New York University Abu Dhabi
CS-UH 2012: Software Engineering
Bayan Assali
Muaath Assali
Hyun Woo Lim
Muhammad Hasin Shabbir
Software Requirements Specification
for
Medifind

Version 1.0 **Date: (10/14/2022)**

Table of Contents

1. Introduction

1.1 Purpose

The purpose of this document is to specify the user and system requirements for the web-based database of drugs called *Medifind*. The document will describe everything that the development team should follow in order to ensure that the needs and requirements of all stakeholders are satisfied. The documents will be used by, but not limited to, the following parties.

a) The stakeholders of Medifind will use this document to
   i) Ensure that the requirements specifications meet the stockholders' needs
   ii) Identify changes needed to improve the system (if applicable)
b) The project managers will use this document to
   i) Plan a bid for the project
   ii) Create Project Development Work Plan
   iii) Make estimation of the completion
   iv) Schedule deliverables
   v) Track project progress
c) The system engineers will use this document to
   i) Develop validation tests
d) The system maintenance engineers will use this document to
   i) Understand the system and relationship between different parts of the system
   ii) Maintain and fix software code (if required)
   iii) Make improvements to the existing system

1.2 Scope

The document is created for a web-based database of drugs - Medifind.

Medifind is created to provide easily accessible information for the end-users of drugs. This system will create drug description pages based on the Admin's input of data on drugs. Medifind's product description page supports multimedia information to provide a detailed description of drugs with a user-friendly design to increase the readability of drug information.

The system will create a QR code with the URL of the respective drug. The QR code can be printed and attached to the drug product or packaging for future reference. Users will be able to search for specific drugs from the database.

The digital database of the drugs enables a variety of potential applications that were not possible with traditional printed information provided with the purchase of drugs.

The main benefits of Medifind for its users are as follows.

a) Easier access to drug information.
b) Access to drug information with higher readability and legibility.
c) Better understanding of drug usage through multimedia content.
    i) Decreased Medication Error due to lack of drug knowledge.
d) Increased user satisfaction with drug use.

1.3 Definitions, Acronyms, and Abbreviations

Definitions:

- Admin: Administrator responsible for the system management.
- User: Any individual authorized to create an account on the system.
- Drugs: Any medication including prescription and Off-the-Counter (OTC) drugs
- Frontend/client-side: The project containing the website with HTML/CSS/JS that the end-user interacts with. It handles input, initial data processing and providing the user with information and system output. Frontend, client-side, and website are used interchangeably in this document.
- Backend/server-side: The project that exposes API endpoints for the front-end, processes any user requests from the end-user, and operates on the database. Backend and server-side are use interchangeably in this document.

Abbreviations:

- HTML: Hypertext Markup Language
- HTTP: Hypertext Transfer Protocol
- HTTPS: Hypertext Transfer Protocol Secure
- DB: Database
- API: Application Programming Interface
- UI: User Interface
- GUI: Graphical User Interface
- ER: Entity Relationship

1.4 References

Frontend/Client-side:

- CSS documentation provided by Mozilla Corporation: https://developer.mozilla.org/en-US/docs/Web/CSS

- Javascript documentation provided by Mozilla Corporation: https://developer.mozilla.org/en-US/docs/Web/JavaScript
- ReactJS documentation provided by Meta: https://reactjs.org/docs/getting-started.html

Backend/Server-side:

- C# documentation provided by Microsoft: https://learn.microsoft.com/en-us/dotnet/csharp/
- .NET documentation provided by Microsoft: https://learn.microsoft.com/en-us/dotnet/
- ASP.NET documentation provided by Microsoft: https://learn.microsoft.com/en-us/aspnet/core/?view=aspnetcore-6.0
- Npgsql documentation provided by Npgsql development team: https://www.npgsql.org/

Database:

- PostgreSQL documentation provided by PostgreSQL global development group: https://www.postgresql.org/docs/

1.5 Overview

In this subsection: (1) Describe what the rest of the SRS contains (2) Explain how the SRS is organized Don't rehash the table of contents here. Point people to the parts of the document they are most concerned with. Customers/potential users care about section 2, developers care about section 3.

The following *Overall Description* section is created for the owners of Medifind and potential users. This section is written in plain language to provide a detailed description of the system and functionalities. The section is used to establish the context for the Specific Requirements section.

The *Specific Requirements* section is created mainly for the developers. It is written with technical terms to identify the requirements of the system in development context.

Both Overall Description and Specific Requirements sections cover all necessary details regarding Medifind, but are written for different audiences.

The information regarding the process of development and additional information are provided at the end of this document.

2. The Overall Description

2.1 Product Perspective

Medifind is an online drug database that allows users to view all relevant information on their prescribed drugs. Users of the application are able to view their drug information from a QR code which they scan on the drug or by accessing the drug through the website. The drug information includes instructions on use, precautions, drug dose, warnings, multimedia content on method of drug application, and more. Medifind provides an easy and accessible way to get drug information for end-users.

Another major user of Medifind is the system admin. The Admin account will be responsible for providing drug details from credited sources and manage the system as a whole. This entity is mostly likely to be a stakeholder of Medifind, for example a member of the government, who has authorized access to drug details databases. However, as pharmacies are added as a potential member managing the application, they will also have access to an Admin account. The Admin will be able to add, manage, and modify drug details from the existing database. As well as view accounts of end-users.

As Medifind is an online application, it is a web-based system that is accessible to any end user with an account. Special access is needed to upload drug information and this is only permissible through an admin account. The drug information will be taken from an existing database provided by the stakeholder. The following block diagram shows how Medifind interacts with other systems to perform the required functionalities.

Figure 1: Block Diagram showing interaction of user with Medifind and other systems

As shown in the block diagram (Figure 1), Medifind is an application dependent on other systems and external interfaces. The user first interacts with a web browser that provides the user interface on the client side. Medifind will be hosted on a web server and it will interact with other systems and APIs such as the drug database to access drug details.

2.1.1 System Interfaces *

Medifind will interact with several interfaces to meet the system requirements. Those include:

- A website that provides the user interface. The user will be able to login in and access all drug information.
- A web server to send messages and data from the DB and APIs to the website.
- An API to interact with the system's own drug DB to store, add, and search drug information.
- Registration API to create Admin and user accounts and login system.
- An API to generate unique QR codes for each drug.
- An API for screen reading drug information

2.1.2 Interfaces

The end-users will interact directly with the website. Through the browser the user will be able create or login to their account through which they can access their profile information. The profile information will contain the saved drugs and those from the QR code. The user will also interact with the web server when searching the drug information, specifically accessing the drug DB. If the user is an admin, they will be able to interact with the web server differently by editing drug pages in the DB.

To make the application accessible to all types of users, the website will aim to achieve ADA compliance through several ways. The website will be equipped with screen readers such that the users can benefit from the information without having to read it. The website will also provide multimedia content with captions for easier understanding while maintaining the same depth of content. The website will provide two languages for the drug descriptions as used by the majority in the country, those languages include both English and Arabic. More languages will be added as the product develops to increase the reach of the product.

2.1.3 Hardware Interfaces

The system is only expected to interact with a keyboard, mouse, and touch screens, all of which are natively supported. Other than that there are no hardware interface requirements.

2.1.4 Software Interfaces

- The systems must ensure cross-browser compatibility and work the supported versions of the browsers list in the following table:

Table 1: A list of Medifind's Supported Browsers

Browser name	Google Chrome	Safari	Microsoft Edge	Mozilla Firefox

Minimum Supported Version	100.0.0 or above	15.0 or above	96.0.0.0 or above	102.0.0 or above
Engine	Blink	WebKit	Blink	Gecko

- For interacting with databases, the system must use the API's in their latest version to ensure proper functionality as requested by the customer.
- The system will use PostgreSQL for the database. Communication between the backend server and the database will be done using the Npgsql library.

2.1.5 Communications Interfaces

- For communication between the client (browser) and the server, the protocol to be used will be HTTPS.
- The system will make use of HTTP POST and GET messages to receive and send responses from the server.
- A certificate will be obtained from a certificate authority for establishing and maintaining HTTPS connection throughout the deployment life of the system.

2.1.6 Memory Constraints

There are no memory constraints of the system. The memory constraints placed on the user will be based on browser support and the ability of the user to download the minimally supported browser in their devices.

2.1.7 Operations

- Backup will be done in stages on a weekly basis to prevent bringing the system down.
- The drug DB will be updated on a daily basis to provide users with the latest drug details.
- Website maintenance will be performed every two months of operation.

2.1.8 Site Adaptation Requirements

- New data tables created for Medifind must be installed on the existing DB server and populated prior to system activation.
- QR codes must be mapped to drugs from the DB prior to the system activation.

2.2 Product Functions

As there are two different intended users of Medifind, Admin and regular user (access drug information), the system will perform different functions accordingly:

Regular user

- Allow users to register in the system.
- Allow users to login into the system using their credentials.
- Allow users to access drug details by scanning QR code and redirecting the user to the correct page.
- Allow users to search for drugs using the drug name, manufacturer, and ingredients.
- Allow users to save searched drugs into their profile.
- The client-side should be able to show the user's previously saved drugs in the profile page.
- Allow users to log out from the system.

Admin

- Allow user to register in the system
- Allow users to login into the system using their credentials
- Allow users to add new drugs to the database
- Allow user to edit stored drug information in the DB
- Allow users to search for drugs using the drug name, manufacturer, and ingredients.
- Allow user to upload multimedia content to the website
- Admin should have unrestricted system-wide access to control user privileges
- and make manual or automatic adjustments.
- Allow user to log out of the system
- Allow user to delete account from the system

2.3 User Characteristics

The *regular* user should be an individual who has basic knowledge on how to navigate through common interfaces on websites. The user should have a valid email address through which they can login into the system. The user should have a basic understanding of the English language or Arabic language.

The *admin* user should be an authorized member from the stakeholder, for example from a governmental organization. The user should have a valid email address which they can use to create and login into an account on Medifind. The member should be able to navigate smoothly through interfaces on websites. The member should be proficient in the English language. The member should have a strong understanding of the medical field, graduating with at least a medical degree, as they will be editing drug details from the DB if required.

Forms of disability support will be provided within the system if necessary, such as screen-readers and multimedia content.

2.4 Constraints

- Ensuring that the system is able to import a large amount of data from the drug DB
- Ensuring fast processing and searching through the DB
- Ensuring that the system is cross-browser compatible
- Ensuring that the system allows for real time updates
- Ensuring Admin privacy and security
- Ensuring DB privacy and security

2.5 Assumptions and Dependencies

- The browsers versions listed in section 2.1.4 will be supported without design changes when the application is running

2.6 Apportioning of Requirements

All system requirements will be implemented as communicated to the customer and project management at Medifind in the time frame agreed on. The system will be running once all functionalities are deployed. After the delivery of the system, there will be future updates occasionally, but not requirements-related, unless intended by the customer.

3. Specific Requirements

3.1 External interfaces

3.1.1 User Interface

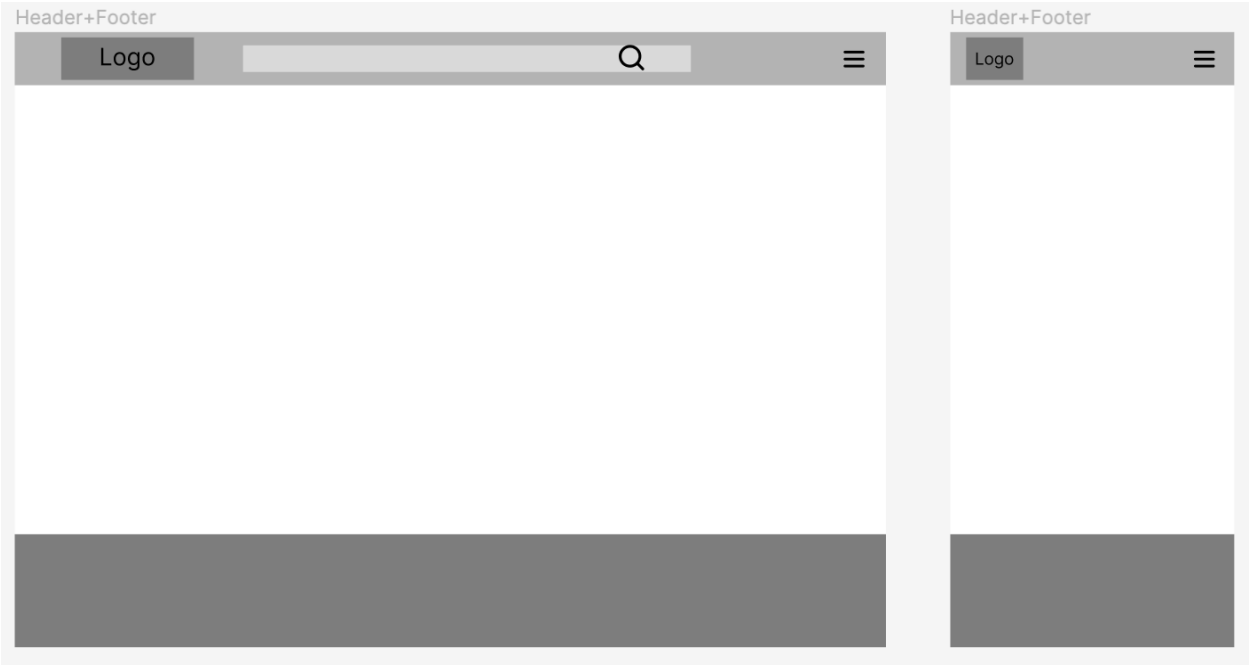
This section details the interface that the end user will interact with in order to use the system. The interface consists of web pages that show the user the system output and some forms to obtain user input.

3.1.1.1 Common Structure Across Pages

Description:
All user interfaces will share the same properties and structure described here unless specified otherwise in the specific user interface subsection.

- Common header and footer across all pages.
- Clicking on the logo redirects the user back to the main landing page.

Wireframe: Desktop and mobile format



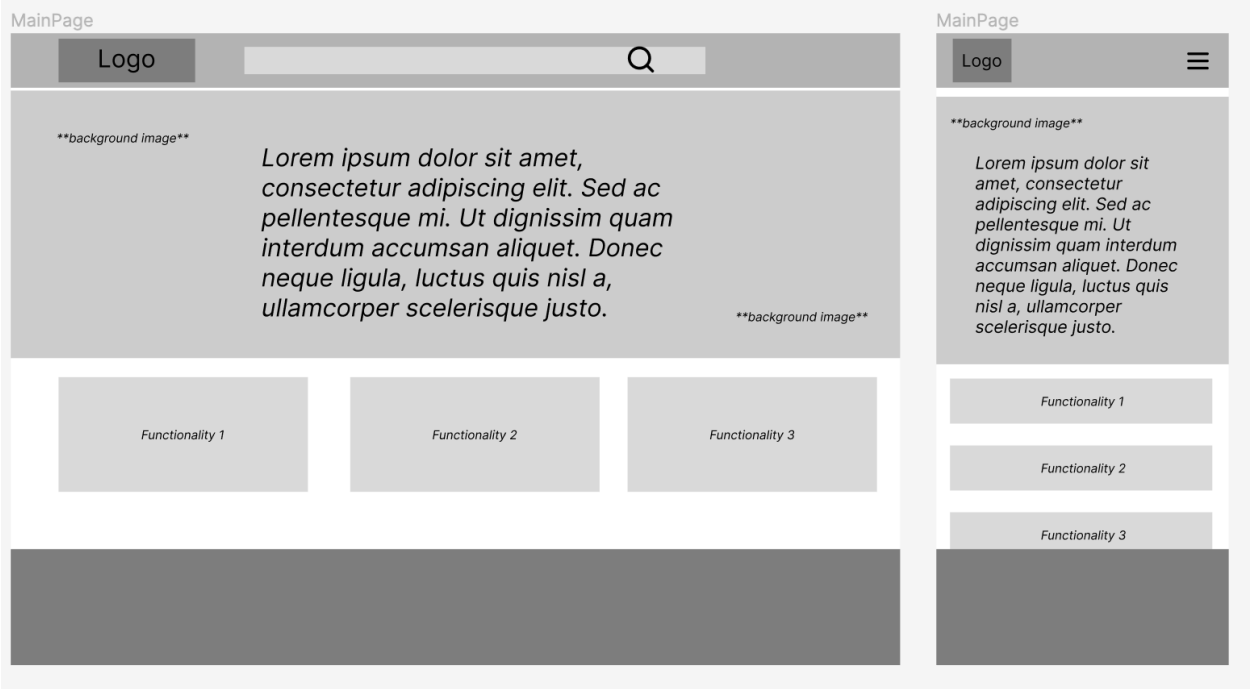
3.1.1.2 Main Page

Description:
This will be the landing page for the website that will provide some description about the system. It includes a search bar at the top to enable users to look through different drugs and their information. Submitting the search input will redirect the user to the interface 3.1.1.3 to display the search results.

Valid input:

- Search bar text input has a maximum length of 100 characters.

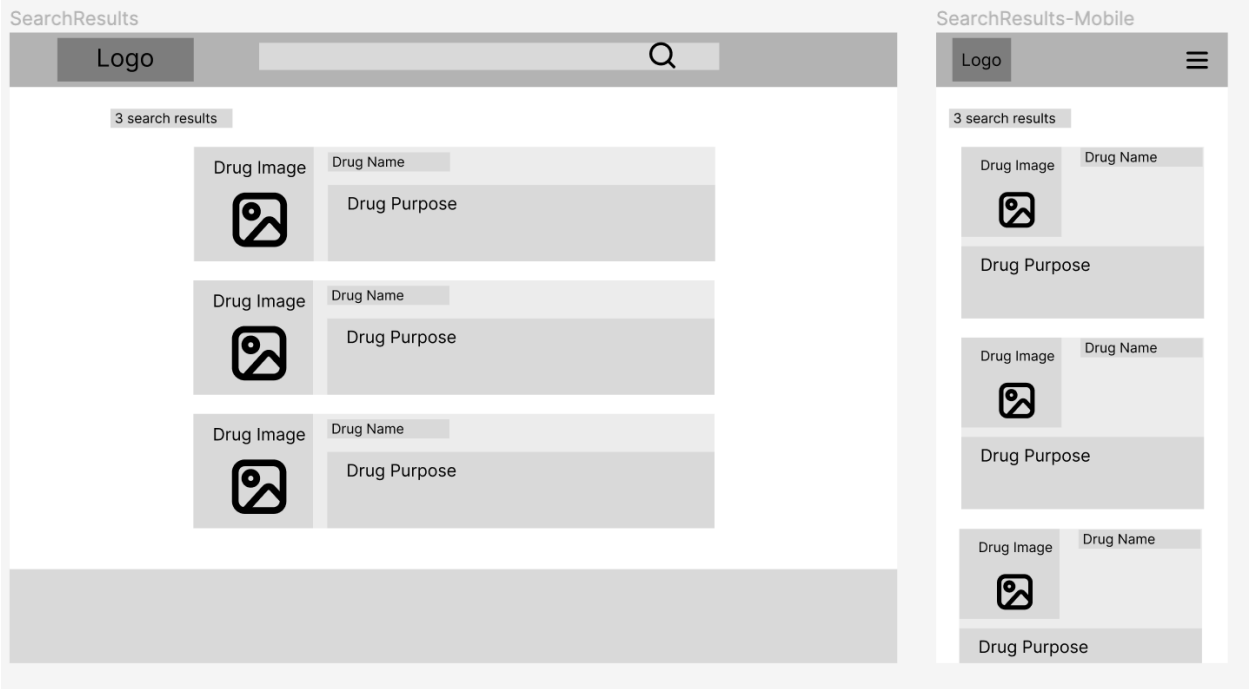
Wireframe: Desktop and mobile format



3.1.1.3 Search Results Page

Description:
This page will display the results found based on the user's search input from interface 3.1.1.2. The maximum number of results displayed per page is 10. Clicking on any of the found results will redirect the user to interface 3.1.1.4 to show the full details of the selected result.

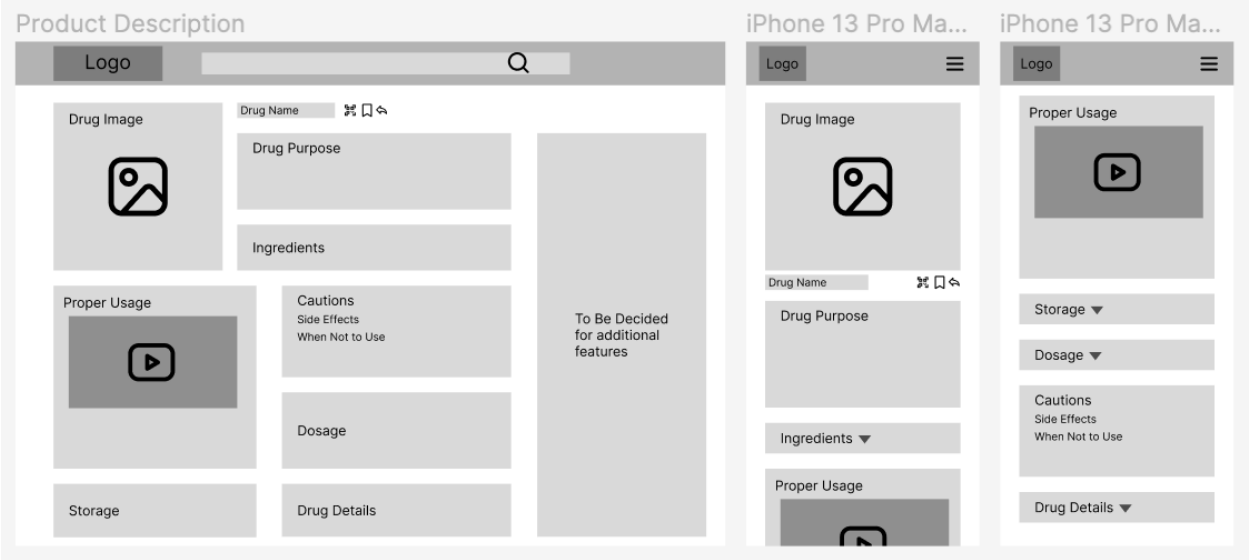
Wireframe:



3.1.1.4 Drug Information Page

Description:
This page will show all the information about a specific drug to the user. Regular users have the ability to bookmark a certain drug, generate a QR code and share the drug's URL using three buttons. Admin can also edit the drug's information or delete the drug. If an admin clicks on "edit" then they will be redirected to the interface 3.1.1.5 for the drug input form.

Wireframe:



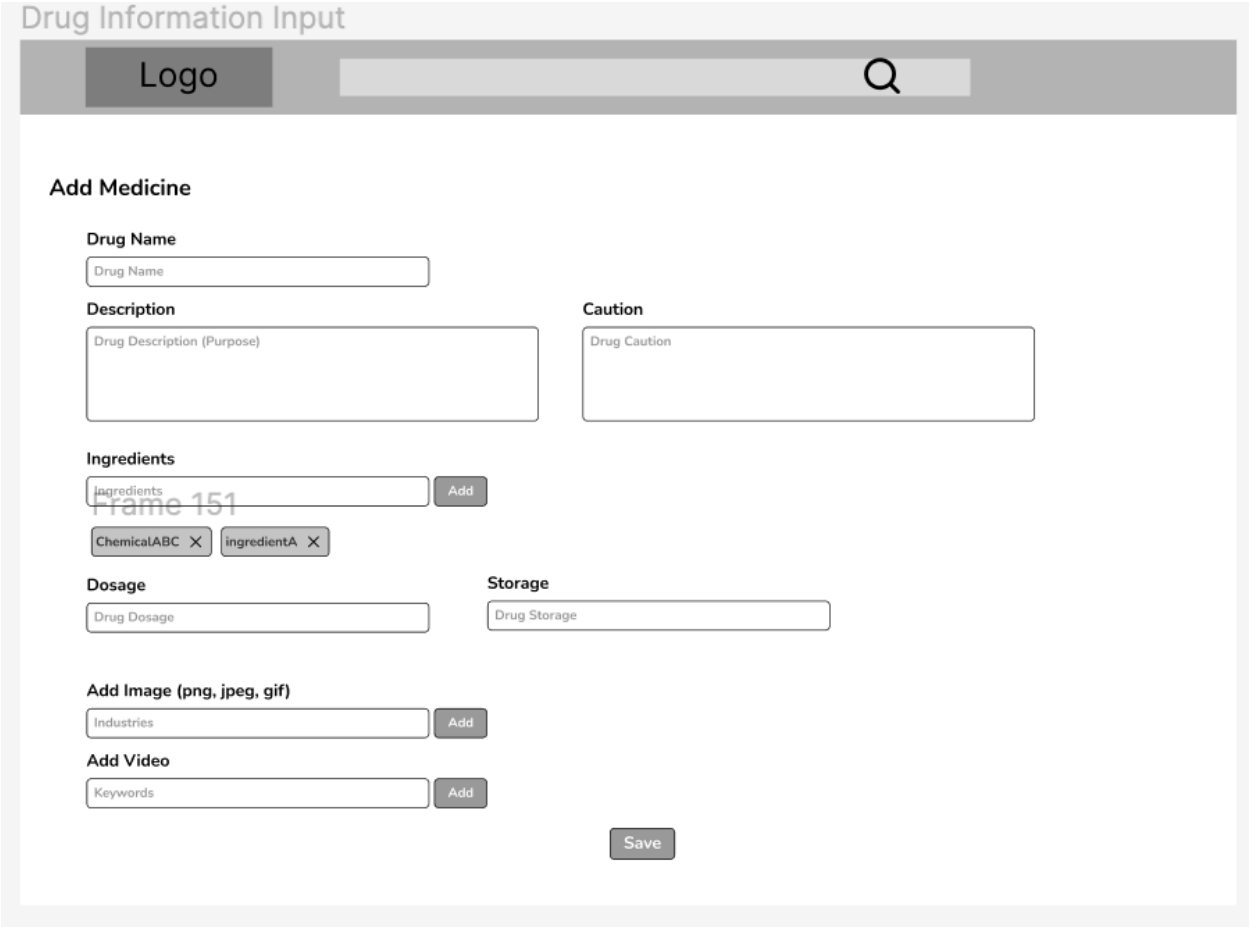
3.1.1.5 Drug Input Form Page

Description:
This page is accessible by admins only and is used to add new drugs to the system or edit existing ones. Admins can input or edit drug name, manufacturer, ingredients, drug image (optional), drug usage, drug usage video (optional), side effects, when not to use, dosage, storage, and drug details.

Valid Input:

- Drug name is required and has a max length of 100 characters.
- Manufacturer is required and has a max length of 100 characters.
- Purpose is required and has a max length of 10,000 characters.
- Ingredients are required with a minimum of 1 element including the name and amount.
- Image is optional but should be 4MB at max per image.
- Drug usage is required with a max length of 10,000 characters.
- Video is optional and a URL should be provided only.
- Side effects are required and max length of 10,000 characters.
- “When not to use” field is required and has a max length of 10,000 characters.
- Storage is required and has a max length of 10,000 characters.
- Drug details are required and have a max length of 10,000 characters.

Wireframe:



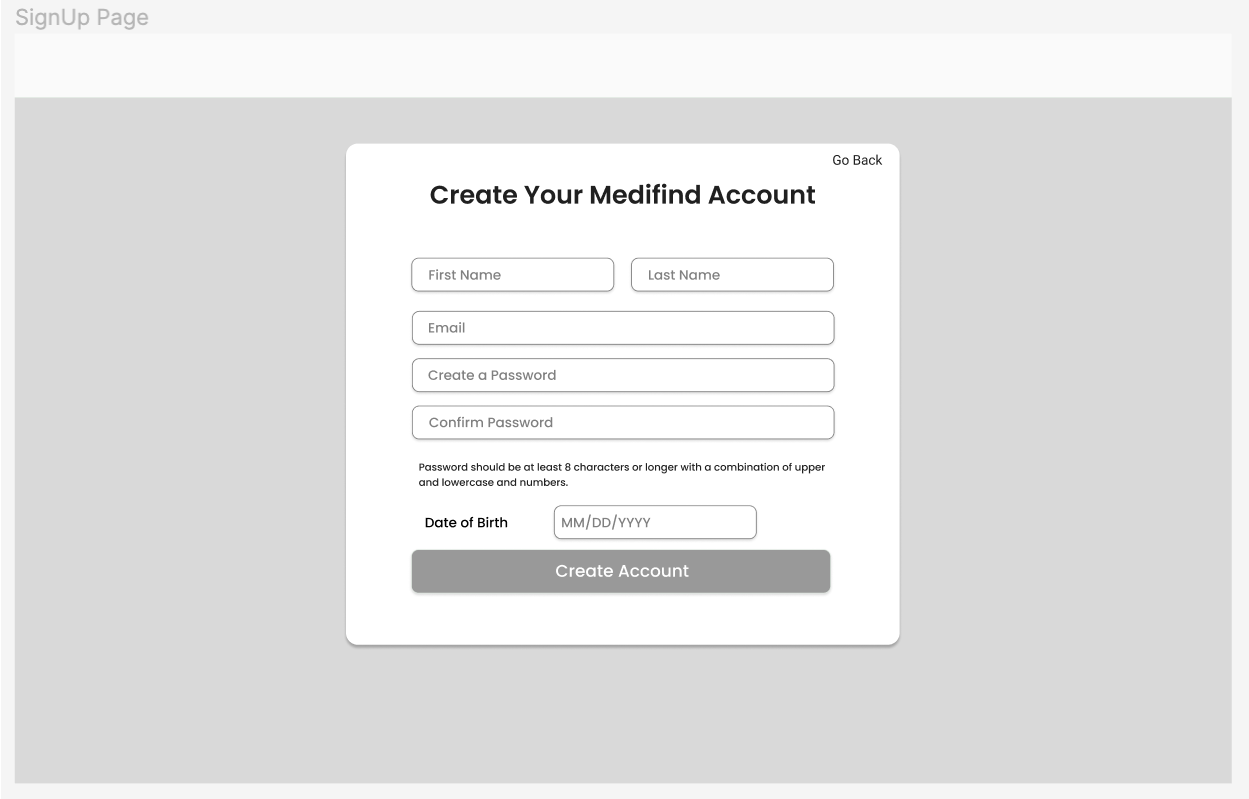
3.1.1.6 Registration Page

Description:
This page allows users to create their own accounts, requiring an email, password, full name, and date of birth. Users will be redirected to interface 3.1.1.2 for the main landing page after registering.

Valid Input:

- Email is required and has a max length of 200 characters.
- Password is required and has a minimum length of 7 characters, minimum of 1 capital letter, minimum of 1 number, and minimum of 1 symbol.
- Full name is required and has a max length of 200 characters.
- Date of birth is required.

Wireframe:



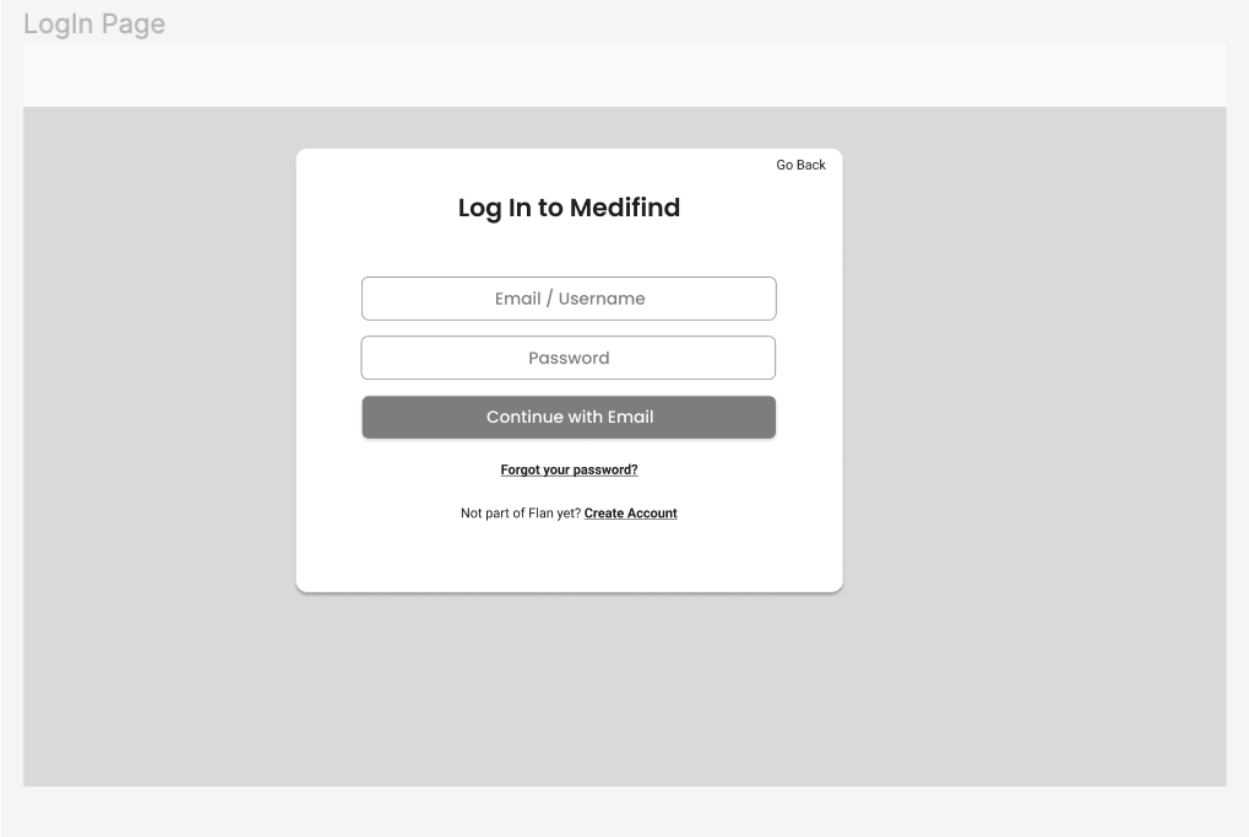
3.1.1.7 Login Page

Description:
This page allows users to login into their existing accounts by inputting their email and password. Users will be redirected to interface 3.1.1.2 for the main landing page after logging-in.

Valid Input:

- Email is required
- Password is required

Wireframe:



3.1.1.8 User Page

Description:
This page shows the users their account information and allows them to edit it. It also shows their bookmarked drugs. Clicking on any of the bookmarked drugs redirects the user to interface 3.1.1.5 for the drug's specific page.

3.1.2 Software Interface

This section describes the backend server endpoints that will be used by the front-end user interface to carry out the requests of the user.

3.1.2.1 Drug Search Request

Command Format:

HTTP GET:
`{url}/api/drugs?name={val1}&manufacturer={val2}&ingredients={val3}&search={val4}`

Valid Input:
- `val1, val2, val3`: required, true or false
- `val4`: required, string, max length of 100 characters

Response/Output:
200 OK response with an array of the results found:
```
[{
    drugId: integer,
    name: string,
    manufacturer: string,
    ingredients: array with elements {name: string, amount: double}
}]
```

3.1.2.2 Specific Drug Get Request

Command Format:
`HTTP GET: {url}/api/drugs/{id}`

Valid Input:
- id: integer

Response/Output:
200 OK response with the following body:
```
{
    drugId: integer
    name: string
    manufacturer: string
    ingredients: array with elements {name: string, amount: double}
    image: string[],
    usage: string,
    video: string[],
    sideEffects: string,
    avoidReasons: string,
    storage: string,
    details: string
}
```

3.1.2.3 Creating Drug Request

Command Format:
HTTP POST: `{url}/api/drugs`
Authentication header: session id

Body:
```
{
    name: …,
    manufacturer: …,
    purpose: …,
    ingredients: [{…}, …, {…}],
    image: string[],
    usage: string,
    video: string[],
    sideEffects: …,
    avoidReasons: …,
    storage: …,
    details: …
}
```

Valid Input:

- name: required, string, max length of 100 characters
- manufacturer: required, string, max length of 100 characters
- purpose: required, string, max length of 10,000 characters
- ingredients: required, minimum size of 1, array with elements {name: string, amount: double}, max length of name is 100 characters
- image: optional, string[], max length of 1,000 characters
- usage: required, string, max length of 10,000 characters
- video: optional, string[], max length of 1,000 characters
- sideEffects: required, string, max length of 10,000 characters
- avoidReasons: required, string, max length of 10,000 characters
- storage: required, string, max length of 10,000 characters
- drugDetails: required, string, max length of 10,000 characters

Response/Output:
200 OK response

3.1.2.4 Editing Drug Request

Command Format:
HTTP POST: `{url}/api/drug/{id}`
Authentication header: session id

Body:
```
{
    name: …,
    manufacturer: …,
    purpose: …,
    ingredients: [{…}, …, {…}],
    image: [...],
    usage: …,
    video: [...],
    sideEffects: …,
    avoidReasons: …,
    storage: …,
    details: …
}
```

Valid Input:
- id: required, integer
- manufacturer: optional, string, max length of 100 characters
- purpose: optional, string, max length of 10,000 characters
- ingredients: optional, array with elements {name: string, amount: double}
- image: optional, string[], max length of 1,000 characters
- usage: optional, string, max length of 10,000 characters
- video: optional, string[], max length of 1,000 characters
- sideEffects: optional, string, max length of 10,000 characters
- avoidReasons: optional, string, max length of 10,000 characters
- storage: optional, string, max length of 10,000 characters
- details: optional, string, max length of 10,000 characters

Response/Output:
200 OK response

3.1.2.4: Create Account Request

Command Format:

HTTP POST: `{url}/api/users/create`
Body:
```
{
    email: …,
    password: …,
    fullName: …,
    dob: …
}
```

Valid Input:
- email: required, string, max length of 200 characters
- password: required, string, minimum length of 7 characters, minimum of 1 capital letter, minimum of 1 number, minimum of 1 symbol
- fullName: required, string, max length of 200 characters
- dob: required, date-time string, ISO 8601 format

Response/Output:
200 OK response with the following body:
```
{
    userId: integer,
    email: string,
    fullName: string,
    dob: string,
    sessionId: integer
}
```

3.1.2.5 Login Request

Command Format:
HTTP POST: `{url}/api/users/login`
Body:
```
{
    email: string,
    password: string
}
```

Response/Output:
200 OK response with the following body:
```
{
    userId: integer,
    email: string,
```

```
    fullName: string,
    dob: string,
    sessionId: integer
}
```

3.1.2.6 Logout Request

Command Format:
HTTP POST: `{url}/api/users/logout`
Authentication header: session id

Response/Output:
200 OK response

3.1.2.7 Validate Session Request

Command Format:
HTTP GET: `{url}/api/users/validate-session`
Authentication header: session id

Response/Output:
200 OK response with the following body:

```
{
    userId: integer,
    email: string,
    fullName: string,
    dob: string,
}
```

3.2 System Functions/Features

3.2.1 Common Error Handling

This subsection describes error handling that is common across many features to avoid repetition. Other error handling requirements that are specific to a single feature will be described in the respective subsection rather than here.

- For all input validation errors at the client side, the user should be notified within the form about each error, so they can correct it. Only when there are no validation errors should the client send a request to the server.
- For all input validation errors at the server side, the user should receive back an error message in the body of the response. The client-side should also inform the user of errors that are received in the body of the response from the server.
- All errors at the server-side along with the request details should be logged for future reference and easier maintainability.
- Validation errors at the server-side should return “400 Bad Request” along with the relevant error message in the body.
- Authentication and authorization errors at the server-side should return “401 Unauthorized” along with the relevant error message in the body.
- Other errors at the server-side (unless specified later in the features) should just return “500 Internal Server Error”.

3.2.2 Drug Input

3.2.2.1 Drug Input Form Web Page

Medifind shall have an input form as a web page for drug information that can be accessed by admin users only. The drug information form should have the same inputs as described in the user interface in 3.1. The input validation on the client side is as described in 3.1 too. Finally, the information will be sent to the server with a POST request. Any error received from the server should be shown to the admin user, following section 3.2.1.

3.2.2.2 Drug Input to the Server

The server must check for authentication and authorization when receiving a POST request for drug information input. The same validity checks as for the client-side must be applied on the server-side as well. The server will then save the data into the database according to the entity relationship diagram (ERD) shown in 3.4. If any validation error occurs, follow section 3.2.1.

3.2.3 Drug Information

3.2.3.1 Drug Information Web Page

Medifind shall have a dedicated web page for every drug in the database as shown in 3.1. The drug's information will be retrieved from the database through a GET request to the server and presented on the page. Users should be able to bookmark the drug with a bookmark button press and share the page's URL with a share button. In addition to the drug's information, admins will be presented with two buttons for editing and deleting. Pressing on edit will direct them to the drug input form with all the fields filled out with the already saved information in the database and allow them to make any changes. Validation and error handling will be the same as that of the drug input form.

3.2.3.2 Drug Search

The client-side shall have a search functionality to search using the categories: drug's name, manufacturer, and ingredients. The user should be able to select which category they want to search with. By default, all categories are checked, displaying results that appear in either of the three categories. The user can then uncheck any of the categories.

3.2.3.3 Drug QR Code Generation

The client-side shall have a QR code generation option for every drug. The QR code is generated at the client-side by anyone and should link to the drug's page URL on the system.

3.2.4 User Accounts

3.2.4.1 Registration

The client shall have a registration page for regular user accounts. The client-side should ask for email, password, full name, and date of birth. Validation of inputs is described in 3.1, and error handling is described in 3.2.1. However, the server must also verify if the email is already used for another account. If so, the server should return "400 Bad Request" and inform the user of the error.

3.2.4.2 Login

The client shall have a login page for all users asking for email and password. The input validation requirements are described in 3.1.. After submission, the server should verify the email and password. If the credentials provided are valid, the server should authenticate the user.

3.2.4.3 Session-based Authentication

Session IDs shall be used for user authentication. A session ID is generated at the server-side, saved to the database, and sent to the client when logging-in. The session ID should expire in 30 days by deleting it from the database. The session ID should be stored at the client as a secure httpOnly cookie. Whenever the user re-opens the website, they should be automatically authenticated by sending the session ID to the server to verify if it is still valid.

3.2.4.4 Logout

The client shall have a logout feature. All information stored on the client-side should be deleted after the user logs out.

3.2.4.5 User Profile Page

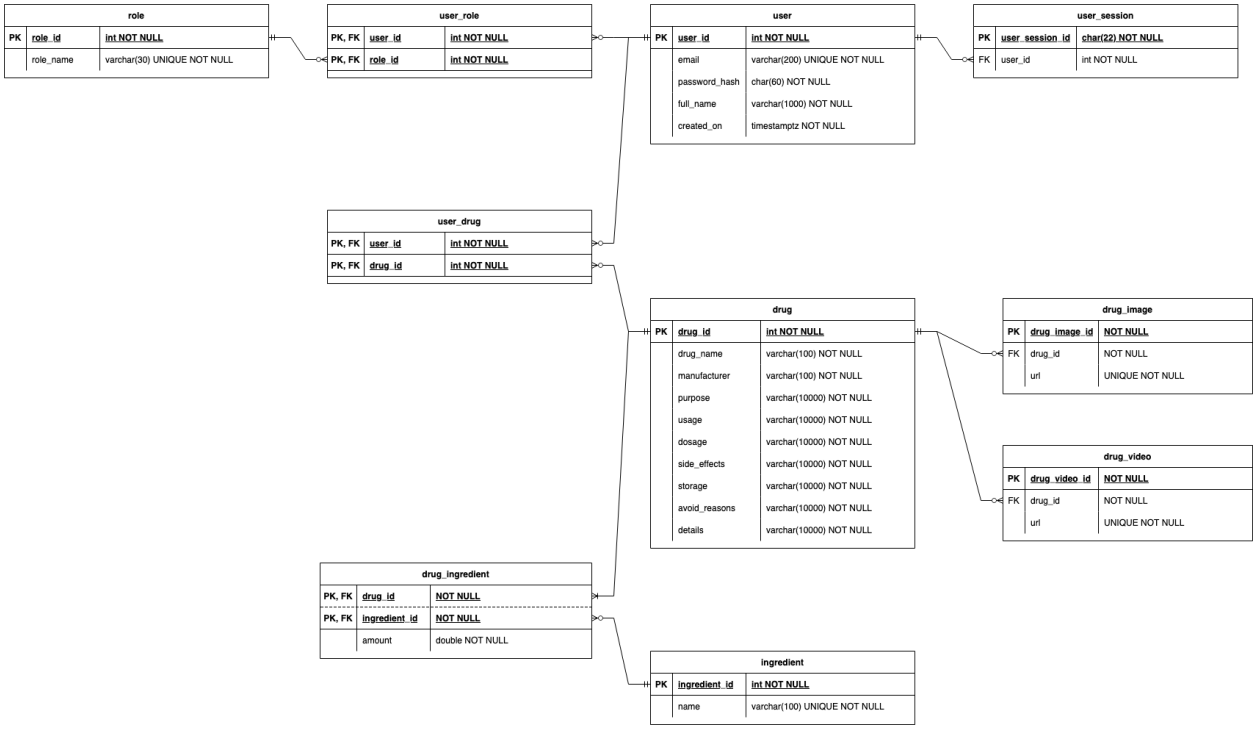
Users shall have a profile page with their account details and bookmarked drugs. The user must be able to edit their accounts details, which includes email, password, full name and date of birth. They should also be able to remove bookmarked drugs.

3.3 Performance Requirements

- The server should be able to handle at least 1000 API calls per second.
- The system should be able to process up to 50MB of images for upload.
- The system should be able to process up to 500MB of video for user viewing.
- The server should be able to handle at least 500 visitors at the same time.

3.4 Logical Database Requirements

The database system used will be PostgreSQL, and Npgsql will be used to interface with the database from the backend server. The entity relation diagram (ERD) for the database is provided below:



3.5 Design Constraints

- The system must comply with the WCAG accessibility guidelines to make sure the system is accessible by everyone, especially since people with vision problems and other disabilities might need to access drug information through the system.
- An audit trail is needed to make sure any changes are traced back. Since the system is provided by the government, users will trust the drug information displayed to them, which means that it must be ensured that any tampering in the system can be reviewed immediately and traced back to the person who did the change.
- The system shall also not use any elements of CSS and JavaScript not supported by the browser versions and types listed in 2.1.4.

3.6 Software System Attributes

3.6.1 Reliability

- The system should not crash in the case of any exception raised by the server.
- The system should not crash when receiving requests with large payloads >50MB.
- The system should not stop working when data backup is performed.
- The system should recover in < 1ms in of a system failure.

3.6.2 Availability

- The system should be available 24/7 with zero downtime as long as the hosting services are fully functional.
- The system should remain available during maintenance through continuous integration and development.
- If absolutely necessary, any maintenance that requires the system to shut down should be done between 12AM and 6AM.

3.6.3 Security

- The system should use session-based authentication.
- Sessions stored on the client-side should be stored as HTTP-only cookies.
- Cryptographically secure random number generators must be used to generate session IDs with 128-bit security.
- Passwords should be hashed and salted using BCrypt before being stored in the database. The input cost should be 11 (2^{11} iterations).
- The system should make sure that all data sent to the database is safe from SQL injections.
- Only allow HTTPS requests.

3.6.4 Maintainability

- The system must log all errors along with the request details that can be reviewed if needed in the future. Logs should be stored for a month, and can be deleted afterwards.
- The system must be maintained with Git and utilize a version control system. This will allow for access to all commits and previous versions.
- Unit tests and integration tests should be developed constantly for all functionalities of the system.
- End-to-end tests that ensure the expected behavior occurs across all parts of the system must be developed as well when all aspects of a specific function are fully developed.
- The database should be backed up bi-weekly, and a yearly backup should be stored for the long-term.

3.6.5 Portability

The system's database and backend must be developed to be hosted on Linux mainly, but it must be able to run on Windows as well. The backend of the system shall be developed with ASP.NET framework in C#. Host-dependent code must only be included when the same functionality requires different code to achieve the same behavior. Otherwise, all code should be host-independent.

4. Change Management Process

- Any further details needed or modifications or updates to the specific requirements must be based on a discussion between developing team leads, project managers, and the client's project collaborators.
- The changes to the requirements may only be made if they are feasible within the specified timeline by a majority vote of the panel as described.
- Alternatively, the team may look into extending the project deadline and altering the costs if the requirement changes requested are critical but unfeasible within the specified deadline. Such changes must be requested formally in writing by the client or the development team.
- The updated requirements must be reflected in an updated version of the SRS with any changes, additions, or deletions being highlighted.

5. Document Approvals

Muhammad Hasin Shabbir - 14 Oct, 2022
Evan - 14 Oct, 2022
Moaaz Assali - 14 Oct, 2022
Bayan Assali - 14 Oct, 2022

6. Supporting Information

Requirement Gathering Process:
The requirements for this project were gathered through the following processes:

- For inception, a basic identification of the problem was noted alongside information about who the solution would be for and what would be the form of the provided solution i.e. the Medifind system.
- The goals of the system were identified and brought forward alongside an understanding of targets from a business perspective.
- For further elaboration, understanding, and refining of system requirements, the following were conducted:
    - Interview of pharmacy employees to understand the perspective of one kind of stakeholder/potential user around the challenges they face and how they feel about the proposed solutions.
    - A survey questionnaire for end-users who would be using the Medifind system to understand the problems they usually face and what would be their preference for the proposed system. To get a trend of what problems users encounter in planning their medications and our system attempting to solve those.

- A team meeting to discuss the requirements from a product and a development perspective.
- Analysis of design challenges in putting medicine information on medicine packages. The main problems identified were the limited space on packaging and the very distributed online systems.
- An observation of similar tools like online medicine wikis (WebMD etc.) was conducted from a technical perspective to understand what might be required for our system's functionality.
- A study of different tech stacks and modern tools, libraries, and technologies used to develop most modern web based applications. This was then discussed within the context of the team's expertise to decide on the final tech stack.

Appendix

Item A - Pharmacy employee interview questions:

1. Do you get calls or inquiries regarding use of drugs or medication?
    1. How often does this happen?
    2. How do you resolve such issues?
2. Have you had incidents of users having Medication Error?
    1. How often does this happen?
    2. How do you resolve such issues?
3. Do you see value in a system that allows information retrieval for medicines through a QR code on the pack?
4. Do you see value in a system that allows information retrieval for prescriptions through a QR code on the pack?

Note: Responses to interview questions may be shared upon request.

Item B - End-user survey questions:

1. Age
2. Occupation
3. What kind of Drug Information is most useful to you for prescription drugs?
4. What kind of Drug Information is most useful to you for Over-the-Counter drugs(non-presrciption)?
5. How many Drugs do you currently own?
6. Do you have experience with Medication Error? (Medication Error includes misuse of medication, overdosing, etc - basically any misuse of medicine with lack of understanding or misunderstanding of the drug). If so, please provide a short overview.
7. Do you look at the medicine packaging for relevant information?
8. Do you search online for information regarding a particular medicine?
9. How do you track your prescription medicines and relevant information?
10. Are there any inconveniences or struggles that you face with finding information about medicine or prescriptions through the methods listed above in the questionnaire?
11. On a scale of 1 to 10 with 1 being the lowest preference and 10 being the highest preference, how much would you prefer having an application that provides you with the information about a medicine through a QR code on the packaging instead of your current method?
12. On a scale of 1 to 10 with 1 being the lowest preference and 10 being the highest preference, how much would you prefer having an application that provides you with the information to track your prescription through a QR code on the packaging instead of your current method?

Note: Responses to survey questions may be shared upon request.