**VIETNAM NATIONAL UNIVERSITY OF HOCHIMINH CITY**

**THE INTERNATIONAL UNIVERSITY**

**SCHOOL OF COMPUTER SCIENCE AND ENGINEERING**

****

A Telehealth Application on Android Platform

By

PHẠM CÔNG TUẤN (ITITIU19060),

PHẠM HOÀNG MINH (ITITIU19031),

ĐỖ MINH HOÀNG (ITITIU17015)

NGUYỄN THIÊN BẢO (ITITIU17035)

Ho Chi Minh City, Vietnam  
2023 – 2024

A Telehealth Application on Android Platform

APPROVED BY:

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ ,   
Le Duy Tan

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_   
(*Typed Committee name here*)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_   
(*Typed Committee name here*)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_   
(*Typed Committee name here*)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_   
(*Typed Committee name here*)

REPORT COMMITTEE   
(Whichever applies)

# ACKNOWLEDGEMENTS

It is with deep gratitude and appreciation that I acknowledge the professional guidance of teacher Le Duy Tan. His constant encouragement and support helped me to achieve my goal. My gratitude goes to the all members of the team, who are Pham Cong Tuan and Pham Hoang Minh, Do Minh Hoang and Nguyen Thien Bao. Their technical help and good humor made thí project become a great learning experience. I am grateful to the faculty of the School of Computer Science of the International University for the support since the beginning of my studies. Gratitude is also expressed to the members of my reading and examination committee.

**TABLE OF CONTENTS**

[ACKNOWLEDGEMENTS 3](#_Toc156175686)

[LIST OF FIGURES 5](#_Toc156175687)

[LIST OF TABLES 6](#_Toc156175688)

[ABSTRACT 7](#_Toc156175689)

[CHAPTER 1: INTRODUCTION 8](#_Toc156175690)

[1.1. Background 8](#_Toc156175691)

[1.2. Problem Statement 8](#_Toc156175692)

[1.3. Scope and Objectives to Solution 9](#_Toc156175693)

[1.4. Report’s Structure 10](#_Toc156175694)

[CHAPTER 2: RELATED WORK 12](#_Toc156175695)

[2.1. VTelehealth 12](#_Toc156175696)

[2.2. VNPT Health 12](#_Toc156175697)

[CHAPTER 3: METHODOLOGY 14](#_Toc156175698)

[3.1. User Requirement Analysis 14](#_Toc156175699)

[3.2. System Design 14](#_Toc156175700)

[3.2.1. Design Architecture 14](#_Toc156175701)

[3.2.2. Database Design 15](#_Toc156175702)

[3.2.3. Application Design 17](#_Toc156175703)

[CHAPTER 4: IMPLEMENTATION AND RESULTS 18](#_Toc156175704)

[4.1. Google Firestore 18](#_Toc156175705)

[4.2. Activity classes 19](#_Toc156175706)

[CHAPTER 5: DISCUSSION AND EVALUATION 23](#_Toc156175707)

[5.1. Discussion 23](#_Toc156175708)

[5.2. Comparison 23](#_Toc156175709)

[5.3. Evaluation 23](#_Toc156175710)

[CHAPTER 6: CONCLUSION 25](#_Toc156175711)

[6.1. Conclusion 25](#_Toc156175712)

[6.2. Future Work 25](#_Toc156175713)

[REFERENCES 26](#_Toc156175714)

# LIST OF FIGURES

***Figure 1***: The structure of the report with six chapters 11

***Figure 2***: The demonstration of the VTelehealth application on Google Play 13

***Figure 3***: The demonstration of the VNPT Health application on Google Play 14

***Figure 4***: The use-case diagram represents the user’s requirement analysis 15

***Figure 5***: The MVVM architecture diagram 16

***Figure 6***: The database architecture drawn by the entity-relationship format 17

***Figure 7***: The data model, DAO, and the repository in our codebase 17

***Figure 8***: The hierarchical structure of these fragment, viewmodel, adapter, and activity classes in our codebase 18

***Figure 9***: The Google Firestore database 20

***Figure 10***: The login and signup feature 21

***Figure 11***: The profile management feature 21

***Figure 12***: The appointment scheduling and live video call feature 22

***Figure 13***: The live messaging between user and doctor feature from the user’s UI 22

***Figure 14***: The administrator interface which involves the doctor’s account creation and appointment approval feature 23

***Figure 15***: The appointment scheduling and live messaging feature from the doctor’s interface 23

# LIST OF TABLES

# ABSTRACT

The advancement of remote healthcare applications is crucial in addressing the challenges posed by time-consuming and inconvenient commutes to healthcare facilities. This type of application is substantially benefited when the commute between the patient’s house and the hospital or any emergency service is frustrating and taking a lot of time. By the request of the Mobile Application Development course, we designed a simple application that support the communication and chatting with the doctor, and booking system. The application leverages the Google Firebase as the remote database system, along with the Agora SDK to support these base business functionalities.

# INTRODUCTION

## Background

In the second decade of the 21st century, we have witnessed a shift in healthcare, driven by rapid technological advancements. Remote healthcare, more commonly known as telehealth, has emerged as a critical component of modern medical practice, especially the accessibility to healthcare is a global concern. A common example of the difficulty is that most people do not store the hospital’s contact as a daily routine, and communication in the emergency is less efficient, leading to a misunderstanding of the patient's scenario and resulting in them being in a danger zone. The advancement of digital technology, especially the online communication platform, robust data management systems, and real-time communication platforms has facilitated the development of telehealth applications that promise to bridge the gap between healthcare providers and patients, especially in remote or underserved areas. While the integration of technology in healthcare is not a new solution, we expect this to have a more inclusive healthcare system.

## Problem Statement

While the telehealth system has shown great promise, its full potential is yet to be realized, especially when the downside of the application should not be underrated such as complex user interfaces, inefficient appointment scheduling processes, inadequate communication channels between patients and healthcare providers, and along with the integration between the client-oriented (mobile) system and server-oriented system. One of the significant challenges is the inefficiency of appointment scheduling processes. Many telehealth platforms lack a streamlined approach to booking and managing appointments, resulting in confusion, time wastage, and often, missed appointments. Additionally, the communication channels sometimes raise concerns about the confidentiality and privacy of sensitive medical information. Furthermore, there is a gap in the integration of these systems with broader healthcare infrastructures. Many telehealth solutions operate in isolation, without proper synchronization with other medical records or systems used by healthcare providers. For example, the development of the hospital software, which is usually operated on a computer or laptop, is different from the mobile system which focuses more on the user experience, convenience, and data security, resulting in fragmented care, under-utilized resources, and money for the hospital. Another point from the hospital perspective is that the hospital cannot charge or profit the patient based on their usage of the application but their health only, while they still need a local or third-party team to maintain their system.

In general, the lack of a streamlined, easy-to-use telehealth system poses a significant barrier to the adoption of remote healthcare services, thus impeding the goal of achieving universal healthcare accessibility. By this, a simple, efficient, and user-centric telehealth application is not just a great opportunity and challenge to address these shortcomings, but a necessary step to pave the way for a telehealth solution that is accessible, reliable, and seamlessly integrated into the broader healthcare ecosystem.

## Scope and Objectives to Solution

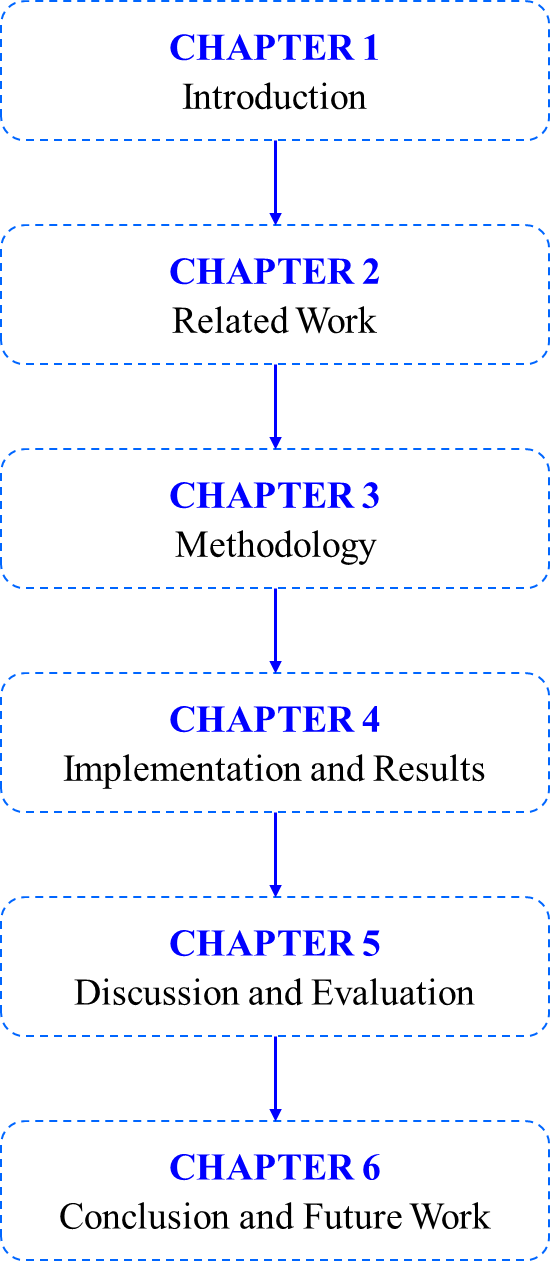
Recall from the previous section, that our scope is to demonstrate a simple way to bring the idea of a telehealth system online without incurring too much cost. This study is focused on the technological aspects of developing a telehealth application, showing that there are only several functionalities that the user wants. Specifically, the objectives are

* **Developing a User-Friendly Interface**: This is one of the uttermost important factors in having a diverse user base, including those with limited technological skills. The aim is to simplify the navigation and operation of the application, making telehealth services more accessible and less intimidating to the average user. But to achieve this, unless the team is comprised of high-tech engineers with proper skill set and service mindset, it would require
* **Minimal Business Functionalities**: This helps to simplify the unnecessary information or functionalities that are not or rarely used by the user such as the communication between patients. This example would bring great annoyance to patients when not only being exploited to scam the patients but also the mental privacy and insecure data protection; whilst the social media platforms are well-handed on these types of scenarios. By this, we reduced the number of functionalities into two which are
  + **Efficient Appointment Management**: The system should efficiently manage appointments, reducing the likelihood of errors and miscommunications that can lead to missed appointments. But to prevent spammers, the system would require medical-operation experts to accept the appointment, which would also utilize and balance the medical system resources and make the platform manageable.
  + **Secure and High-Quality Communication**: The application will prioritize establishing a secure and seamless communication channel, involving high-quality video and audio capabilities with stable connectivity. However, to ensure the best treatment the patient could receive while adapting to the production-grade medical standard, the doctors must have passed an online advisor course, which helps the patients how to use the medicine properly and diagnose any adversarial symptoms without concluding the sick.

In this study, we focused on the technological aspects of the solution where we developed a telehealth system that highlights the use of Google Firebase and Agora SDK in crafting a solution. The solution would detailed exploration of the design process and technical development of the application, but acknowledge its limitations towards existing healthcare systems in real-world healthcare settings.

## Report’s Structure

This report consists of six chapters in which the first chapter is an overview of the real-world consequence, the motivation, and the proposed solution of this report. The second chapter is a literature review of some existing related works found on Google Play. The third chapter presents how methods and approaches used in the development of the telehealth application are described, including the technical specifications, design principles, and development tools. The fourth chapter will present the demo and instructions for our initial deployment and testing. The fifth and sixth chapters finalize the report by discussing the topic, the validity of the application, and the development direction of the report.

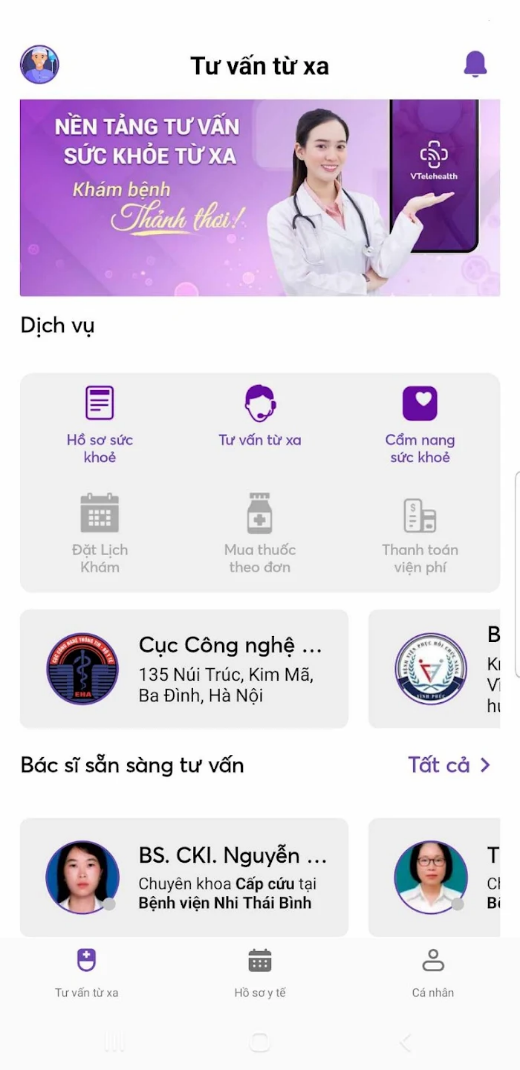
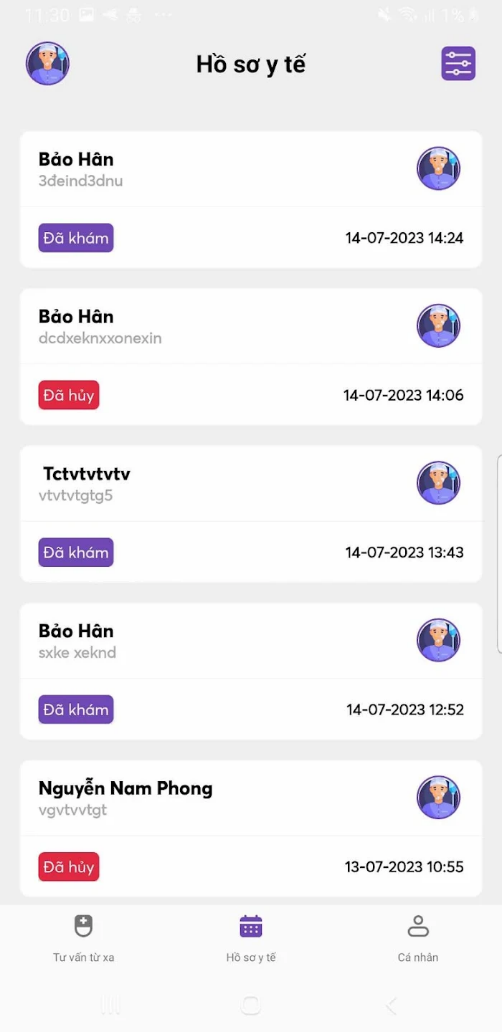


***Figure 1***: The structure of the report with six chapters.

# RELATED WORK

## VTelehealth

VTelehealth is also a telehealth system that is developed recently by the Vietnamese Ministry of Health. In general, the application supports the health profile and live communication, along with several integrations. Similarly, our application works on a small scale and support the live video communication as an additional feature.

***Figure 2***: The demonstration of the VTelehealth application on Google Play.

## VNPT Health

Another application is the VNPT Health which is developed the VNPT company. However, the application is developed under the COVID-19 pandemic which adds additional support to monitor the user’s health such as body mass index (BMI), breathing, along with the report of the number of Vietnamese patients who infected by the Corona virus.

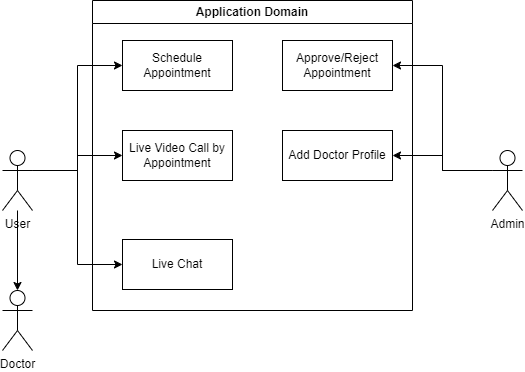


***Figure 3:*** The demonstration of the VNPT Health application on Google Play.

# METHODOLOGY

## User Requirement Analysis

As mentioned in Section 1.3, the two main functionalities related to the appointment scheduling and live communication is splitted into this use-case diagram. From the perspective of the application domain, the user and doctor share similar functional usecase, as opposed to the administrator. The user and doctor can schedule the appointment and live video call on the accepted appointment from the administrator. They can also perform live messaging, a new functionality that enable direct communication between users and doctor. On the other hand, the administrator would support the user to either approve or reject the appointment, and create the doctor’s profile, which adds another management layer. The doctor’s profile can be seen when the user wants to perform live communication with any doctors provided by the application. The detailed diagram can be displayed on *Figure 4*.

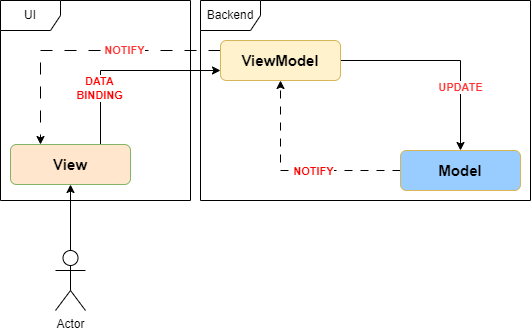


**Figure 4**: The use-case diagram represents the user’s requirement analysis

## System Design

### Design Architecture

In this section, we will discuss how our application delivered. By our nature, our application involves separate parts following the MVVM design patten (model-view-viewmodel). The overall architecture can be shown by the below *figure 5*, which is adapted from the ByteByteGo channel [1]. Our application consists of three separate components: the **view** which is responsible to receive the operation of the UI from the user, the **model** which contains the reference to the database connection and control over the data logic, and the view model to control how the data is managed, queried, and used to adapt with the business and control logic.



**Figure 5**: The MVVM architecture diagram

### Database Design

From the database perspective, our simple usecase can be interpreted by these four tables under the following entity-relationship diagram in *Figure 6*. The *Profile* and *Doctor* table has the ID stored as the varchar datatype which associates with the user’s and doctor’s information. Meanwhile, the appointment scheduling requires another table (*Appointment*) which is to store the datetime field to record the date and time of the meeting, and the status field which is to allow the administrator to either accept or reject this request. Another functionality, the live chat, which also has the *Chat* table to store the user ID as the credentials to communicate with the Agora SDK. Those IDs on the four table are automatically managed by the system and it would be an hidden field from the application developer’s perspective.

A diagram of a data flow

Description automatically generated with medium confidence

**Figure 6**: The database architecture drawn by the entity-relationship format.

From the coding packages, these tables can be found under the *data* package (*Figure 7*), which contains the DAO (data access object) to define centralized operations and SQL query over the database, the dataclass or data model to represent a row of data in the database server, and the repository to contains the database’s instance and methods for executing the SQL queries defined by the respective DAO.

A screenshot of a computer

Description automatically generated A screenshot of a computer program

Description automatically generated

A screenshot of a computer program

Description automatically generated

**Figure 7**: The data model, DAO, and the repository in our codebase.

### Application Design

From the application’s perspective, we have these *fragment* classes which contain the logic code for features, listen for user interaction and update the UI, the *viewmodel* classes which manage data received from the repository such as the mutation of live data or the asynchronous programming that requests the data from the repository (database). In each feature’s fragment, we initialize the instance of revelant viewmodels to get and manage the data. Moving onto a higher level design, we meet some *adapter* classes which is used for recycler view and dropdown as they match a list of data to the list of UI items in the dropdown/scrolling list. Finally all of the fragment and adapters would be redirected to the *activity* classes as a whole operation. The hierarchical structure of these classes are displayed on *Figure 8*.

A screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generated

**Figure 8**: The hierarchical structure of these fragment, viewmodel, adapter, and activity classes in our codebase.

# IMPLEMENTATION AND RESULTS

## Google Firestore

At the beginning of the project we intend to combine the SQLite database hosted on a specific website, and leverage the idea of transaction log to proceed the database. But since that idea is too complex, we switched that platform into the Google Firestore database. The user interface of how the database is managed is displayed under *Figure 9*.

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

**Figure 9**: The Google Firestore database.

## Activity classes

The development of the application requires five different activities, and these are the screenshots of those features including the login/signup (*Figure 10*), the user’s profile management (*Figure 11*), the appointment scheduling and live video call feature (*Figure 12*), the live messaging between user and doctor feature (*Figures 13 and 15*), and the doctor’s account creation and appointment approval feature from the administrator perspective (*Figure 14*).

A screenshot of a cell phone

Description automatically generatedA screenshot of a phone

Description automatically generated

**Figure 10**: The login and signup feature.

A screenshot of a phone

Description automatically generatedA screenshot of a phone

Description automatically generated

**Figure 11**: The profile management feature.

A screenshot of a phone

Description automatically generated A screen shot of a cell phone

Description automatically generated

**Figure 12**: The appointment scheduling and live video call feature.

A screenshot of a phone

Description automatically generated A screenshot of a phone

Description automatically generated

**Figure 13**: The live messaging between user and doctor feature from the user’s UI.

A screenshot of a phone

Description automatically generatedA screenshot of a phone

Description automatically generated

**Figure 14**: The administrator interface which involves the doctor’s account creation and appointment approval feature.

A screenshot of a phone

Description automatically generatedA screenshot of a phone

Description automatically generated

**Figure 15**: The appointment scheduling and live messaging feature from the doctor’s interface.

# DISCUSSION AND EVALUATION

## Discussion

The discussion section of the telehealth application report delves into the findings, implications, and broader context of the project. It provides a comprehensive analysis of the telehealth application's performance, addressing key aspects such as usability, effectiveness, and impact on healthcare outcomes. The discussion explores how the application aligns with the initial objectives outlined in the project, highlighting any deviations or unexpected outcomes. Moreover, this section facilitates the interpretation of results in relation to existing literature, emphasizing the contributions of the telehealth application to the field. It delves into potential limitations, offering insights into areas that may require further refinement or investigation in future iterations. Engaging in a dialogue about the implications of the project's outcomes on the broader healthcare landscape, policy considerations, and potential scalability is also a crucial component of the discussion.

## Comparison

In the comparison section, the telehealth application report evaluates the performance and features of the developed application against existing telehealth solutions or industry standards. Comparative analysis provides valuable insights into the uniqueness and competitive edge of the developed application. It may involve benchmarking against similar applications in terms of user interface, functionality, security protocols, and overall user experience. Furthermore, a comparative examination might extend to the contextual landscape, comparing the telehealth application's implementation in different healthcare settings or regions. This section aims to identify best practices, lessons learned from other telehealth initiatives, and areas where the developed application excels or requires enhancement in comparison to its counterparts.

## Evaluation

The evaluation section consolidates the overall assessment of the telehealth application, drawing on the discussion and comparison elements. It provides a summative analysis of the project's success in achieving its goals and objectives. The evaluation encompasses both quantitative and qualitative metrics, measuring the application's impact on healthcare accessibility, efficiency, and user satisfaction. Critical evaluation involves addressing challenges encountered during the development and implementation phases, offering insights into the strategies employed to overcome these obstacles. It also considers the potential long-term sustainability and scalability of the telehealth application, taking into account evolving technologies, regulatory changes, and user needs. This section concludes with actionable recommendations for future enhancements or adaptations, ensuring that the telehealth application continues to evolve in response to the dynamic healthcare landscape.

# CONCLUSION

## Conclusion

In conclusion, the initial phase of developing the telehealth application, focusing on login and appointment scheduling functionalities, marks a significant stride towards fostering accessible and patient-centric healthcare delivery. The successful implementation of these foundational features lays the groundwork for a more comprehensive telehealth ecosystem. The user-friendly login interface and streamlined appointment scheduling process contribute to an efficient and convenient experience for both healthcare providers and patients. However, it is crucial to acknowledge the limited scope of the current iteration and recognize that the full potential of telehealth is yet to be realized. While the implemented features form a robust foundation, the telehealth application is poised for expansion and augmentation to encompass a broader range of functionalities in subsequent phases.

## Future Work

Moving forward, the telehealth application project must evolve to meet the escalating demands and expectations of modern healthcare. Future work will involve the integration of additional features, such as video consultations, secure messaging, and real-time health monitoring. These enhancements aim to facilitate more comprehensive and personalized healthcare interactions, transcending the limitations of traditional in-person consultations. Furthermore, addressing interoperability challenges with existing electronic health record (EHR) systems and ensuring compliance with evolving healthcare regulations will be pivotal. Collaborations with healthcare professionals, user feedback sessions, and iterative testing will be essential in refining the application's usability and functionality. The expansion of the telehealth application also necessitates considerations for scalability and adaptability to diverse healthcare settings. As the project progresses, exploring partnerships with healthcare institutions and incorporating telehealth into broader healthcare infrastructure will be imperative for widespread adoption.

# REFERENCES

1. ByteByteGo. (2024, January 9). *Everything You NEED to Know About Client Architecture Patterns* [Video]. YouTube. <https://www.youtube.com/watch?v=I5c7fBgvkNY>