

VIET NAM NATIONAL UNIVERSITY HO CHI MINH CITY  
INTERNATIONAL UNIVERSITY  
INFORMATION TECHNOLOGY - DEPARTMENT OF COMPUTER SCIENCE



## MOBILE APPLICATION DEVELOPMENT

---

### Project report

## DEVELOP A MOBILE TELEHEALTH SYSTEM

---

Lecturer: Dr. Le Duy Tan

Team: H3T-Solution

No	Student name	Student ID
1	Nguyen Thi Mai Huong	ITITIU19128
2	Ho Thi Thu Hoa	ITITIU19120
3	Tran Nam Tuan	ITITIU19230
4	Tran Tan Tai	ITITIU19047
5	Do Hoang Tuan	ITITIU19229
6	Tran Long Gia Huy	ITITIU19131

Ho Chi Minh City, January 1st 2024



## TABLE OF CONTENTS

<b>1</b>	<b>Introduction</b>	<b>5</b>
1.1	Background and Motivation . . . . .	5
1.2	Problem statement . . . . .	5
1.3	Project objectives and scope . . . . .	6
<b>2</b>	<b>System analysis and design</b>	<b>6</b>
2.1	Limitations of traditional health system . . . . .	6
2.2	Our proposed system . . . . .	7
2.2.1	System functional requirements . . . . .	7
2.2.2	System non-functional requirements . . . . .	8
2.2.3	Resources requirements . . . . .	9
2.3	System artifacts . . . . .	10
2.3.1	Database analyze and design . . . . .	10
<b>3</b>	<b>Applied tools, techniques, and frameworks</b>	<b>12</b>
3.1	Kotlin . . . . .	12
3.2	NodeJS . . . . .	12
3.2.1	Introduction about NodeJS and its operation method . . . . .	12
3.2.2	NodeJS advantages . . . . .	13
3.3	Mongodb . . . . .	14
<b>4</b>	<b>Demonstration</b>	<b>15</b>
4.1	Register/Login Layout . . . . .	15
4.2	Homepage/logout and View user Profile Layout . . . . .	18
4.3	Features related to user symptoms managing . . . . .	19
4.4	View list covid-19 result and add self-test result Layout . . . . .	23
4.5	Features related to vaccination history management flow . . . . .	25
<b>5</b>	<b>Conclusion</b>	<b>30</b>



5.1	Short Retrospective . . . . .	30
5.1.1	Lesson learned . . . . .	30
5.1.2	Limitations in the project . . . . .	30
5.2	Conclusion . . . . .	31
<b>6</b>	<b>References</b>	<b>31</b>



## LIST OF FIGURES

1	Database Design . . . . .	10
2	Registration user account screen . . . . .	15
3	Registration user account screen [fig 2] . . . . .	16
4	Login screen with button Register and save login option . . . . .	17
5	Homepage layout with logout button screen . . . . .	18
6	View user Profile screen . . . . .	19
7	Layout showing list symptom tracking . . . . .	20
8	Add new symptom screen . . . . .	21
9	Edit a symptom entry screen . . . . .	22
10	View list covid-19 result Screen . . . . .	23
11	Add self-test result Screen . . . . .	24
12	Vaccination History Screen with exceptional popup - User not vaccinated . . . .	25
13	Succeed added vaccination popup on Vaccination History Screen . . . . .	26
14	Vaccination list history Screen with vaccinated popup . . . . .	27
15	Vaccination registrations record Screen . . . . .	28
16	Vaccination registration Screen . . . . .	29



# 1 Introduction

## 1.1 Background and Motivation

The era of digital technology 4.0 along with the strong development of the internet has promoted the development of many platforms/ applications serve for healthcare management sectors in the world in general as well as in Vietnam particularly. This gradually be more and more concentrated on building healthcare applications after recognised the significant impact of Covid-19 pandemic, a global epidemic caused by the new coronavirus. according to post *WHO Coronavirus (COVID-19) Dashboard* in WHO Homepage: it has affected and became a concern in many countries around the world, causing hundreds of millions of infections (767,364,883 confirmed cases) and millions of deaths (6,938,353 deaths)[1]. The global impact of the COVID-19 pandemic has disrupted lives and economic conditions across many countries. This has led to economic crises, the closure of multiple businesses and factories, and widespread public concern.

Since the vital of the COVID-19 pandemic, it has become a global concern on how to digitalize all important processes in one applications from providing accurate and up-to-date information to facilitating registration, symptom tracking, accessing to test results, vaccine registration/status, and potential offering remote medical guidance. Many big tech companies over the world analysing and developing software that most fit for global needs, especially user-friendly application that easily used on mobile. In current decade, particularly from 2023 onward, we witness the emergence of numerous mobile telehealth applications. These applications have become ubiquitous globally, signifying a significant shift towards leveraging technology for healthcare solutions.

## 1.2 Problem statement

Although, the COVID-19 pandemic has been controlled and the world is recovering from the consequences caused by the disease in a positive way. However, the World Health Organization (WHO) released important information on March 16 on the situation of variants of SARS-CoV-2, WHO listed many variants as multiple variants of concern (VOCs) such as the Omicron viruses. And a number of variants with worrying symptoms have been identified by WHO since the beginning of the COVID-19 pandemic, that makes we cannot be negligent our health during the serious covid-19 epidemic. Bear that in mind, we created an Kotlin application that aims to empower users to take control of their health during these challenging times. Providing this application, it is really convenient for user to tracking his/her own symptom, the COVID test results from a testing facility, Vaccination registration and history.



### 1.3 Project objectives and scope

Our Telehealth App primarily aims to provide a comprehensive mobile solution for managing their health, with a particular focus on tracking their own symptom history in the context of COVID-19. Due to time limitation, we mainly focus on several essential functionalities outlined in the minimum viable product first such as user registration, symptom tracking, COVID-19 test results, vaccine registration/status, and optional features like remote medical guidance, text chat, and video chat.

Our application is developed using Kotlin, with the UI crafted utilizing Jetpack Compose featuring Material 3 components. For API communication, the client uses Retrofit 2, and Kotlinx is employed for efficient JSON serialization. Regarding data, crucially personal characteristics, are collected exclusively from registered users of our application. This approach ensures a user-centric concentration, offering a meaningful and tailored health management experience.

## 2 System analysis and design

### 2.1 Limitations of traditional health system

The traditional health tracking systems in place before the digital revolution were primarily dependent on manual data entry, paper-based records, and in-person consultations. This approach had several disadvantages. First of all, there was a significant chance of human mistake in record-keeping and data entry. Secondly, inefficiencies in patient treatment were frequently caused by the inability to obtain real-time data and the difficulties in quickly sharing information across various healthcare practitioners. Additionally, because paper records are more prone to loss and destruction by nature, these systems struggled with data security and privacy issues. In addition, patients' capacity to actively participate in their own health management was hampered by their restricted access to their own medical records.

COVID-19 forced health tracking systems around the world to quickly adapt, bringing in a variety of digital technology designed to track the virus's progress. However, these recently put into place mechanisms have their own set of difficulties. Since many COVID-19 tracking systems relied on self-reported data from many sources that adhered to different standards, they suffered from problems with data consistency and accuracy. Effective data exchange and analysis was further hampered by interoperability problems across various platforms and geographical areas. [2]

In addition to these technological difficulties, privacy issues intensified as a result of the gathering of private health information, posing important queries about consent and data use. Moreover, the digital divide became more pronounced, as individuals without access to technology or the internet were often left out of these tracking efforts. This exclusion resulted in data gaps, undermining the overall efficacy of these systems in managing the pandemic. [2]



## 2.2 Our proposed system

### 2.2.1 System functional requirements

Name	Detail
User Registration Module	<ul style="list-style-type: none"><li>• Secure sign-up process.</li><li>• Data privacy compliance.</li><li>• User-friendly interface.</li></ul>
Symptom Tracking Feature	<ul style="list-style-type: none"><li>• Daily input of health data.</li><li>• Tracking of COVID-19 specific symptoms.</li><li>• Analytical tools for monitoring symptom progression.</li></ul>
COVID-19 Test Results Management	<ul style="list-style-type: none"><li>• Upload and view test results.</li><li>• Notifications for test updates.</li></ul>
Vaccine Registration and Status Tracking	<ul style="list-style-type: none"><li>• System for vaccine registration.</li><li>• Real-time status updates on vaccination.</li></ul>
Remote Medical Guidance Tools	<ul style="list-style-type: none"><li>• Reliable text chat functionality.</li><li>• High-quality video chat for consultations.</li></ul>



Data Security and Compliance	<ul style="list-style-type: none"><li>• Secure storage of user data.</li><li>• Compliance with healthcare regulations and standards.</li></ul>
------------------------------	--

### 2.2.2 System non-functional requirements

Name	Detail
Performance Requirements	<ul style="list-style-type: none"><li>• High responsiveness and minimal latency in user interactions.</li><li>• Efficient handling of simultaneous user connections.</li></ul>
Security Requirements	<ul style="list-style-type: none"><li>• Advanced data encryption for user privacy.</li><li>• Compliance with healthcare data protection standards.</li></ul>
Scalability	<ul style="list-style-type: none"><li>• Capability to scale resources based on user demand.</li><li>• Adaptability to growing number of users.</li></ul>
Usability	<ul style="list-style-type: none"><li>• Intuitive user interface design.</li><li>• Accessibility features for diverse users.</li></ul>





Reliability	<ul style="list-style-type: none"><li>• Consistent app functionality with minimal down-time.</li><li>• Robust error handling and data recovery mechanisms.</li></ul>
Compliance	<ul style="list-style-type: none"><li>• Adherence to relevant legal and regulatory standards.</li><li>• Regular updates to comply with evolving health-care policies.</li></ul>

### 2.2.3 Resources requirements

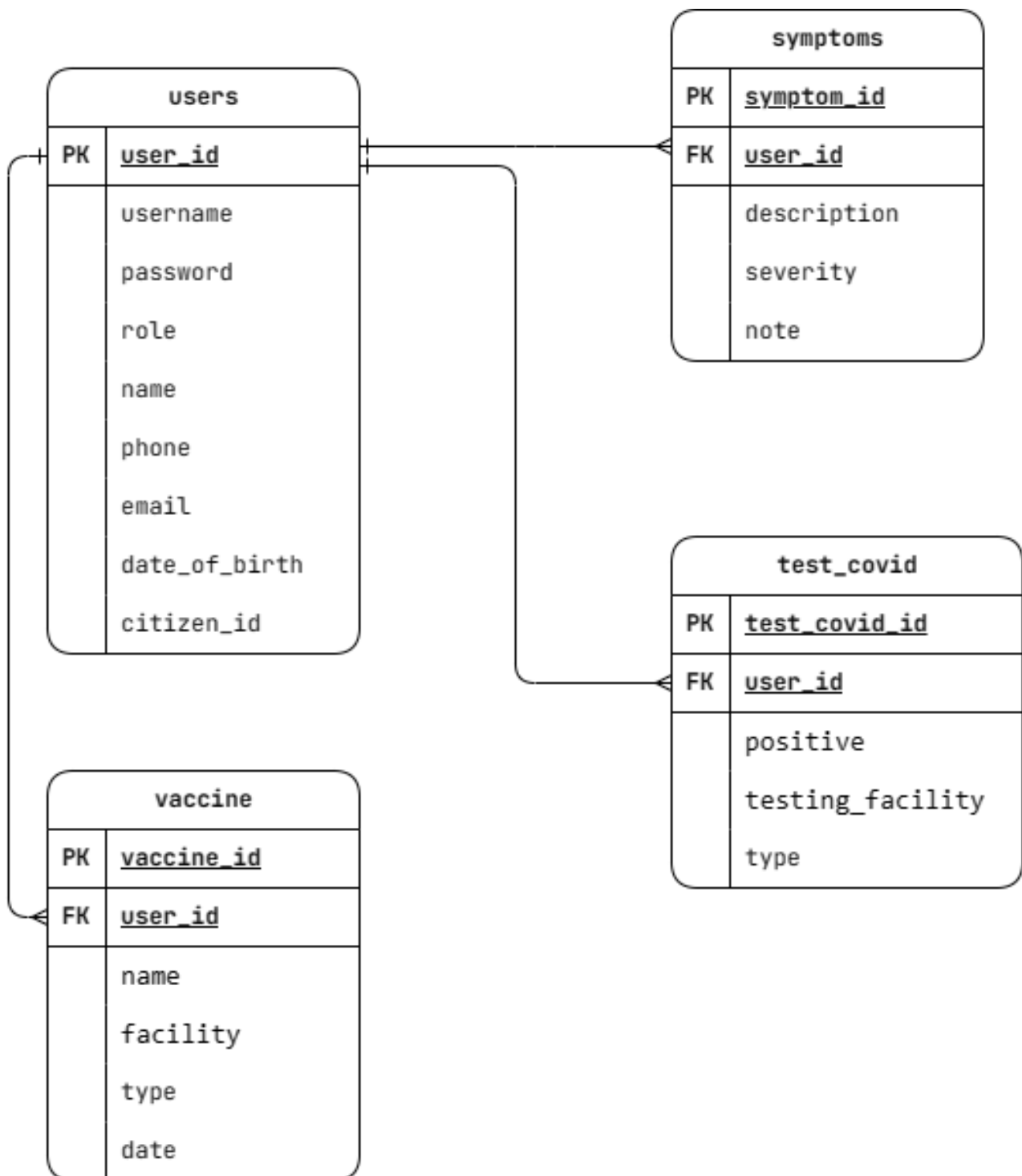
For our Tele-health App project, we recognize the need for a diverse range of resources to ensure its success. Initially, we will focus on establishing a highly skilled development staff. This team, which includes myself and other professionals in healthcare app development, UI/UX design, database management, and security, will be in charge of creating the app's architecture, designing the user interface, and maintaining strong security standards. Moreover, our plan would engage medical professionals to provide essential healthcare insights and validate the app's medical functionalities.

In terms of infrastructure, the project requires substantial server capacity for hosting the app and managing user data. We will rely on reliable cloud services for data storage and scalability, and implement robust cybersecurity measures to protect sensitive health information. Our investment extends to testing equipment and environments, ensuring the app's compatibility across various devices. Additionally, ongoing maintenance, updates, and user support are crucial for addressing any technical challenges and adapting to changing healthcare guidelines. The project's success also hinges on integrating the app into existing healthcare systems and ensuring compliance with regulatory standards, a process that necessitates collaboration with health authorities and IT specialists, including our team.



## 2.3 System artifacts

### 2.3.1 Database analyze and design



Hình 1: Database Design



#### 1. users

- Central table in the schema, with other tables related to it.
- Contains user information such as username, password, role, contact details, and unique identifiers like date of birth and citizen ID.
- The **user\_id** is the **primary key (PK)** that uniquely identifies each user.

#### 2. symptoms

- Tracks the symptoms experienced by users.
- Includes a description of the symptom, its severity, and an additional note for any extra information.
- The **symptom\_id** serves as the PK, with **user\_id** as a **foreign key (FK)** referencing the **users** table.
- Establishes a **one-to-many** relationship as one user can report multiple symptoms.

#### 3. test\_covid

- Holds the COVID-19 test results for users.
- Records whether the test was positive, the facility where the test was conducted, and the type of test administered.
- The **test\_covid\_id** is the **PK**, with **user\_id** as an **FK** linked to the users table.
- Implies a **one-to-many** relationship since a user might undergo multiple tests over time.

#### 4. vaccine

- Keeps track of vaccination details for users.
- Logs the name of the vaccine administered, the facility where the vaccination took place, the vaccine type, and the date of vaccination.
- The **vaccine\_id** is the **PK**, with **user\_id** as an **FK** connecting back to the users table.
- Indicates a **one-to-many** relationship since a user can receive multiple vaccine doses.

The system is designed to facilitate the tracking of individual users' health data, specifically pertaining to COVID-19. It enabled users to document various instances of symptoms and undergo multiple COVID tests, ensuring a comprehensive record of their health status in relation to the virus. Additionally, the system maintains detailed vaccination records, which are essential for monitoring the distribution and administration of vaccines to the populace. This capability is crucial for managing public health efforts and ensuring the effective control of vaccine uptake.

## 3 Applied tools, techniques, and frameworks

The app is written in Kotlin. The User Interface is constructed using Jetpack Compose with Material 3 components. The client uses Retrofit 2 for API communication. Kotlinx is used for JSON serialization. Regarding server, it is written with TypeScript and run on NodeJS. The database uses MongoDB.

### 3.1 Kotlin

Kotlin is a concise, modern and mature programming language that is interoperable with Java and other languages, and offers various ways to boost productivity, code safety, and developer satisfaction. It is a statically typed programming language that contributes on reducing language verbosity relative to Java. Its interoperability characteristic helps developers develop applications utilizing Kotlin together with other JVM languages that makes them can be applied libraries written in different language such as jars.

To the best of my knowledge, a large percentage of Kotlin developers community suggests migrating from Java to Kotlin due to several reasons:

- Java interoperability
- Better lambdas, Null safety
- Destructuring declaration, String interpolation, Operator overloading Familiar syntax, Smart casts, Intuitive equals, Extension Functions

### 3.2 NodeJS

#### 3.2.1 Introduction about NodeJS and its operation method

Nodejs is an independent development platform (Platform) built on V8 JavaScript Engine – an interpreter that executes JavaScript code that makes it possible to build web applications such as video clips, forums and especially is a narrow social networking site that quickly and easily expands. Node.js has been built and developed since 2009, sponsored by Joyent company.

NodeJS can run on many different operating system platforms from Windows to Linux, and OS X so that is also an advantage. NodeJS provides rich libraries in the form of various Javascript Modules that simplify programming and reduce time to a minimum.

#### *NodeJS's operation method:*

- The main idea of Node js is to use non-blocking, directing data input and output through real-time tasks quickly. Because Node js is rapidly scalable and capable of handling a large number of concurrent connections by high throughput.



- If in traditional web applications, requests create a new request processing thread and occupy the system's RAM, the system's resources will be used inefficiently. Therefore, the solution that Node.js offers is to use single-threaded (Single-Threaded), combined with non-blocking I/O to execute requests, allowing tens of thousands of concurrent connections to be supported.

### 3.2.2 NodeJS advantages

NodeJS is widely applied in many enterprise projects because it meets many different uses:

- NodeJS does not need to wait for the API to return data, so any APIs in the NodeJS library are not synchronized.
- NodeJS is a Platform, not a Framework. Therefore, Node JS allows you to build websites independently and faster.
- NodeJS can run on multiple platforms including Windows, MacOS, and Linux.
- NodeJS is considered a single-threaded server and cannot support multithreading.
- NodeJS is not considered a programming language, so newbies must have a solid grasp of basic programming knowledge such as protocols, Javascript, etc. to be able to use NodeJS. However, the NodeJS community is usually very large, and ready to support you anytime, anywhere.
- NodeJS can create, open, read, write, delete, and close files while they are on the server.
- The core part of NodeJS is usually known in C++ language, so its performance and processing speed are relatively high. As a result, most NodeJS applications are capable of responding to real-time running on cross-platform, multi-device, etc.
- Build content for dynamic websites.
- Perform data collection according to specific requirements.
- Perform query, edit, delete, add data in basic management systems such as: Microsoft SQL Server, MySQL, MongoDB, PostgreSQL.

**In this project, I build APIs using the NodeJS framework**



### 3.3 Mongoddb

Mongoddb is a versatile and robust NoSQL database that seamlessly integrates with Kotlin applications powered by Node.js. It offers document-oriented architecture and scalability features, that enabling to handle diverse data types and work with data efficiently. In other words, it brings high availability, high performance and easy scalability.

MongoDB doesn't store data in a table like traditional relational databases, instead it utilizes **collections and documents** to save data in BSON (Binary JSON) format, mirroring a JSON-like structure. A MongoDB server can contain many databases.

In MongoDB, ***Document*** is a set of key-value pairs that can be organized in the flexible schema. That offers the ability to hold different data types in documents' common field.

Collection can be considered as an RDBMS table, it is a set of MongoDB documents that have a close purpose. Collections do not enforce a schema which means documents in the same collection can have different fields.

This resilience in data representation aligns well with the dynamic and evolving nature of our Kotlin application, allowing us to store and retrieve data in a manner that fits the diverse needs of our users.



## 4 Demonstration

### 4.1 Register/Login Layout

17:57 100% X [Icons] [Icons] [Icons]

17:57 [Icons] VoD LTE1 7%

## Register an account

Username

Username must 8 characters in length, but no more than 32.

Password

Password must contain:

- At least one digit.
- At least one lowercase character.
- At least one uppercase character.
- At least one special character (\*.!@#\$%^&(){}[];,<>,?/~\_+|=|\).
- At least 8 characters in length, but no more than 32.

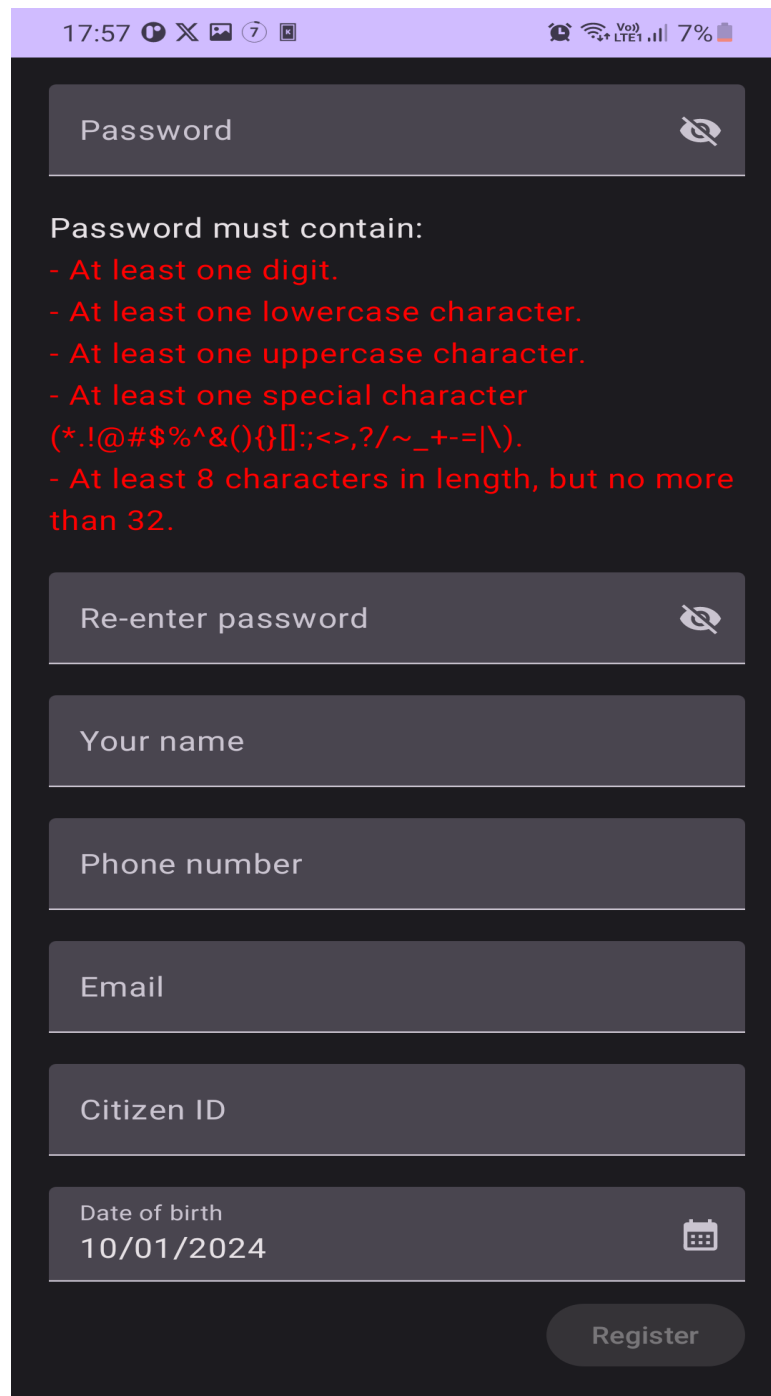
Re-enter password

Your name

Phone number

Email

Hình 2: Registration user account screen



17:57 [status icons] 7%

Password [toggle icon]

Password must contain:

- At least one digit.
- At least one lowercase character.
- At least one uppercase character.
- At least one special character (\*, !, @, #, \$, %, ^, &, (, ), {, }, [, ], ;, <, >, ?, /, ~, \_ +, =, |, \).
- At least 8 characters in length, but no more than 32.

Re-enter password [toggle icon]

Your name

Phone number

Email

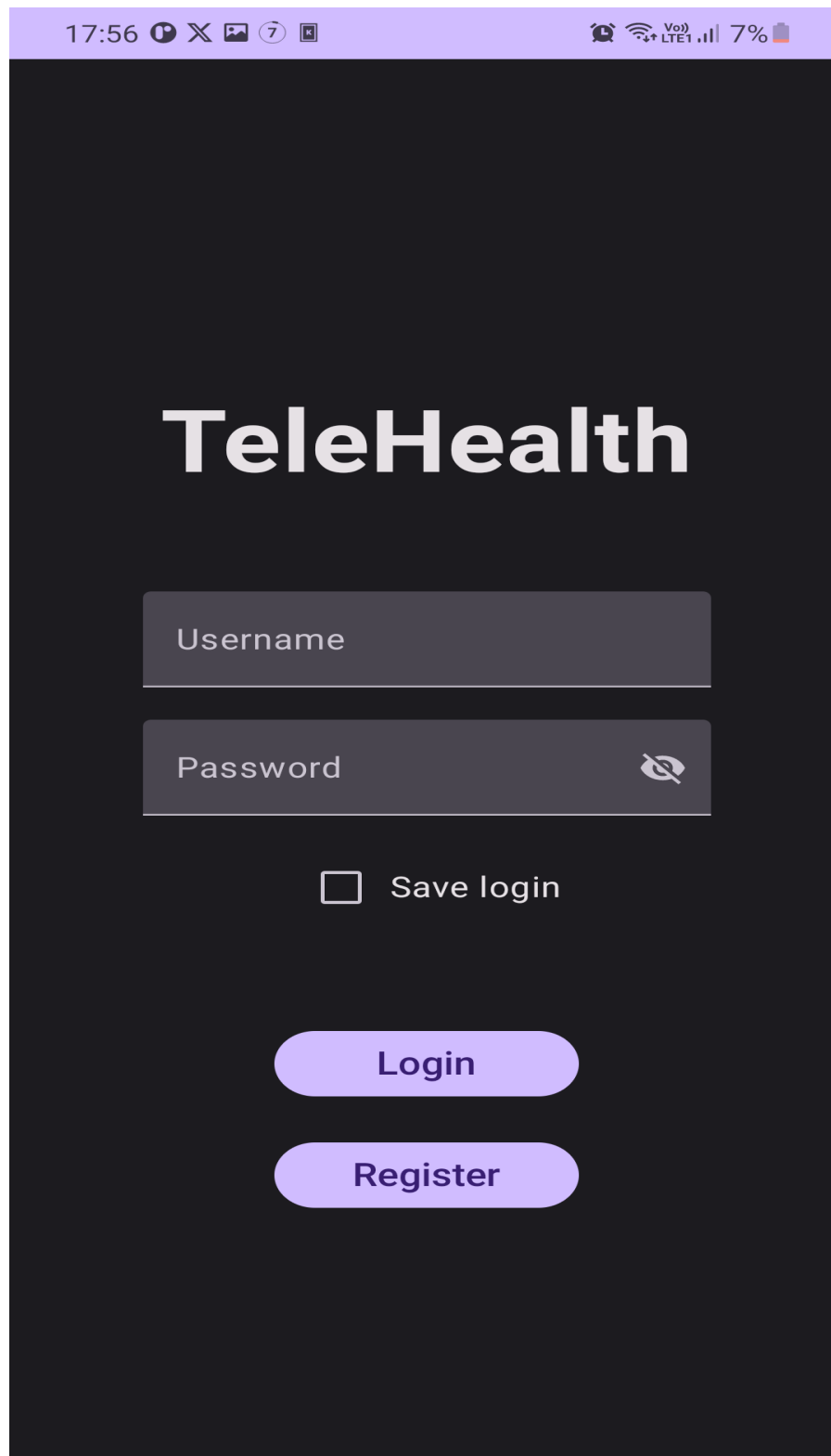
Citizen ID

Date of birth  
10/01/2024 [calendar icon]

Register

Hình 3: Registration user account screen [fig 2]

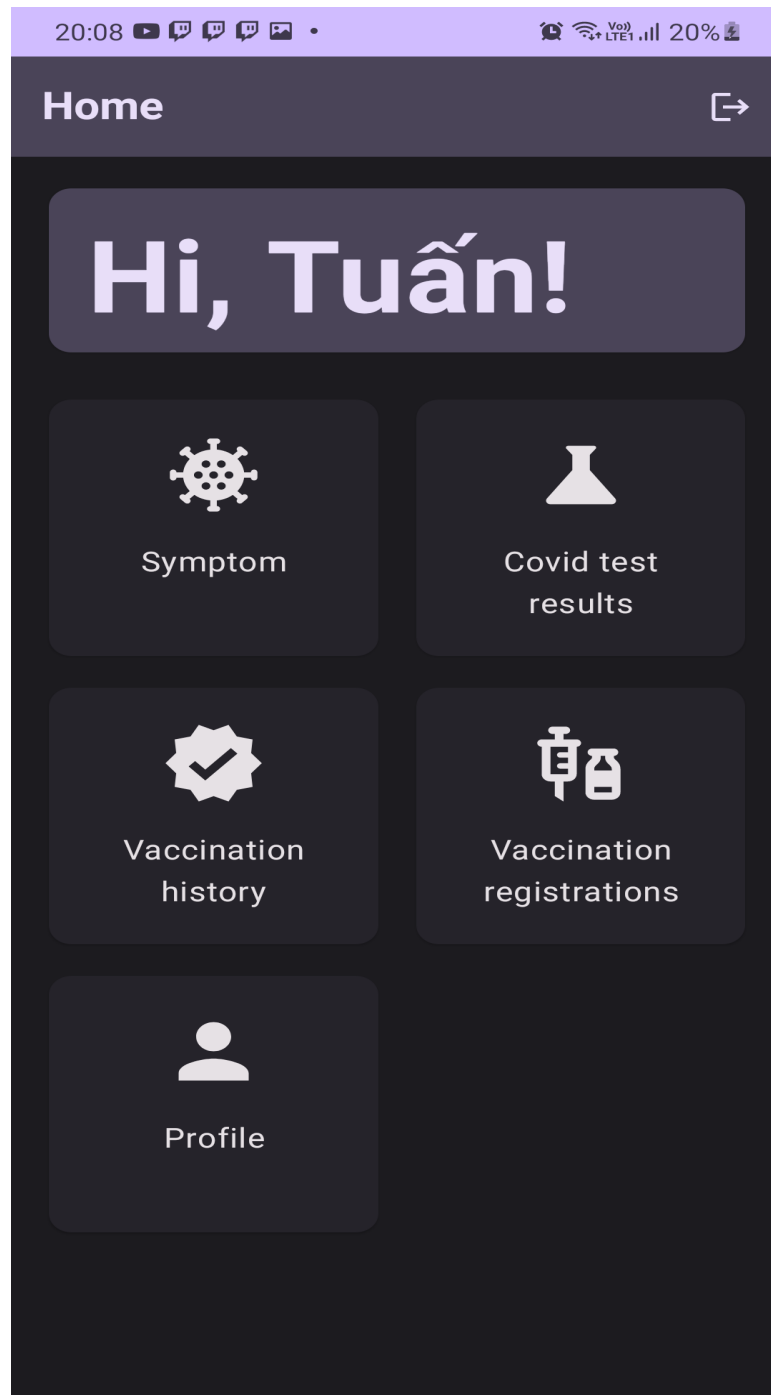




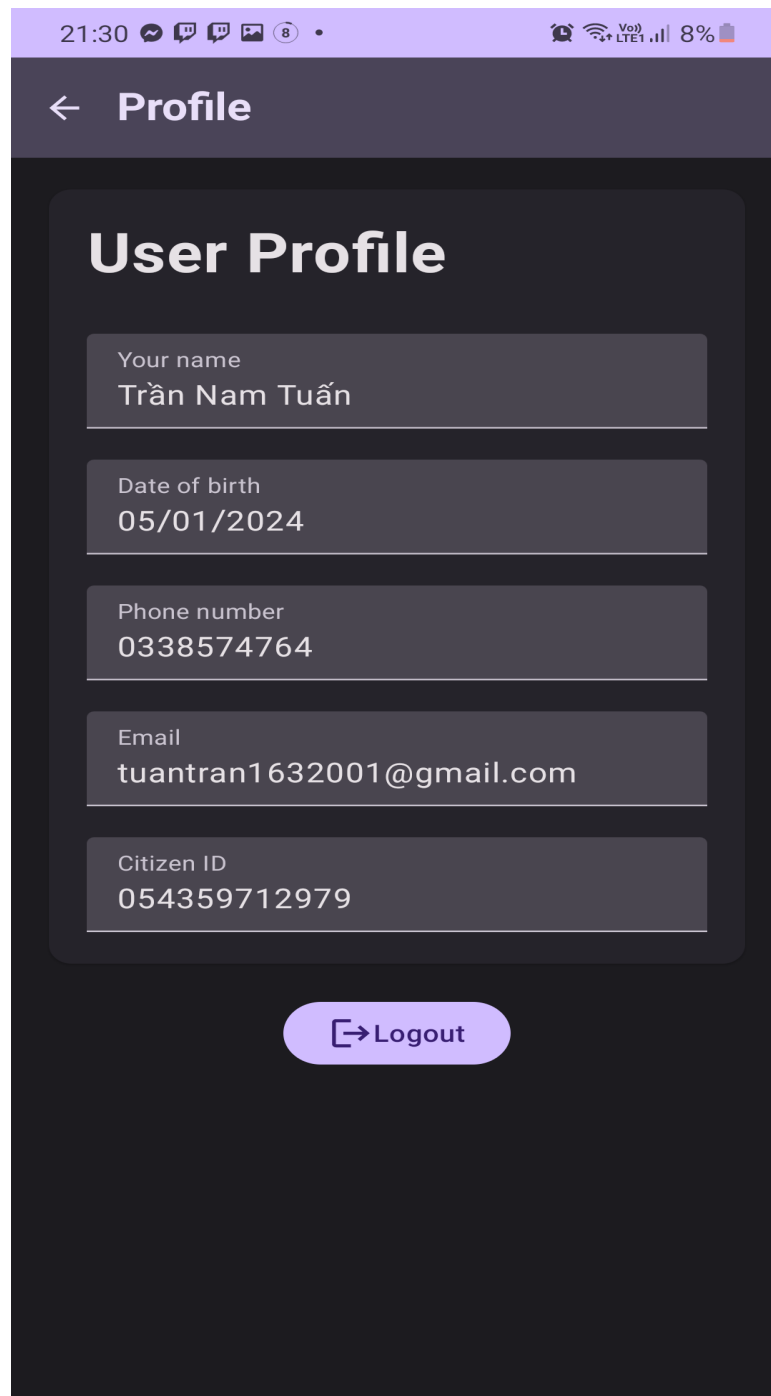
Hình 4: Login screen with button Register and save login option



## 4.2 Homepage/logout and View user Profile Layout



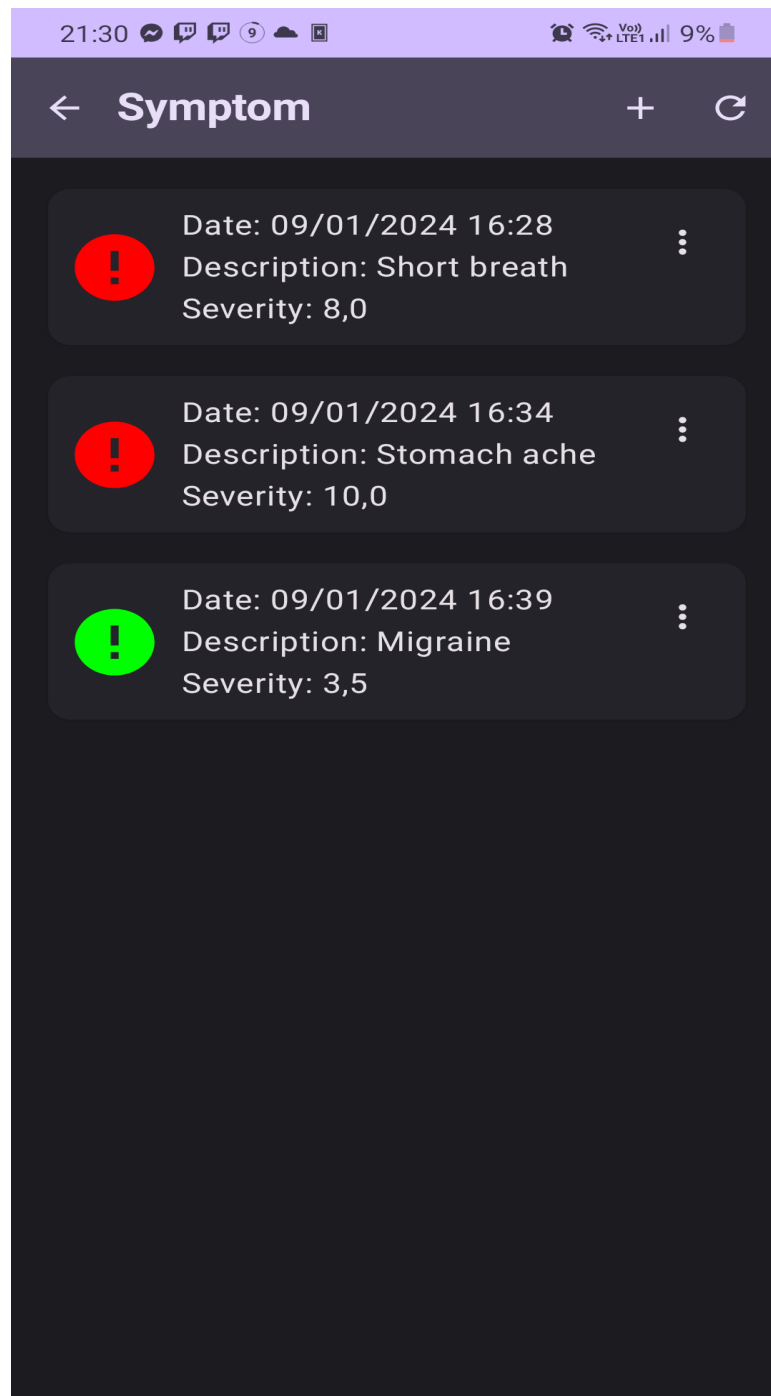
Hình 5: Homepage layout with logout button screen



Hình 6: View user Profile screen

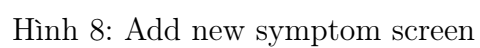
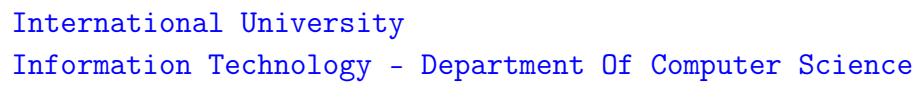
### 4.3 Features related to user symptoms managing

The app allows the user to track their symptom, with a severity rating ranging from 1 (mild) to 10 (extreme).



Hình 7: Layout showing list symptom tracking

Add new symptom layout. The date and time will automatically be set.





18:30 🔊 🔒 📷 📶 6%

← **Edit your symptom** ✎ 🗑

Date  
09/01/2024 16:28

Description  
Short breath

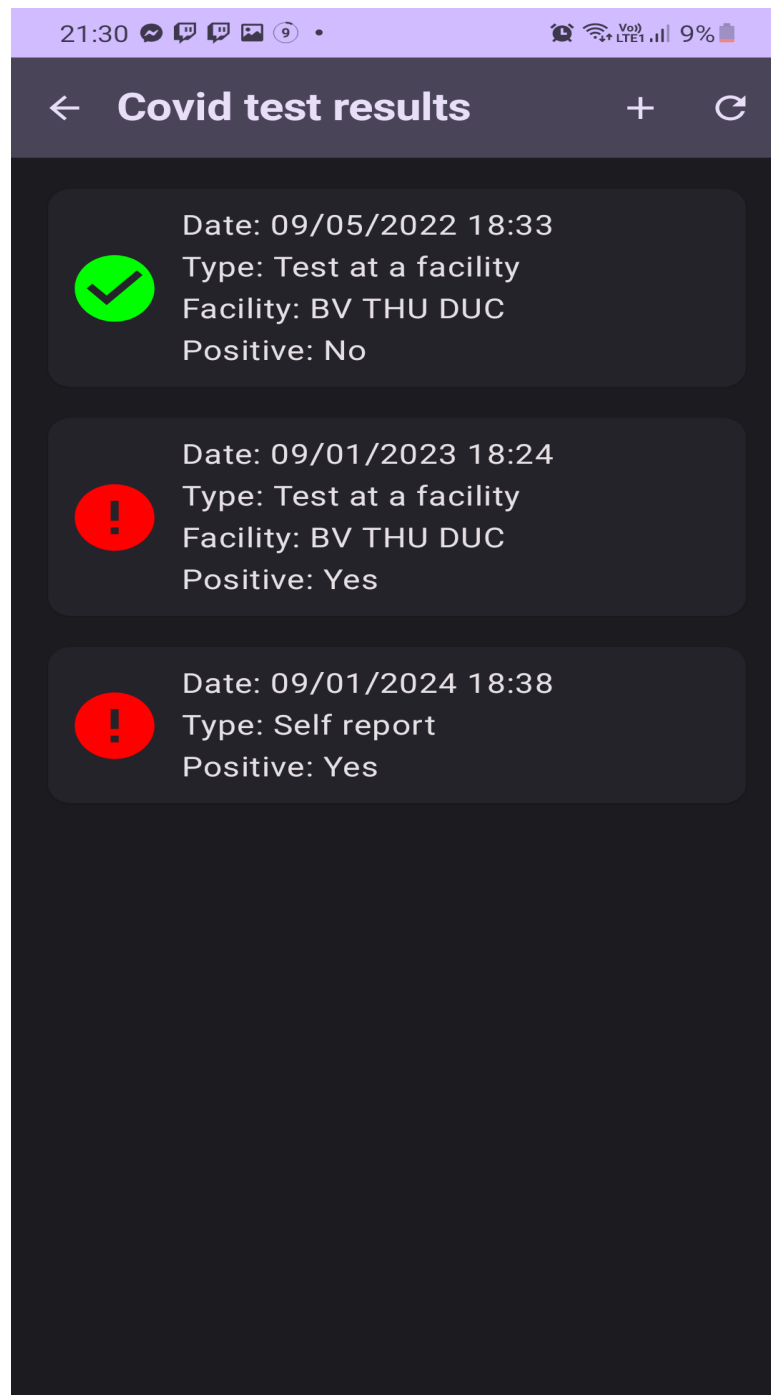
Severity  8,0

Note  
No note

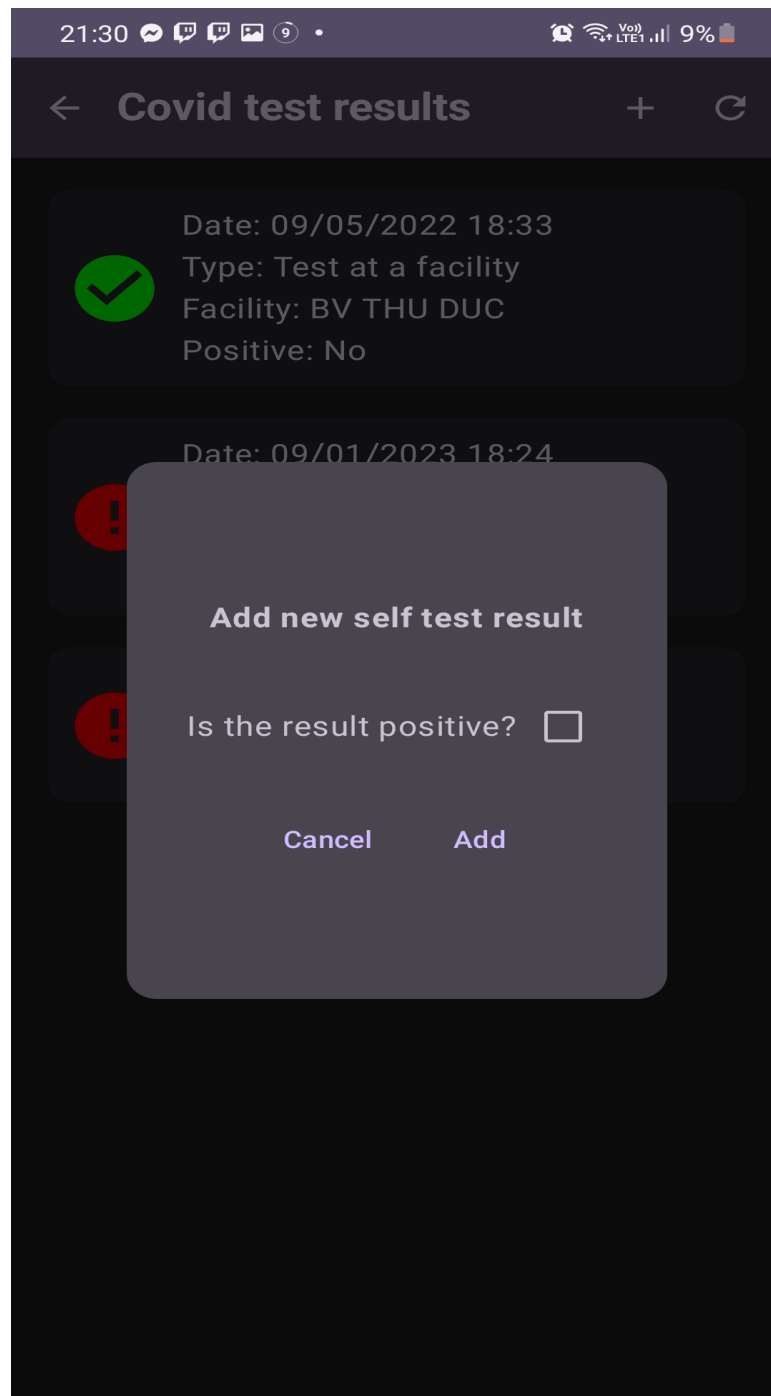
Hình 9: Edit a symptom entry screen



#### 4.4 View list covid-19 result and add self-test result Layout



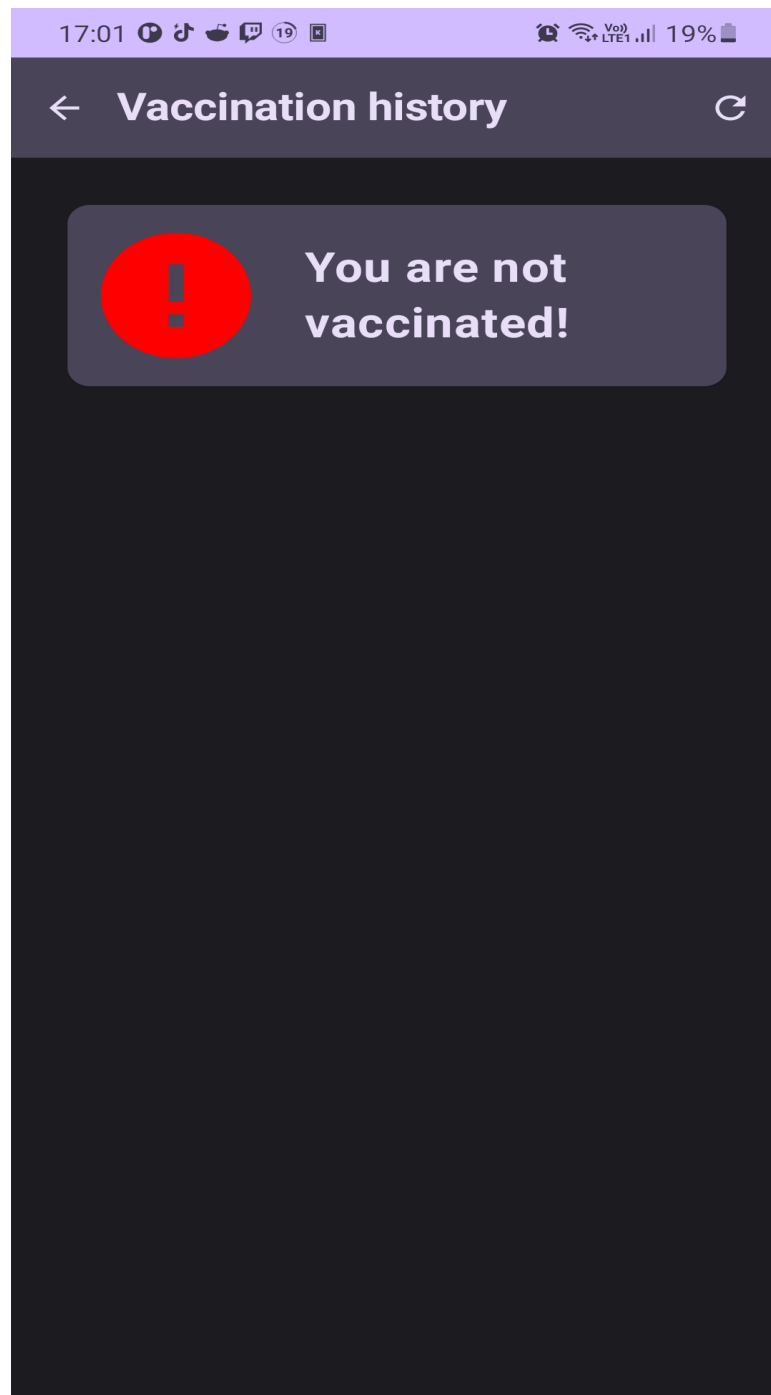
Hình 10: View list covid-19 result Screen



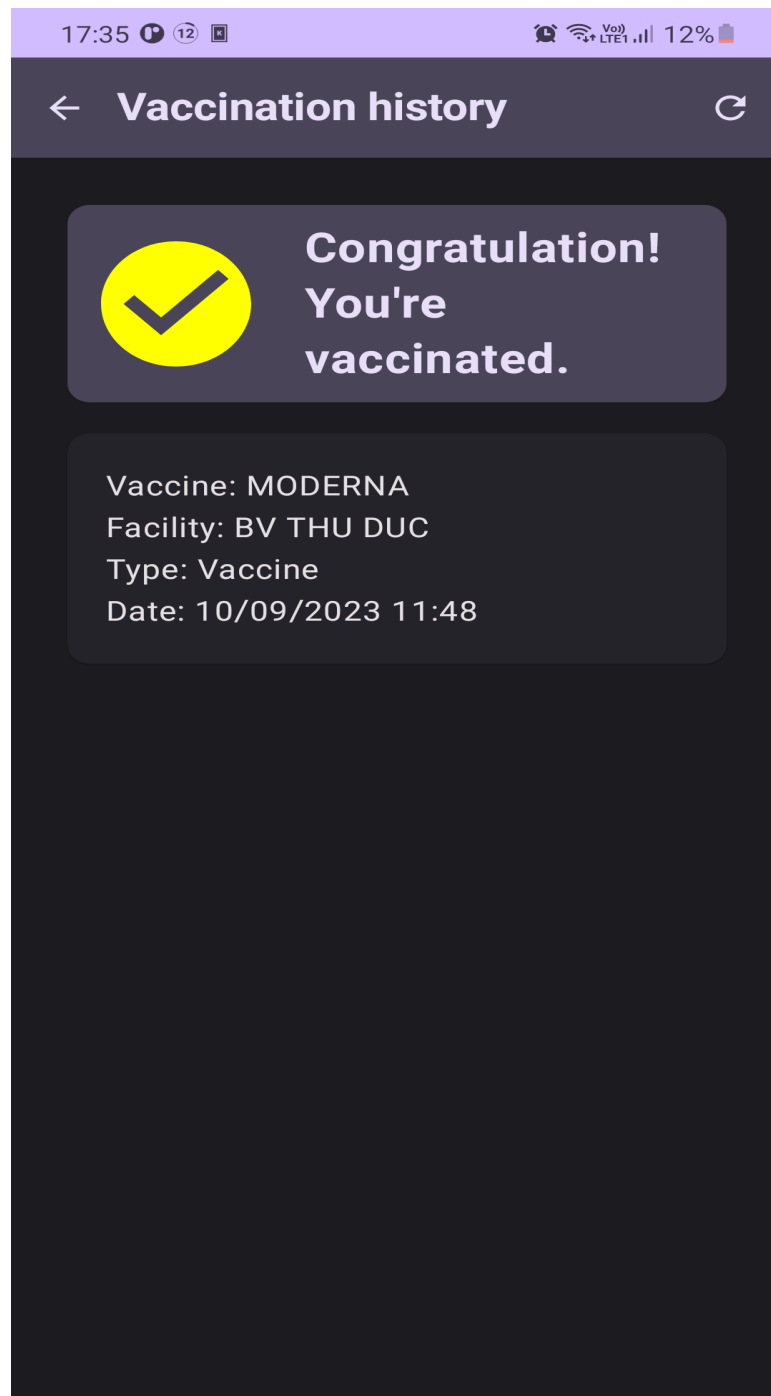
Hình 11: Add self-test result Screen



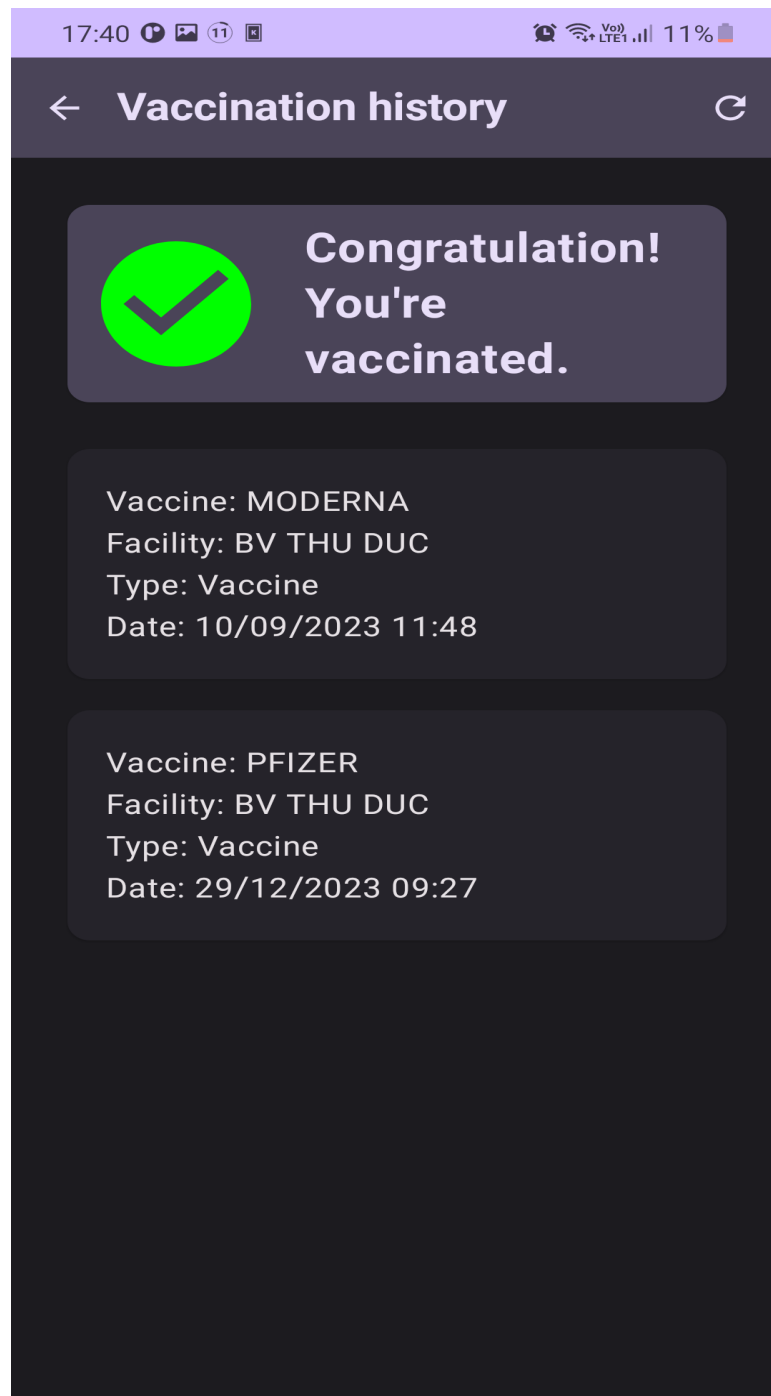
#### 4.5 Features related to vaccination history management flow



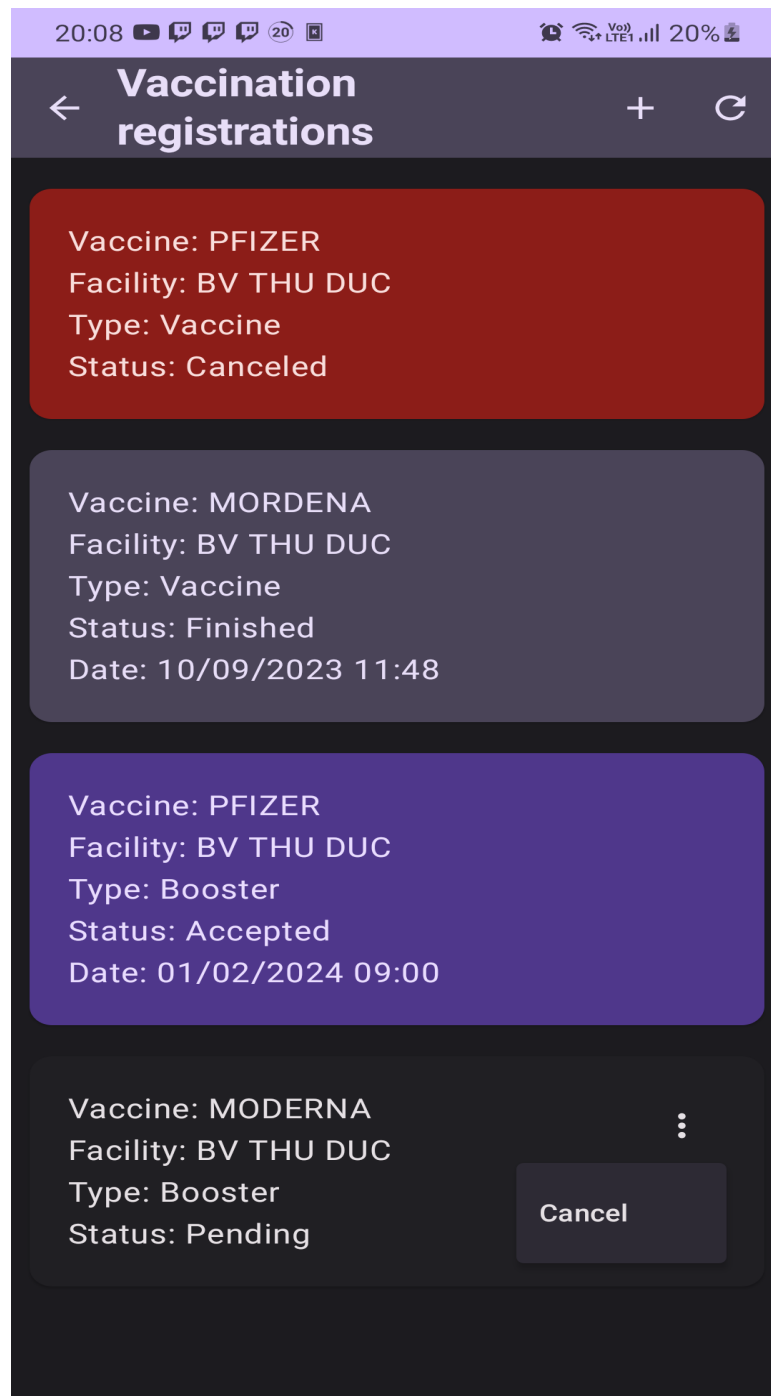
Hình 12: Vaccination History Screen with exceptional popup - User not vaccinated



Hình 13: Succeed added vaccination popup on Vaccination History Screen



Hình 14: Vaccination list history Screen with vaccinated popup



Hình 15: Vaccination registrations record Screen



20:08 20%

Register for vaccination

Vaccine

Type  
Vaccine

Facility

Hình 16: Vaccination registration Screen

## 5 Conclusion

### 5.1 Short Retrospective

#### 5.1.1 Lesson learned

During our journey through the Telehealth App project, we learned a multitude of essential insights that have significantly influenced our approach to healthcare technology. Direct engagement with end users, including patients and health practitioners, has proven critical. Their feedback has been invaluable in developing the app's functionality, ensuring that the UI is not only user-friendly but also closely linked with the intricacies of healthcare delivery.

The team and I have learned the value of an agile process in project development, which enables us to respond quickly to criticism and new technical options. Furthermore, the project highlighted the importance of data security. We handled the complexities of healthcare regulatory compliance, which was both instructive and practical, establishing in us a strong regard for user data privacy and integrity.

Reflecting on the Telehealth App project, it is clear that it was more than just a software development exercise; it was an invaluable learning experience. It enhanced our belief in the power of collaboration and underlined our commitment to expanding healthcare services using innovative digital technology. This project not only increased our technical skills, but it also expanded our understanding of the critical role that technology plays in the healthcare system.

#### 5.1.2 Limitations in the project

During the development of our Telehealth App, we encountered several professional hurdles. First, interoperability emerged as a serious barrier. As we attempted to link our app with several healthcare infrastructures, the differences in their standards and processes posed a hurdle to seamless adoption. As a result, it became clear that unifying various systems was critical for the app's widespread acceptance in the healthcare sector.

Additionally, the digital divide presented a formidable obstacle. The team recognized that equitable access to technology is not universal, and this gap hindered the app's potential reach. Therefore, it became a priority to devise strategies that would make our service more accessible to those with limited digital resources.

Furthermore, the team observed a variance in user digital literacy, which led to different levels of ease and efficiency in using the app. This variability highlighted the necessity for us to invest in comprehensive user education and develop a robust support system, ensuring that all users could navigate the app with confidence and ease.



## 5.2 Conclusion

Finally, the Telehealth App project has been a diverse activity that has provided valuable insights into the development of digital health solutions. We overcame hurdles such as interoperability and the digital divide, learning vital lessons about the need of flexible, inclusive design and collaborating with varied technical ecosystems.

The experience has highlighted the importance of constant improvement, particularly in user education and assistance, to ensure that the app is accessible and easy for all users.

Reflecting on the project's progress, the team and I have gotten a better understanding of the complexity of healthcare technology. We understand the crucial importance of user-centered development and the influence of digital literacy on software uptake.

Moving forward, these findings will help us strengthen our strategy and develop stronger links between technology and healthcare. This initiative not only increased our technological capabilities, but it also reaffirmed our dedication to bridge the gap between innovation and user demands in the ever-changing field of healthcare technology.

## 6 References

- [1] *WHO Coronavirus (COVID-19) Dashboard*, WHO (COVID-19) Homepage: <https://covid19.who.int/>
- [2] *Budd, J., Miller, B.S., Manning, E.M. et al. Digital technologies in the public-health response to COVID-19. Nat Med 26, 1183–1192 (2020). <https://doi.org/10.1038/s41591-020-1011-4>*