IFS4205: Information Security Capstone Project
Team 2

Semester 1 AY2019/2020

Final Project Report

Authors:
Sean Yap Yu Rong
Lim Ding Heng
Toh Yunqi Cheryl
Sun Shengran

Contents

1 Project Introduction

This project, **NUSMed**, is a prototype health care system. At this stage, it is not possible to achieve relationships expected within a proper health care system. Therefore, to achieve a meaningful and functional health care system, the health care system was designed to be condensed and compact, with core and essential functionalities. The system was streamlined to achieve the best user experience without neglecting core functionalities while retaining strict and robust security mechanisms.

To give a summary, the system relies on trusted administrators to perform registration of users, such as patients, therapists and researchers. The administrator will invigilate the download, installation and first login to the Web App via the Mobile App; to associate the Mobile App to his or her account. At this point, the user is assigned and given a physical NFC tag that is associated to his or her account. After registration, users are able to login via MFA through the combination usage of his or her token ID, device ID, password and NRIC. To protect patient confidential information such as personal information and medical records, a permissions system is implemented where therapists are required to request permissions from patients to obtain consent and perform his or her medical duties. Permission requests involve record types permissions and patient read permissions are implicit as it is assumed that account permissions are necessary for any therapist to access records. More information about permissions will be discussed in more detail in the coming sections.

Similarly to permissions, there is also a system for diagnosis. As explained above, it is not possible to produce a fully encompassing system such that patients have treatments and treatments are performed by doctors. However, we find that it is absolutely necessary in a health care system for therapists to view and set a patient's diagnosis; as such, there are implementations in place for diagnosis to be attributed to a patient to show some illness inflicted him or her for some time period. This is important for therapists to assess patients as diagnoses tend to be influenced by previous or past diagnoses. Additionally, records can be attributed to multiple diagnosis to show which records are associated and or relevant for treating which illness.

To facilitate the functionalities in this system. there are two databases, one "main" database and one "logging" database. The "main" database performs typical operations related to the system while the "logging" database facilitates the storing of application logs. The "logging" database is also used to backup transactional and query logs of the "main" database. Enhanced Entity Relationship (EER) diagrams will help with visualizing the tables and relationships and can be found in the appendix.

2 Infrastructure and Tools

This system consists of two smaller projects with two separate source code repositories; one to facilitate the web application and another to facilitate the mobile application.

Code repository for Web App:
https://github.com/seanieyap/IFS4205-AY1920-S1-Team02-NUSMed-WebApp

Code repository for Mobile App:
https://github.com/seanieyap/IFS4205-AY1920-S1-Team02-NUSMed-AndroidApp

Firstly, this section will explain the individual components that make up the infrastructure of the system. Secondly, this section will expand on the first point by summarising the tools that are used in the software development process of both the Web and Mobile App, the language and frameworks used during development and the configurations made to support the system.

2.1 Infrastructure

The infrastructure can be summarised as utilizing the ubiquitous "WISA" solution stack with MSSQL database server replaced with MySQL database server, in order to create a complete Web Application platform with efficient security risk mitigation. The web server (server 1) is the only intended entry point to enter the infrastructure; it is the only endpoint while administrator entry point would be the CI/CD server (server 4), utilizing the concept of jump host to access all other machines in the infrastructure.

Details of the individual components in the infrastructure of the can be found in Appendix B.

2.2 Tools

This section serves to inform on-boarding developers and pen-testers about the tools used during the software development process. In addition to the description of the tools and rationale for the choice of tools, there will also be additional software engineering related information. Where relevant, code structures will be explained implicitly via detailed documentation of framework implementations, software engineering design considerations and design patterns.

Details of the individual tools used can be found in Appendix C.

3 Subsystem 1 (Multi-Factor Authentication)

3.1 Overview

The multi-factor authentication (MFA) subsystem involves both what the user knows (password) and what the user has (smartphone & NFC tag). As mentioned in the Tools Assessment Report, the NFC tag is a factor in this MFA subsystem so that it can also facilitate the requirement of validating a data entry of an emergency patient. Using NFC tags would address the ease-of-use concern without neglecting the authentication strength. Another reason NFC tags was chosen as a factor in this MFA subsystem was because NFC tags could be embedded underneath the skin and an attacker would have to be within an uncomfortably close range to read or clone the NFC tag.

3.2 Assumptions

The administrator mentioned in the entire report is the most trustworthy individual and refers to a clerk, receptionist or nurse. The administrator will always be responsible for user registration, including the assigning of NFC tags.

Every user owns an Android smartphone with an in-built NFC reader and, hence, will be able to utilise the services this health care system provides.

3.3 Tools and Software

For the MFA subsystem to function, the user is required to own an Android smartphone running on at least version 6.0 with an in-built NFC reader. The user also needs the NFC tag (NXP MIFARE Ultralight - NTAG213) issued to him by the administrator.

3.4 Implementation

This section briefly describes the MFA subsystem implementation that involves the user registration on both the web and mobile app, the different login mechanisms for both the web and mobile app, as well as the authentication to upload data for an emergency patient.

For more details on the implementation, please refer to Appendix D.

There are two parts to the user registration, with the registration via the web app being the first part. The process begins with the user approaching an administrator to register for an account. This mirrors the common procedure in Singapore whereby a patient first registers him or herself at the registration counter; or perhaps during the induction of a new employee, therapist or researcher. The user will key in his credentials and other details while the administrator will select the appropriate roles to be given to the user. The administrator will then issue an NFC tag with a unique token ID for the user.

In the second part of the user registration, the user will download and install NUSMed's mobile app. After installing the mobile app, the user authenticates with the same credentials used when registering via the web app and scans the issued NFC tag. Each installation of the mobile app will generate a unique device ID which would be tied the user. This ensures only a single device is permitted to scan the NFC tag for a particular user. The user registration process is now complete and the user also remains logged in on the mobile app unless the user's mobile session expires due to inactivity.

The web login is implemented via the use of session and cache mechanisms to store and utilize stateful variables that are invisible to client-side. The user will first authenticate with his username and password. Upon successful authentication, a modal will be shown with a countdown timer of 30 seconds and the user is required to scan his issued NFC tag via the mobile app in that time-frame. If successful, the user will then be logged in on the web app and brought to the "Role Selection" page to select his role from the roles registered for him by the administrator.

To login on the mobile app, the user will key in the same credentials used when registering an account for the web app. The “Scan NFC” page will then be shown and the user is required to scan the NFC tag issued to him. If all parameters are validated successfully by the web server, the web server returns a JSON Web Token (JWT) and the user will be successfully logged in on the mobile app. The user will then be directed to the “Select Role” page to select his role from the roles registered for him by the administrator.

For an emergency patient, he/she would be unable to create/upload any records personally or accept requests from therapists to view/create/upload his/her records. Hence, a therapist has to do so on behalf of the patient. As such, it is necessary to authenticate the identity of the therapist who is doing so on behalf of the patient, as well as the identity of the patient. Upon admission of an emergency patient, an administrator would assign an appropriate and available therapist to the patient through the administrator portal. The assigned therapist will then scan the emergency patient’s NFC tag with his mobile app. As only the assigned therapist is able to scan the emergency patient’s NFC tag successfully, this verifies the identity of both the therapist and patient. After the successful scan, the assigned therapist would be granted permissions to view/create/upload any record for the emergency patient. This access can be revoked when the patient recovers and sets the permission manually.

3.5 Security Considerations

By using TLS1.3, the authentication of the web server to the client is provided. TLS1.3 protects the confidentiality and integrity of any data in transit between the web server and client. Thus, an attacker will not be able to sniff the password during transmission. It also prevents a captured stream of TLS data from being replayed at a later time as seen in replay attacks.

With regard to password storage security, the web app will utilize the key derivation function, Password-Based Key Derivation Function 2 (PBKDF2), to generate a 256 bytes hash from the password. PBKDF2 works by taking in the password and a salt generated by a secure random number generator (RNGCryptoServiceProvider), passing them through a pseudo-random function based on HMAC-SHA512, and iterate 10 000 times to generate the final 256 bytes hashed password. Both the hashed password and salt will be stored into the database. Utilizing PBKDF2 which is slow by design mitigates the risk of successful brute force attacks on the hashed password.

With regard to the generation of the unique token ID on the NFC tag, Java’s UUID class is used to generate a 16 bytes random unique ID. Java’s UUID class uses a cryptographically strong pseudo random number generator to generate the UUID thus the chance of collision is extremely low especially in this use case as at most 8 billion UUIDs will be generated to tag everyone in this world. In the intended implementation if this system is in production, the token ID can be generated and written to the NFC tag by the tag’s manufacturer. The token ID does not have to be personally generated by NUSMed as the multi-factor authentication system involves another unique ID tied to each user’s mobile app. Hence, the tag will serve its purpose as long as the token ID is unique and it is up to the tag manufacturer to implement security mechanisms such as preventing cloning or writing to the tag. For the unique device ID for the mobile app, it will be generated via Java’s UUID class as well. The device ID will be a 16 bytes random unique ID.

With regard to the storage of the device ID, it will be stored in the mobile app’s encrypted shared preferences (EncryptedSharedPreferences) and the encryption algorithm used is AES-256 GCM. The encryption key would be stored securely in Android’s keystore. This secures the data stored in the shared preferences from other applications on the user’s mobile device.

4 Subsystem 2 (Admins, Therapists and Patients Functionalities)

4.1 Overview

This subsystem includes several components that drives the core system of NUSMed. There are many components to this subsystem, mainly, the web app, web services and database. Details of the permissions system will also be housed here.

4.2 Assumptions

Assumptions for this subsystem are split separately according to individual components due to the large amount of items related to each components.

4.2.1 Accounts and Roles

The decision was made to assume that a single user would sometimes possess multiple roles akin to real life situations where a therapist may sometimes fall sick and be a patient. Hence, a single account is able to possess multiple roles at the same time. However, accounts may only be logged into via a single role at a time. Additionally, since there is no business need for concurrent logins for any user, it is not possible for a single user to log in to the system on two separate browsers or machines.

Due to the powerful abilities available to administrators, it is assumed that administrators are unable to perform administrative actions on themselves. They require another administrator to perform actions on their account, ensuring separation of duties. The ability for users of certain roles to edit certain information is also tuned to maintain logic; we assume that while all roles are able to edit their own contact information, other sets of data is restricted, similar to Singapore's medical system. Patients are able to edit their next-of-kin information and no one else would be able to do so, including administrators. Therapists and researchers are unable to edit their job title and department, this ability is instead given to administrators.

4.2.2 (Medical) Records

There is a need to make certain assumptions due to the decision to employ a consent based permissions system; it is assumed that therapists innately require patient's details to proceed with treatment of any kind. The justification here is that the patient's details are crucial to medical diagnosis and treatment. Hence, requests for record permissions will innately request for permission to view patient information. Additionally, this consenting system impacts the process and requirements of the storing of private information and hence there is also the need to assume that records always belong to patients; even when created by therapists, in that, therapists do not possess a record store of his or her own. This is because therapists should not upload a patient's record when he has not been permitted to do so. Patient's personal and record information may only be viewed by therapists if granted permissions by and only by the specific patient.

To make records logical in the medical context, they are assumed to be redacted when viewed by a therapist that do not have the permissions to view them, instead of being removed from view entirely. This is important as therapists needs to know if some information has been omitted to perform medical duties. It is also assumed that permissions, access control and consent system will utilize permission groups by way of record types.

Records can either be plain text, referred to as "content" or be a file such as an image, movie or text. This determination is made innately by the Web App according to the specified record type. Record types and specifications are as follows:

1. Height Measurement (content only): format: Centimetre, cm. values: 0 - 280
2. Weight Measurement (content only): format: Kilogram, kg. values: 0 - 650
3. Temperature Reading (content only): format: Degree Celsius, °C. values: 0 - 100

4. Blood Pressure Reading (content only): format: Systolic Pressure (mmHG) over Diastolic Pressure (mmHG). values: 0 - 250 / 0 - 250)
5. ECG Reading (file only): formats: .txt, .csv. max size: 0.5MB
6. MRI (file only): formats: .jpg, .jpeg, .png. max size: 5MB
7. X-ray (file only): formats: .jpg, .jpeg, .png. max size: 5MB
8. Gait (file only): formats: .txt, .csv, .mp4. max size: 0.5MB (for .txt & .csv), 50MB (for .mp4)

4.2.3 (Medical) Diagnosis

Medical Diagnoses have been explained earlier in this document as crucial to creating a valid world view akin to a healthcare medical system. To achieve this, it is assumed that all patients can be attributed a diagnosis that serves as a sort of treatment history; the history of therapists that assign the diagnosis is recorded and is permanent. Similarly, records can be tagged with diagnoses to aid research that require correlation between diagnosis and record information; also providing patients more information regarding their treatment history.

4.3 Tools and Software

This section has been covered in detail in the Tools Assessment Report. As such, only client side frameworks and security related and related to the interface will be shown here. These frameworks were not disclosed in the earlier report.

Client-side Frameworks:

- JQuery v3.4.1
- Twitter Bootstrap v4.3.1: front-end framework for web development and design
- Font Awesome: font-end framework for icons
- Toastr v2.1.4: JavaScript library for non-blocking notifications

Security Related Frameworks:

- BouncyCastle v1.8.5: Collection of cryptography APIs for encryption; used for encryption of session data, hashing of passwords, encryption cache values and etc.
- HTMLSanitizer v4.0.217: Cleans HTML from constructs to prevent cross site scripting (XSS)
- NWebsec v5.1.1: Security library to configure security headers, detect re-directions and control cache headers.

4.4 Implementation

This section entails the implementation of the Web Application, mainly the interfaces, functionalities and other quirks involving patient, therapist and administrator roles as well as the permissions system.

Details of the individual components in the implementation and the permissions system can be found in Appendix E.

4.5 Security Considerations

The first and most important security consideration is to secure the transportation of data between the web server and end users. TLS1.3 is implemented via NUS SOC reverse proxy to to ensure confidentiality and integrity from and to the web-server, Android app and user's web browser. Likewise, to prevent sniffing that reveals MySQL commands and return queries between the web app and database servers, database servers and the web app will need to be configured to only accept remote connection via TLS1.3 and nothing lower.

Authentication needs to be strongly enforced such that modification to account states will permeate effects globally and instantaneously. Hence, every request is validated for authorization via values in the form ticket to enforce role based access control to determine access on the directory level and then functional level. Authentication is also strengthened by utilizing application cache to detect and prevent concurrent login sessions.

All user related parameters that modifies SQL queries are parameterized to prevent SQL injection of all forms. Cross site scripting (XSS) is mitigated by sanitizing all inputs and outputs. Additionally, placing outputs from database into HTML controls that do not permit browser execution adds on an additional layer of mitigation in event of issues with sanitization. For instance, placing outputs into disabled text-boxes when possible instead of plain text. Cross-Site Request Forgery (CSRF) is another vulnerability that could allow unauthorized actions. To deter attackers, viewstate mechanism and page token placed on the every page will validate pages and prevent CSRF.

Lastly, software engineering, design and code mistakes are prevented from being leaked by not displaying error codes, debug information and status messages through the redirection and use of custom error pages that are static and obtuse.

5 Subsystem 3 (Researcher's Functionalities)

5.1 Overview

This subsystem allows researchers to view and download the anonymised records of patients which fit their search criteria. It comprises three parts - the generation of a large data set, the implementation of a k-anonymity algorithm and the retrieval of relevant data.

5.2 Assumptions

Medical records are assumed to have been de-identified. Diagnoses and record related information are treated as data and not identifiers to a patient.

5.3 Tools and Software

The tools and software used for this subsystem will be similar to the ones listed under Subsection 4.3. Python and pandas library were also used for data generation.

5.4 Implementation

This section gives a brief overview of the approach used in the generation of data, the anonymisation of records and the retrieval of records that fit the filters set. For a detailed description of the implementation, please refer to Appendix F.

5.4.1 Data Generation

Generation of the basic information of user accounts was done via a Python script with data, such as names and addresses, pulled from the Internet. Patients were then assigned diagnoses taken from ICD-10 2016.

5.4.2 Anonymisation

Two anonymisation techniques - *Generalisation* and *Suppression* - were used. *Generalisation* involves replacing an individual value of an attribute with a broader range of values while *Suppression* refers to the omission of records whose set of quasi-identifiers appear in less than k record.

The Datafly algorithm was implemented in our application to achieve k-anonymity. In this algorithm, the quasi-identifier with the greatest number of distinct values is generalised. Records with quasi-identifiers which occur less than k times will then be suppressed. In our application, the value of k and the suppression threshold has been set to 3 and 10% respectively.

5.4.3 Queries

Researchers are able to filter for patients and records by specifying patient data, record data, or both. Patient data include age, postal code, sex, gender, marital status and diagnoses while record data comprise diagnoses and record type.

5.5 Security Considerations

The retrieval of anonymised records on the fly could allow for the deduction of certain sensitive information as attackers could input values for the different quasi-identifiers separately. To mitigate this problem, the k-anonymity algorithm will be run on all the records once a day.

6 Subsystem 5 (Data Collection from Sensors)

6.1 Overview

This subsystem allows patient and therapist to upload medical records from mobile device. It consists of three parts: patient uploads records for himself, therapist uploads records for his patients, and therapist uploads records for emergency patient.

6.2 Assumptions

All medical records to be uploaded, including files and sensor data, are already stored in user's device.

The therapist in charge of the unconscious patient is able to get the patient's NFC tag so that he can obtain permission to upload records for this patient.

6.3 Tools and Software

This subsystem requires Android smartphones running on at least version 6.0 with an in-built NFC reader which have been registered in NUSMed's system. It also involves the NFC tag of the patient or therapist.

6.4 Implementation

This section explains the implementation of record upload from NUSMed's mobile app. In this subsystem, user is able to securely upload medical data (i.e. sensor readings, images and videos) from his local device (Android phone) to the database. Regarding the design purpose of this subsystem, only two types of users, patient and therapist, are allowed to upload medical records to database from their devices. Specifically, a patient can upload records for himself, while a therapist can upload records for his patients.

A patient is able to upload all types of medical records from his device, including sensor readings, time series data, images, and videos. A therapist can upload records for his patients only if the type of record to be uploaded is permitted by the patient. For unconscious patients, after admin has assigned an emergency therapist to the patient, only this therapist is able to scan the patient's NFC to authenticate himself and accept the association, and then he can upload all types of medical records regarding to this patient.

For each medical type of records that user can upload, the content and format are restricted by the system via the record type stipulated, on top of system wide policies. The checking of all record information are performed on both mobile application side and web server side.

For the detailed implementation of this subsystem, please refer to Appendix G.

6.5 Security Considerations

Since the system is using TLS1.3 for communication and data transfer, it ensures the authentication of the web server to the client, as well as the confidentiality and integrity of data transmitting between the web server and the client. Therefore, attacker is unable to sniff the content of the record during transmission, or to capture the packets and perform replay attacks.

To ensure that the medical record is uploaded by an authenticated patient/therapist, user has to login from his mobile device and obtain a valid JWT before uploading records. When user uploads a record, the device ID and JWT are also sent along with record information for the server to validate. Since attacker is unable to craft a legitimate JWT due to the signature attached by server, he is unable to spoof a record uploading request without authentication.

For therapist uploading records for his patients, after server validates the authenticity and permission of the therapist, web server is responsible for assigning the ownership of the record to the particular patient. Once the record has been successfully uploaded, information related to the record is then expunged from the device.

7 Security Claims

This section details the numerous security related claims that developers of the system are confident of providing users. Claims are coded with a unique prefix and identifier to provide easy identification during bug and issue reporting; and are split into several sections according to their area of relevance within the system. There are a total of 15 security claims.

7.1 Server and Infrastructure Security Claims

This section contains claims whose security mechanisms are implicitly provided by servers and or infrastructure components. There are 3 server and infrastructure security claims in total.

S1-TLS

It is not possible to perform sniffing and man-in-the-middle attacks on the following connections due to TLS implementations. It is not possible to initiate HTTP (non-HTTPs) connections with the following as well. This ensures confidentiality of the information between the relevant parties.

- Between NUS SOC reverse proxy and end users using the Web App and/or Mobile App

  Connection secured via TLS1.3 mechanisms provided via NUS SOC reverse proxy.
- Between web server (server 1) and main database (server 2); and logging database (server 5)

  Connection secured via MySQL TLS1.2; certificates generated via OpenSSL.
- Between web server (server 1) and file server (server 4)

  Connection secured via SMB 3.1.1.

S2-ACCESS

It is not possible to access any systems or services that are not intended to be accessible.

All servers except the web server are configured not be exposed to the public. The database servers are configured to only accept connections from the web server using a specific MySQL account and the file server only accepts connections from a specific service account.

S3-MOBILESTORAGE

It is not possible to retrieve the device ID or JWT from the shared preferences of NUSMed's mobile app via another application installed on the mobile device.

The device ID and JWT is stored in NUSMed's mobile app's encrypted shared preferences (EncryptedShared-Preferences) and the encryption algorithm used is AES-256 GCM. The encryption key would be stored securely in Android's keystore. This secures the data stored in the shared preferences from other applications on the user's mobile device.

7.2 Web / Mobile Application Security Claims

This section contains claims whose security mechanisms are provided by application frameworks and or purposeful developed code that has overall security effects on the entire Web and or Mobile application. There are 4 Web / Mobile application security claims.

W1-SQLINJECT

It is not possible for an attacker to perform SQL injection attacks throughout the entire system.

The following are the several mechanisms that prevents SQL injection throughout the Web App.

- Web Application Firewall, ModSecurity, rules and blacklisting

  ModSecurity is a Web Application Firewall that rejects request that are blacklisted as according to latest rules provided via the open source ModSecurity community. Requests to the Web Server are always filtered by this firewall and dropped if rules are found to be broken.

- ASP.NET page validation

  ASP.NET page validation ensures that any and all requests are validated to not possess any malicious strings. Detected malicious SQL queries in page requests will be dropped.

- Defensive programming, specifically, the use of AJAX and defensive error handling

  The use of AJAX means that page responses are HTML chunks that do not contain explicit information that may allow attackers from analysing page responses and thus guess parameters to perform further attacks.

- Parameterization

  User input parameters are all parameterized before being used in SQL queries. This prevents malicious SQL inputs to be executed due to parsing that will modify the malicious input into safe strings to be used in compiling queries.

W2-XSS

It is not possible for an attacker to perform cross site scripting attacks throughout the entire system.

The following are mechanisms that prevents cross site scripting, and its variations, throughout the Web App.

- Web Application Firewall, ModSecurity, rules and blacklisting

  ModSecurity is a Web Application Firewall that is packaged as an IIS module that enables IIS to reject request that are blacklisted as according to latest rules provided via the open source ModSecurity community. Requests to the Web Server are always filtered by the ModSecurity module and requests are dropped if some blacklisted rule are found to have been broken.

- ASP.NET page validation

  ASP.NET page validation ensures that any and all requests are validated to not possess any malicious strings. This prevents malicious scripts from being inputted into the database or written onto pages, hence, hinders the process of stored XSS.

- Defensive programming

  The lack of usage of query strings in the Web App prevents rudimentary non-persistent (reflected) XSS from taking place. The surface area of persistent (stored) XSS is also reduced by using HTML tags that prevents the execution of scripts whenever possible to display information; this prevents execution of malicious scripts and thus mitigates even the possibility of victims executing malicious stored XSS attacks. All data seen by the client are also HTML encoded to reduce the possibility of script execution.

- Use of HttpOnly cookie

  To mitigate risk of the possible XSS exploits, the HttpOnly flag of the authentication cookie is explicitly set. This prevents client side scripts from accessing the protected cookie, which in this case, holds authentication information.

W3-CSRF

It is not possible for an attacker to perform cross site request forgery attacks throughout the entire system.

Anti CSRF Tokens / page tokens are present via master-page, as such are included on all pages. These tokens are randomly-generated values that are verified on every request preceding the first request. If the anti CSRF tokens cannot be verified, the request does not proceed and an exception is raised. This prevents attackers from crafting a malicious form / request that would be successfully be verified by the Web App and thus prevent malicious requests from taking action.

W4-SESSION

It is not possible for any single user to initiate 2 concurrent sessions at any time; in that an account is able to be logged in more than once to achieve 2 concurrent sessions.

A single user is allowed to log into the Web App and maintain only one session at any time. If the same user tries via another location, no matter the browser, computer or system, the prior session associated with the user will be terminated. This does not apply to replay attack via obtaining authentication and session cookie. Replay attack is re-mediated via the secure communication channel implemented via TLS1.3.

7.3 Functional Security Claims

This section contains claims whose security mechanisms are provided by specific and purposeful developed code. There are 8 functional security claims.

F1-JWT

It is not possible for attackers to craft or modify a JWT that enables him/her to authenticate and login.

The JWT is signed with the private key of the server and it is thus impossible for an attacker to craft a JWT since the signature will never be valid without the right private key. The signature also indicates the integrity of the data in the JWT since any changes to the JWT contents results in a different signature.

F2-FORMAUTH

It is not possible for an attacker to craft their own Form Authentication Cookie that enables him/her to authenticate and login into the web application.

Form authentication cookie is encrypted and contains a GUID that is given during the login process and verified on every subsequent request.

F3-MFA

It is not possible for an attacker to perform unauthorized actions without all three secrets: password, Device ID, Token ID.

The MFA system is designed in such a way that a user (and his password) is associated with a single mobile device and a single NFC tag and all three are necessary to login to both the web and mobile app. Even if an attacker knows the token ID of an NFC tag, he/she would be unable to do anything without knowing whom the tag belongs to, which device is associated with the tag, and what password to use to sign in.

This also mitigates attacks via social engineering or phishing sites. It is highly difficult for an attacker to convince a user to give his/her password, mobile device and NFC tag all at once. A phishing site is also at most able to phish a user's password unless the attacker goes the extra mile to create an app identical to NUSMed's mobile app. However, that is detectable as well since the fake app would have been signed with a different certificate from the legitimate app.

F4-KANON

It is not possible to identify a patient from the quasi-identifiers.

This is achieved by running a modified Datafly algorithm on all patients with at least one medical record. The Datafly algorithm ensures k-anonymity as it continues generalisation should k-anonymity not be satisfied and suppresses the records when further generalisation is not required or not possible. Records are assumed to have been de-identified and it is to be noted that record details and diagnoses should not be treated as identifiers to a patient as that is not within the scope of the project.

F5-ACCESSCONTROL

It is not possible for therapist, patients or admins to perform any action outside of their given roles and permissions as according to functional specifications. This includes viewing unauthorized information.

F6-PASS

It is not possible to obtain any user account password via cracking, guessing or other means.

The account lockout feature prevents password guessing by locking out user accounts after 3 failed login attempts made after 5 minutes.

F7-RECORD

It is not possible to modify any record that had been previously uploaded; be it owned by him/herself and or others. This includes the record's attached records, contents and other related data such as title, description and etc.

F8-FILE

It is not possible for an attacker to upload file types or file sizes that is not specified to be allowed by the system; and thus are restricted by the system. This extends to exploiting the file upload to perform remote code execution, remote file inclusion and other related attacks.

ModSecurity Web Application Firewall (WAF), use of Windows file server with File Resource Server Management (FRSM) accessed via UNC and defensive programming techniques prevents file types and sizes outside of functional specifications to be uploaded and execution of files located in the file server.

Appendices

A Enhanced Entity Relationship (EER) Diagram

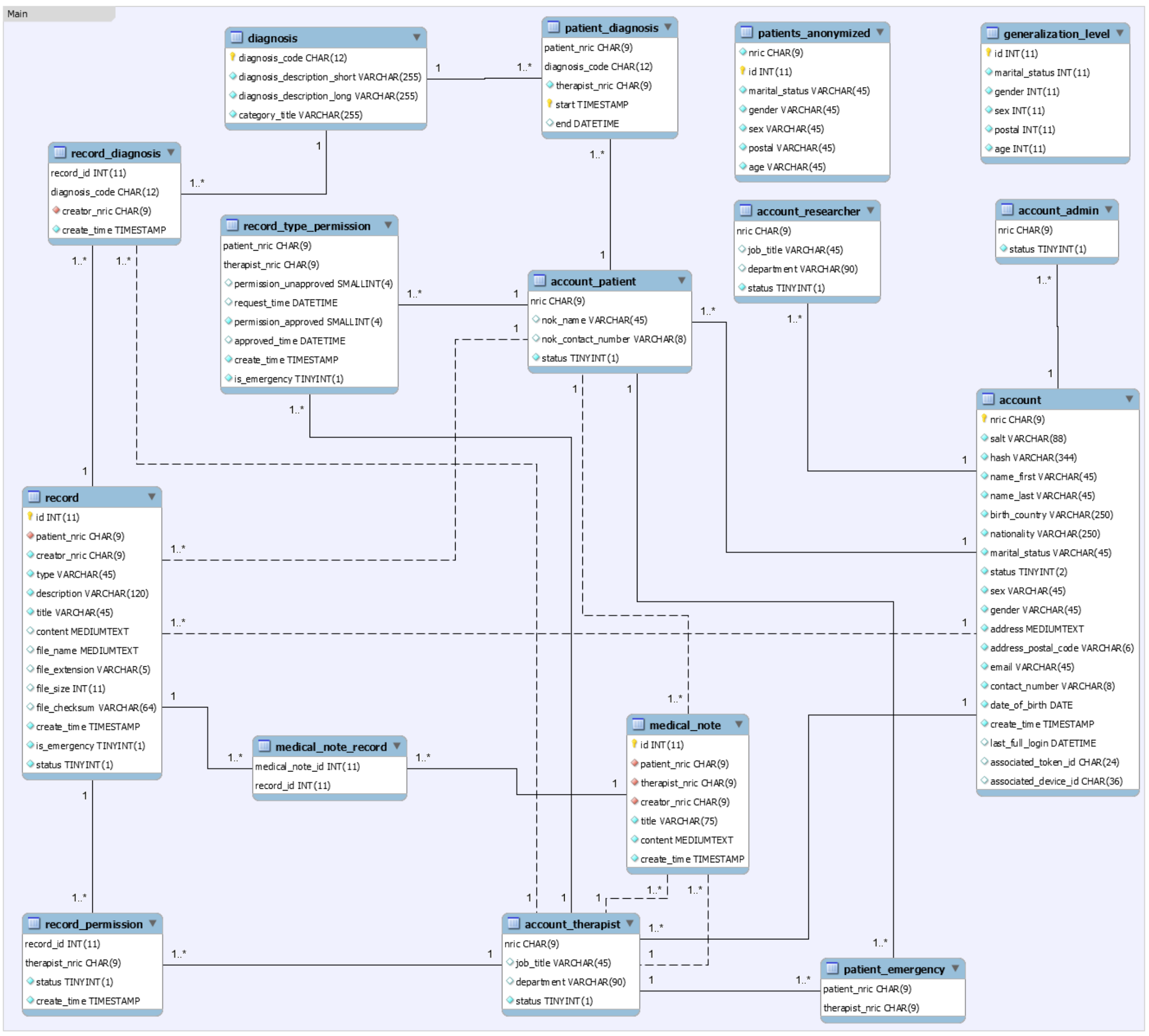

Figure 1: Enhanced Entity Relationship (EER) diagram of the "main" database

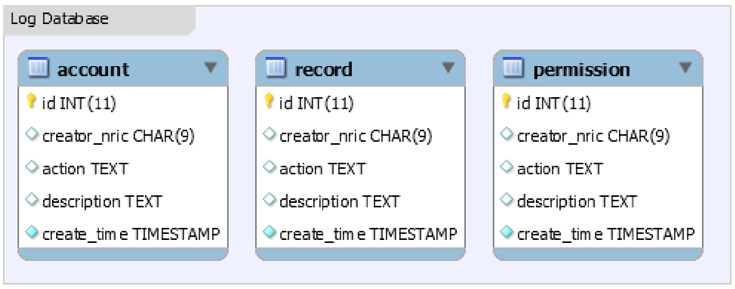

Figure 2: Enhanced Entity Relationship (EER) diagram of the "logging" database

B Infrastructure Components

This appendix section seek to provide detailed information on the individual components present in the infrastructure. An architecture diagram is shown below to help visualise the infrastructure of the system.

Note that red arrows show communications essential to operations while blue arrows show admin related communications.

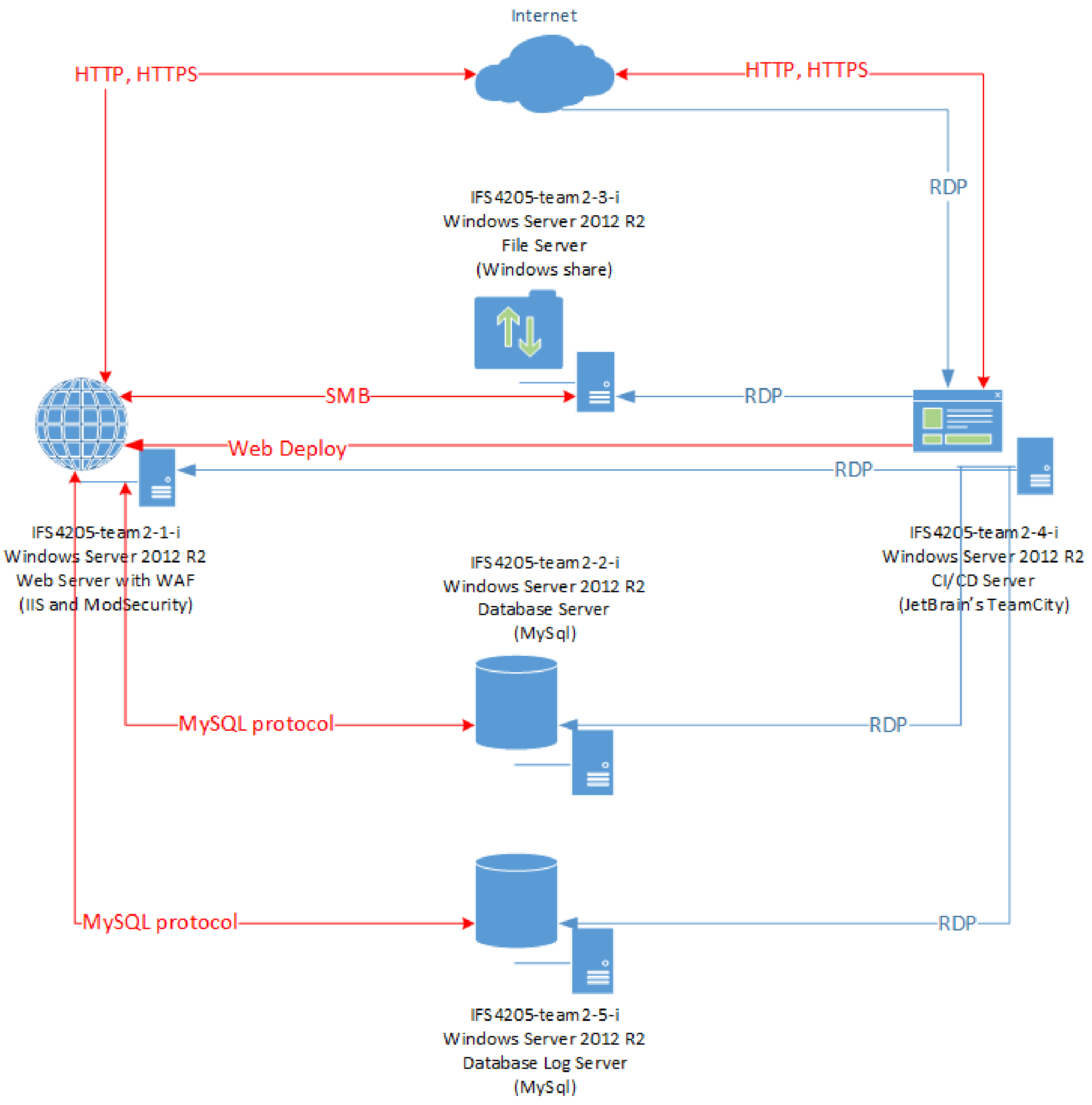

Figure 3: Architecture Diagram of entire system

B.1 Server Operating Systems

Chosen: Windows Server 2012 R2 Standard

All 5 machines in the infrastructure run on Windows Server 2012 R2. It was chosen as the operation system as firstly, it completes the solution stack resembling the “WINS” stack. This means that there are no additional software needed to support the system/application such that the solution of this system is at its minimal set, reducing the surface area of attack due to lesser and redundant application or services. Secondly, Windows

Server 2012 R3 Standard comes in-built with Remote Desktop Protocol (RDP) that provides a graphic interface for connecting to these Virtual Machines (VM); allowing for more efficient administrative control. Thirdly, Windows Server 2012 R3 Standard reduces the amount of third-party components needed to be installed manually from external sources; for example, Internet Information Services 10 (IIS) is in-built and File Server via Windows share or IIS File Server is in-built. The File Server is also easily support by File Server Resource Manager (FSRM) to filter files in and out of the system.

B.2 Web Software / Server

Chosen: Internet Information Services (IIS) 10 with ModSecurity Web Application Firewall (WAF)

The IIS instance on the web server is configured via web.config settings from the web application; hence it does not have nor require any specific configuration other than to run the default application pool using a specific Active Directory (AD) service account and having a web deploy installed and configured to allow deployment via a specific web deploy user account.

Additionally, a Web Application Firewall (WAF), ModSecurity, is also installed as an IIS module. ModSecurity was chosen because of its ease of use and open source community that provides free and up-to-date firewall rules. It filters requests through a series of rules, mostly black-list rules, to provide some measures against SQL injection, XSS attacks and more. This is discussed more in detail in the security claims section.

B.3 Database Software / Server

Chosen: MySQL version 8

There are two database servers in the infrastructure, one called "main" as it houses main production data and the other called "logging" as it houses logging related data. MySQL was chosen for this project as it checks all the boxes of being open-source, free and has the ability to perform database modeling. It is also the easiest and quickest database system to install and configure out of the alternatives. TLS1.2 is enabled and configured using certificates generated via OpenSSL. The databases are also configured to accept connections only from local and the web server via specific database user accounts; used by the Web App, which credentials are injected via TeamCity.

B.4 File Server

Chosen: Windows Shared Folder, Server Message Block (SMB) 3.1.1

Server Message Block (SMB) is a network communication protocol for providing shared access to files, printers, and serial ports between nodes on a network. It works through a client-server approach, where a client makes specific requests and the server responds accordingly. It supports the same levels of security features as FTPS. However, utilizing Windows shared folder in conjunction with granting access to a specific service account for the web application to login to the folder would be a stronger authentication mechanism due to reliance on Windows Inter-process Communication (IPC) mechanisms to perform to facilitate communication between processes and computers over the shared nodes.

Additionally, compared to FTPS, SMB is more efficient for the project needs as SMB, being a network communication protocol, allows for clients to randomly read or write from a file residing on a different node. FTPS on the other hand, requires the client to retrieve the entirety of the file. On the other hand, transfer speeds of FTPS tends to exceed SMB due to lesser overhead. Regardless, SMB 3.1.1 is chosen as it is more purposefully build for retrieving files for users over web applications whereby entire files need not be downloaded fully before other actions may take place; such as in the case of video files. It is a more reliable protocol to communicate within a private network. Additionally, File Server Resource Manager is installed and enabled on the file server to white-list only files of specific file types to be created. The shared folder that is in use is also configured to be read and written by a specific Active Directory (AD) service account; the account does not have execute permissions.

B.5 Build Management and Continuous Integration (CI) Software / Server

Chosen: TeamCity

TeamCity is a propriety build management and continuous integration server from JetBrains that was chosen for this project due to its value. It is famous for being reliable, high quality, easy to use, easy to navigate and, essentially, easy to configure; having provisions for popular development systems in-built, such as the ability to build and publish via MSBuild. It also has the ability to install plugins and increase its number of features. Most importantly, the free version of TeamCity has the generous restriction of a maximum of 100 build configurations and 1 build agent; more than enough for this project without worrying for expanding team size. Being said, TeamCity is notably a compromise of both Jenkins and Azure DevOps Server Express. Whilst it does not perform better in the niche areas of either Jenkins nor Azure DevOps Server Express, it does not have the drawbacks present in both alternatives too.

Builds and deployments are executed if changes are detected from the Web App's Github repository. First, the server runs several scripts to replace the development related keys with production values in the web.config of the Web App source code. After which, the server builds and checks. Lastly and if previous processes were successful, the server deploys the built deployment package via a specific web deployment user account that has been configured on the IIS web server. The TeamCity server is accessible outside of the infrastructure for administrators to manage builds and deployments.

C Tools

This appendix section seek to provide detailed information on the tools used during the software development process. Tools are split into two sections, "Frameworks and Languages" and "Development Tools" in order to compartmentalise information. Individuals not interested in the development tools used during the initial implementation period do not need to visit the latter section. In contrast, the former section is crucial to be read in order to understand the inner workings of the Web and Mobile App.

C.1 Frameworks and Languages

This appendix section provides detailed information to understand the rational leading to the initial implementation of the system as well as provides some guidance to understanding the code base; perhaps in order to review or perform modifications to the system.

C.1.1 Web Application Framework and "Main Programming Language"

Chosen: ASP.NET Web Forms, C#, with Form Authentication and Membership.

To remove the steep learning curve, ASP.NET Web Forms instead of MVC was chosen for its Modelview-viewmodel (MVVM) software architectural pattern that all team members are familiar with, also it separates the graphical user interface (GUI), in this case web pages, from the business logic (back-end logic). This allows for both simple and accelerated development process, if properly planned for, thus making up for the lack of manpower. This pattern, however, does come with drawbacks, such as confusing UI related code in large pages and data bindings that result in relatively larger memory consumption. Regardless, since it is not expected for any web page in the planned application to be absurdly large; also, the project would not have many UI components, this issue should not pose a problem. Another factor in the consideration is the Web Forms view engine that allows for a more traditional approach to rendering pages that does not require more than rudimentary experience and understanding of client-side technologies; it abstracts it away and thus takes care of some part of the development.

On the topic of security, there is little that Web Forms, despite being an older framework, does worse at as compared to the other alternate frameworks listed above. This framework simply does things in a different method such as in the case of software design pattern and server-side rendering components and etc. These are feature sets that we opt to make use of to aid in software engineering, ease development and most importantly, lessen man hours to focus on implementing security features. ASP.NET provides some mechanisms out of the box that helps mitigate many vulnerabilities such as page validation, viewstate and Anti XSS encoding. It is notable that while ASP.NET is regarded as open-source, the single component of Web Form controls are proprietary. The version of ASP.NET Framework that was utilized is 4.8, the most updated and recommended version to utilize.

The Web Application has a code structure typical of a ASP.NET Web Forms application, utilizing the "3 layer architecture", the most popular subset of the multi-tier architecture (also often referred to as n-tier architecture) or multilayered architecture. The architecture is a client-server architecture in which presentation, application processing, and data management functions are physically separated. In context of the Web App and typical ASP.NET Web Forms applications, the 3 layers are as follows.

- Presentation Layer consists of the web pages and their attached code behind.
- Business Logic Layer performs functional business actions and are housed in the "BLL" code folder.
- Data Access Layer performs database related actions and are housed in the "DAL" code folder.

Web server configuration are provided by the "web.config" file hosted at the root of the source code which is expected to be taken in by the deployed IIS server. It is to be noted that keys and secrets in the open source repository in the configuration file are dummy keys for development environment. Production related keys and secrets are currently injected before build execution and deployment via TeamCity pre-build scripts.

C.1.2 Multi-Factor Authentication Framework

Chosen: Near-field communication (NFC) tags (NXP MIFARE Ultralight - NTAG213)

NFC tags typically covers a range of up to 10 centimeters which makes it much more apt for our authentication system due to the short range. There are passive NFC tags which are powered from an NFC reader which essentially means these tags can last forever and work as long as there is an NFC reader to power it. These passive NFC tags are cheap and it is also possible for these tags to be implanted under the skin. The NFC tags can also be read by smartphones with an in-built NFC reader and is thus more feasible and practical since a majority of the population owns a smartphone.

C.1.3 Mobile Application Framework

Chosen: Android application for Android phones (at least version 6.0) with in-built NFC reader

As we have chosen NFC technology as part of the MFA process, we thus built a mobile application for it since smartphones can function as a NFC reader as well. As such, phones minimally require an in-built NFC reader. Choosing to build a mobile application for the MFA process also makes it convenient for potential users since the mobile application could integrate functions such as uploading patient data like photos and videos.

We do not have phones running on different operating systems (OS) or the necessary development environment for applications on other OS (e.g. XCode). Hence, we have chosen to only develop the application on Android and make the assumption that all potential users have and use an Android phone with an in-built NFC reader.

C.2 Development Tools

This appendix section provides detailed information to understand the tools that were used in the software development process in developing the initial release. These tools are development tools, and as such, may be replaced by alternatives. These tools can be thought of as subjective and up to personal preference.

C.2.1 Integrated Development Environment (IDE) for C# Development

Chosen: Visual Studio Community 2019

Visual Studio Community 2019 was used to develop the web application. It is the recommended IDE by Microsoft and is community supported. This community version is selected as it is free and supports the features that we require for the development of this project. Visual Studio invaluably includes in-built capabilities that would otherwise result in more work to include, as plugins, in other IDEs; for example, Git support, memory debugger, code publishing, code analysis, express server support and more.

C.2.2 Integrated Development Environment (IDE) for Android Application Development

Chosen: Android Studio 3.5

Android Studio was used to develop an Android application which works with NFC. Android Studio is the official recommended development environment for Android developers. The application is developed with API level 23.

C.2.3 Integrated Development Environment (IDE) for SQL development, administration and database design

Chosen: MySQL Workbench

MySQL Workbench is the propriety SQL management tool provided by Oracle, developed by the community. It is provided free and comes bundled with MySQL community database server. It is not only used as a database management tool but as a database design tool to model diagrams. It is to be noted that Visual Studio has an inbuilt database management tool but MySQL Workbench is used for more highly intensive tasks.

D Implementation of subsystem 1

This section details the implementation of the multi-factor authentication system. It is split into 4 parts: User Registration (Web), User Registration (Mobile), User Login (Web), User Login (Mobile).

D.1 User Registration (Web)

The following details out the user registration process via the web app in steps.

1. User approaches an administrator to register for an account.
2. Administrator logs in to the NUSMed web app, selects the administrator role, and navigates to the registration page.
3. Administrator refers to the user to key in his details including his NRIC and password.
4. Administrator selects the roles for the user and submits the form.
5. Administrator will manually set the user's associated token ID to the value of the unique token ID from a NFC tag.
6. Administrator will issue the NFC tag to the user.

D.2 User Registration (Mobile)

The following details out the user registration process via the mobile app in steps.

1. User will download and install NUSMed's mobile app.
2. User will log onto the mobile app with the same NRIC and password he provided during registration with the web app.
3. User will then scan the issued NFC tag from the mobile app.
4. Upon successful authentication, the user will remain permanently logged in on the mobile app unless the user's mobile session expires due to inactivity.
5. The user registration process is now complete.

D.3 User Login (Web)

The following details out the user login process via the web app in steps.

1. User logs in on the web app with his NRIC and password.
2. If the credentials are correct, a modal will be shown with a countdown timer of 30 seconds.
3. Within the 30 seconds, the user is required to scan his issued NFC tag via the mobile app.
4. If the authentication is successful, the user will be brought to the "Role Selection" page and will be able to select from his/her registered roles for the current session.

D.4 User Login (Mobile)

The following details out the user login process via the mobile app in steps.

1. User logs in on the mobile app with his NRIC and password.
2. The "Scan NFC" page will then be shown and the user is required to scan the NFC tag issued to him.
3. Upon successful authentication by the web server, a JWT token is returned to the mobile app and the user is successfully logged in onto the mobile app.

4. The user remains permanently logged in on the mobile app unless the JWT token expires after 15 minutes of inactivity.
5. The user is then brought to the “Select Role” page to select his role from the roles registered for him by the administrator.

E Implementation of subsystem 2

Subsystem 2 involves the implementation of a web interface for patients and therapists to mainly manage medical records via a permission system that integrates the act of requesting and giving consent. This subsystem has implementations that not only involves patients and therapists but all functionalities involving records due to the permissions system. Furthermore, there are also administrator functionalities to manage the system.

The permission system involves patient information and medical records that is assessed through a number of items. For medical records, a therapist is allowed access if and only if he or she has both requested and received consent, has permissions to the specific record type of the medical record, has no fine-grain blacklist permission associated with the therapist (specified from the patient) and if the record has no global blacklist permission set. Permissions regarding patient information is much simplified, in that, a therapist is allowed access if and only if he or she has requested and received consent. The patient might not have given any permissions to the therapist for any record type but permission to patient information will be given implicitly.

The sub-sections here will go into detail the implementation of the interfaces available in the Web App. It is separated into 5 sub-sections. They are namely, implementation of the role section feature, shared interface, interface for patients, interface for therapists and interface for administrators.

E.1 Role Selection

Upon login, if an account possess multiple roles that are "active", they will be redirected to the Role Selection page. The role the web app assigns to the user, upon login, would be called "Multiple" instead of any role. To be specific, the ticket encrypted in the cookie would have that role instead of the other real and proper roles. This ensures that the user is unable to access any pages or functionalities except to visit this Role Selection page to select another role.

In the event that a user, such as patients, only possess one single role, this whole process will be omitted and the user will be able to obtain the correct role straight away without interruption. In both cases, upon obtaining a proper role, the user will be assigned a proper ticket with the selected role that will be encrypted in a cookie and will be redirected to the dashboards of the specific role.

E.2 Shared Interface for authenticated users

All authenticated users share the same interface but with restricted access to directories and pages. This section details the implementation for functionalities shared across roles.

All authenticated users with proper roles are able to view the "My Account" drop-down which displays pages with functionalities related to account information and authentication. The drop-down contains the following features.

- My Profile

  This page allows users to view and edit, certain, information related to them, including their personal information, contact information and etc. Here, patients are also additionally able to edit his or her next-of-kin information while therapists and researchers are able to edit his or her job information. Fields not related to the selected role at the moment of time are not shown to the user.

- Change Password

  This page allows users to change their password. Password generation follows the login and registration process of password security, checking and storing.

- Switch Role

  "Switch Role" is not a web page but a button that allows users to return to the role selection page to select another role instead of going through the entire process of logout and login simply to switch roles.

- Logout

  "Logout" is not a web page but another button that simply signs the user out via removing the form authentication ticket and updating database values to indicate user logout.

In the event of concurrent logins, the older session will be kicked out and redirected to the home page with a popup shown.

E.3 Interface for Patients

Patients have access to these pages:

- Dashboard

  Upon login, patients will first be presented with a dashboard showing statistics such as how many therapists have access to his information, number of records and etc. Most importantly, his past and present diagnosis are presented in a timeline view. Lastly, important notifications are presented here such as in the event for therapists requesting for permissions.

- My Therapists

  Patients are able to visit this page to view a list of therapists. Both therapists that have been granted or are requesting permissions are shown in a list format. The listing here are sorted in descending order by the date and time of which the permissions has been requested for.

  Here, many actions can be performed such as changing permissions of therapists; for example, the removal of all permissions or granting some permissions. Therapists can also be revoked of permissions to remove his or her access to patient information; this is intended to be used at the end of treatment. Additionally, there is also a button to view the therapist's non-sensitive information; such as first name, last name, job title and department he or she belongs to.

  Therapists whom permissions have not been approved/granted or received via the emergency system will be displayed with a non-intrusive informational sign. These signs serves to inform the user the abnormal event and to invoke action to either grant consent or revoke permissions. Detailed event information can be viewed via tool-tips by hovering on the signs.

  All these actions are performed via Ajax and do not feature any full page loads. All of these actions also restrict patients to a world view without knowledge of other patients or therapists. He or she are unable to grant therapists of permissions unless requested for and are not able to view any other therapists.

- My Diagnosis

  Patients are able to view the diagnoses that their therapists have attributed to them via this page. They are not able to modify nor delete any attributed diagnosis.

- My Records

  - View

    This page displays all of the patient's records (records that he or she owns) in a table format; this includes information such as title, record type, date of creation, person who created record and description. Fine-grain permissions and global record permissions can also be modified here.

  Records that have been created by a therapist that has permissions received via the emergency system will be displayed with non-intrusive informational signs. These signs serves to inform the user the abnormal event and to invoke action to either grant consent or revoke permissions of the therapist. Similarly, detailed event information can be viewed via tool-tips by hovering on the signs.

  Records can be downloaded or viewed in the browser. Text, images and videos are supported for viewing in the browser via HTML 5 player that is inbuilt into modern browsers and is displayed to the patient via modal popup.

  - Upload New Record

    This page is solely for patients to upload records, via a full post-back form. This form is dynamically altered via Ajax depending on the record type chosen. This is because, as stated above, certain record types only requires text content while others requires a file to act as the record and thus be uploaded. For instance, if the record type chosen is of type "Temperature Reading", the form will be altered to display a multi-line textbox instead of upload file control.

E.4 Interface for Therapists

Therapists have access to these pages:

- Dashboard

  Upon login, therapists will first be presented with a dashboard showing statistics such as the number of patients he currently has permissions to and the number of patients that has not granted him or her permissions. To facilitate usability, instructions on how the system works with regards to permissions and record handling are displayed on this page along with links to other pages that therapists have access to.

- My Patients

  - View

    This page serves all functionalities related to a therapist's patients. There are a few subsets of functionalities here that can be seen in 3 groupings.

    This page allows therapists to view his or her patients. Both patients that have granted or pending permissions are shown in a list format. The listing here will be sorted in descending order by the date and time of which the permissions has been requested for. Here, there are several buttons on each row and patient, each performing a functionality. The permissions button performs permission related functions such as requesting a new set of permissions, via a collection of record types, or revoke patient permission completely (akin to treatment has ended). The view information button displays the patient's private information; such as personal, contact and next of kin information. Additionally, the view records button displays the patients records. This will be further explained below.

    If permissions have not been granted, the normal patient item will not be shown, instead, the item will be replaced with a different item that does not show restricted patient information and information informing the user of the lack of permissions.

    Lastly, via the list of patients, records can be viewed through selecting the "view records" button which triggers a modal. At this stage, the web app retrieves and updates the underlying grid view on the modal, populating it with entries that would be the selected patient's records. Each record can then be interacted with in similar methods as with the patient's list. This includes information such as title, record type, date of creation, person who created record and description. Therapists only has read access to the records and are unable to edit nor delete records. Text, images and videos are supported for viewing in the browser via HTML 5 player that is inbuilt into modern browsers and is displayed to the patient via another modal popup.

    All these actions are performed via Ajax and will not feature any full page loads. This allows for reusable functionalities to be developed for use in other pages as well.

  – Submit New Request

    This page provides therapists with the functionality of requesting for permissions from a patient that is, perhaps, not under his care. This can be seen as the initial action in getting consent. For instance a newly registered patient first meets his therapist. In this page, the therapists is allowed to only search through a list of all the patients via NRIC to identify the patient of interest and select "request". Using only NRIC and not first nor last name will prevent therapists from searching for others other than the patients he or she comes into contact with.

    A modal popup will once again be used to allow the therapist to select what record types he would like to request permissions for and select "submit", to submit the request.

- My Medical Notes

  This section has two pages to view and create medical notes. Therapists are able to create medical notes that is associated with a certain patient and specific records to be shared amongst fellow therapists. The process of created notes are similar to other functionalities that have been covered above. The permissions approved by patients onto therapists persists here and should persist such that therapists should not be able to view those records of patients if they do not have permissions to do so; even when shared.

E.5 Interface for Administrators

Administrators have access to these pages:

- Dashboard

  Upon login, administrators will first be presented with a dashboard showing the functionalities he or she has access to.

- Manage Accounts

  – View Accounts

    This page provides administrators the ability to view and manage all accounts. First, on this page, administrators are able to perform crucial actions such as changing the status of roles, changing the account status, changing MFA token IDs (but unable to view) and changing staff (therapists and researchers) information of accounts. All account information may be viewed by the administrators. However, personal, contact and next-of-kin information can only be edited by the account holder; administrators are prevented from editing these information to prevent misuse.

    To ensure data privacy, administrators are unable to search the list of accounts in any other methods except via NRIC. All information about the accounts that returns in a search are displayed in a format that hides all information except the NRIC. Administrators need to choose and click buttons specific to certain functionalities to trigger the modal popup to display the specific user interface relevant for those actions. This prevents simple incidents such as patients shoulder-surfing administrators such as clerks working at their desks.

  – Register Account

    This page as stated in subsystem 1 facilitates the registration of a user. Administrators will be filling in all the items in this form except for the password fields. Similar to some services in

Singapore, such as hotels, the administrators would use this form in registering an account for the user in question and and oversee the user in changing his or her password.

- View Logs

  The following three pages displays a table displaying logs ordered in descending order by date and time. Administrators are able to filter the table by subject, action, date from and date to. Each type of logs are segregated into three pages to allow for easy viewing of logs.

  – Account Logs

  – Records Logs

  – Permission Logs

- Researcher Data

  This page serves as a page to automatically trigger the k-anonymization algorithm that is detailed in subsystem 3. The page contains a single button, "Re-Generate Data" to start the algorithm that will overwrite existing k-anonymize related data in the database that is provided to researchers. The process takes up to a minute; during this period of time, mitigations put in place prevents other administrators from triggering the process.

F Implementation of subsystem 3

Subsystem 3 involves the implementation of a web interface for researchers to access the medical records of anonymised patients. This subsystem requires the generation of a large dataset, the implementation of a k-anonymisation algorithm and the retrieval of relevant medical records. This section provides a description of the implementation of this subsystem.

F.1 Data Generation

To populate the database, a series of MySQL INSERT statements were generated and then executed in MySQL Workbench.

These series of INSERT statements were generated using a Python script with the help of pandas library.

For clarity purposes, *record* in this context refers to a record in the database while *medical record* refers to a medical reading or scan of a patient.

Information that had to be generated for the creation of records in the tables were NRIC, first name, last name, date of birth, birth country, nationality, marital status, sex, gender, address, email, contact number, hospital department, medical record creation time and treatment dates.

The list of possible first names, last names, birth countries, nationalities, addresses, diagnoses and hospital departments were obtained from online sources. Each record was assigned a random value from these lists with the exception of birth country and nationality.

Since our application assumes a local context, a majority of the records have 'Singapore' as their birth country and 'Singaporean' as their nationality. The remaining records were assigned a random birth country and nationality.

NRIC numbers were generated randomly and were prefixed and suffixed with a character that is dependent on their nationality while contact numbers were generated in the same way except that they were prefixed with either 6, 8 or 9.

A random date generator was created to facilitate the assignment of date-related information to a a record, for example the date of birth and medical record creation time. A medical record was assigned a creation time that lies within a treatment time period. In addition, the starting and ending treatment date of a patient by a particular therapist was assigned such that the therapist was at a suitable age during the time of treatment.

The relevant information were then stored in either a list or a dataframe and then iterated through to generate the SQL statements.

F.2 Data Anonymisation

Anonymisation of data was achieved by utilising the Datafly algorithm. The Datafly algorithm uses generalisation and suppression to achieve k-anonymity. Generalisation involves replacing an individual value of an attribute with a more general range of values while Suppression refers to the omission of records whose set of quasi-identifiers appear in less than k records. The Datafly algorithm is a greedy heuristic algorithm that begins generalisation from the quasi-identifier that is most diverse. Since the suppression of records is generally discouraged, a modification of the algorithm has been made to include a suppression threshold. The different quasi-identifiers will first be generalised. Eventually, once the number of records that can be suppressed falls below than the threshold, suppression will occur. However, if it is not possible to generalise further, suppression will still occur even if the number of records to suppress exceeds the threshold.

In our application, the quasi-identifiers are age of patient, sex, gender, marital status and postal code. The Value Generalisation Hierarchies of each of the quasi-identifiers are shown in the following figures. These Value Generalisation Hierarchies were created with reference to the Value Generalisation Hierarchies used in relevant research papers.

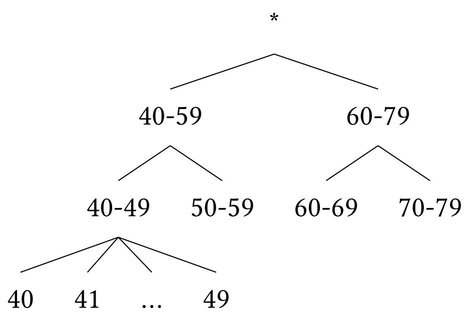

Figure 4: Value Generalisation Hierarchy of Age

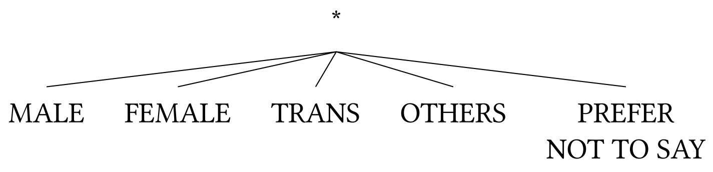

Figure 5: Value Generalisation Hierarchy of Gender

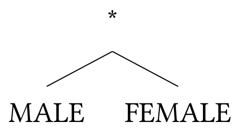

Figure 6: Value Generalisation Hierarchy of Sex

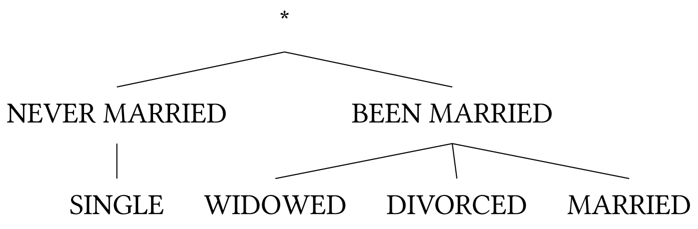

Figure 7: Value Generalisation Hierarchy of Marital Status

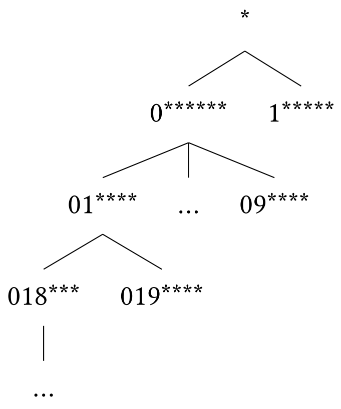

Figure 8: Value Generalisation Hierarchy of Postal Code

F.3 Data Querying

Researchers are able to select multiple values for each of the filters. Filters are provided for both patient data and record data. Filters for patient data include age, sex, gender, postal code and diagnosis. Filters for record data include record type and diagnosis. Multiple values selected from the same filter will be concatenated by an 'OR' condition while values selected from different filters will be concatenated by an 'AND' condition in the MySQL query used.

Upon clicking of the View or Download button, the relevant patients and their records will be retrieved from the database.

F.4 Interface for Researchers

Researchers have a dashboard page with a link that directs them to the 'Search Data' page where they can filter for anonymized records. Filters for quasi-identifiers are marked with a tiny information logo next to it.

Each filter has a multi-select dropdown menu and this enables researchers to select multiple values for it. For example, a researcher can view or download records of patients whose marital status are either single or widowed.

The View button leads to the display of a table consisting of all patients who fit the filters set. Researchers are then able to view the diagnoses and records of each patient in a modal popup.

In the modal popup for records, researchers are able to view and download individual medical records.

Clicking the Download button on the Search Data page triggers the download of a CSV file containing information of the patients and their records.

G Implementation of subsystem 5

This section details the implementation of the data collection from sensors via the NUSMed's mobile app. It is split into 4 parts: Data Collection/Retrieval, Data Upload by Patient, Data Upload by Therapist for Normal Patient, Data Upload by Therapist for Emergency Patient.

G.1 Data Collection / Retrieval

Medical data files are assumed to be originally taken and stored in the mobile device instead of retrieving from other remote sources. For example, image and video records are taken using phone's camera and video recorder, and therefore auto-stored in the local storage of the phone.

Sensor data can be collected using sensor applications in the phone or remote sensor devices (i.e. Fitbit). These types of records may not necessarily be stored somewhere in the phone since they are just values instead of contained in a file. User can just get the value and enter it manually in the application when uploading data.

G.2 Data Upload by Patient

In order for a patient to upload records for himself, he needs to first login using his credentials and NFC tag, select his role as "Patient", and select "Upload Record". In the patient's upload page, patient needs to enter record title, record description, record type and data. If all the fields are valid, patient can upload the record successfully.

As mentioned in Subsystem 2, there are in total eight medical types. Each type has restrictions on the data being uploaded:

1. Height Measurement (content only): format: Centimetre, cm. values: 0 - 280
2. Weight Measurement (content only): format: Kilogram, kg. values: 0 - 650
3. Temperature Reading (content only): format: Degree Celsius, °C. values: 0 - 100
4. Blood Pressure Reading (content only): format: Systolic Pressure (mmHG) over Diastolic Pressure (mmHG). values: 0 - 250 / 0 - 250)
5. ECG Reading (file only): formats: .txt, .csv. max size: 0.5MB
6. MRI (file only): formats: .jpg, .jpeg, .png. max size: 5MB
7. X-ray (file only): formats: .jpg, .jpeg, .png. max size: 5MB
8. Gait (file only): formats: .txt, .csv, .mp4. max size: 0.5MB (for .txt & .csv), 50MB (for .mp4)

There are two types of checks on the mobile application side as well as one check on the web server side before the records has been successfully uploaded to database. The mobile application checks patient's input every time when the patient keys in a character or selects a medical file, and displays an error for any invalid input. When patient clicks the "Upload" button, the application performs an overall check on all the patient inputs again, encodes them and sends them to the server along with the JSON Web Token (JWT) and device ID. Server first checks the JWT and device ID to ensure that the patient is uploading record using his registered device and that the session is still active. Then, server decodes the patient inputs and checks if all the fields are valid. For medical files, server decodes the content of the file and checks if the actual size matches the size claimed by the patient in order to detect any maliciously crafted request. If all the checks are passed, server adds the record into the database. For medical files such as time series data, images and videos, server also creates a directory in the file server and puts the file into the directory.

G.3 Data Upload by Therapist for Normal Patient

To upload records for patients, therapist needs to first login from his registered device, select his role as "Therapist", and select "Upload Record (for patient)". Apart from all the input fields mentioned in the patient's record upload page, therapist also needs to select the patient that he wants to upload record for. If all the input fields are valid and the medical type is permitted by the selected patient, the therapist can successfully upload the record for this patient.

Therapist is able to get a list of his patients once he has logged in as therapist. If the therapist does not have any patients at the moment, he will get an empty patients list such that he is unable to select any patient to upload records for. If a patient used to be the therapist's patient but revoked the therapist's access after the treatment, the therapist can still get the patient's NRIC as a proof of previous treatment, but he is unable to upload any type of records for this patient, as all the medical types have been removed from the selection field. As for the therapist's active patients, a subset of medical types are shown in the selection field based on patient's permissions for the therapist.

Similar to patient record upload activity, there are two types of checks on the mobile application side as well as one check on the web server side before the records has been successfully uploaded to database. Although therapist is able to only upload records with types that are permitted by the patient, server also checks whether the therapist has permission on this medical type in order to detect maliciously crafted request. If server detects that the upload request is spoofed by attacker with disallowed medical type, it will not add the record into the database but still let the request pass so that attacker is unable to gain information on the permissions given by the patient.

G.4 Data Upload by Therapist for Emergency Patient

As mentioned in Subsystem 1, since the patient is unconscious and therefore unable to upload records by himself or accept permission requests from therapists, a therapist has to be associated with the patient such that he can get full access to the patient's information and records. In order to achieve that, admin first assign the emergency patient to the therapist. After that, the therapist logs into his account from his mobile device, select role as "Therapist" and select "Scan NFC (for emergency patient)". Only this therapist can successfully scan the patient's NFC to confirm the association, and therefore gain full access to the patient information. From both his mobile device and web, the therapist is able to upload any type of records for the patient during the period that the patient is unconscious. After the patient becomes conscious again, he can reset the therapist's permission, and deny the ownership of records uploaded by the therapist when he was unconscious.

-End-