high risk of heart stroke and 0 indicates low risk.
- **Frontend** – The user interface of the system that allows interaction with the application.
- **Backend** – The server-side logic that processes data, communicates with the machine learning model, and returns results.

1.4.2 Acronyms and Abbreviations

- **AI** – Artificial Intelligence
- **API** – Application Programming Interface
- **AWS** – Amazon Web Services
- **CSV** – Comma-Separated Values
- **DBMS** – Database Management System
- **EHR** – Electronic Health Record
- **GDPR** – General Data Protection Regulation
- **GUI** – Graphical User Interface
- **HIPAA** – Health Insurance Portability and Accountability Act
- **HTTP** – Hypertext Transfer Protocol
- **JSON** – JavaScript Object Notation
- **JWT** – JSON Web Token
- **ML** – Machine Learning

- **NoSQL** – Non-Relational Structured Query Language
- **RBAC** – Role-Based Access Control
- **REST** – Representational State Transfer
- **RDBMS** – Relational Database Management System
- **SRS** – Software Requirements Specification
- **TLS** – Transport Layer Security
- **UML** – Unified Modeling Language
- **UI** – User Interface
- **UX** – User Experience

1.5 Document Conventions

1.5.1 Formatting Conventions

- The document follows **IEEE formatting requirements**.
- The primary font used throughout the document is **Arial, size 11 or 12**, for all textual content.
- Section and subsection headings follow the predefined structure of this template, using **bold formatting** for clear distinction.
- Comments or placeholders for additional information are written in **italics**.
- The document maintains **single-line spacing** and **1-inch margins** on all sides for consistency and readability.

1.5.2 Naming Conventions

- Functional requirements are prefixed with **"FR"** followed by a number (e.g., **FR1, FR2**).
- Performance requirements are prefixed with **"P"** followed by a number (e.g., **P1, P2**).
- Safety and system requirements are prefixed with **"S"** followed by a number (e.g., **S1, S2**).
- Database-related requirements are prefixed with **"D"**, while legal and compliance requirements are prefixed with **"L"**.

1.5.3 Terminology and Definitions

- **"System"** refers to the software product being developed.
- **"User"** refers to any individual interacting with the system, including patients, doctors, and administrators.
- **"Model"** refers to the pre-trained machine learning model used for predictions.
- **"Prediction"** refers to the result generated by the system based on user input.

1.6 References and Acknowledgments

1.6.1 References

This Software Requirements Specification (SRS) document is based on the following sources:

1. **IEEE Std 830-1998** – Recommended Practice for Software Requirements Specifications.
2. **User Interface Style Guide** – Internal design document defining UI components and accessibility standards.
3. **Machine Learning Model Documentation** – Technical details on model architecture, training dataset, and evaluation metrics.
4. **Database System Documentation** – PostgreSQL database schema and indexing strategy.
5. **Cloud Service Provider Documentation** – AWS EC2 and S3 configuration guidelines for deployment and storage.
6. **API Documentation** – REST API endpoints for system communication and data exchange.
7. **Legal and Compliance Standards** – General Data Protection Regulation (GDPR) and Health Insurance Portability and Accountability Act (HIPAA) compliance guidelines.

1.6.2 Acknowledgements

This document was created with contributions from the following:

- **Professor Manish Hurkat** – For providing valuable insights into software engineering principles and guiding the structuring of this SRS document.
- **Development Team: Yash Upadhyay, Raajit Singh, Guni Sharma and Gupta Devesh** – For their extensive research, discussions, and efforts in defining the functional and technical requirements of the system.
- **Beta Testers and End-Users** – Yash Upadhyay and Gupta Devesh for testing the working of the model to see how accurate it gives the result.

2 Overall Description

2.1 Product Overview

The Machine Learning Model for heart stroke prediction is designed to analyze user-provided health and lifestyle data to assess the likelihood of a heart stroke. By leveraging predictive analytics, the model provides a binary classification (0 or 1), indicating whether an individual is at risk. This system aims to assist healthcare professionals, researchers, and individuals in early risk assessment, enabling proactive medical interventions and lifestyle modifications.

The product is designed to be efficient, accessible, and scalable, making it suitable for integration into healthcare applications, wellness platforms, and research initiatives. It processes multiple health parameters, applies advanced algorithms, and delivers results in an interpretable format. The system's accuracy, usability, and reliability make it a valuable tool for improving cardiovascular risk evaluation and preventive care strategies.

2.2 Product Functionality

The Machine Learning Model for heart stroke prediction is designed to process user-input health and lifestyle data to assess the likelihood of a heart stroke. The system performs the following major functions:

- **User Data Input:** The system collects essential health-related information in terms of numbers from the user.
- **Data Validation and Preprocessing:** Ensures that the input data is correctly formatted, within acceptable ranges so that it can be processed by the product to predict accurate results. It also makes it free from inconsistencies.
- **Predictive Analysis:** Utilizes machine learning algorithms to analyze the input data and generate a stroke risk prediction based on the input given by the user about their health..
- **Risk Classification:** Produces a binary output (0 or 1), indicating whether the user is at risk of experiencing a heart stroke.
- **Result Presentation:** Displays the prediction result in an interpretable format, allowing users or healthcare professionals to take necessary actions.
- **System Integration:** Provides interfaces for integration with external healthcare applications, research platforms, or decision-support systems.
- **Security and Privacy Management:** Implements data encryption and access controls to ensure the confidentiality, authentication and integrity of user information.

These core functionalities enable the system to support healthcare professionals and individuals in early stroke risk assessment, facilitating preventive care and timely medical intervention.

2.3 Design and Implementation Constraints

The development of the heart stroke prediction system is subject to several design and implementation constraints to ensure efficiency, scalability, and compliance with industry standards. These constraints cover hardware and software limitations, required methodologies, and regulatory considerations.

2.3.1 Hardware Constraints

- The system must be optimized for **low-end devices**, ensuring accessibility for users with limited processing power and memory.
- The prediction model must execute efficiently within **2 seconds**, even on devices with **a minimum of 4GB RAM and a dual-core processor**.
- The application should be compatible across **mobile, tablet, and desktop platforms**.

2.3.2 Software and Database Constraints

- The system will use a **relational database management system (RDBMS)** such as PostgreSQL or MySQL for structured data storage.
- Machine learning models will be implemented using **Python with TensorFlow or Scikit-learn**.

- The front-end will be developed using **React.js or Angular**, while the back-end will use **Node.js or Django**.
- The system should support **cloud deployment** on platforms such as AWS, Azure, or Google Cloud.
- The software will provide **RESTful APIs** for external system integration.

2.3.3 Methodological Constraints

- The system will be designed using the **Agile Software Development Methodology**, ensuring flexibility, iterative development, and continuous user feedback.
    - **Reference:** Beck, K. Extreme Programming Explained: Embrace Change, Addison-Wesley, 1999.
- The software architecture and system behavior must be documented using **UML diagrams** such as use case diagrams, sequence diagrams, and class diagrams.
    - **Reference:** Booch, G., Rumbaugh, J., & Jacobson, I. The Unified Modeling Language User Guide, Addison-Wesley, 2005.

2.3.4 Regulatory and Compliance Constraints

- The system must comply with **healthcare industry regulations** for data management.
- It must meet **ISO/IEC 27001 (Information Security Management) standards** for data integrity and system reliability.
- The software should follow the **Open Data Protocol (OData) for secure API communication** with external healthcare platforms.

2.3.5 Performance and Scalability Constraints

- The system must support **at least 500 concurrent users** without performance degradation.
- It should implement **multi-threaded processing** for real-time predictions while optimizing CPU and memory usage.
- The software must be **cross-platform compatible**, functioning seamlessly on **Windows, macOS, Linux, Android, and iOS**.
- The UI should be **responsive and optimized for both desktop and mobile devices**.

These constraints define the technical boundaries for development, ensuring that the system meets scalability and regulatory requirements while maintaining optimal performance.

2.4 Assumptions and Dependencies

The successful development and deployment of the Machine Learning Model for heart stroke prediction rely on several assumptions and dependencies. These factors, while not confirmed, are expected to remain stable throughout the project lifecycle. Any deviation from these assumptions may impact the system's design, implementation, or overall feasibility.

2.4.1 Assumptions

- **Availability of Quality Data:** It is assumed that the training dataset used for model development is representative, accurate, and sufficiently large to ensure high predictive accuracy.
- **Stable Development Environment:** The system is expected to be developed in a controlled environment with access to the necessary **machine learning libraries, development tools, and computing resources (e.g., GPUs for model training)**.
- **User Compliance with Input Standards:** It is assumed that users will provide accurate and complete input data within the expected ranges for effective stroke risk prediction.
- **Regulatory Stability:** Healthcare regulations such as **HIPAA and GDPR** will remain unchanged during the development phase, ensuring that compliance measures do not require significant rework.
- **Third-Party Service Reliability:** If external APIs (e.g., cloud-based ML services or healthcare databases) are used, they will remain operational and provide consistent performance.

2.4.2 Dependencies

- **Machine Learning Frameworks:** The system relies on libraries such as **TensorFlow, PyTorch, or Scikit-learn** for model training and deployment. Any changes in these frameworks may impact compatibility or performance.
- **Database Management System (DBMS):** The project depends on **PostgreSQL or MongoDB** for data storage. Performance or compatibility issues in these databases may require modifications to the system architecture.
- **Cloud Infrastructure:** If cloud-based training or deployment is used, the system will depend on platforms like **AWS, Google Cloud, or Microsoft Azure**. Downtime or policy changes in these services may affect the system's availability.
- **Regulatory and Compliance Requirements:** The model's implementation is dependent on adherence to **healthcare data privacy regulations**. Any changes in compliance standards may require security and data-handling modifications.
- **Hardware Performance:** Efficient execution of the model is dependent on the availability of **sufficient processing power (CPUs and GPUs)**. Performance issues may arise if deployed on low-end hardware.

By acknowledging these assumptions and dependencies, the project can proactively mitigate risks and ensure that the system remains adaptable to potential changes in external factors.

3 Specific Requirements

3.1 External Interface Requirements

3.1.1 User Interfaces

The system will provide an intuitive and user-friendly **web-based graphical interface**, accessible through modern web browsers. The interface will be designed for ease of use, ensuring that users can efficiently input data and receive prediction results.

3.1.1.1 Main UI Components

- **Login and Authentication Screen:** If required, the system will include a secure login page with email/password authentication.
- **Data Input Form:** Users will enter necessary health parameters through labeled text fields, dropdown menus, and radio buttons.
- **Prediction Output:** The system will display results in a **clear text format**, accompanied by a **color-coded risk indicator (e.g., Green for Low Risk, Red for High Risk)**.
- **Error Handling:** Any invalid inputs will trigger **real-time validation messages**, guiding users to correct their entries.
- **Navigation & Accessibility:** The interface will follow **WCAG accessibility standards**, with proper color contrast, tooltips, and keyboard shortcuts.

3.1.1.2 UI Design Considerations

4 The UI will be built using **React.js (or any front-end framework)** for responsiveness and scalability.
5 A clean and minimalistic design will ensure ease of interaction for both medical professionals and general users.
6 The system will support both **desktop and mobile views**, ensuring adaptability across different screen sizes

3.1.2 Hardware Interfaces

The system will interact with various hardware components to ensure seamless data processing and user accessibility. The following list outlines the key hardware interfaces required for system operation.

3.1.2.1 Supported Device Types

- **Personal Computers (PCs) & Laptops**: The system will be accessible via web browsers on Windows, macOS, and Linux-based machines.
- **Mobile Devices & Tablets**: The system will be optimized for Android and iOS platforms, ensuring compatibility with touchscreen devices.
- **Cloud Computing Resources**: If deployed on cloud-based services (e.g., AWS, Google Cloud), the system will leverage virtual computing instances for model execution.
- **Database Server**: The software will interact with a relational or NoSQL database to store and retrieve patient input data and model predictions.

3.1.2.2 Data and Control Interactions

- The system will collect **user-provided input data** through the front-end interface and process it via the machine learning model.
- The **output predictions** will be displayed to the user in text-based or graphical formats.
- The **hardware interfaces will support standard data validation mechanisms** to prevent incorrect or inconsistent user inputs.

3.1.3 Software Interfaces

The system will interact with various software components to facilitate data processing, model execution, and user interaction. These software interfaces ensure seamless communication between the machine learning model, user interface, and external applications.

3.1.3.1 External Software Components

- **Web Application (Frontend):**
  - The system will feature a **web-based graphical user interface (GUI)** developed using **React.js (or an equivalent frontend framework)**.
  - It will communicate with the backend via **RESTful APIs** or **GraphQL** to send user input data and receive prediction results.
  - Supports standard web browsers like **Google Chrome, Mozilla Firefox, and Microsoft Edge**.
- **Backend Application & API Layer:**
  - Built using **Python (Flask/Django/FastAPI)** or **Node.js (Express.js)** to handle user requests, process data, and interact with the machine learning model.
  - The API will expose endpoints for **data submission, model execution, and result retrieval**.
  - Implements **authentication mechanisms (JWT/OAuth2)** for secure access control.
- **Machine Learning Model Integration:**
  - The predictive model will be developed using **TensorFlow, PyTorch, or Scikit-learn**.
  - It will be deployed as a **standalone microservice** or embedded within the backend API.
  - Model inference will occur either **locally on the server** or via **cloud-based ML services (AWS SageMaker, Google AI Platform, etc.)**.
- **Database Management System (DBMS):**
  - A **relational (PostgreSQL/MySQL)** or **NoSQL (MongoDB/Firebase)** database will be used to store patient input data, prediction history, and user credentials.
  - The database will support **data validation, indexing, and encryption** to ensure security and efficiency.

3.1.3.2 Mobile Application Interface (if applicable in future versions)

- A **mobile app** may be developed using **React Native (cross-platform) or native development (Kotlin/Swift)**.
- The mobile app will **send user health data to the backend API**, receive prediction results, and provide real-time feedback.
- Supports **push notifications** for reminders or health alerts.

3.1.3.3 Third-Party API Integrations

- **Cloud Storage & Data Processing:** Integration with **Google Cloud Storage, AWS S3, or Firebase** for storing user input data securely.
- **Medical APIs:** If required, the system may connect with external health data sources (e.g., **FHIR, Apple HealthKit, Google Fit**) to fetch real-time medical data.

The software interfaces ensure smooth interaction between different system components, providing a reliable and scalable environment for both web and mobile users.

3.2 Functional Requirements

This section details the functional requirements of the system, describing the specific tasks and operations it must perform to meet user needs. These requirements define the expected behavior of the system, ensuring that it provides accurate predictions based on input data while maintaining usability and efficiency.

3.2.1 User Authentication and Access Control

- The system shall allow users to register and log in securely using email and password authentication.
- The system shall implement role-based access control (RBAC), distinguishing between patients, doctors, and administrators.
- The system shall support password recovery via email verification.

3.2.2 User Data Input and Management

- The system shall allow users to enter required health parameters through a structured form on the web/mobile interface.
- The system shall validate user inputs to ensure they meet predefined constraints (e.g., age must be a positive number, BMI should be within a valid range).
- The system shall allow users to edit and update their previously entered health data.
- The system shall store user input data securely in a database.

3.2.3 Machine Learning Model Execution

- The system shall process the user's input data and predict heart stroke chances using a pre-trained machine learning model.
- The system shall ensure that predictions are generated in real-time, with a response time of fewer than 2 seconds.
- The system shall provide explanations of the prediction results, including which factors contributed to the final output.
- The system shall log each prediction request along with timestamps for analysis and future improvements.

3.2.4 Results Display and Interpretation

- The system shall display the prediction results in a clear and understandable format, such as:
    - **Textual output**: "Low/Moderate/High risk of heart stroke."
    - **Graphical visualization**: Charts or progress bars indicating risk factors.

- The system shall provide preventive recommendations based on the prediction results.

3.2.5 Data Storage and Management

- The system shall store all user input and prediction history securely in a database.
- The system shall automatically delete old data after a predefined period (e.g., 6 months) unless the user opts to retain it.
- The system shall comply with data privacy regulations (GDPR, HIPAA where applicable) to protect user confidentiality.

3.2.6 Reporting and Analytics

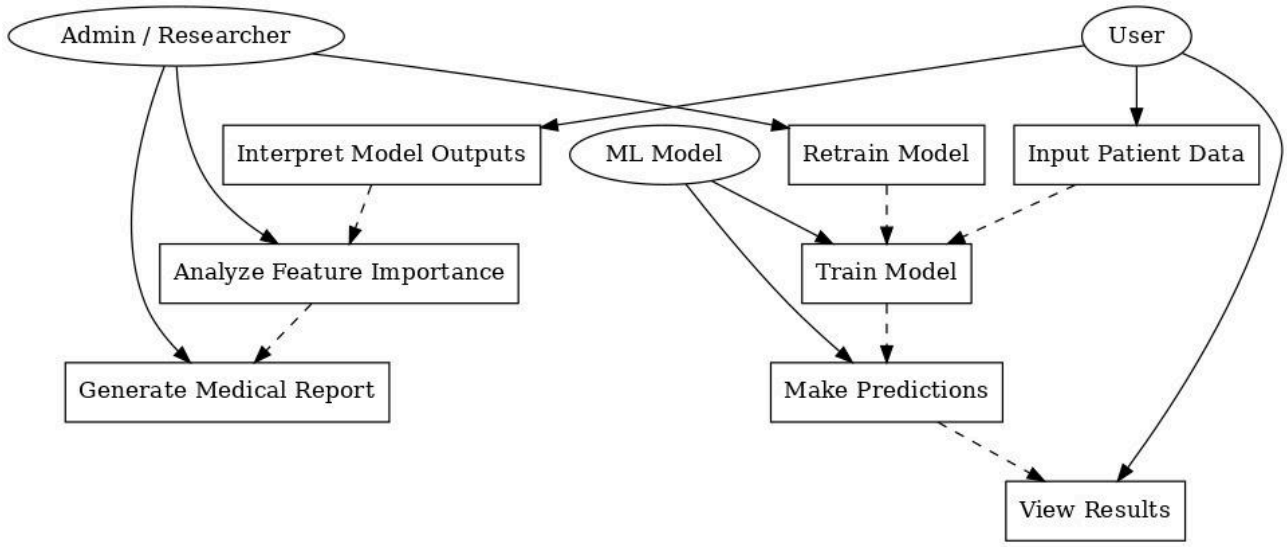
- The system shall generate statistical reports based on user health trends.
- The system shall allow users to download their prediction history in PDF or CSV format.
- The system shall provide administrators and doctors with anonymized analytics for research purposes.

3.2.7 System Performance and Availability

- The system shall be available 24/7 with 99.9% uptime on a cloud-based or on-premise server.
- The system shall handle simultaneous user requests without performance degradation (support for at least 500 concurrent users).
- The system shall provide error handling and failover mechanisms to ensure continuous operation.

These functional requirements define the core behavior of the system, ensuring a seamless and efficient user experience.

3.3 Use Case Diagram



4 Other Non-functional Requirements

4.1 Performance Requirements

The following performance requirements define the system's expected efficiency, responsiveness, and reliability under various conditions. These requirements ensure the system can handle multiple concurrent users while maintaining optimal speed and accuracy.

4.1.1 General Performance Requirements

- **P1.** The system shall provide prediction results within **2 seconds** after the user submits the required health parameters.
- **P2.** The system shall support at least **500 concurrent users** without significant performance degradation.
- **P3.** The system shall maintain **99.9% uptime**, ensuring continuous availability for users.

4.1.2 User Authentication and Data Processing

- **P4.** User login and authentication shall be processed within **1 second** under normal server load.
- **P5.** User registration shall be completed within **3-5 seconds**, including data validation and storage.
- **P6.** Retrieving and displaying previously entered user health data shall take no longer than **2.5 seconds**.

4.1.3 Machine Learning Model Execution

- **P7.** The system shall process user input and generate predictions in **less than 10 seconds** for optimal user experience.
- **P8.** The system shall maintain a prediction accuracy of at least **85%**, based on pre-trained model evaluation metrics.

4.1.3 Data Storage and Retrieval

- **P9.** The system shall retrieve stored prediction history within **5 seconds** when requested by the user.
- **P10.** The system shall allow users to download reports (PDF/CSV) within **5 seconds** for up to **1 year of stored data**.

4.1.4 System Load Handling and Availability

- **P11.** The system shall handle **peak loads** of up to **500 requests per minute** without crashing.
- **P12.** If the system exceeds the supported load, it shall **queue requests and process them in order** while displaying a message to users about potential delays.
- **P13.** The system shall automatically recover from failures within **30 seconds** by restarting affected services.

These performance requirements ensure the system remains **responsive, scalable, and reliable**, providing users with a seamless experience across various conditions.

4.2 Safety and Security Requirements

4.2.1 Safety Requirements

- **S1.** The system shall provide clear **warnings and disclaimers** stating that the predictions are **not medical diagnoses** and should be validated by a certified healthcare professional.
- **S2.** The system shall prevent data corruption by implementing **automatic error handling and validation** mechanisms.
- **S3.** If the system encounters a failure or an invalid input, it shall display **clear error messages** and allow users to retry without losing previously entered data.
- **S4.** The system shall maintain **automated backups of critical data** at regular intervals to prevent data loss due to unexpected failures.
- **S5.** The system shall not store or process **real-time medical data from external sensors**, ensuring compliance with data protection policies.

4.2.2 Security Requirements

- **SC1.** The system shall enforce **secure user authentication** for login, including a unique **email and password-based authentication** mechanism.
- **SC2.** The system shall **limit failed login attempts** to prevent brute force attacks and lock an account temporarily after **five consecutive failed attempts**.
- **SC3.** User sessions shall expire automatically after **15 minutes of inactivity**, requiring re-authentication to prevent unauthorized access.
- **SC4.** The system shall encrypt all communication between the mobile app and the backend using **TLS (Transport Layer Security) 1.2 or higher** to prevent data interception.
- **SC5.** The system shall restrict **data access based on user roles**, ensuring that only authorized personnel (e.g., doctors, administrators) can access specific information.

These safety and security requirements help ensure that the system is **reliable, secure, and compliant with necessary standards**, minimizing risks while providing accurate and efficient functionality.

4.3 Software Quality Attributes

The following software quality attributes ensure that the system meets both **user expectations and technical requirements**. These attributes are designed to enhance usability, maintainability, and overall system performance.

4.3.1 Reliability

- **R1.** The system shall maintain **99.9% uptime**, ensuring that users can access the platform at any time with minimal service disruptions.

- **R2.** The system shall implement **automated failure recovery mechanisms**, such as retry logic for failed requests and failover systems for critical operations.
- **R3.** To prevent data loss, the system shall store user input and prediction logs in a **redundant database with daily automated backups**.

4.3.2 Usability

- **U1.** The system shall have an **intuitive user interface** with a maximum of **three clicks** required to complete any primary function (e.g., submitting health data, viewing predictions).
- **U2.** The interface shall be designed following **WCAG 2.1 accessibility guidelines**, ensuring support for **screen readers and keyboard navigation**.
- **U3.** The system shall provide **real-time input validation** with immediate feedback to assist users in entering accurate data.

4.3.3 Maintainability

- **M1.** The system shall follow a **modular architecture**, ensuring that individual components (e.g., authentication, data processing, model execution) can be updated or replaced independently.
- **M2.** The system shall include **detailed documentation** covering API endpoints, database schema, and machine learning model integration to support future enhancements.
- **M3.** The codebase shall be written in a **consistent coding style** (e.g., PEP 8 for Python) and shall include **unit tests for at least 80% of critical components**.

4.3.4 Adaptability

- **A1.** The system shall be designed to **support multiple deployment environments**, including **on-premise servers, cloud services (AWS, GCP), and mobile applications**.
- **A2.** The system shall allow easy integration of **new machine learning models** by using a standardized API for model predictions.
- **A3.** If future versions require external sensor integration, the system shall provide a **generic data ingestion framework** to handle diverse input sources with minimal code changes.

4.3.5 Interoperability

- **I1.** The system shall use **RESTful APIs** to enable seamless communication between the backend, web interface, and mobile app.
- **I2.** The database shall support **structured (SQL) and unstructured (NoSQL) data** formats to accommodate different types of storage needs.
- **I3.** The system shall allow users to **export their prediction history** in **CSV and PDF formats** for easy sharing with healthcare professionals.

These quality attributes ensure that the system remains **scalable, efficient, and user-friendly**, while also being easy to maintain and extend for future requirements.

Appendix A – Data Dictionary

<Data dictionary is used to track all the different variables, states and functional requirements that you described in your document. Make sure to include the complete list of all constants, state variables (and their possible states), inputs and outputs in a table. In the table, include the description of these items as well as all related operations and requirements.>

Variable Name	Data Type	Possible Values / Description	Related Operations
Patient_ID	Integer	Unique ID assigned to each patient	Data Storage, Retrieval
Age	Integer	Patient's age in years (e.g., 18–100)	Data Input, Analysis
Gender	String	Male, Female, Other	Data Input, Classification
Blood_Pressure	Float	Measured in mmHg (e.g., 120/80)	Data Input, Risk Assessment
Cholesterol_Level	Float	Measured in mg/dL (e.g., 200)	Data Input, Risk Assessment
Heart_Rate	Integer	Beats per minute (e.g., 60-100 bpm)	Data Input, Monitoring
ECG_Results	String	Normal, Abnormal	Data Input, Diagnosis
Model_Status	String	Training, Trained, Predicting	Model Execution
Prediction_Result	String	Low Risk, Medium Risk, High Risk	Model Output
Diagnosis_Outcome	String	Confirmed Heart Attack, No Heart Attack	Doctor Review, Final Decision
Treatment_Recommendation	String	Medication, Lifestyle Changes, Surgery	Medical Recommendation
Timestamp	Date/Time	Record of when data was entered or modified	Logging, Tracking