VIETNAM NATIONAL UNIVERSITY HO CHI MINH CITY
UNIVERSITY OF TECHNOLOGY FACULTY OF COMPUTER SCIENCE AND ENGINEERING

HOSPITAL MANAGEMENT SYSTEM

Student: Nguyễn Thanh Tùng ID: 2152337

Feb - March 2024

TABLE OF CONTENT

1. REQUIREMENT ELICITATION

1.1. Project description

A hospital X needs to build a management information system to manage the information of their patients, doctors, and nurses.

The database of hospital X needs to store the information of employees (doctors and nurses) including: a unique code, full name consisting of first name and last name, date of birth, gender, address, start date (first day of work), phone number(s), and speciality with its related name and degree's year. The hospital has many departments. Each department has a unique code, a title, and a dean who is a doctor.

The employees have to belong to a specific department. A department has at least one or many employees. The dean must hold a specific speciality and has had more than 5 years of experience since the date he or she was awarded the speciality degree.

The patients have to provide with the hospital their information such as: full name (first name and last name), date of birth, gender, address, and phone number. After receiving their information, the system will store them into the database, and generate a unique code to identify each patient simultaneously. Patients are divided into two types: outpatients and inpatients.

- For outpatients, the information of the examining doctor needs to be stored. The outpatients can have many examinations with their examining doctor. The hospital needs to store the details of each examination such as: examination date, diagnosis, the next examination date if any, medications, and fee.

- For inpatients, some information is added such as: date of admission, treating doctors, caring nurse, diagnosis, sickroom, date of discharge,

and fee. After admission to the hospital, a patient can receive treatment from at least one doctor.

A doctor can treat many patients at the same time, or sometimes, he has no patients to treat. The hospital needs the details of each treatment such as: treatment period (start date and end date), result, and medications. Each inpatient is taken care of by a nurse; a nurse can take care of many inpatients at the same time. Furthermore, when a patient is recovered and his or her last treatment has been confirmed as "recovered" by the doctor, he or she will be discharged from the hospital. As a result, the discharge date must be recorded by the system. The information of a medication is also stored in the database. This information consists of a unique code, name of the medication, effects, price, and expiration date. A medication is provided by one or more providers, and one provider may provide many types of medication. A provider is tracked by its unique number, name, address, and phone. Moreover, the hospital also wants to keep the imported medication information including imported date, price, and quantity. In case one medication is out-of-date, it will be automatically marked so in the database.

1.2. Describe all functional and non-functional requirements of the project.

1.2.1. Functional Requirements

- **Administrators**
  - Administrators shall be able to access, review and remove information of the hospital via corresponding buttons.
  - Administrators shall be able to add medication and provider of that medication to the database
- **Doctors**
  - Doctors shall be able to report patient treatment / examination to the database.
  - Doctors shall be able to view the department they belong to.
- **Nurses**
  - Nurses shall be able to view patients that are assigned to them.
  - Nurses shall be able to view the department they belong to.
- **Receptionists**
  - Receptionists shall be able to add patient information to the database and assign patients to doctors and nurses.
- **All stakeholders**
  - Stakeholders shall be able to login into their account after entering the correct code and password.
  - Stakeholders shall be able to filter results via search bar.
  - Stakeholders shall be able to navigate between pages via the sidebar.

1.2.2. Non-Functional Requirements

- **Performance:**
    - The system should respond to user interactions within 2 seconds, even under peak load conditions.
    - Database queries should be optimized for efficient retrieval and manipulation of data.
- **Scalability:**
    - The system should be scalable to accommodate an increasing number of patients, doctors, nurses, and other entities over time.
- **Security:**
    - Access to different functions of the database like add, delete is restricted based on user roles and permissions.
    - Passwords should be encrypted to protect it from unauthorized access or tampering.

1.3. Draw use case diagram for the system and main module

1.3.1. System use case diagram

1.3.2. Component Module

1.3.2.1. Use case diagram

1.3.2.2. Table description

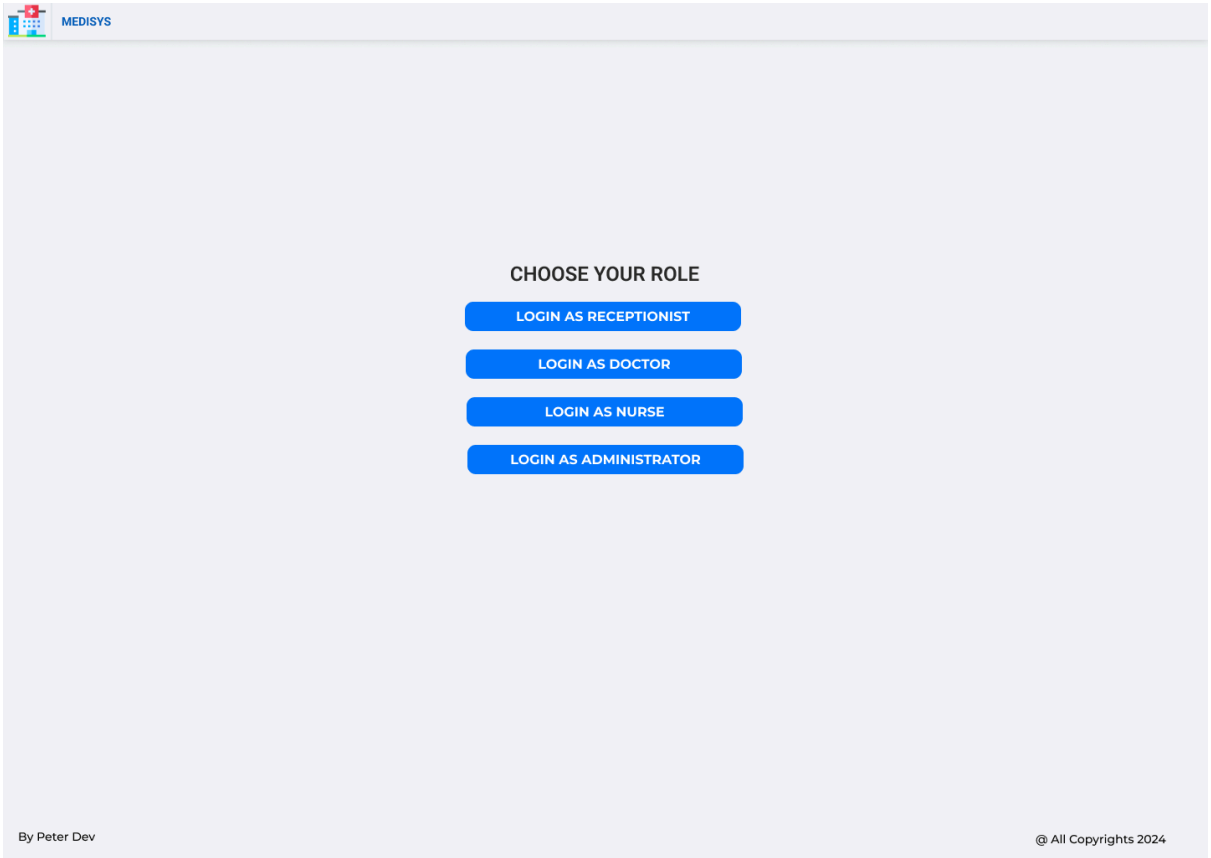
2. SYSTEM MODELLING

2.1. Draw activity diagrams

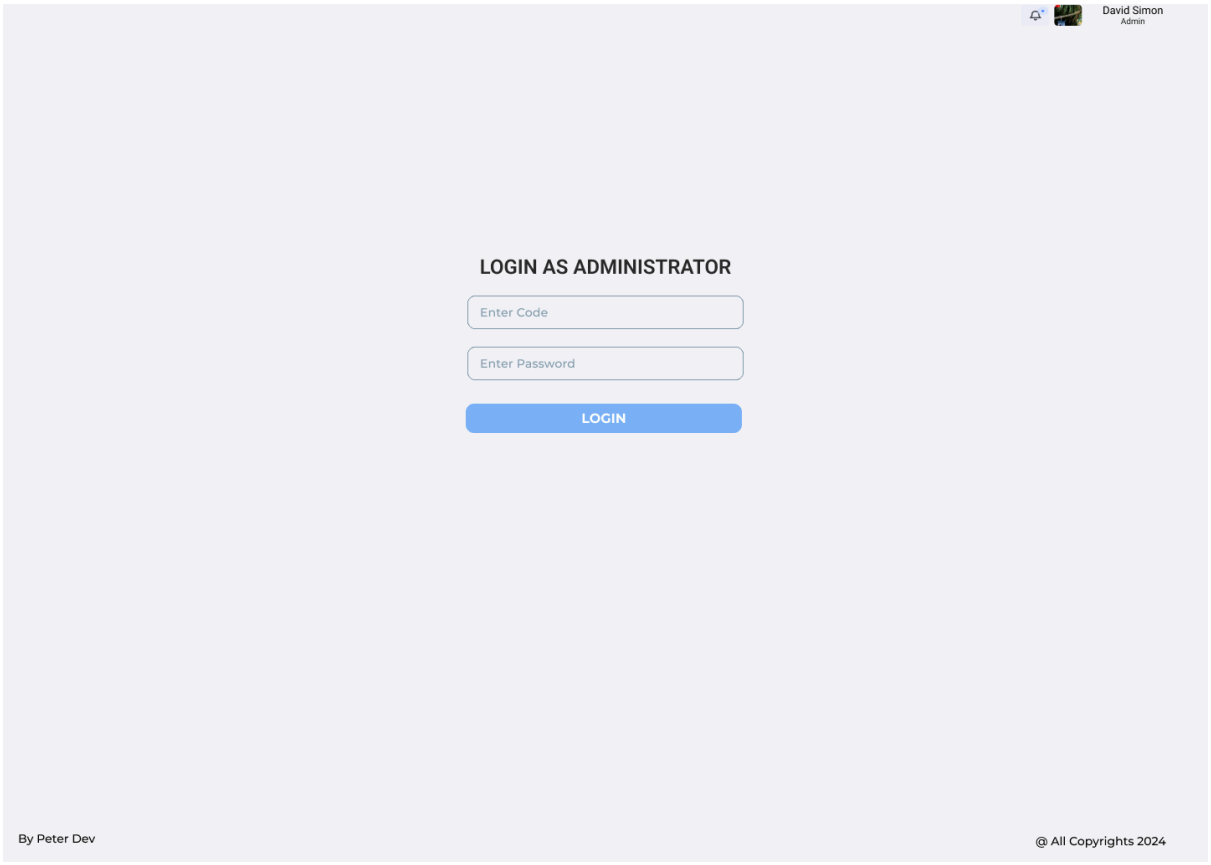
2.2. Draw User Interface

User Interface

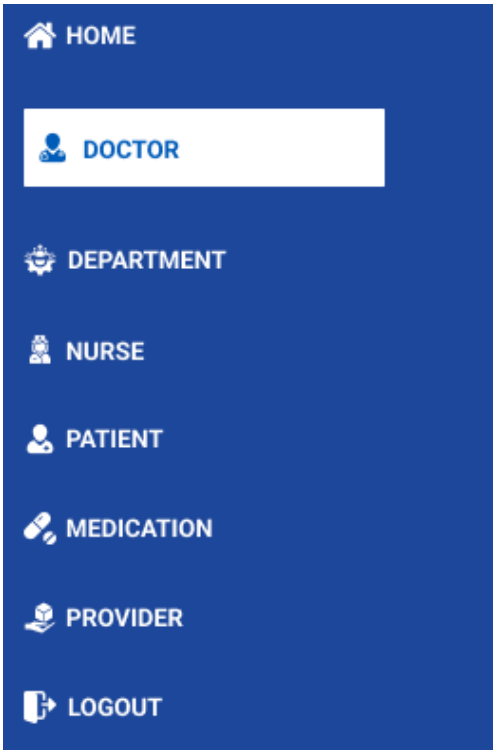
Login Page: User chooses the role to login.



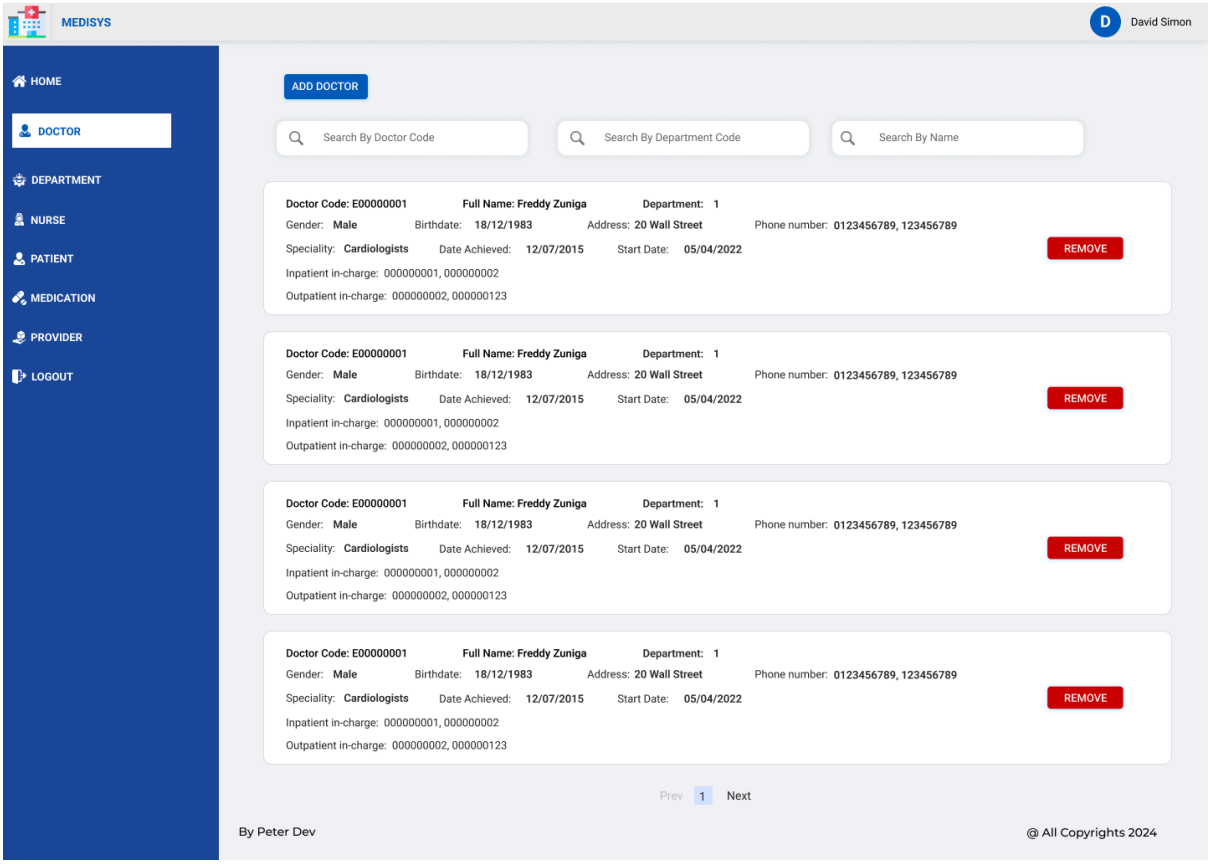
Authentication: Enter user code and password



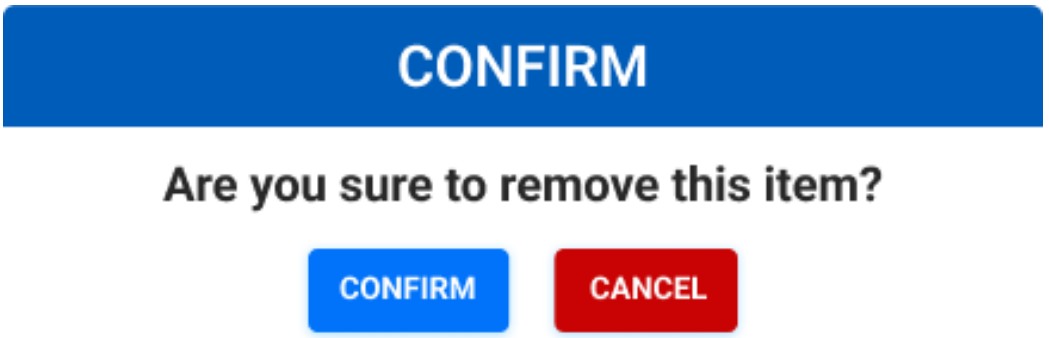
Navbar: displayed on the left side, is included in all pages to navigate between different pages by clicking on corresponding buttons.



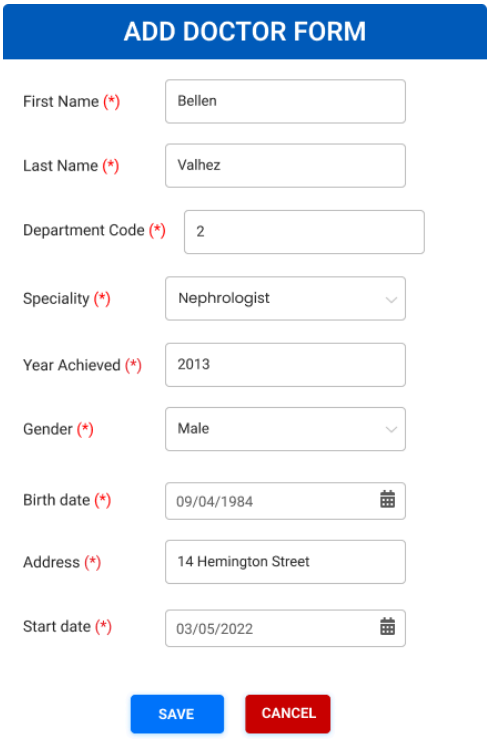
Sample page: Below is Doctor Page for admin, contains a button to Add doctor, a button to Remove doctor, and some search bar for searching. Pagination is also supported for better view.



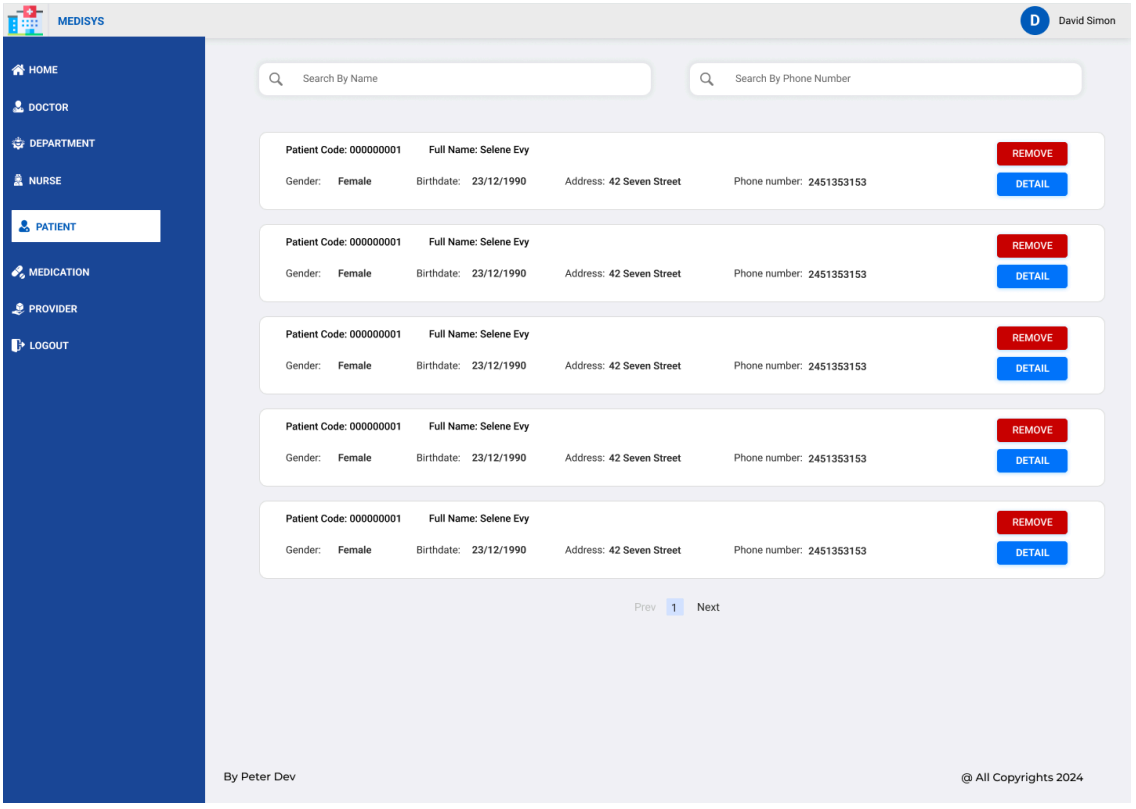
Confirm Modal: displayed when clicking the "Remove" button for confirmation.

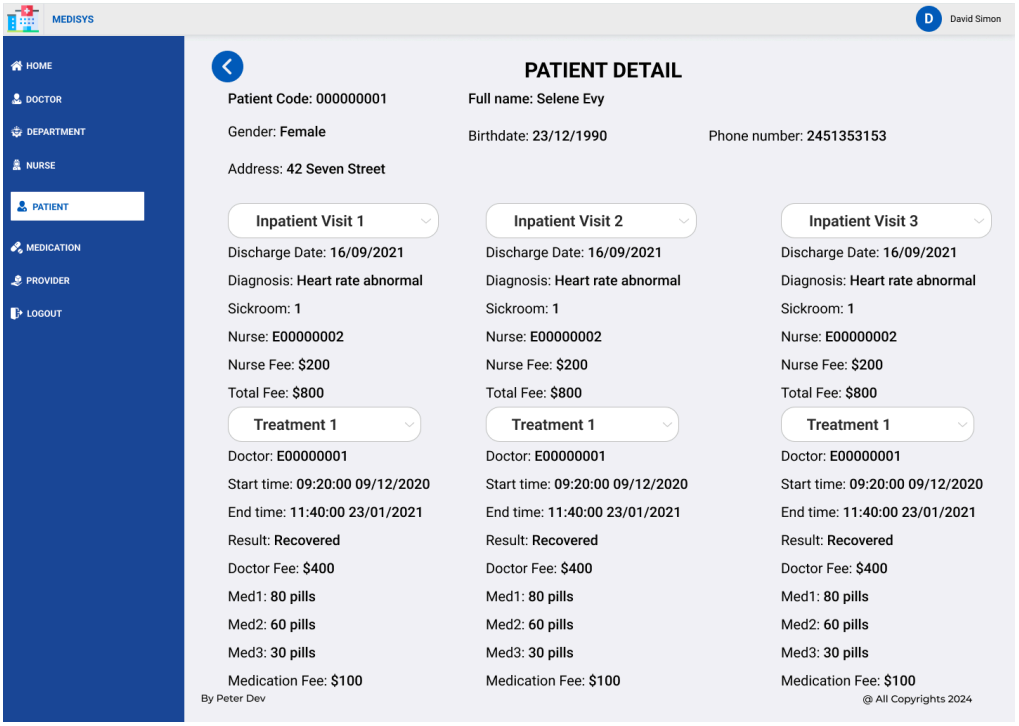


Add Form: displayed when clicking the "Add" button for filling data.

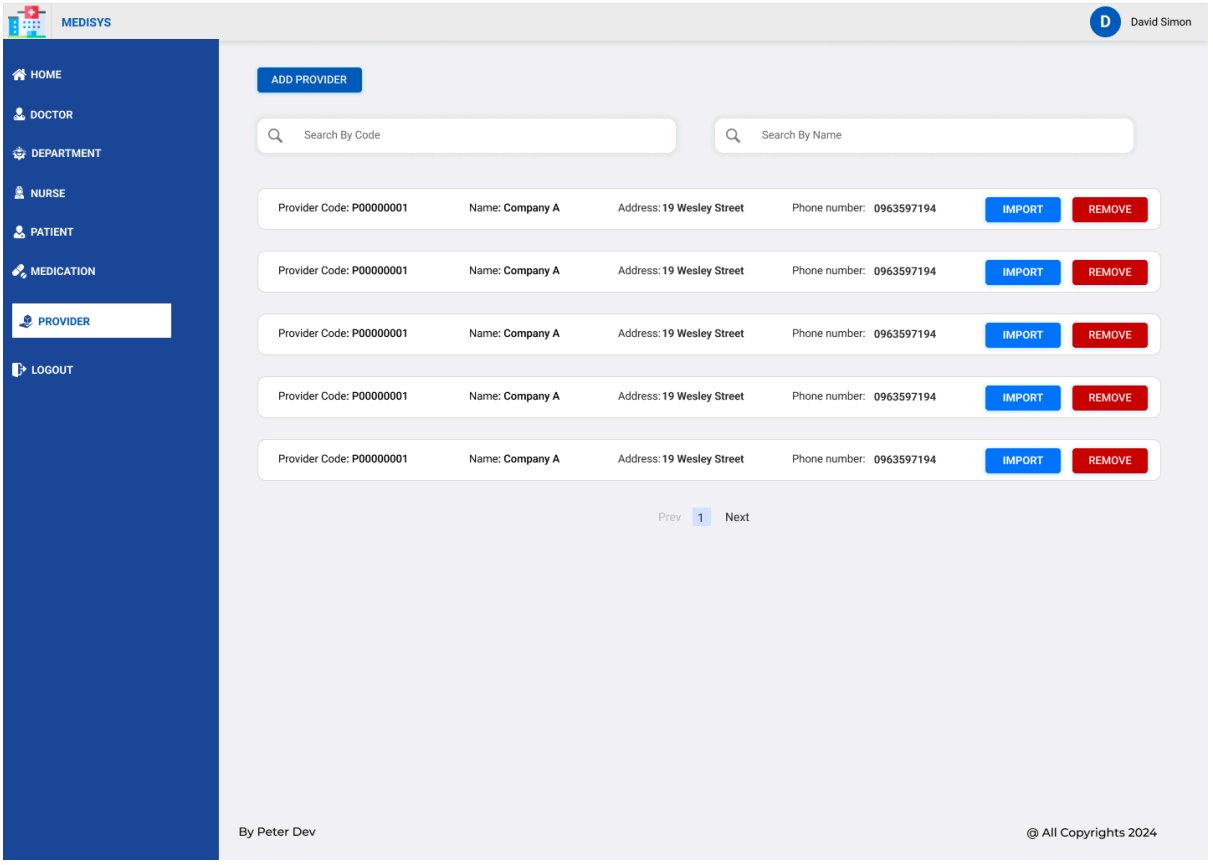


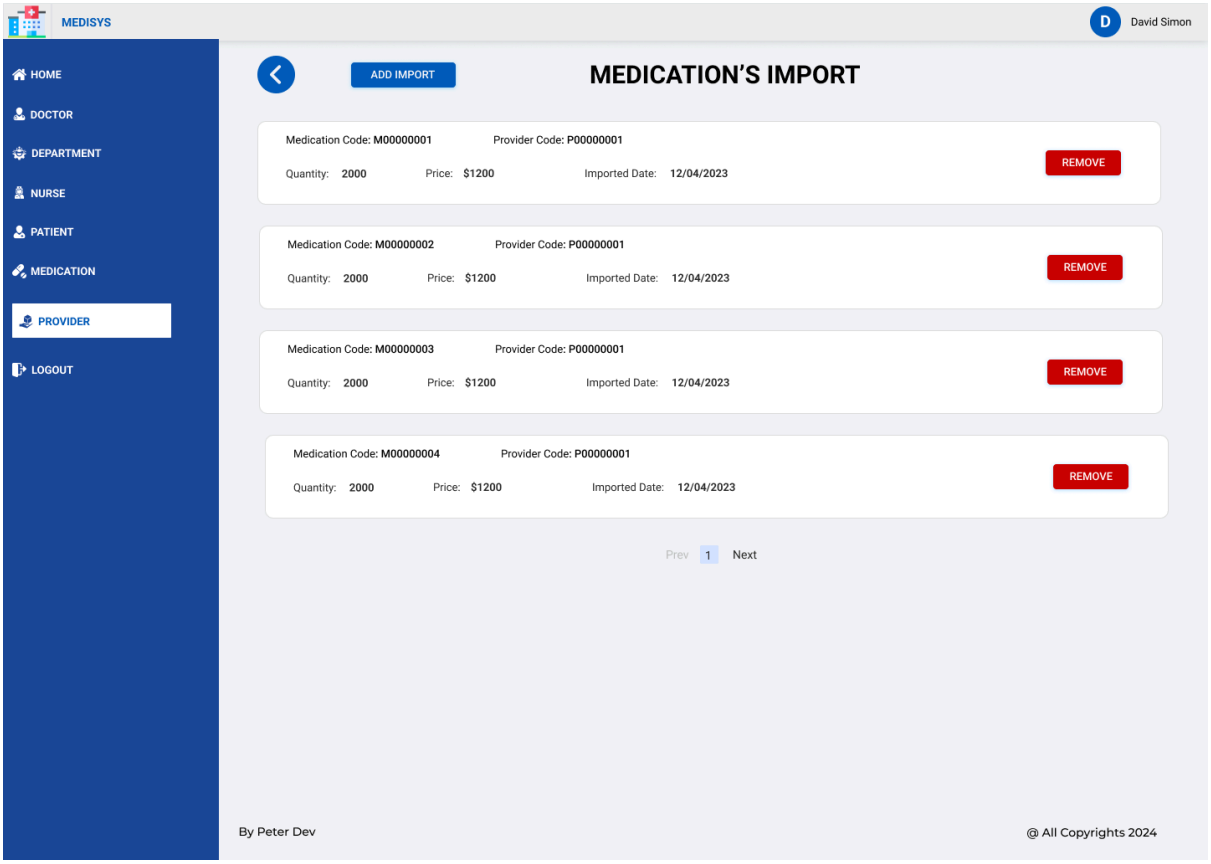
Patient detail: displayed when clicking on the "Detail" button in the Patient page. The visit logs of each patient will be displayed by dropdown form for better responsiveness to the page.





Medication Import: displayed when clicking on the "Import" button in the Provider page. The importation logs of each provider will be displayed and a button to Add Import.





3. ARCHITECTURE DESIGN

3.1. Presentation Strategy

3.2. Data Storage Approach

3.3. External services / APIs

4. IMPLEMENTATION - SPRINT 1

4.1. Setting up an online repository (github) for version control.

The version control system chosen is Git, and the remote repository service chosen is GitHub. The official repository is available here:
Link Github Project

5. IMPLEMENTATION - SPRINT 2

5.1. MVP 2

Some compelling reasons why React.js was chosen for this project:

- **Component-Based Architecture**: React breaks down the interface into small, easy-to-handle parts, making it simpler for developers to create, manage, and test different pieces of the Student Interface Module.
- **Virtual DOM**: React's Virtual DOM only updates what's needed, helps the student interface work smoothly.
- **Declarative Syntax for Readability**: React simplifies code readability, helping developers to describe the desired UI state. This makes it easier to reason about how the UI should behave and predict component behavior.
- **Active Community and Ecosystem**: Many developers use React, so there's a big group of people ready to help and share useful tools, making it easier for us to build the interface.

This MVP mainly resembles the look and feel of the system as in the user view, following with some illustrations of the user actions.

6. CONCLUSION

6.1. Challenges

6.2. Solutions

6.3. Lessons Learned