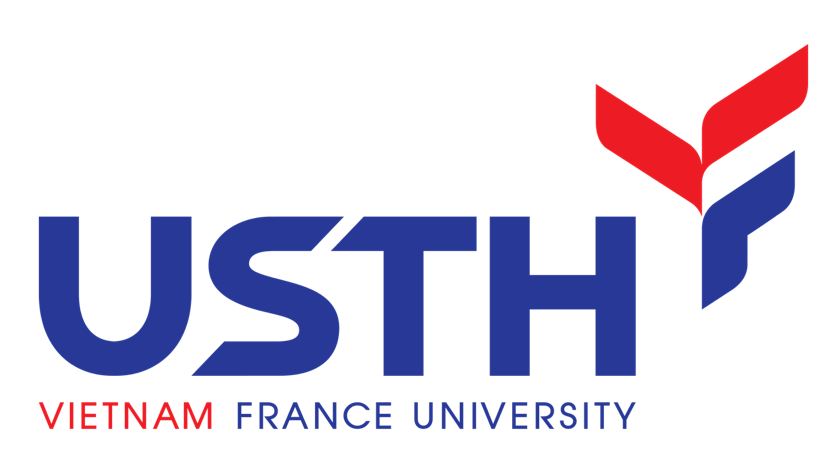
University of Science and Technology of Hanoi

ICT department



**REMOTE LUNG CANCER DIAGNOSIS**

Group members:

Nguyen Hoang Son (BI10-155)

Do Thanh Dat (BI10-026)

Nguyen Khang Thai (BI10-158)

Nguyen Tan Dung (BI10-041)

Trinh Quoc Hieu (BI10-060)

Supervisor: Dr. Tran Giang Son

University of Science and Technology of Hanoi

Hanoi, Jan 2021

Table of Contents

[1. Introduction 4](#_Toc95273956)

[1.1. Context and Motivation 4](#_Toc95273957)

[1.2. Objective 4](#_Toc95273958)

[2. Methodology 5](#_Toc95273959)

[2.1. Functional Requirements 5](#_Toc95273960)

[2.2. Sequence Diagrams 5](#_Toc95273961)

[2.2.1. Use-case diagram 6](#_Toc95273962)

[2.2.2. Sequence Diagrams 7](#_Toc95273963)

[2.2.3. System architecture 10](#_Toc95273964)

[3. Implementation 11](#_Toc95273965)

[3.1. Tools & Technical Choices 11](#_Toc95273966)

[3.1.1. Google Firebase 11](#_Toc95273967)

[3.1.2. REST and API Technology 11](#_Toc95273968)

[3.1.3. Firebase authentication function 12](#_Toc95273969)

[3.1.4. Node Package Manager (npm) and Yarn 12](#_Toc95273970)

[3.1.5. MockAPI 12](#_Toc95273971)

[3.1.6. JavaScript 13](#_Toc95273972)

[3.1.7. Cascading Style Sheets (CSS) 14](#_Toc95273973)

[3.1.8. Figma 14](#_Toc95273974)

[3.1.9. Thunder Client for Visual Studio Code 14](#_Toc95273975)

[3.2. Client site 14](#_Toc95273976)

[3.2.1. User Interface 15](#_Toc95273977)

[3.2.2. Combine with back-end 15](#_Toc95273978)

[4. Deployment 19](#_Toc95273979)

[4.1. Back-end Deployment 19](#_Toc95273980)

[4.2. Front-end Deployment 19](#_Toc95273981)

[5. Results 20](#_Toc95273982)

[6. Conclusions and Future works 20](#_Toc95273983)

[6.1. Conclusions 20](#_Toc95273984)

[6.2. Future Works 20](#_Toc95273985)

[Preference: 21](#_Toc95273986)

***Acknowledgement***

We would like to thanks all the teachers in ICT department in USTH, who have helped us a lot during our project.

We are greatly thankful to our supervisor Dr. Tran Giang Son for his kind support and guidance to successfully complete our project. We had highly benefited by his help and had gained a lot of knowledge about the various process and techniques employed.

We also would like to thanks Dr. Nghiem Thi Phuong for teaching us and helping us choose the most suitable project for us and Dr.Son for pointing out our shortcomings in our project report and how to make it better.

Group 15

# **1. Introduction**

## 1.1. Context and Motivation

Lung cancer is still one of the most frequent malignancies globally, killing more people each year than breast, colon, and prostate cancers combined.

* Lung cancer is by far the most common cause of cancer mortality in both men and women, accounting for over 25% of all cancer fatalities. Lung cancer kills more people each year than colon, breast, and prostate cancer combined. Worldwide in 2020, there were 2.21 million new cases of lung cancer and 1.80 million deaths from lung cancer.
* Lung cancer has the lowest 5-year survival rate at only 18% compared to breast at 90%, prostate at 99%, and colorectal at 65%.

In Vietnam, in 2018, according to the World Health Organization (WHO), we had the rate of 21.7 cases per 100,000 people, ranked 56 out of 185 countries and territories. We had the highest rate in South-East Asia. In the same year, Vietnam recorded more than 33,600 new cases and nearly 21,000 deaths.

According to Global Cancer Statistics, lung cancer has a high incidence and fatality rate. Most lung cancer patients are diagnosed when it has progressed to an advanced stage. This is because most lung cancer patients have no particular evident symptoms at the start of the disease, and many people are discovered at an advanced stage. According to statistics, the prognosis of lung cancer is closely related to the clinical stage, implying that early detection might enhance the patient's prognosis. Efficient diagnosis and screening procedures must be chosen to increase the early detection rate and forecast of lung cancer. The 5-year survival rate for lung cancer drops dramatically from a stage 1 diagnosis (68-92% survival) to a stage 4 diagnosis (0-10% survival). Only 16% of the eligible population will be diagnosed at an early stage when lung cancer is when the disease is most likely to be cured. Therefore, our project has the aim to build a mobile application to help doctors can remotely diagnose lung cancer.

## 1.2. Objective

First and foremost, we are to study about the anatomy and attributes of the lungs and lung nodules in order to attain a solid medical background. Using that knowledge, we try to create a Mobile Application to support doctors to remote diagnosis of lung cancer and management.

# **2. Methodology**

## 2.1. Functional Requirements

This Mobile Application is developed with basic services:

* The *Sign-Up* function allows users to create a new account in the system.
* The *Sign-In* and *Sign-Out* functions allow users to login or logout of the system.
* The *Home Screen* tab has 7 main features:
* Doctors: allows users to view famous doctors’ information and contact, admin can view, update or add doctors’ information in the database system.
* *Hospital*: show list of hospitals and contact information of each hospital.
* *Patients*: show list of patients and their information.
* *Labs*: (ongoing)
* *Insurance*: (ongoing)
* *Related Articles*: News, blog and doctors’ advices
* The *Chat* system: Allow users to contact doctors and supporters.
* The *Calendar* system: Users can check if they have an appointment schedule.
* The Account system: User’s account information.
* The system limited users with some features:
* The admin can only update system information through Database.

2.2. Sequence Diagrams

### 2.2.1. Use-case diagram

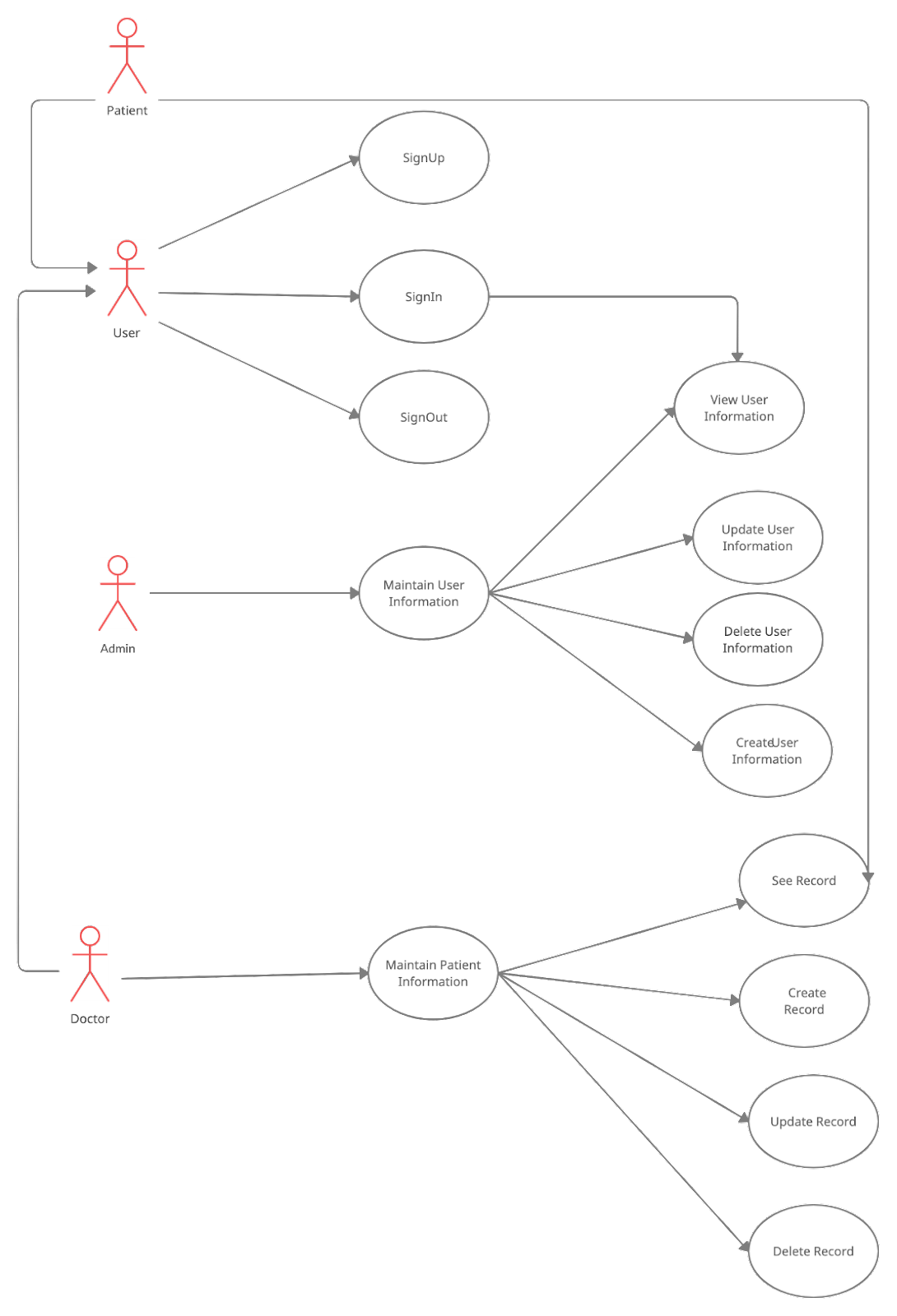


Figure 1: Use-case diagram

* Sign-Up: This use case describes how an user signs up to the Mobile App. The user can be a Patient or a Doctor.
* Sign-In : This use case describes how an Admin or a User signs in the Mobile App.
* Sign-Out: This use case describes how an Admin or an User signs out the Mobile App.
* Maintain user information: This use case describes how an Admin changes his/her personal information. With the role Admin, he/she can update information or delete other existing accounts besides his/her account.
* Update user information: This use case allows admin to edit the user information which appears in the dashboard.
* Delete user information: In this use case, the user has the role admin can delete the user account.
* Create user information: In this use case, the user has the role admin can create the user account.
* View User Information: The user can see their personal information.
* Maintain patient record: This use case describes how Doctors and Admins can create, update or delete medical records of patients. Patients can only view their own medical record.
* See record: This use case allows Patients to view their medical records and doctors to view patient records by searching patient ID in the Patients List.
* Create record: This use case allows Doctors to create new medical records for their Patients.
* Update record: Only the user who has a role as a Doctor can update the patient records information.
* Delete record: Only Doctors can delete Patient records.

### 2.2.2. Sequence Diagrams

2.2.2.1. Sign in

The user will have to press the login button. From there, the system will check the data on the database; if that data already exists, the system will allow login.

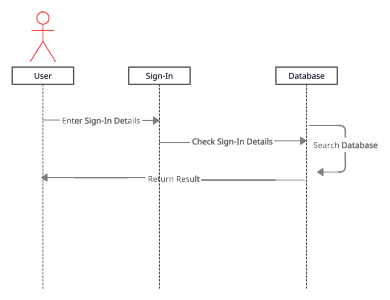


Figure 2: Sign-in sequence diagram

2.2.2.2. Sign Up

The user creates an account with the system by filling out a form with personal information. After completing the registration, he/she creates an account by clicking the "Sign Up" button. The "Sign Up" button will not be available to connect if the user is missing any information. If the user inputs duplicate data, the system will display an error notice and prompt the user to re-enter the information. If the user already has an account, he/she can select "Already have an account?" from the drop-down menu. To go to the login screen, click "Log in Now."

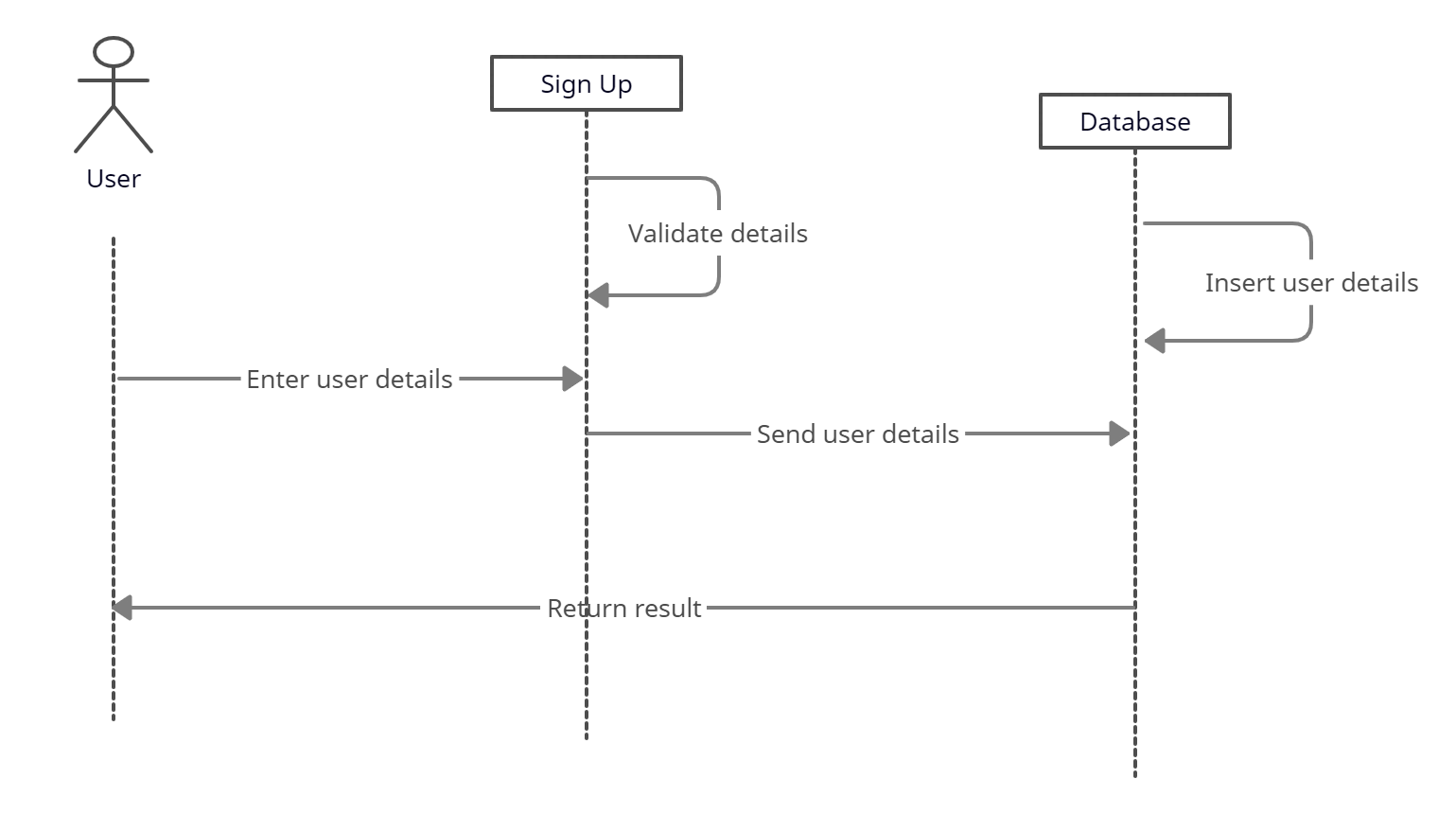


Figure 3: Sign-up sequence diagram

2.2.2.3. Chat

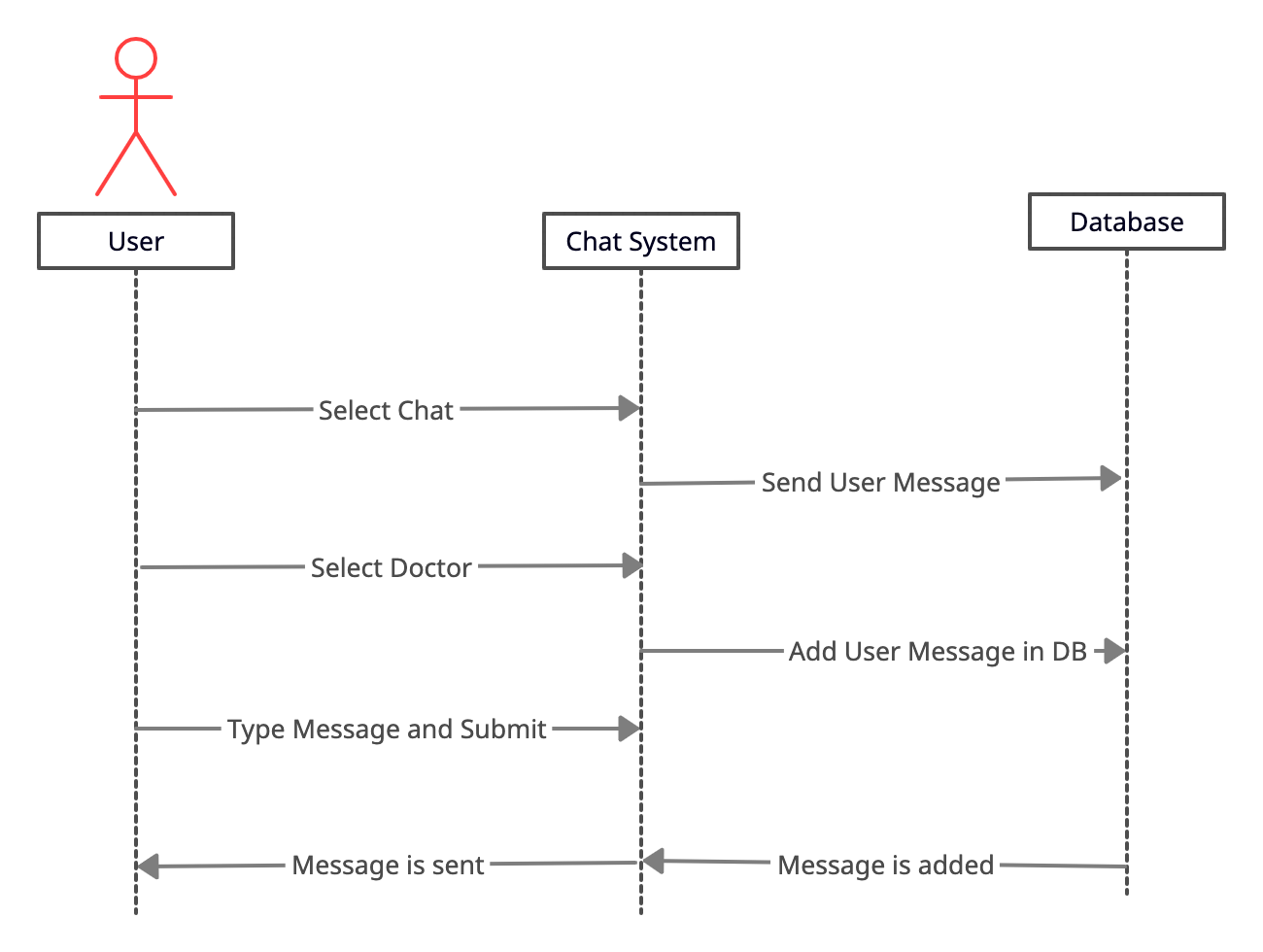


Figure 4: Chat sequence diagram

2.2.2.4. Search

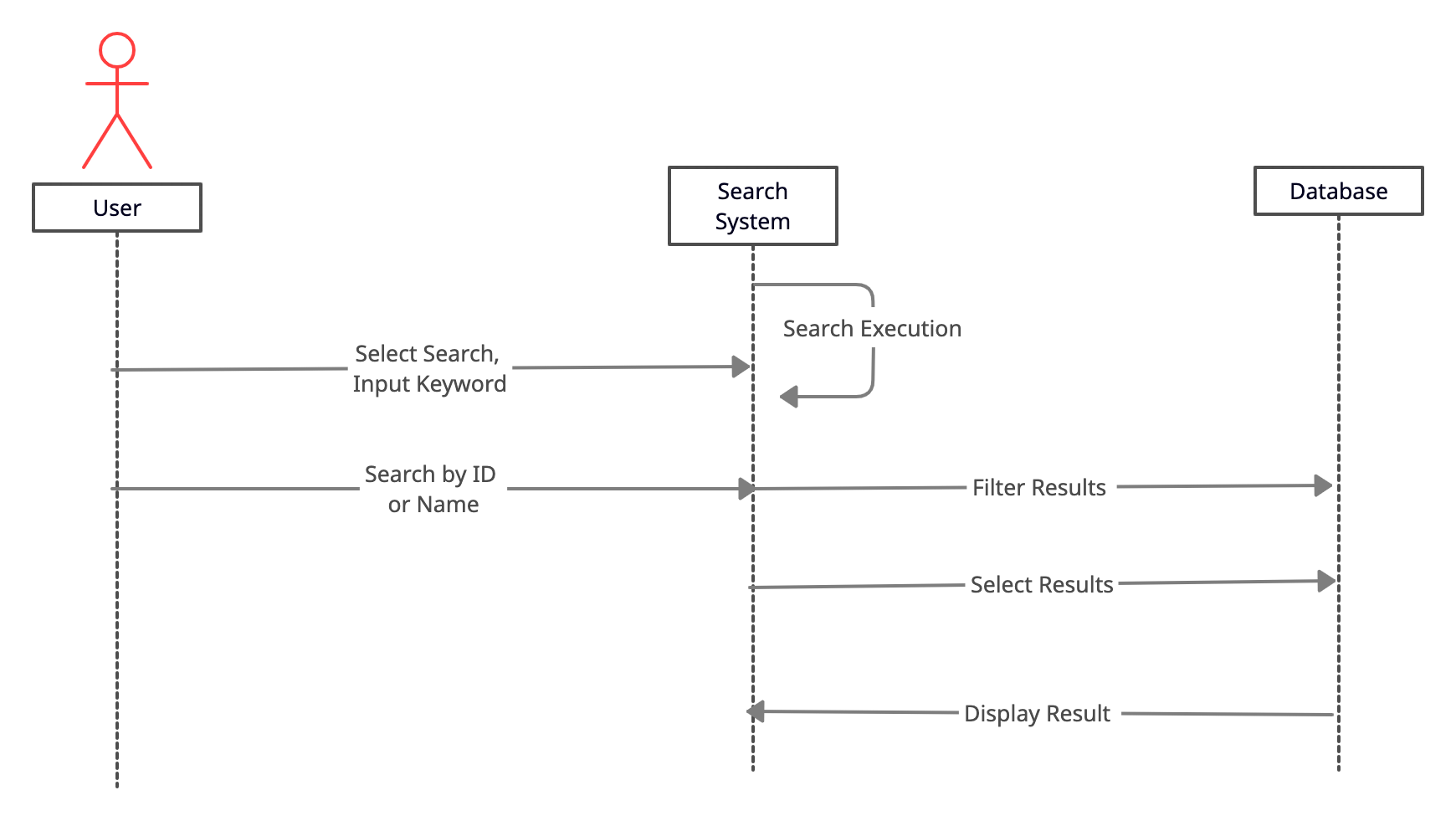


Figure 5: Search sequence diagram

* + 1. System architecture

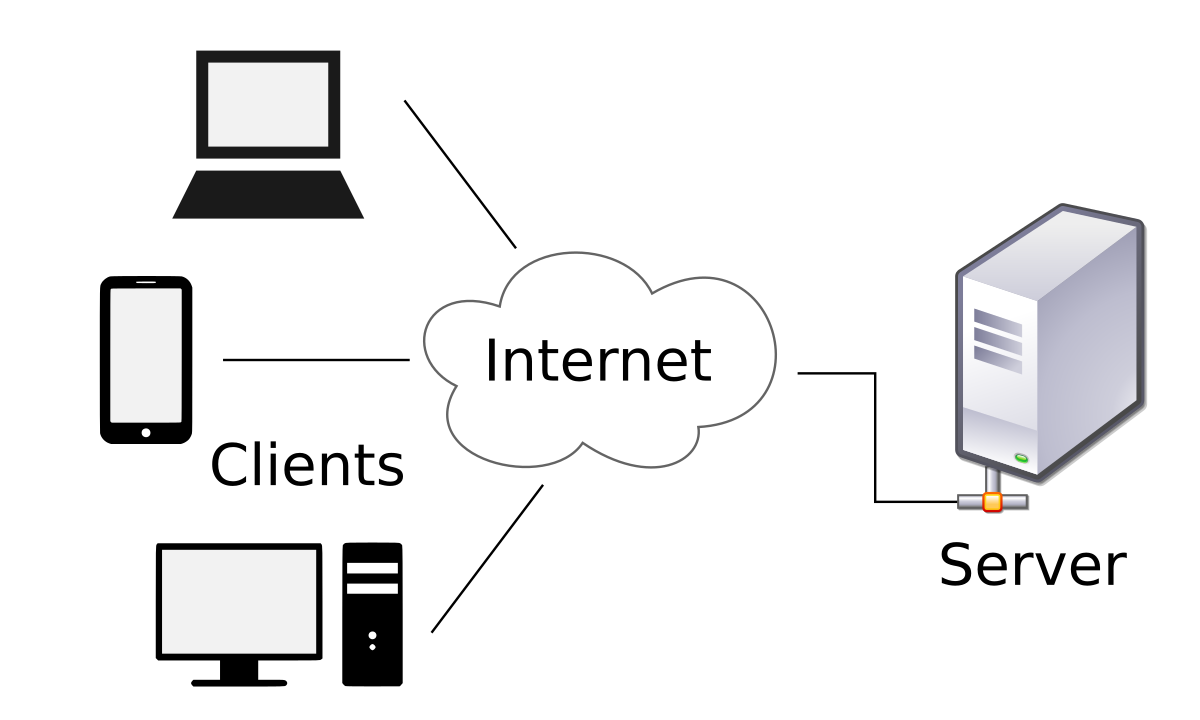


Figure 6: System diagram

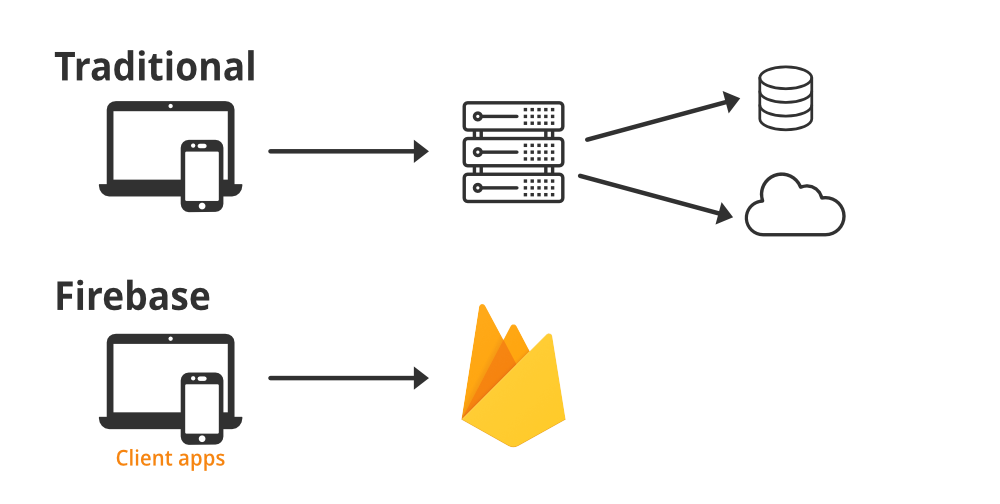
The Client-Server paradigm is a communication model where the server offers resources and services to one or more client devices. The client, any machine that wants to receive help or use services from a server, will send a request to that server under this paradigm. The server may then handle the client's request and response data. The benefit of using an architect is because a single server may serve a large number of clients at the same time, and the client can connect to the server over a network, eliminating the need for both parties to be in the exact location. This is the most prevalent architecture used in developing and sharing resources since it allows more effective communication and resource sharing.

# **3. Implementation**

## 3.1. Tools & Technical Choices

### 3.1.1. Google Firebase

Firebase is a product of Google which helps developers to build, manage, and grow their apps easily. It helps developers to build their apps faster and in a more secure way. No programming is required on the firebase side which makes it easy to use its features more efficiently. It provides services to android, ios, web, and unity. It provides cloud storage. It uses NoSQL for the database for the storage of data.



In our system, we use Firebase to contain a database of user accounts, patient information, doctor information, hospital information, documents in the *Related Articles* field.

### 3.1.2. REST and API Technology

Restful API is a standard used in the design of APIs for software, applications, and web services to facilitate the management of resources. System resources such as text files, images, videos, audio, or mobile data are targeted, including resource states that are formatted and transmitted over HTTP. The RESTful API commonly uses four methods of HTTP to perform the request:

* GET: to retrieve resource from server
* POST: to create a resource on the server
* DELETE: to remove a resource from the server
* PUT: to update a resource

Restful API helps the application clearly, return many types: xml, json,..., code is simple and easy to understand, focus on systems resources.

3.1.3. Firebase authentication function

Firebase Authentication provides backend services, easy-to-use SDKs, and ready-made UI libraries to authenticate users to your app. It supports authentication using passwords, phone numbers, popular federated identity providers like Google, Facebook and Twitter, and more.

Firebase Authentication integrates tightly with other Firebase services, and it leverages industry standards like OAuth 2.0 and OpenID Connect, so it can be easily integrated with your custom backend.

### 3.1.4. Node Package Manager (npm) and Yarn

Node Package Manager (npm) is a package manager for Node JavaScript platform. It helps put modules in specific folders so that node can find them, manages dependency library intelligent NPM consists of 2 main parts which are CLI (Command Line Input) tool for publishing, downloading packages and the other is a repository that hosts JavaScript packages.

### 3.1.5. MockAPI

### MockAPI is a simple tool that lets you easily mock up APIs, generate custom data, and perform operations on it using RESTful interface.

### MockAPI is free for 1 project, using JSON file,  easy to use  its URL to connect ThunderClient in VS Code with GET, PUT, DELETE, UPDATE,... to NoSQL Database.

### We use data format JSON, which stands for JavaScript Object Notation, it is a lightweight data-interchange format that is used to send data between computers.

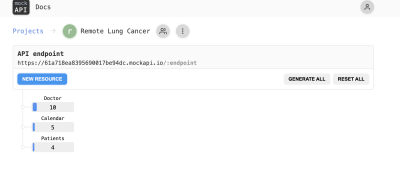


Figure 7: NoSQL database

### 3.1.6. JavaScript

* As one of the core technologies of the World Wide Web, JavaScript makes it capable to interact within web pages and most browsers already have a built-in engine to execute it. Following are some JavaScript libraries we used:
* TypeScript: TypeScript is JavaScript with syntax for types. TypeScript is a strongly typed programming language that builds on JavaScript, give better tooling at any scale.
* React Native: “A framework for building native apps with React". React Native enables you to build world-class application experiences on native platforms using a consistent developer experience based on JavaScript and React. React Native allows us to create reusable UI components. It is currently one of the most popular JavaScript libraries and has a strong foundation and large community behind it.

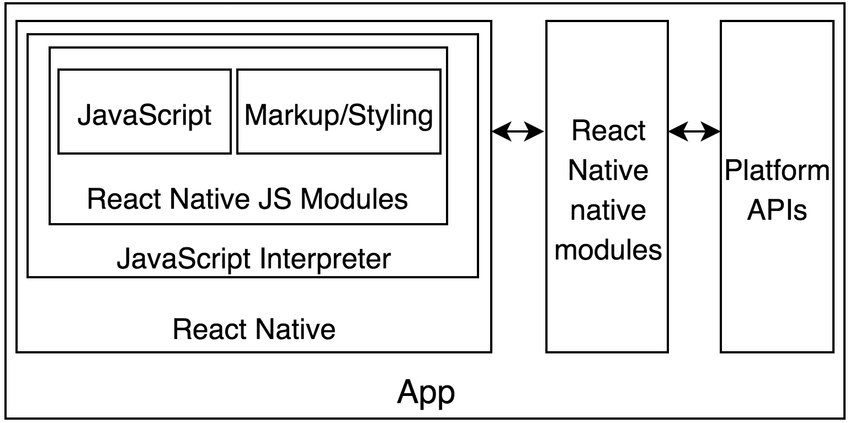


Figure 8: Application structure

* React Hooks: The React hook APIs provide an alternative to writing class-based components, and offer an alternative approach to state management and lifecycle methods. Hooks bring to functional components the things we once were only able to do with classes, like being able to work with React localstate, effects and context through useState, useEffect, useContext.
* Date-fns: provides the most comprehensive, yet simple and consistent toolset for manipulating JavaScript dates in a browser & Node.js.

3.1.7. Cascading Style Sheets (CSS)

Cascading Style Sheets (CSS) - Cascading Style Sheets (CSS) is designed to separate content from presentation for improving content flexibility and accessibility. With the use of React Native, we can directly set the styles for the components inside the TypeScript file, however, if we need to have some global styling in the application, we may define them CSS.

### 3.1.8. Figma

Figma is a browser-based UI and UX design application, with excellent design, prototyping, and code-generation tools. It’s currently (arguably) the industry’s leading interface design tool, with robust features which support teams working on every phase of the design process.

### 3.1.9. Thunder Client for Visual Studio Code

Send http/https API requests using any of the methods GET, POST, PUT, DELETE, PATCH, HEAD and OPTIONS.

* The Response data supports syntax highlighting using ACE Editor which can handle large responses easily. We just need the link of the API and use methods GET, POST, PUT, DELETE… with VS Code.
* It can show the status, time to connect the API, size of the API.

3.1.10. Android Emulator and Xcode’s simulator

Both Android Emulator (by Android Studio) and Xcode’s simulator (from Xcode) help in creating an Android Virtual Device (ADV) and IOS’s device on PC. It can be used for running and testing our application in both IOS and Android devices.

## 3.2. Client site

### 3.2.1. User Interface

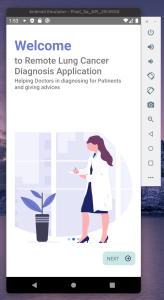
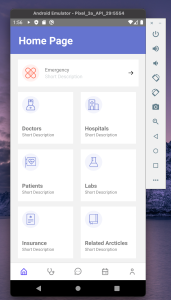
 

Figure 9: Welcome and Home Page Screens

### 3.2.2. Combine with back-end

We combine the API that is produced by the back-end team with the support of the networking library, to send requests and then display results so the user can easily interact with them.

3.2.3. Screens

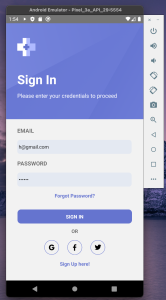


Figure 10: Sign in Screen

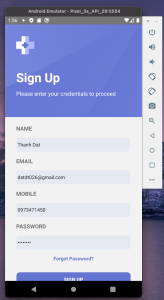


Figure 11: Sign up Screen

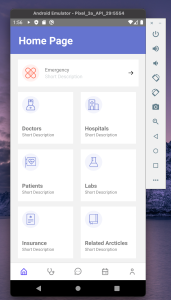


Figure 12: Home Page Screen

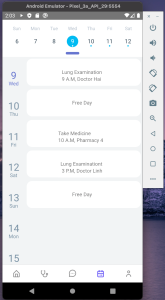
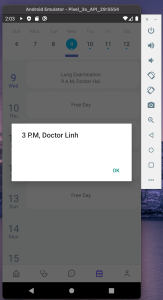
 

Figure 13: Calendar Screens

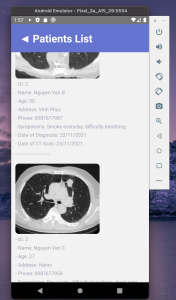


Figure 14: Patient list Screen

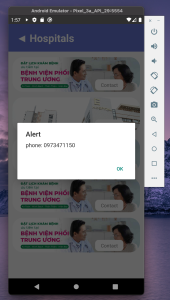
****

Figure 15: Hospital list Screen

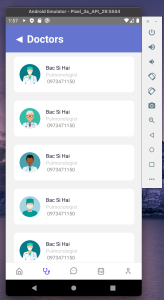


Figure 16: Doctor list Screen

# **4. Deployment**

## 4.1. Back-end Deployment

We use NoSQL json to store data. Besides, we also use Google Firebase to keep account information and check information related to login accounts chats. And we use MockAPI to store doctors, patients, calendar information.

## 4.2. Front-end Deployment

We use Figma with available free template to design our UI. And then, we follow that design to code our application. Moreover, we also use the UI design’s library.

# **5. Results**

We have implemented a Mobile Application for Remote Lung Cancer Diagnosis with these:

* Sign-Up function
* Sign-In/Sign-out function
* Maintain user information function (include View, Delete and Update, Create)
* Maintain patient record function (include View, Delete and Update, Create)
* Create appointment function
* Friendly User Interface

# **6. Conclusions and Future works**

## 6.1. Conclusions

* In conclusion, the purposes of Remote Lung Cancer Diagnosis Mobile Application have been accomplished. The contact between doctors and patients becomes easier by friendly User Interface and Chat System. Database management becomes easier and more effective.
* The main functions of the Remote Lung Cancer Diagnosis Mobile App have already been implemented:
* Users can register, log in to the system.
* Patients can view their own patient records.
* Doctors can view, create patient records.
* Chat Systems

## 6.2. Future Works

* To improve this Mobile Application in the future, the following tasks need to be done:
* Improve security.
* Improving the performance of the application to be faster
* Doctors can create schedules.
* Import and Export files
* Detect Node Module by camera

Preference:

UI design: <https://freedesignresources.net/healthcare-digital-clinic-ui-kit/>

Lung cancer statistics: <https://www.thoracic.org/about/global-public-health/firs/resources/world-lung-cancer-day-fact-sheet-2021.pdf>