CSC131 - Section #4
Spring 2024
Deliverable #2
Team Sloth Thinkers

Gavin Kabel
Jeffrey Sardella
Bhabesh Phuyal
Harpreet Kaur
Mustafa Ahmady
Rajan Bains

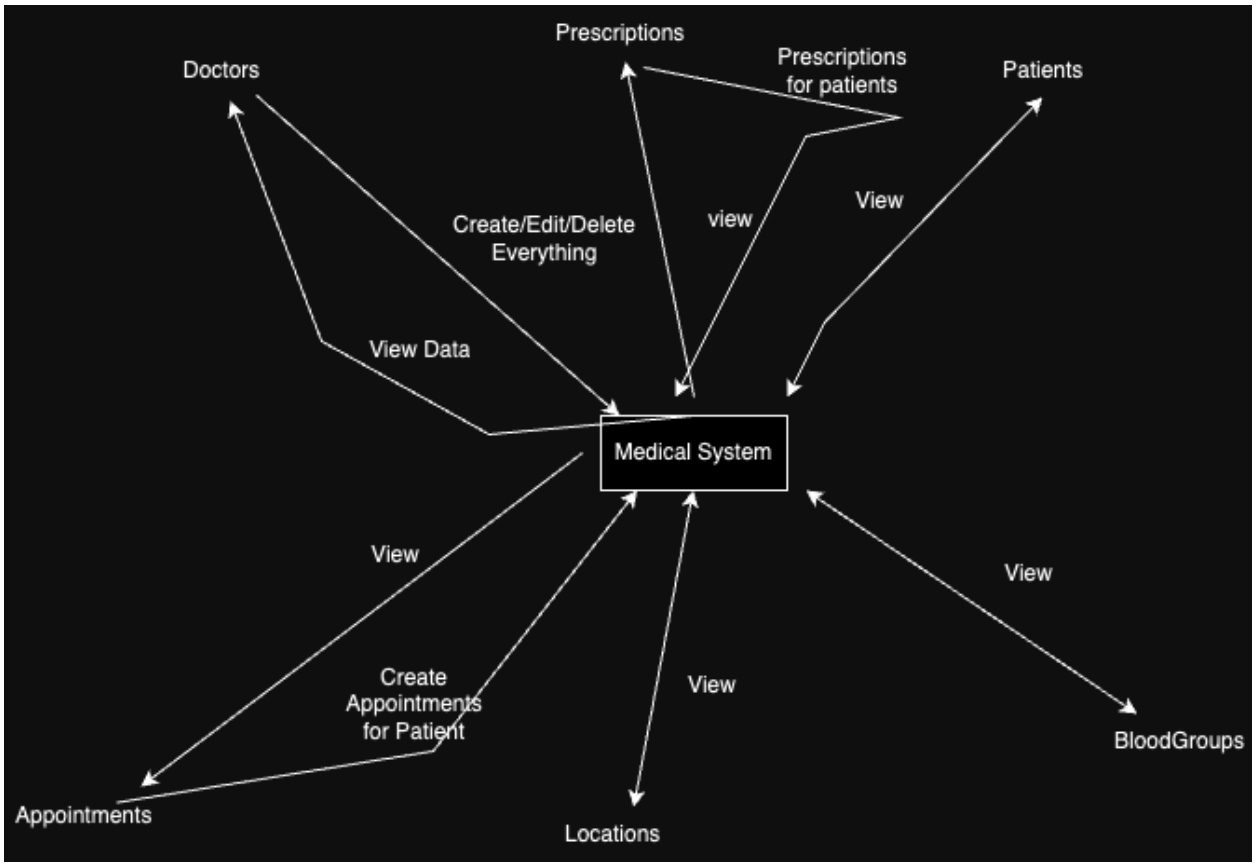
Table of Contents

History revision table

Date	Commit Message	Committed By
Apr 25, 2024	Location Directions	gbkabel
Apr 24, 2024	fixed major bug in messaging and when creating convo redirects u to messages	poopoop60
Apr 24, 2024	swappped messages pos	poopoop60
Apr 24, 2024	messaging works	poopoop60
Apr 24, 2024	added conversation and messages backend	poopoop60
Apr 23, 2024	rh factor dropdown now remove some nav stuff	poopoop60
Apr 12, 2024	fixed registration screen labels	poopoop60
Apr 9, 2024	Added more search stuff to prescriptions	poopoop60
Apr 8, 2024	Back buttons now work properly and fixed bug when deleting appointments	poopoop60
Apr 8, 2024	Added search to everything	poopoop60
Apr 8, 2024	Fixed photo bug, added rest of appointment stuff	poopoop60
Apr 7, 2024	Finished patients(details and update)	poopoop60
Apr 3, 2024	Lot of style changes. Added patient details and kinda edit	poopoop60
Apr 3, 2024	Profile/Settings Pages	gbkabel
Apr 2, 2024	Added stuff to worker details, and fixed details tables	poopoop60
Apr 2, 2024	Can now create appointments	poopoop60
Apr 2, 2024	Fixed deleting users remove update data in edit, added appoint results	poopoop60
Apr 1, 2024	Added my medication,appointment also changed style	poopoop60
Apr 1, 2024	Dashboard now updates data	poopoop60
Mar 31, 2024	Added prescription details and style cleanup	poopoop60
Mar 31, 2024	Added option to update work data everywhere add patient remove and update	poopoop60
Mar 30, 2024	Added some dal stuff for bulk updating when a doctor is deleted	poopoop60
Mar 30, 2024	Add to home may work. Added app icons	poopoop60
Mar 29, 2024	Prescriptions work now patients work now with storing photos in db	poopoop60
Mar 28, 2024	Merge branch 'main' of https://github.com/gbkabel/MedicalApp	poopoop60
Mar 28, 2024	Added deleting, details, edit user accounts	poopoop60
Mar 27, 2024	Delete WebApplication1 directory	poopoop60
Mar 27, 2024	Added first and last name to register and user account	poopoop60
Mar 27, 2024	You now need a role and birthdate to make account and other stuff	poopoop60
Mar 27, 2024	Finished folder structure	poopoop60
Mar 27, 2024	Setting up folder structure locations	poopoop60
Mar 26, 2024	Add prescription	poopoop60
Mar 26, 2024	Clickable patient name	gbkabel
Mar 26, 2024	Hides sidebar based on width now	poopoop60
Mar 25, 2024	More app name fixes	gbkabel
Mar 25, 2024	AddToHomeScreen Names	gbkabel
Mar 25, 2024	Update dal and added blood group	poopoop60
Mar 24, 2024	CreatePrescription front end	gbkabel
Mar 24, 2024	AddPatient page front end work	gbkabel
Mar 20, 2024	Built out patient database	poopoop60
Mar 19, 2024	Added alerts to update/edit location and error code page works now	poopoop60
Mar 19, 2024	You can now delete and edit locations	poopoop60
Mar 18, 2024	Remove location	poopoop60
Mar 15, 2024	Push	poopoop60
Mar 15, 2024	Merge branch 'main' of https://github.com/gbkabel/MedicalApp	MustafaAhmady
Mar 15, 2024	Can now add locations working on deleting them	poopoop60
Mar 14, 2024	Error code page	poopoop60
Mar 13, 2024	Added more pages and added locations	poopoop60
Mar 13, 2024	Added razor pages for all task bar items	poopoop60
Last Month	Know using layout for main hub	poopoop60
Last Month	Added tables to all appointment	poopoop60
Mar 12, 2024	Added models	poopoop60
Mar 11, 2024	Removed controllers they are dumb	poopoop60
Mar 11, 2024	Fixed logging out, added DAL, added home controller	poopoop60
Mar 11, 2024	Backend for register and forgot password	poopoop60
Mar 11, 2024	Proper URL redirecting	gbkabel
Mar 11, 2024	Forgot password page	gbkabel
Mar 11, 2024	Register user display fix	gbkabel
Mar 11, 2024	Register user page	gbkabel
Mar 11, 2024	Added calendar and added DB for appointments, location, patient	poopoop60

Mar 10, 2024	**Calendar prep**	**poopoop60**
Mar 8, 2024	**Front in edits**	**poopoop60**
Mar 8, 2024	**Create .project**	**MustafaAhmady**
Mar 8, 2024	**Login loop**	**poopoop60**
Mar 8, 2024	**Cleanup**	**poopoop60**
Mar 8, 2024	**Added login screen to template**	**poopoop60**
Mar 7, 2024	**Merge pull request #2 from gbkabel/sbMerge**	**gbkabel**
Mar 5, 2024	**sbAdded**	**gbkabel**
Mar 5, 2024	**sbApplied**	**gbkabel**
Mar 4, 2024	**Added userdataexpansion**	**poopoop60**
Mar 1, 2024	**Plz work**	**gbkabel**
Mar 1, 2024	**New key**	**gbkabel**
Mar 1, 2024	**Merge pull request #1 from gbkabel/asp.net**	**gbkabel**
Mar 1, 2024	**Random commit**	**poopoop60**
Mar 1, 2024	**Setup email confirmation for accounts and int commit**	**poopoop60**
Feb 27, 2024	**Update README.md**	**poopoop60**
Feb 26, 2024	**Initial commit**	**gbkabel**

System Context Diagram



These are how the systems interact with each. A doctor has all the power to create/edit/delete data at will for everything. Everything acts as a view for data. Besides prescriptions and appointments which makes data for patients.

Object Oriented Requirements Analysis (OOA) – UML Modeling

Use Case Model

Description: Create / Draw a "Use Case" model that captures the essence of the project
Describe the system Use Cases- include typical and alternate flows for some of the Use Cases.

1. Create Personal Health Profile
Actors: User
Description: Allows users to create their personal health profile by giving relevant information about medical history, allergies, etc.
- Typical Flow:
  1. User logs into the system.
  2. User navigates to the profile section.
  3. User fills out necessary information and saves the profile.
- Alternate Flow:
  - If the user already has a profile, they can choose to edit it rather than creating a new one.

2. Login
- Actors: User
- Description: Allows users to securely log into the system using their information..
- Typical Flow:
  1. User enters their username and password.
  2. System verifies credentials and grants access.
- Alternate Flow:
  - If the credentials are incorrect, the system prompts the user to re-enter.

3. Schedule Appointment
- Actors: User, Healthcare Provider
- Description: Allows users to schedule appointments with healthcare providers.
- Typical Flow:
  1. User selects a healthcare provider.
  2. User chooses an available time slot.
  3. System confirms the appointment and sends a notification to the user and provider.
- Alternate Flow:
  - If the desired time slot is not available, the user can choose an alternative time or provider.

4. View Medical History
- Actors: User
  - Description: Allows users to view their medical history, including past diagnoses, treatments, and medications.
- Typical Flow:
  1. User navigates to the medical history section.
  2. System retrieves and displays the relevant medical information.
- Alternate Flow:
  - If there is no medical history available, the system notifies the user.

5. Search Healthcare Providers
- Actors: User
  - Description: Allows users to search for healthcare providers based on specialty, location, and availability.
- Typical Flow:

1. User enters search criteria (specialty, location, etc.).
2. System retrieves and displays a list of matching healthcare providers.

- Alternate Flow:
  - If there are no matching providers, the system notifies the user and suggests broadening the search criteria.

6. Communicate with Healthcare Provider

- Actors: User, Healthcare Provider
- Description: Allows users to communicate with healthcare providers through a secure messaging system.
- Typical Flow:
  1. User selects the messaging option.
  2. User composes a message and sends it to the provider.
  3. Provider receives the message and responds accordingly.
- Alternate Flow:
  - If the provider is unavailable, the system notifies the user and suggests alternative communication methods.

System Requirements

5.1 System Functional Requirement

FR1: The system shall allow users to create and manage their personal health profile.
FR2: The system shall provide secure login functionality for users.
FR3: The system shall allow users to schedule, modify, and cancel appointments with healthcare providers.
FR4: The system shall provide users with reminders for their scheduled appointments.
FR5: The system shall allow users to view their medical history including past diagnoses, treatments, and medications.
FR6: The system shall allow users to search for healthcare providers based on specialty, location, and availability.
FR7: The system shall allow users to communicate with healthcare providers through a secure messaging system.
FR8: The system shall provide users with access to educational resources related to their health conditions.
FR9: The system shall allow users to track their health metrics (like blood pressure, weight, etc.) over time.
FR10: The system shall ensure all user data is stored and transmitted securely to protect patient privacy.

5.2 Non-Functional Requirements (NFRs)

NFR1: **Usability**: The system shall be user-friendly, with a clear and intuitive interface that can be easily navigated by users with varying levels of technical proficiency.

NFR2: **Reliability**: The system shall have an uptime of 99.9%, ensuring that users can access their health information and communicate with healthcare providers whenever needed.

NFR3: **Performance**: The system shall be able to support up to 100 simultaneous users without significant degradation in response time.

NFR4: **Security**: The system shall comply with all relevant health data security regulations, including HIPAA in the U.S., to ensure the privacy and security of user data.

NFR5: **Scalability**: The system shall be able to handle an increasing number of users and data volume without a significant impact on system performance.

NFR6: **Interoperability**: The system shall be able to interface with other healthcare systems (like EHRs, pharmacy systems, etc.) to exchange relevant patient data.

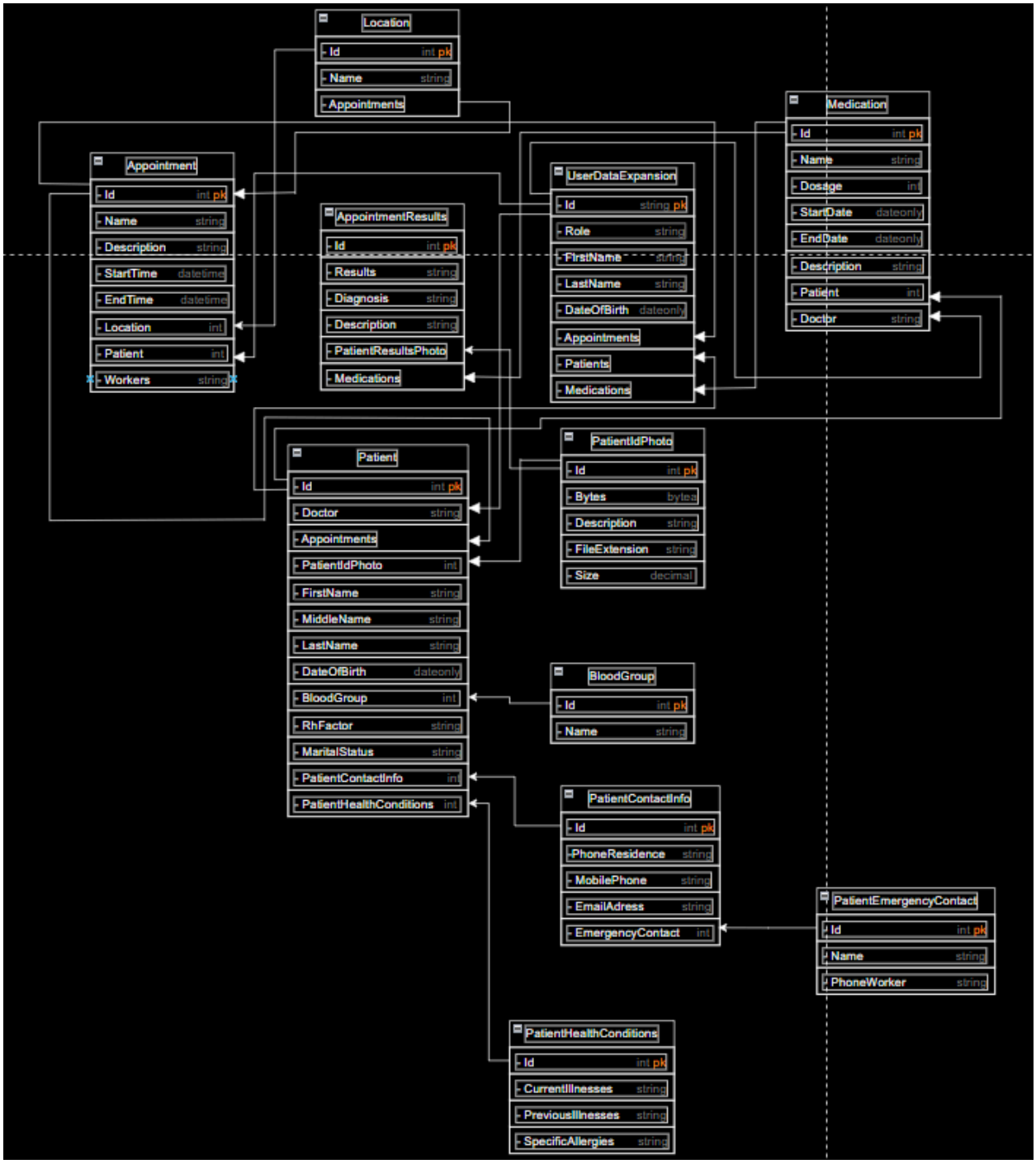
NFR7: **Maintainability**: The system shall be designed in a modular manner allowing for easy updates and maintenance.

NFR8: **Accessibility**: The system shall comply with WCAG 2.1 Level AA accessibility guidelines to ensure that it is accessible to users with disabilities.

NFR9: **Data Integrity**: The system shall ensure the accuracy and consistency of health data over its entire lifecycle.

NFR10: **Disaster Recovery**: The system shall have a robust backup and recovery mechanism to protect against data loss in the event of a disaster.

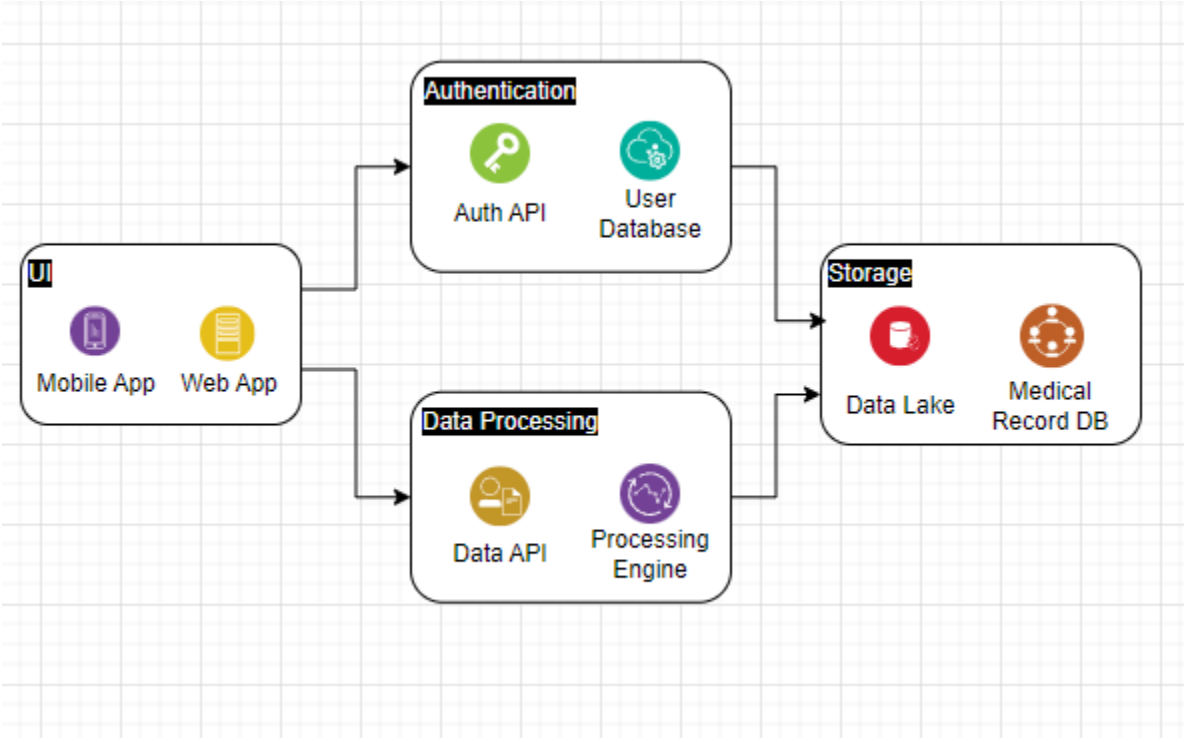
Data Design - Entity Relationship Diagram(ERD)



- The Data Design describes structures that reside within the software. Attributes and relationships between data objects dictate the choice of data structures.

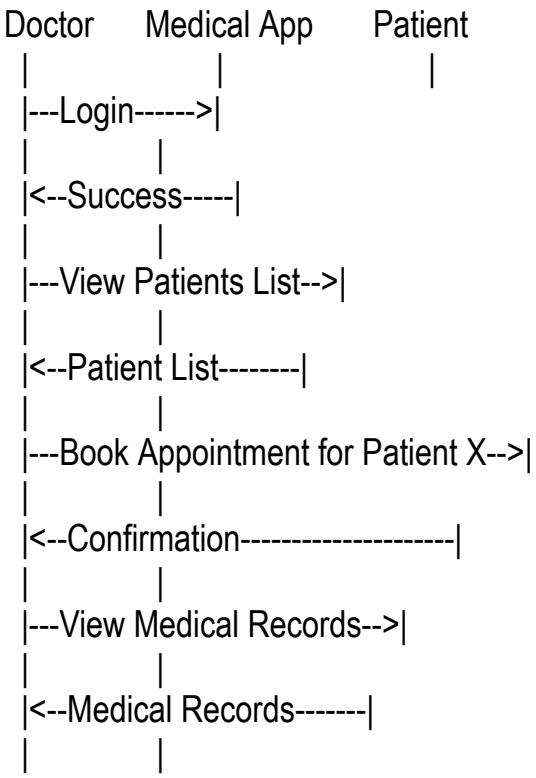
Architectural Design

- Provide a high-level overview of system, its major components and how these components are connected (interactions how u use website)



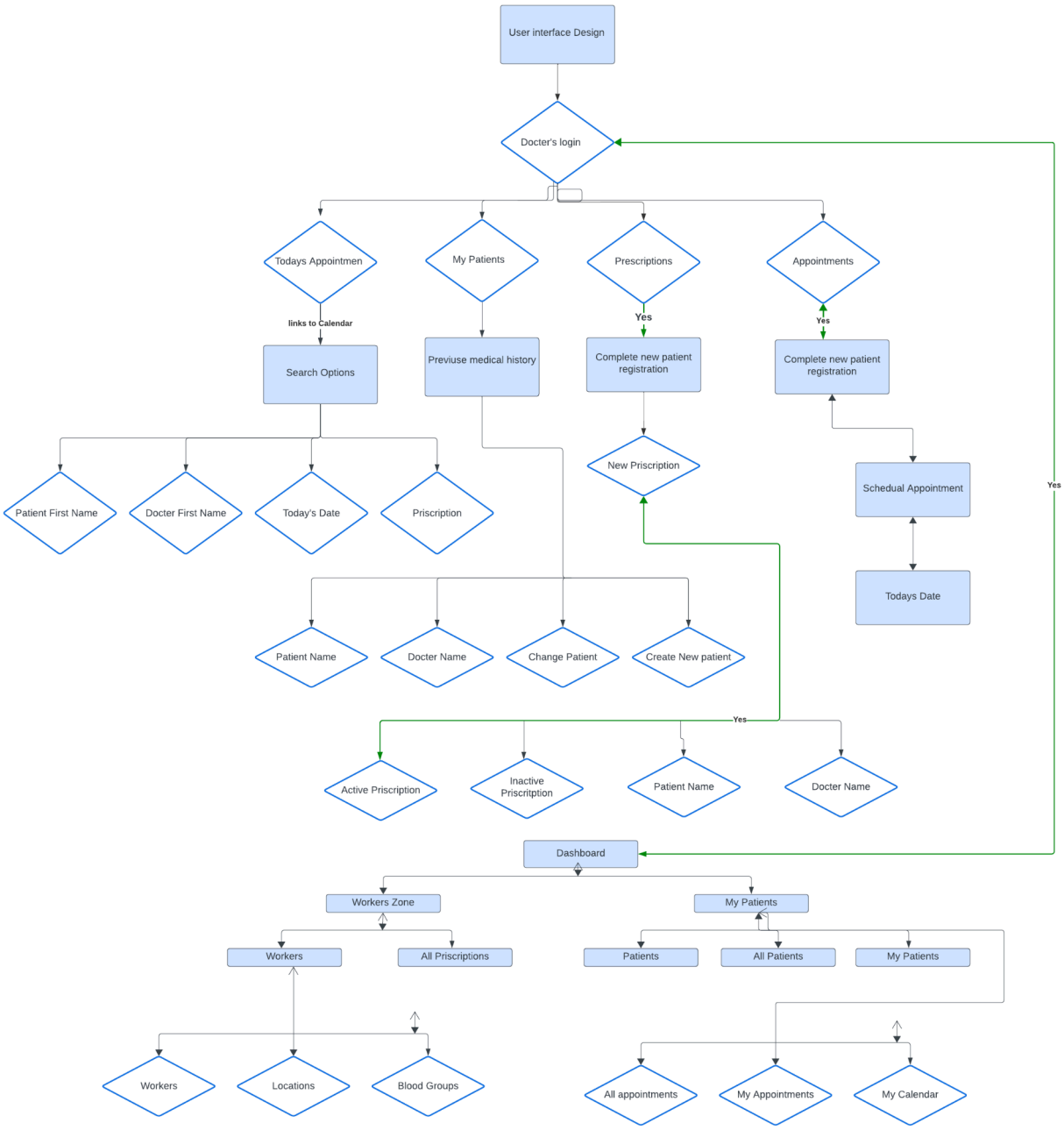
Detailed Design

- Create a set of interaction models (i.e. sequence diagrams) to capture a low level design view of the system.



The Doctor logs into the Medical App.
The Medical App confirms successful login.
The Doctor requests to view the list of Patients.
The Medical App provides the list of Patients.
The Doctor books an appointment with a specific Patient (Patient X in this case).
The Medical App confirms the appointment.
The Doctor requests to view their Medical Records.
The Medical App provides the Patient's Medical Records.

User Interface Design



- The Interface Design describes internal and external program interfaces. Interface designs are based on the information obtained from the analysis models. Use Cases, User Stories, and Sequence diagrams to capture the interface design.

Technology and Tools

- **Front end: Bootstrap: html, css. javascript**
- **Software: Visual Studio**
- **Virtual Phone Software: Android Studio**
- **Web Testing Software: Google Chrome**
- **Backend: Asp.net: C#**
- **Database: SQL**
- **Version control: Github**

Assumption and Constraints

Assumptions:

User Health Data: It's assumed that the app will have access to accurate health data from the users. This could be manually entered by the user or collected through connected devices.

Healthcare Standards: The app is expected to follow healthcare standards like HL7, FHIR, etc., for data exchange, if applicable.

User Consent: It's assumed that users have given their consent to collect, store, and process their personal and health data.

Stable Software Environment: It's assumed that the software environment (like the operating system, database, etc.) will remain stable throughout the development process.

Availability of Resources: It's assumed that all necessary resources (like development tools, testing tools, skilled personnel, etc.) are available and accessible throughout the development process.

Constraints:

Data Privacy and Security: Medical apps deal with sensitive health data, so they must comply with data privacy regulations like HIPAA (in the U.S.) or GDPR (in the EU). This can significantly impact the design and implementation of the app.

Interoperability: The app might need to integrate with other healthcare systems (like EHR systems), which can be a constraint.

Medical Accuracy: The information provided by the app needs to be medically accurate and reliable. This can be a constraint in terms of the resources needed to ensure the accuracy of the information.

User Interface: The user interface should be easy to use for individuals with varying levels of health literacy and digital literacy.

Time: The time frame within which the software needs to be developed and deployed can be a significant constraint. It can impact the scope of the project and the quality of the final product.

Technology: The choice of technology can also be a constraint. For example, if the software needs to be developed in a specific programming language or needs to be compatible with certain hardware or software.