Software Requirement Specifications

DEDOC

Version: 1.0

Project Code	
Supervisor	*Dr Farrukh Shahid*
Co Supervisor	*Shahbaz Siddiqui*
Project Team	*Sarmad (k191116)* *Mansoor(k191114)* *Khizer(k191057)*
Submission Date	*25-11-2022*

Document History

Version	Name of Person	Date	Description of change
			[e.g. Document Created]
			[Added Non-functional requirements]
			[Added Use Case x.x.xx]

Distribution List

Name	Role
DR FARRUKH SHAHID	*Supervisor*
SHAHBAZ SIDDIQUI	*Co- Supervisor*

Document Sign-Off

Version	Sign-off Authority	Sign-off Date

Table of Contents

1. Introduction

1.1. Purpose of Document

The purpose of the document is to collect and analyze all assorted ideas that have come up to define the system, its requirements with respect to consumers. Also, we shall predict and sort out how we hope this product will be used in order to gain a better understanding of the project, outline concepts that may be developed later, and document ideas that are being considered, but may be discarded as the product develops. In short, the purpose of this SRS document is to provide a detailed overview of our software product, its parameters and goals. This document describes the project's target audience and its user interface, hardware and software requirements.

1.2. Intended Audience

The purpose of this document is to give a detailed description of the requirements for our Final year Project. It will illustrate the purpose, scope and complete description for the development of system. It will also explain external interface requirements and system requirements as well as non-functional requirements. This document is primarily intended to be proposed to a customer for its approval and also for further processing such as additions to be developed in later releases. Customers/hospitals can refer to section 3 and 4 for the list of requirements implemented in Version final. This document will also be used as a reference for developing and testing Version final

1.3 Abbreviations

FAQ	Frequently Asked Questions
EHR	Electronic health care Records

1.4 Document Convention

Throughout this document, All the user entities are written in capitalizations i.e., first letter as capital. Also, any significant term which has been described in the glossary is made bold and italic in the text. On the other hand, those terms which are significant (but not described in glossary) are bold in text. The document is prepared using Microsoft Word 2020 and has used the font type 'Times New Roman'. The fixed font size that has been used to type this document is 12pt and for headings 14pt with 1 linespacing

2. Overall System Description

2.1. Project Background

Blockchain is also used in the healthcare management system for effective maintenance of electronic health and medical records. The technology ensures security, privacy, and immutability. This work proposes a framework by integrating the blockchain and Federated Deep Learning in order to provide a tailored recommendation system. Electronic medical records contain personal and confidential information that traditional storage methods must protect against cyberattacks and third-party authentication. To overcome this challenge, a method of distributed storage was proposed in this work. The focus of this work is also on making treatment recommendations to patients by comparing their medical records with historical data. This work is motivated by the limitations of existing work. The current study supports his EHR preservation, but the recommendation system is neither discussed nor incorporated, making it difficult to create a treatment recommendation system. Federated learning may therefore lead to more accurate treatment recommendations.

2.2. Project Scope

Primarily, the scope of our final year project is limited. We will be limiting our scope to the medical sectors where a patient data will be recorded on our system and by means of recording, we will be using that data only to train our machine learning model. We will be using the concept of federated Learning Through Federated learning, multiple organizations or institutions work together to solve a machine-learning problem under the coordination of a central server or service provide. Thus, a deep-learning model is maintained and improved upon within a central server. The model is trained by distributing itself to hospitals which allows these sites to keep their data localized. Data from each collaborator is never exchanged or transferred during training. Instead of bringing the data to the central server, as in conventional deep learning, the central server maintains a global shared model, which is disseminated to all institutions. Each entity subsequently maintains a separate model based on its own patients' data. Thereafter, each center provides feedback to the server based on its individually trained model—either by its weight or the error gradient of the model. The central server aggregates the feedback from all participants, and based on predefined criteria, updates the global model. The predefined criteria allow the model to evaluate the quality of the feedback and therefore to only incorporate that which is value-adding. The feedback from centers with adverse or strange results can thus be ignored. This process forms one round of federated learning, and it is iterated until the global model is trained.

2.3. Not In Scope

We will be only targeting one/two disease for the recommendation and it will be running on local environment not globally deployed. It will be on private blockchain and only people who are connected will have the feasibility to connect to it

2.4. Project Objectives

Our system implementation addresses the three major issues:

1. slow access to medical data
2. system interoperability
3. patient agency

Recent advances in deep learning have shown many successful stories in smart healthcare applications with data-driven insight into improving clinical institutions' quality of care. Excellent deep learning models are heavily data-driven. The more data trained, the more robust and more generalizable the performance of the deep learning model. However, pooling the medical data into centralized storage to train a robust deep learning model faces privacy, ownership, and strict regulation challenges. **Federated learning** resolves the previous challenges with a shared global deep learning model using a central aggregator server. At the same time, patient data remain with the local party, maintaining data anonymity and security.

This method provides decentralized machine learning model training with-out transmitting medical data through a coordinated central aggregate server. Medical institutions, working as client nodes, train their deep learning models locally and then periodically forward them to the aggregate server. The central server coordinates and aggregates the local models from each node to create a global model, then distributes the global model to all the other nodes. It is worth noting that the training data are kept private to each node and never transmitted during the training process. Only the model's weight and parameters are transmitted, ensuring that medical data remain confidential. For these reasons, FL mitigates many security concerns because it retains sensitive and private data while enabling multiple medical institutions to work together. FL holds an excellent promise in healthcare applications to improve medical services for both institutions and patients—for instance, predict autism spectrum disorder, mortality and intensive care unit (ICU) stay-time prediction.

2.5. Stakeholders

Users that will be interacting with our system will be the hospital admin, a super admin and a patient for the development purpose the development team along with the QA team will be responsible for the working of the system and they'll be the one interacting with the system. Stakeholders can only take actions according to the roles providing to them

2.6. Operating Environment

It is a web-based application running on a browser and since it is locally deployed. Only the user with the accessibility can access the system. The user must have a node js version of > 14.00. Ganache should be installed on the system and a web3 provider such as meta mask should be installed on the browser in order to authenticate the system

2.7. System Constraints

- **Software constraints**

  Node js version above 14.00 for running js locally. Ganache along with meta mask extension should be installed on the system

- **Hardware constraints**

No such constraints for hardware

- **Cultural constraints**

Understanding of English language since all the instructions are provided in English, so we're guessing that our end user should have a good grip on English.

2.8. Assumptions & Dependencies

One assumption about the software is that it will always be used on system that have enough resources to run the application. If the system does not have enough hardware resources available for the application, there may be scenarios where the application does not work as intended or not even at all. The application uses IPFS for online storage of information. If the interface changes the application needs to be adjusted accordingly. External libraries of web3 and ether js should be installed on the system along with the JavaScript, so it should be run smoothly

3. External Interface Requirements

3.1. Hardware Interfaces

Since the application must run over the internet, all the hardware shall require to connect internet will be hardware interface for the system. As for e.g., Modem, WAN – LAN, Ethernet Cross-Cable.

3.2. Software Interfaces

Operating System*: We have chosen Windows operating system for its best support and user-friendliness.*

Database*: To save the data of model in an IPFS distributed database system.*

Smart Contracts*: To save and implement the contract which is agreed between two entities in the system*

Ethereum Blockchain*: Ethereum Blockchain used to ensure smoothness of program and decentralization of system*

Global Server*: Global server used to interact with the hospital server to send/receive the training model*

3.3. Communications Interfaces

The DEDOC system shall use the HTTP protocol for communication over the internet and for the intranet communication will be through TCP/IP protocol suite.

4. Functional Requirements

4.1. Functional Hierarchy

[This section will give a big picture of overall system functionality. The main modules/features of system and their sub-functions will be described here in the form of a functional hierarchy so that, before getting into the use case, audience could grab the idea of overall system functions.]

4.2. Use Cases

4.2.1. [Title of use case]

[Use Case Diagram]
[Use Case Description]

<Use case Id: name>		
Use case Id:	Write use case reference number.	
Actors:	<List of actors (external agents), indicating who initiated the use case>	
Feature:	<Feature from which the use case is driven>	
Pre-condition:	<List the assumptions required before this Use Case can be executed. >	
Scenarios		
Step#	**Action**	**Software Reaction**
1.	Numbered actions of the actors	Numbered description of system responses
2.		
Alternate Scenarios: Write additional, optional, branching or iterative steps. Refer to specific action number to ensure understandability.		
1a: **2a:**		
Post Conditions		
Step#	**Description**	
	Sequentially list conditions expected at the completion of the use case.	
Use Case Cross referenced	<Related use cases, which use or are used by this use case>	

5. Non-functional Requirements

5.1. Performance Requirements

The System shall be based on web and has to be run from a web server. The system shall take initial load time depending on internet connection strength which also depends on the media from which the product is run. The performance shall depend upon hardware components of the client/customer

5.2. Safety Requirements

The software is completely environmentally friendly and does not cause any safety violations. The web page will have a flexible font that can be zoomed so as to not over constrain the eyes.

5.3. Security Requirements

Data Transfer

- The system shall use secure sockets in all transactions that include any confidential customer information.
- The system shall automatically log out all customers after a period of inactivity.
- The system shall confirm all transactions with the customer's web browser.
- The system shall not leave any cookies on the customer's computer containing the user's password.
- The system shall not leave any cookies on the customer's computer containing any of the user's confidential information.

Data Storage

- The customer's web browser shall never display a customer's password. It shall always be echoed with special characters representing typed characters.
- The customer's web browser shall never display a customer's credit card number after retrieving from the database. It shall always be shown with just the last 4 digits of the credit card number.
- The system's back-end servers shall never display a customer's password. The customer's password may be reset but never shown.
- The system's back-end servers shall only be accessible to authenticated administrators.
- The system's back-end databases shall be encrypted.

5.4. User Documentation

The software is accompanied by the following materials for further help:

- Software Requirement Document
- Software Design Specification Document

6. References

1. R. Roth, M, Sheller, “The future of digital health with federated
learning
”, September 2022, online: “https://www.nature.com/articles/s41746-020-00323-1”.
2. R. Stoffel et al,” Federated Learning for Healthcare: Systematic
3. Review and Architecture Proposal”, august 2022,
   online: ”https://dl.acm.org/doi/10.1145/3501813”
4. J.Xiu,C.Su,P.Walker, ”Federated Learning for Healthcare
Informatics”, November 2020, online: “https://link.springer.com/article/10.1007/s41666-020-00082-4”

7. Appendices

[This section should include supporting detail that would be too distracting to include in the main body of the document.]