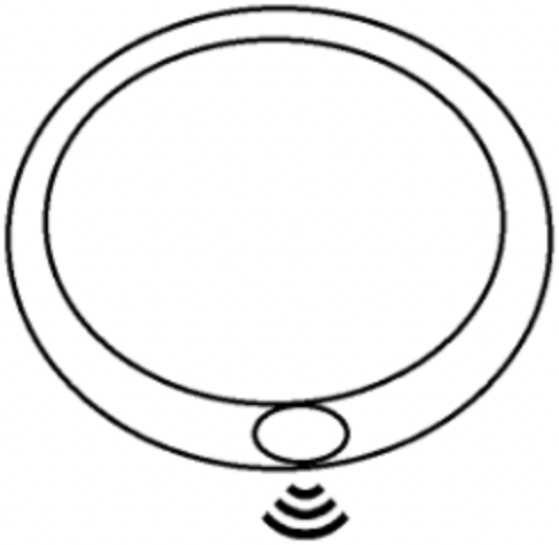
Department of Computer Science & Engineering
The University of Texas at Arlington

Detailed Design Specification
CSE 4317: Senior Design II
Spring 2023



Medi ID Group
Medi ID

Christian Blundell
Ivan Chu
Ahamad Natsheh
Mahmoud Natsheh
Alex Stringer

Revision History

Revision	Date	Author(s)	Description
0.1	2.06.2023	AN, MN	document creation
0.2	2.12.2023	CB, IC, AN, MN, AS	complete draft
0.3	2.13.2023	AN	Finalized
1.0	4.24.2023	IC	Updated Version

CONTENTS

List of Figures

List of Tables

1 Introduction

The product Medical ID bracelet should store important medical information about a user that is helpful to medical professionals. This information includes name, date of birth, gender, allergies, medical background information, primary care physician, and any other important medical information. Our medical identification bracelet will use an NFC chip to hold the link to our website and a key to the user's account. Once the bracelet is tapped with a phone it will redirect them to our website and prompt either the user or the EMT to login to their account. If the user logs in, then it will take them to their account where they can either view or edit their information. If an EMT logs in, then it will just allow them to view the patient's medical information and not edit it. The user of our product will be medical professionals and the users who buy it to store their medical information on it.

2 System Overview

The first layer of the medical identification bracelet is the tap function. The tap function is where a user taps their phone on to the medical identification bracelet which has the NFC chip stored inside it. Once the phone senses the NFC chip it will read it and execute what is on it. The NFC chip will hold the link to our website and a key to the user's account. Thus, through taping the NFC chip it will redirect the user to our website which stores the patient's medical information. This second layer is where all the user's data is stored. All the patient's medical information will be stored in a database which will be written to from the Z layer. This layer will store all the information in order to validate a user and give them access to the medical information, additionally it will have encryption and decryption in order to keep the patients information secure. The third and final layer is the user interface, this is where the user will be able to see all medical information and edit it. This layer has two components to it, the first being the owner of the medical identification view of the user interface and the second component being the medical professionals view of the user interface. For the owner of the bracelet when they are redirected to our website, they will be asked to enter their login information, once verified the user will be able to see all their information that they had stored. Additionally, they will be able to edit their information and change what they have stored in the database. Once the medical professionals tap the bracelet they will be redirected to our website where they will be asked to login to their accounts. When they are verified, they will be logged in and will be able to see the patient's medical information only and will not be able to edit it. Once the medical professional is logged out they will no longer be able to see the patients information, in order to keep it secure.

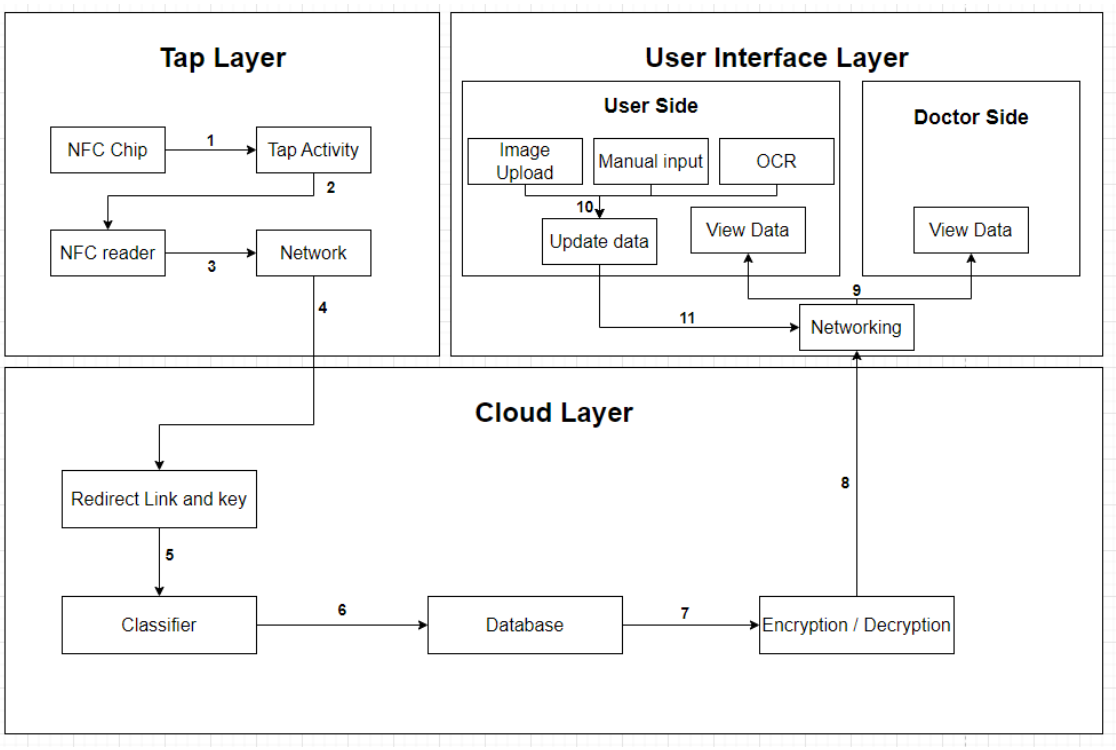


Figure 1: System architecture

3 Tap Layer Subsystems

The Tap layer(NFC Layer) contains the NFC Reader subsystem, Data Transfer subsystem, Security subsystem, and networking. This layer will manage the reading of the NFC chips to allow the users to access their information.

3.1 Layer Hardware

NFC Hardware, NFC Chip, Mobile devices with internet accessible.

3.2 Layer Operating System

Android and IOS 13 or newer

3.3 Layer Software Dependencies

TagXplorer, NFC Tools, Internet Browser

3.4 NFC Reader subsystem

This subsystem is responsible for reading NFC tags and other NFC-enabled devices, such as smartphones or medical equipment.

3.4.1 Subsystem Hardware

NFC Reader.

3.4.2 Subsystem Operating System

Android or IOS 13 above.

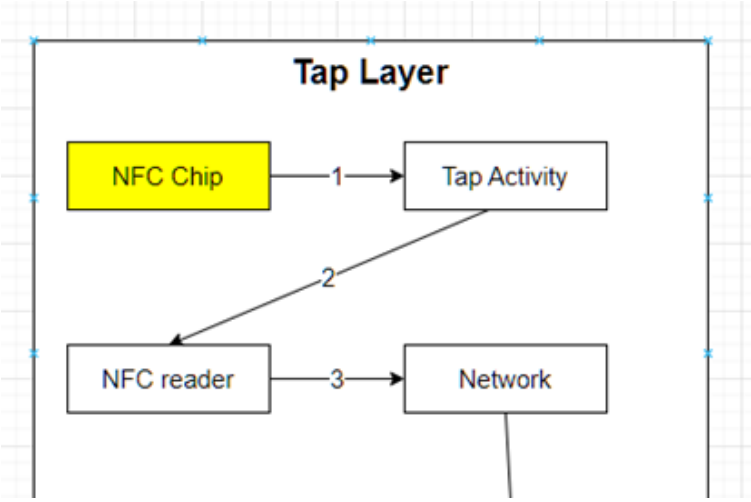


Figure 2: NFC chip and its Relation to Tap Layer Subsystem

3.4.3 Subsystem Software Dependencies

TagXplorer and NFC Tools.

3.4.4 Subsystem Programming Languages

Programming not needed, so no programming language to reference.

3.4.5 Subsystem Data Structures

No data structures to reference.

3.4.6 Subsystem Data Processing

No data processing to reference.

3.5 Data Transfer Subsystem

The Data Transfer Subsystem is responsible for transferring data between the digital medical ID bracelet and other NFC-enabled devices, such as smartphones or medical equipment.

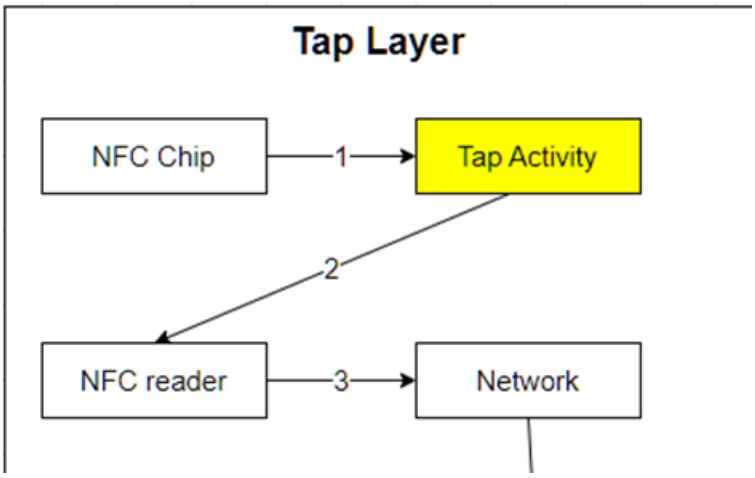


Figure 3: Tap activity and its Relation to Tap Layer Subsystem

3.5.1 Subsystem Hardware

This subsystem relies on the hardware components of the digital medical ID bracelet, including the NFC chip and any associated antennas or sensors.

3.5.2 Subsystem Operating System

The Data Transfer Subsystem may interact with the operating system of the digital medical ID bracelet, such as to manage power consumption or handle user interface interactions.

3.5.3 Subsystem Software Dependencies

This subsystem may depend on software libraries or protocols for handling NFC communication and data transfer.

3.5.4 Subsystem Programming Languages

The Data Transfer Subsystem may be programmed using a variety of languages, such as Java, Python, or C++.

3.5.5 Subsystem Data Structures

The Data Transfer Subsystem may use data structures such as JSON or XML to format and organize data for transfer.

3.5.6 Subsystem Data Processing

The Data Transfer Subsystem may perform data processing tasks, such as encrypting or decrypting data, validating data integrity, or compressing data to optimize transfer speed and efficiency.

3.6 Security Subsystem

The Security Subsystem is responsible for ensuring the confidentiality, integrity, and availability of data stored on the digital medical ID bracelet and during data transfer.

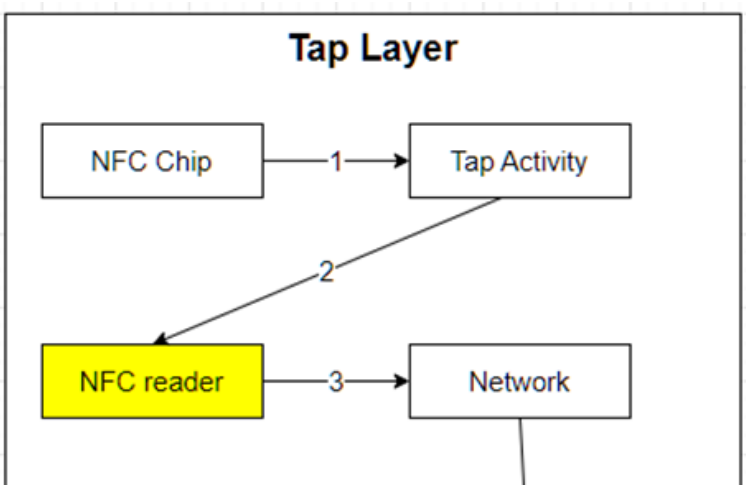


Figure 4: NFC reader and its Relation to Tap Layer Subsystem

3.6.1 Subsystem Hardware

This subsystem may rely on hardware components such as encryption modules, secure storage devices, or biometric sensors.

3.6.2 Subsystem Operating System

The Security Subsystem may interact with the operating system of the digital medical ID bracelet, such as to manage access control or provide secure communication channels.

3.6.3 Subsystem Software Dependencies

This subsystem may depend on software libraries or protocols for encryption, decryption, or secure communication.

3.6.4 Subsystem Programming Languages

The Security Subsystem may be programmed using languages such as C, C++, or Java.

3.6.5 Subsystem Data Structures

The Security Subsystem may use data structures such as hash functions, cryptographic keys, or digital certificates to secure data.

3.6.6 Subsystem Data Processing

The Security Subsystem may perform data processing tasks such as encryption, decryption, or authentication to ensure the security of data on the digital medical ID bracelet and during data transfer.

3.7 Network

A network is required for the NFC reader to request users' data from the cloud.

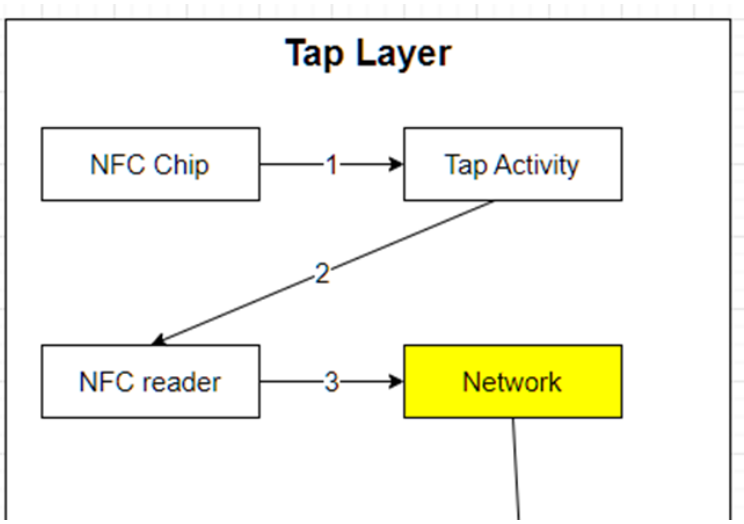


Figure 5: Network and its Relation to Tap Layer Subsystem

3.7.1 Subsystem Hardware

This subsystem may rely on hardware components such as Wi-Fi or Bluetooth modules to establish wireless connections or Ethernet ports for wired connections.

3.7.2 Subsystem Operating System

The Network Subsystem may interact with the operating system of the digital medical ID bracelet, such as to manage network settings or handle network events.

3.7.3 Subsystem Software Dependencies

This subsystem may depend on software libraries or protocols for network communication, such as TCP/IP or Bluetooth protocols.

3.7.4 Subsystem Programming Languages

The Network Subsystem may be programmed using languages such as C, C++, or Java.

3.7.5 Subsystem Data Structures

The Network Subsystem may use data structures such as packets, sockets, or network addresses to manage the network communication.

3.7.6 Subsystem Data Processing

The Network Subsystem may perform data processing tasks such as routing, filtering, or buffering to manage network traffic and ensure reliable communication between devices.

4 Patient Layer Subsystems

The UI Patient Layer (Layer Y) consists of the reading and editing of patient data and the networking communication to the database that connects to the site to store patient data for the bracelet user.

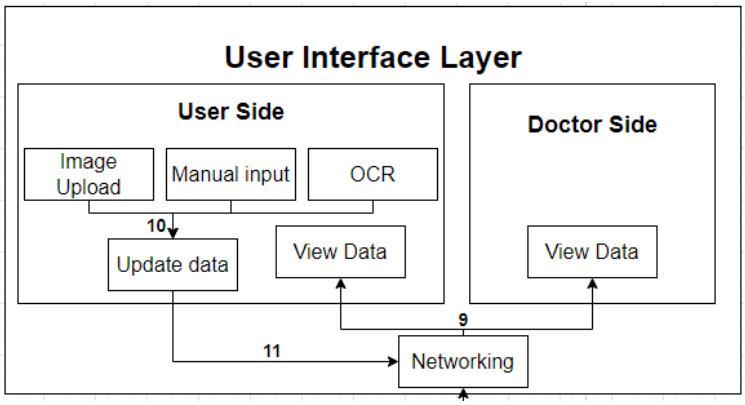


Figure 6: Complete UI Layer (Both User Types)

4.1 Layer Hardware

N/A

4.2 Layer Operating System

No specific OS is required to complete this subsystem; Windows OS was used to integrate and test the user information import.

4.3 Layer Software Dependencies

N/A

4.4 Read Patient Data

The reading of patient data is done by the patient. The user views the data in order to verify its accuracy or fill in information for various needs. This is a use-case action done by both user types and requires a stable connection to both a network as well as the database that has patient information stored. This is implemented using a class that distinguishes types of information and displays information using PHP HTML that displays saved data from the patient's query result.

4.4.1 Subsystem Hardware

N/A

4.4.2 Subsystem Operating System

No specific OS is required to complete this subsystem; Windows OS was used to integrate and test the user information import.

4.4.3 Subsystem Software Dependencies

Valid SQL scripts that reach the patient's key in the database are required in order for the information to be displayed properly. PHP scripts must also allow for the appropriate types of data to be represented properly (list of values, images) using libraries that allow for exporting various data variable types.

4.4.4 Subsystem Programming Languages

PHP, HTTP.

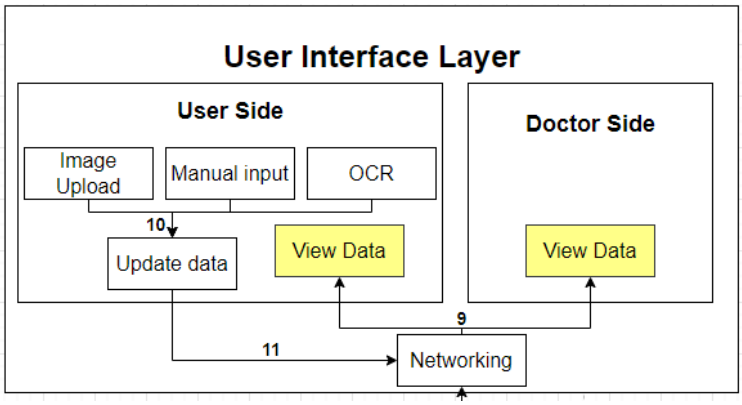


Figure 7: View Data Layer and its Relation to UI Layer Subsystem

4.4.5 Subsystem Data Structures

Classes of different data types must be defined in order to accurately call the right type of data (mentioned in 4.4.3). Tesseract OCR is a software engine that allows files to be read and input the contents as an array of values.

4.4.6 Subsystem Data Processing

Standard PHP script to display data from SQL results based on patient key. This includes image files not filtered using OCR. Tesseract OCR engine is also needed to process image file that stores its contents as array values that can be extracted similar to manually inputted values from the user. Real-time processing method when connected to a network (To access database)

4.5 Network For UI

The network connection allows for the UI of the website to be accessed, so that all types of users may access specific user data. This is a software component that is required for updating information to be validated and stored in the database via the middleware server that operates out of the network. Network setup is also required to view the patient information which can then be redirected to an option to update any information accessible by the user.

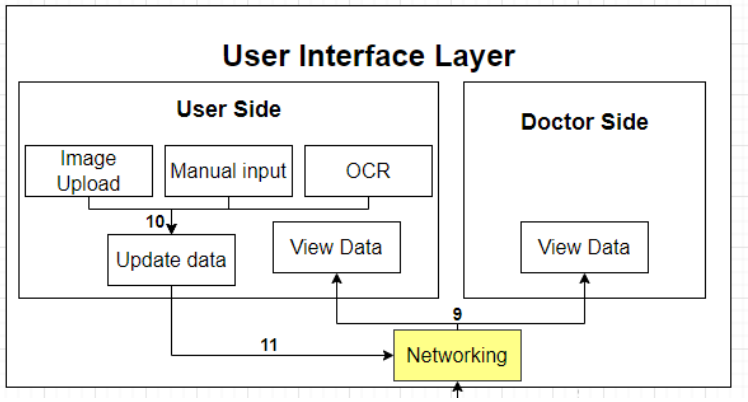


Figure 8: Networking Layer and its Relation to UI Layer Subsystem

4.5.1 Subsystem Hardware

N/A

4.5.2 Subsystem Operating System

No specific OS is required to complete this subsystem; Windows OS was used to integrate and test the user information import. SSMS Running the SQL Server Studio is required for Windows OS testing.

4.5.3 Subsystem Software Dependencies

SSMS (for Windows OS) must be initialized to connect to the SQL Server Studio. Networking is predefined for the scope of this web service and network configuration is not required for the service to work on all devices.

4.5.4 Subsystem Programming Languages

PHP

4.5.5 Subsystem Data Structures

N/A

4.5.6 Subsystem Data Processing

Online processing method when connected to a network (To access data located on database)

4.6 Update User Data

The verified patient (user) can update various medical information sections, which include adding information, removing information, or changing information. This is then saved in the database using the user's appropriate key.

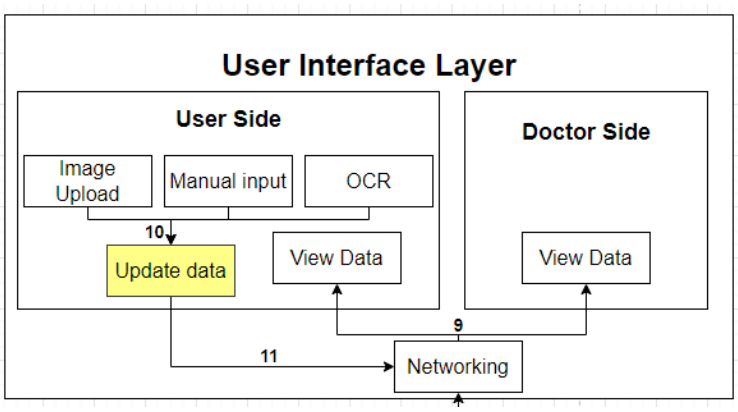


Figure 9: Update User Data Layer and its Relation to UI Layer Subsystem

4.6.1 Subsystem Hardware

N/A

4.6.2 Subsystem Operating System

No specific OS is required to complete this subsystem; Windows OS was used to integrate and test the user information import. SSMS Running the SQL Server Studio is required for Windows OS testing.

4.6.3 Subsystem Software Dependencies

Valid SQL scripts that reach the patient's key in the database are required in order for the information to be accessed properly. PHP scripts must also allow for the appropriate types of data to be represented properly (array of values, images) using libraries that allow for exporting various data variable types. The network must have an accessible server connection to the SQL tables in order to store (update) the new values into the appropriate patient record based on their registered key.

4.6.4 Subsystem Programming Languages

PHP, SQL.

4.6.5 Subsystem Data Structures

Abstraction of classes based on information type that is to be updated by the user.

4.6.6 Subsystem Data Processing

Standard PHP script to display accessed data from SQL results based on patient key. This includes image files not filtered using OCR. All data types must be accessed in order for the user to properly give a new value to this data type.

4.7 Optical Character Recognition

OCR tools are used to read images uploaded by patients and input the data into an array that can be stored in the database as text. EMT can then view the contents of the image as an array of text is stored in the database. OCR Tesseract is an engine that will be used to implement the exporting of values that are gathered through the validated image uploaded by the user.

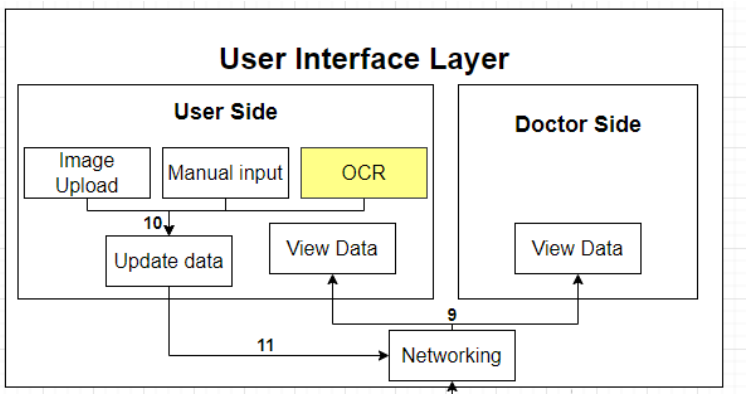


Figure 10: OCR Layer and its Relation to UI Layer Subsystem

4.7.1 Subsystem Hardware

Smartphone with camera (for testing).

4.7.2 Subsystem Operating System

Windows OS was used to integrate and test the user information import. SSMS Running the SQL Server Studio is required for Windows OS testing.

4.7.3 Subsystem Software Dependencies

Valid SQL scripts that reach the patient's key in the database are required in order for the information to be accessed properly. PHP scripts must also allow for the appropriate types of data to be represented properly (array of values, images) using libraries that allow for exporting various data variable types.

The network must have an accessible server connection to the SQL tables in order to store (update) the new values into the appropriate patient record based on their registered key.

4.7.4 Subsystem Programming Languages

PHP.

4.7.5 Subsystem Data Structures

Classes of different data types must be defined in order to accurately call the right type of data (image file). Tesseract OCR is a software engine that allows files to be read and input the contents as an array of values.

4.7.6 Subsystem Data Processing

Standard PHP script to display accessed data from SQL results based on patient key. All data types must be accessed in order for the user to properly give a new value to this data type if necessary. The data must be processed as values similar to those manually inputted by the user so that they can be modified (see Update User Data section).

4.8 Manual Input

The user manually inputs values and information into their account details, which consist of various medical information and documents. The manual input is done through manual inputs created using PHP that allows for query searching by the patient. The classes offer abstraction and some hierarchies in order for the user to distinguish information types.

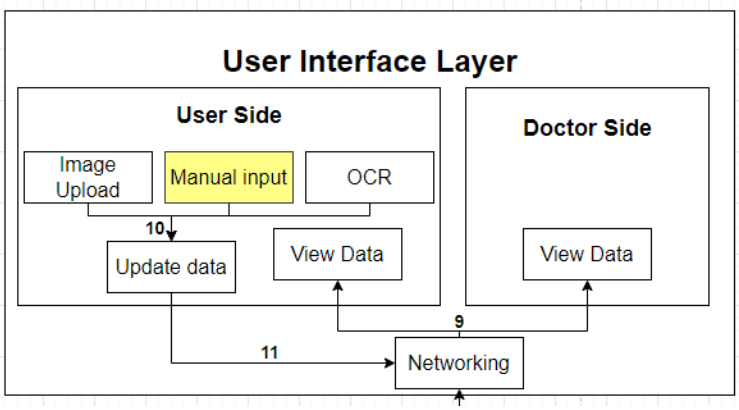


Figure 11: Manual Input Layer and its Relation to UI Layer Subsystem

4.8.1 Subsystem Hardware

N/A

4.8.2 Subsystem Operating System

No specific OS is required to complete this subsystem; Windows OS was used to integrate and test the user information import. SSMS Running the SQL Server Studio is required for Windows OS testing.

4.8.3 Subsystem Software Dependencies

Valid SQL scripts that reach the patient's key in the database are required in order for the information to be accessed properly. PHP scripts must also allow for the appropriate types of data to be represented properly (array of values, images) using libraries that allow for exporting various data variable types.

The network must have an accessible server connection to the SQL tables in order to store (initialize) the new values in the appropriate patient record based on their registered key.

4.8.4 Subsystem Programming Languages

PHP, SQL.

4.8.5 Subsystem Data Structures

Abstraction of classes based on information type that is to be entered by the user. Class hierarchies are initialized to distinguish specific information from generalized information from the user's input.

4.8.6 Subsystem Data Processing

Standard PHP script to display accessed data from SQL results based on patient key. This includes image files not filtered using OCR. All data types must be accessed in order for the user to properly give a value to this initialized data type.

4.9 Image Uploading

Image uploading is for documents that verify medical or insurance information of the user, which can be viewed by the patient and verified EMT. Image uploading is done manually by the user, and is implemented using standard PHP

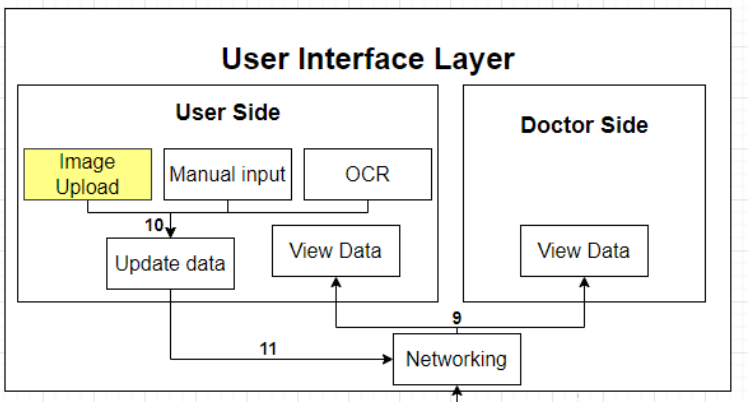


Figure 12: Image Uploading Layer and its Relation to UI Layer Subsystem

4.9.1 Subsystem Hardware

Smartphone with camera (for testing).

4.9.2 Subsystem Operating System

Windows OS was used to integrate and test the user information import. SSMS Running the SQL Server Studio is required for Windows OS testing.

4.9.3 Subsystem Software Dependencies

Valid SQL scripts that reach the patient's key in the database are required in order for the information to be accessed properly. PHP scripts must also allow for the appropriate types of data to be represented properly (array of values, images) using libraries that allow for exporting various data variable types. The network must have an accessible server connection to the SQL tables in order to store (update) the image file into the appropriate patient record based on their registered key.

4.9.4 Subsystem Programming Languages

PHP.

4.9.5 Subsystem Data Structures

Classes of different data types must be defined in order to accurately call the right type of data (image file). Uploading files inputted using PHP must be followed and modified to store into the PHP scripts that access user information (patient-end only).

4.9.6 Subsystem Data Processing

Standard PHP script to display accessed data from SQL results based on a patient key in real-time. All data types must be accessed in order for the user to properly give a new value to this data type if necessary. The data must be processed as an image file that is not read through OCR (stored as an image file in database).

5 EMT Layer Subsystems

The UI EMT Layer (Layer Y) consists of the reading of patient data and the networking communication to the database that connects to the site to view patient data for the bracelet user.

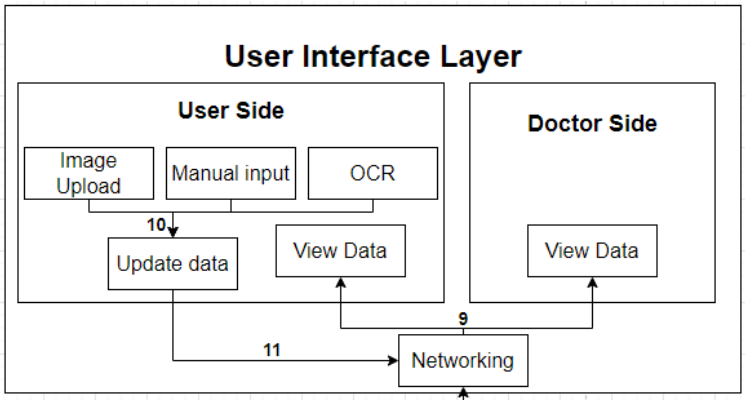


Figure 13: Complete UI Layer (Both User Types)

5.1 Layer Hardware

N/A

5.2 Layer Operating System

No specific OS is required to complete this subsystem; Windows OS was used to integrate and test the user information import.

5.3 Layer Software Dependencies

N/A

5.4 Read Patient Data

The reading of patient data is done by the EMT. The user views the data in order to verify its accuracy or fill in information for various needs. This is a use-case action done by both user types and requires a stable connection to both a network as well as the database that has patient information stored. This is implemented using a class that distinguishes types of information and displays information using frontend scripts that display saved data from the patient's query result.

5.4.1 Subsystem Hardware

N/A

5.4.2 Subsystem Operating System

No specific OS is required to complete this subsystem; Windows OS was used to integrate and test the user information import.

5.4.3 Subsystem Software Dependencies

Valid SQL scripts that reach the patient's key after accessing the NFC Chip in the database are required in order for the information to be displayed properly. PHP scripts must also allow for the appropriate types of data to be represented properly (array of values, images) using libraries that allow for exporting various data variable types.

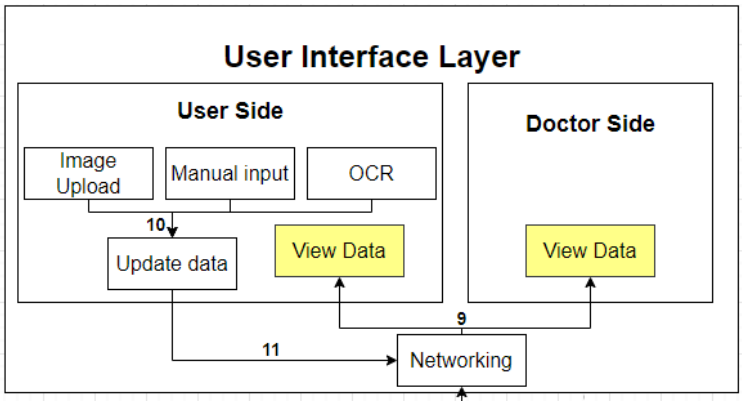


Figure 14: View Data Layer and its Relation to UI Layer Subsystem

5.4.4 Subsystem Programming Languages

PHP, HTTP.

5.4.5 Subsystem Data Structures

Classes of different data types must be defined in order to accurately call the right type of data (mentioned in 4.4.3). Tesseract OCR is a software engine that allows files to be read and input the contents as an array of values. JSON files are needed to display the contents package after reading an uploaded image.

5.4.6 Subsystem Data Processing

Standard PHP script to display data from SQL results based on patient key. This includes image files not filtered using OCR. Tesseract OCR engine is also needed to process image file that stores its contents as array values that can be extracted similar to manually inputted values from the user.

6 Cloud Layer Subsystems

The Cloud Layer consists of all encrypted patient data stored on the database, this also includes securing user Login information. The Cloud Layer communicates with the Tap layer to link the NFC chip to the proper patient data, and also communicates with the User Interface Layers to allow the patient to edit their data.

6.1 Layer Hardware

N/A

6.2 Layer Operating System

No specific OS is required to complete this subsystem; Windows OS will be used to integrate and test the data storing and retrieval.

6.3 Layer Software Dependencies

JSON Package Dependencies

6.4 Redirect Link and Key

The Tap layer communicates with the Cloud layer by sending a link that directs the device to the patients account page that is protected by a Login verification. The Login credentials are

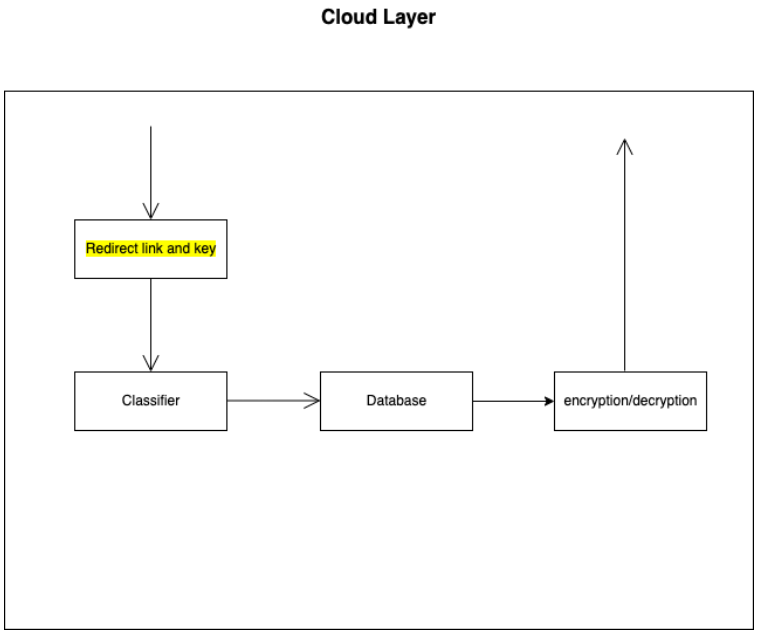


Figure 15: Redirect link and key and its Relation to Cloud Layer Subsystem

6.4.1 Subsystem Hardware

N/A

6.4.2 Subsystem Operating System

No specific OS is required to complete this subsystem; Windows OS will be used to integrate and test the data storing and retrieval.

6.4.3 Subsystem Software Dependencies

JSON libraries and SQL scripts that retrieve patient data.

6.4.4 Subsystem Programming Languages

SQL, JSON, JS

6.4.5 Subsystem Data Structures

NFC chip is scanned by the built-in NFC scanner of the mobile device.Each NFC holds a URL that leads to a specific page for each individual.

6.4.6 Subsystem Data Processing

SQL scripts that retrieve patient data from the database and displays it on the opened page

6.5 Classifier

The classifier is responsible for giving each user specific permissions to the data on the cloud layer. Patients are allowed to add and remove data, EMT and hospital personnel are only allowed to view data of a specific patient.

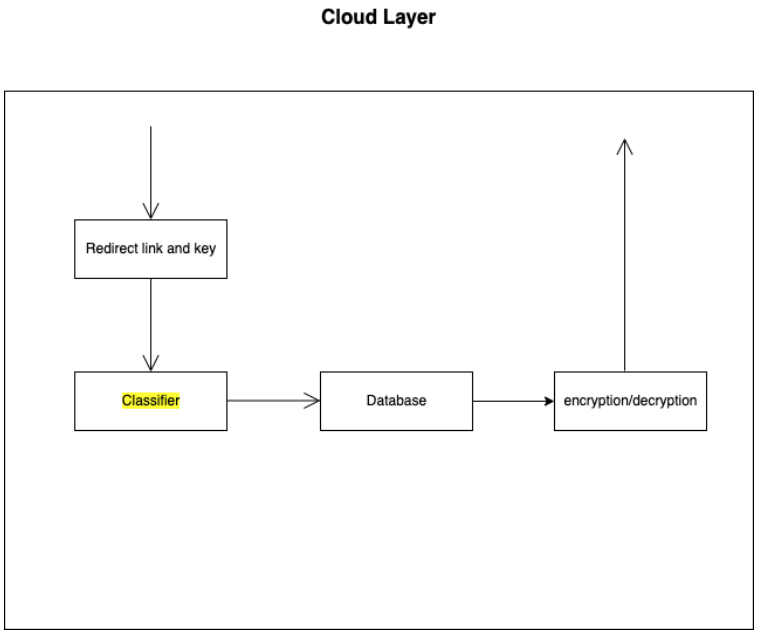


Figure 16: Classifier and its Relation to Cloud Layer Subsystem

6.5.1 Subsystem Hardware

Mobile device

6.5.2 Subsystem Operating System

No specific OS is required to complete this subsystem; Windows OS will be used to integrate and test the data storing and retrieval.

6.5.3 Subsystem Software Dependencies

JSON libraries and SQL scripts that retrieve patient data.

6.5.4 Subsystem Programming Languages

SQL, JSON, JS

6.5.5 Subsystem Data Structures

N/A

6.5.6 Subsystem Data Processing

N/A

6.6 Database

Database contains all data the patient would like to upload to the server. Data stored include allergies, blood type,images, and medical records. All patient information will be protected to prevent any security breaches.

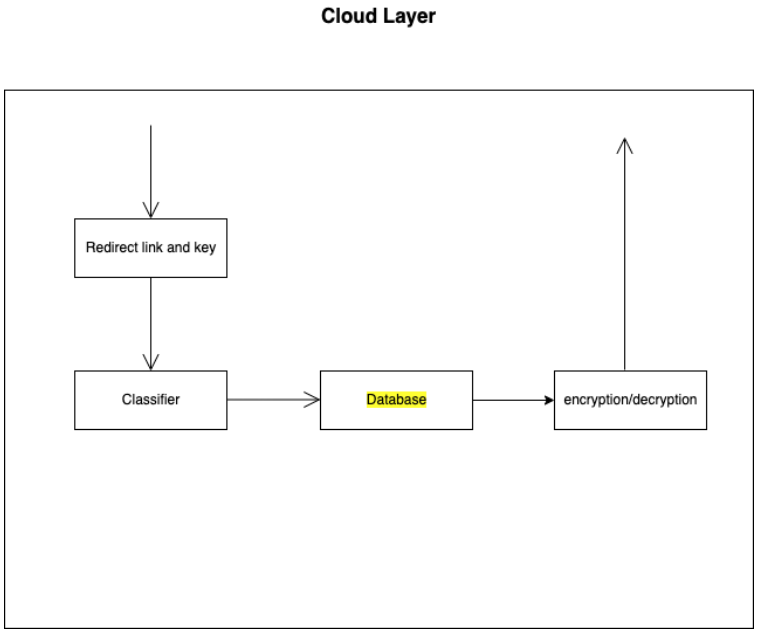


Figure 17: Database and its Relation to Cloud Layer Subsystem

6.6.1 Subsystem Hardware

N/A

6.6.2 Subsystem Operating System

No specific OS is required to complete this subsystem; Windows OS will be used to integrate and test the data storing and retrieval.

6.6.3 Subsystem Software Dependencies

JSON libraries and SQL scripts that retrieve patient data.

6.6.4 Subsystem Programming Languages

SQL, JSON, JS

6.6.5 Subsystem Data Structures

All data that is saved on the database will be stored as JSON objects. The database uses a JSON tree data structure to store the JSON objects.

6.6.6 Subsystem Data Processing

N/A

6.7 encryption/decryption

Encrypts each patients login credentials and well as secures each users data within the database so that it is unreadable. The patients data will be later decrypted when the right personnel make a request and displayed on the User Interface layer.

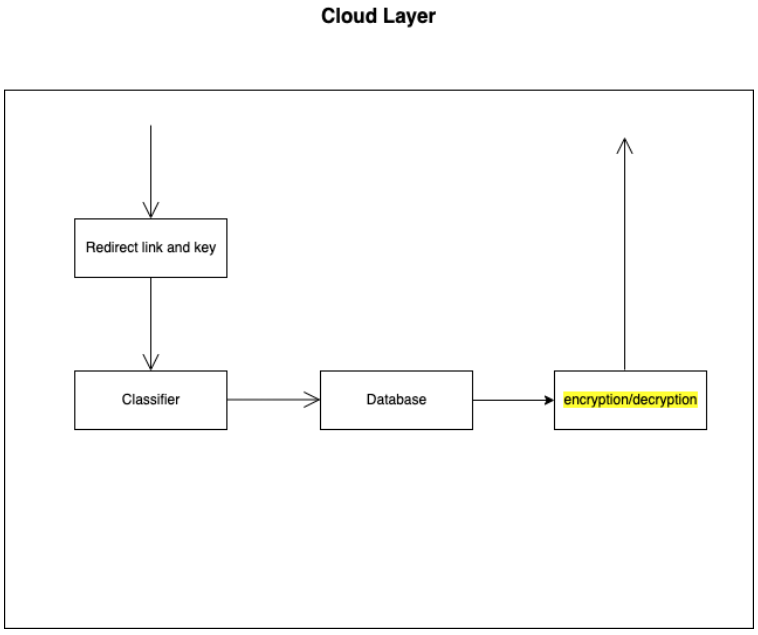


Figure 18: Encryption/decryption and its Relation to Cloud Layer Subsystem

6.7.1 Subsystem Hardware

N/A

6.7.2 Subsystem Operating System

N/A

6.7.3 Subsystem Software Dependencies

JSON libraries and SQL scripts that retrieve patient data.

6.7.4 Subsystem Programming Languages

SQL, JSON, JS

6.7.5 Subsystem Data Structures

All data that is saved on the database will be encrypted and stored as JSON objects. The database uses a JSON tree data structure to store the JSON objects.

6.7.6 Subsystem Data Processing

N/A

7 APPENDIX A

Include any additional documents (CAD design, circuit schematics, etc) as an appendix as necessary.

REFERENCES