THU Bridge Mobile Application

A Project Report
Presented to
The Faculty of the College of
Engineering

San Jose State University
In Partial Fulfillment
Of the Requirements for the Degree
Master of Science in Software Engineering

By

Chirag Arora
Jigar Jigneshbhai Shah
Sai Prashanth Guthula
Sai Vamsi Dutt Patibandla

05/2024

APPROVED

Gheorghe Guzun, Project Advisor

Dan Harkey, Director, MS Software Engineering

Rod Fatoohi, Department Chair

ABSTRACT

THU Bridge Mobile Application

By
Chirag Arora
Sai Prashanth Guthula
Sai Vamsi Dutt Patibandla
Jigar Jigneshbhai Shah

The THU Bridge mobile application is a solution to the communication problem in healthcare. The application bridges language gaps between patients, interpreters, and administrators. The app is user-friendly and secure, and it benefits patients, interpreters, and administrators in several ways: It enhances communication between patients and healthcare providers, It improves healthcare accessibility for patients who cannot communicate effectively with their healthcare providers due to language barriers, It reduces costs associated with healthcare interpretation services, It elevates levels of patient satisfaction. The THU Bridge mobile application caters to three distinct user roles: patients, interpreters, and administrators. Patients: Patients can use the app to request interpreter services, access appointment schedules, and communicate with interpreters. Interpreters: Interpreters can use the app to accept or decline interpreting service requests, view appointment schedules, and communicate with patients. Administrators: Administrators can use the app to manage interpreters, oversee appointment schedules, and track app usage. In the future, the THU Bridge application will connect patients with healthcare providers and volunteers across borders. Telehelp

Ukraine has 120 providers, 40-50 active weekly, and 60-80 volunteers. The goal of the collaboration is to expand the reach of both initiatives and connect patients globally with healthcare providers and volunteers. This will improve healthcare accessibility and break down language barriers even further. The THU Bridge app and Telehelp Ukraine are innovative solutions to healthcare communication challenges. By collaborating and leveraging technology, they can enhance healthcare accessibility and improve patient outcomes and satisfaction globally.

Acknowledgments

We are deeply indebted to Professor Gheorghi Guzun for his invaluable comments and assistance in the preparation of this study. His guidance has been a beacon, steering us through complex theoretical frameworks and practical applications with ease and clarity. We are also immensely grateful to our classmates and team members for their dedicated collaboration, diverse perspectives, and tireless efforts in ensuring the success of this project. Their commitment to excellence, creative problem-solving, and ability to work under challenging circumstances have been inspiring.

Table of Contents

List of Figures

Chapter 1. Project Overview

1.1 Introduction

The importance of mobile apps for public health was highlighted by the COVID-19 pandemic. These apps had to strike a difficult balance between individual privacy concerns and public health surveillance, which was crucial for tracking and spreading information about the infection. These apps make use of Bluetooth and GPS technology to find potential viral exposures and give users instructions. However, there are serious privacy concerns with the collection and use of sensitive health data. Finding the ideal balance between efficient disease surveillance and preserving people's right to privacy continues to be a major obstacle in the creation of such apps.

THU Bridge Mobile App represents a pioneering endeavor in healthcare communication. At its core, THU Bridge is designed to provide an efficient, user-friendly platform that bridges the communication gap among patients, interpreters, and healthcare administrators.

1.2 Proposed Areas of Study and Academic Contribution

The THU Bridge app and its partnership with Telehelp Ukraine are transforming healthcare communication by bridging language gaps, improving accessibility, reducing costs, and elevating patient satisfaction.

1. Bridging Language Barriers: The THU Bridge app is a powerful solution for bridging language gaps within the healthcare sector. It directly tackles the challenge of effective communication among patients, interpreters, and administrators. By facilitating seamless communication, the application contributes significantly to overcoming linguistic barriers that often hinder healthcare interactions.

2. Improved Healthcare Accessibility: A key aspect of the project's justification is its role in improving healthcare accessibility. The THU Bridge app acts as a gateway to better healthcare services for patients who struggle to communicate with healthcare providers due to language barriers. It ensures that individuals from diverse linguistic backgrounds can access the care they need without hindrance.

3. Cost Reduction: The project also addresses the economic aspect of healthcare communication. By streamlining interpreter services and offering a user-friendly

platform, the application has the potential to reduce the costs associated with traditional interpretation services. This cost reduction benefits both healthcare facilities and patients.

4. Elevated Patient Satisfaction: Patient satisfaction is a crucial metric in healthcare. The THU Bridge app directly contributes to higher levels of patient satisfaction by removing language-related frustrations and ensuring that patients can effectively express their needs and concerns.

5. Global Expansion: The project's forward-looking approach envisions global connectivity between patients and healthcare providers. Collaborating with Telehelp Ukraine further strengthens this initiative, promising a broader impact and breaking down language barriers worldwide.

6. Innovation in Healthcare Communication: The THU Bridge app and its collaboration with Telehelp Ukraine represent innovative solutions to longstanding healthcare communication challenges. By harnessing technology and fostering collaboration, these initiatives have the power to transform the way healthcare is accessed and delivered.

1.3 Current State-of-the-Art

THU Bridge Mobile App represents a pioneering endeavor in healthcare communication. At its core, THU Bridge is designed to provide an efficient, user-friendly platform that bridges the communication gap among patients, interpreters, and healthcare administrators.

This ambitious undertaking utilizes state-of-the-art technologies and innovative methodologies to address the complex challenges within the healthcare communication landscape. Notably, the project is built upon a Flask backend deployed on the Heroku platform, which ensures a robust and scalable infrastructure. Coupled with a PostgreSQL database, this framework is meticulously designed to manage healthcare information securely and efficiently.

The commitment to excellence is underscored by comprehensive documentation that sheds light on the intricacies of the project. Moreover, the creation of a Product Requirements Document (PRD) tailored for Quality Assurance Engineers demonstrates a dedication to upholding high-quality standards. The integration of Continuous Integration/Continuous Deployment (CI/CD) workflows streamlines development, enabling swift and efficient updates and enhancements for users.

What sets THU Bridge apart is its unyielding focus on eliminating language barriers. Through innovative technological solutions and a user-centric approach, the project ensures that patients from diverse linguistic backgrounds can access healthcare services without hindrance. This forward-looking project aims to transform healthcare communication practices and outcomes, promising a brighter and more accessible future for patients, interpreters, and administrators alike.

Chapter 2. Project Architecture

2.1 Introduction

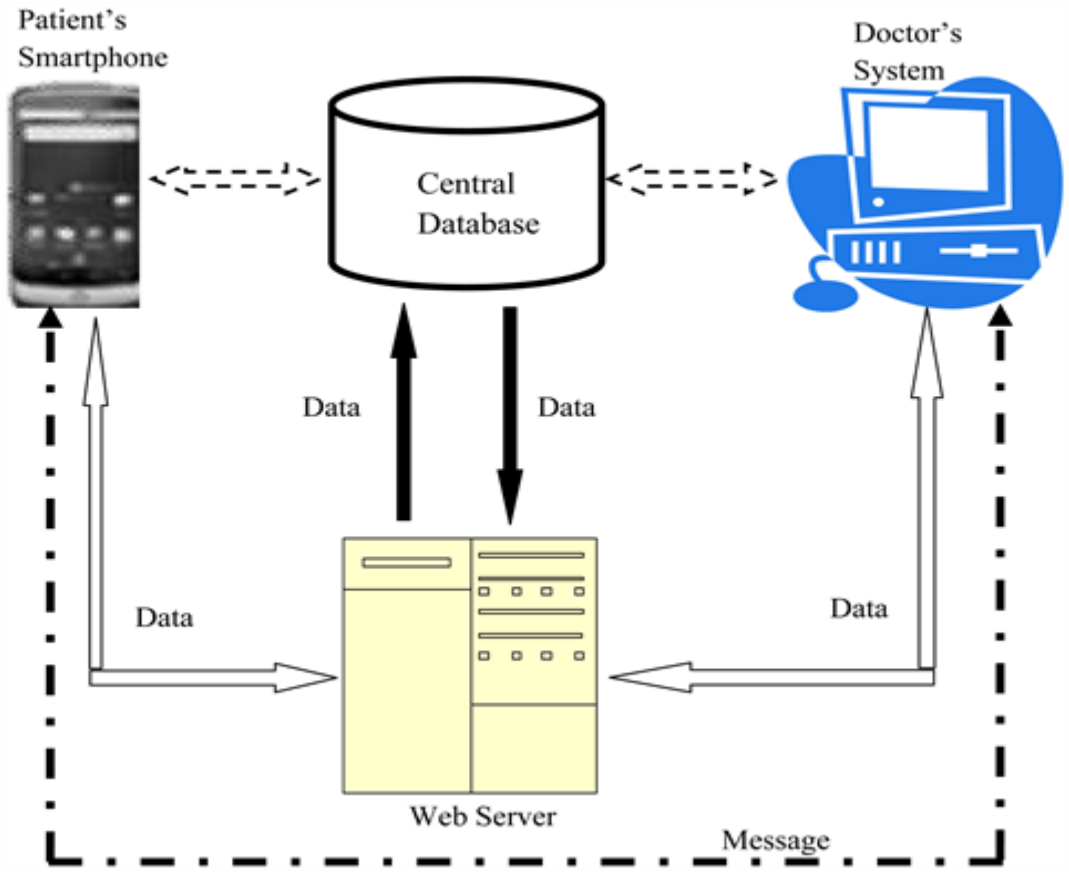


Fig1. Basic Architecture

The TeleHelp Ukraine Android app is designed to facilitate healthcare appointments and consultations. The project architecture follows modern software engineering principles, focusing on modularity, scalability, and clean separation of concerns.

Flow Chart: Patient as User

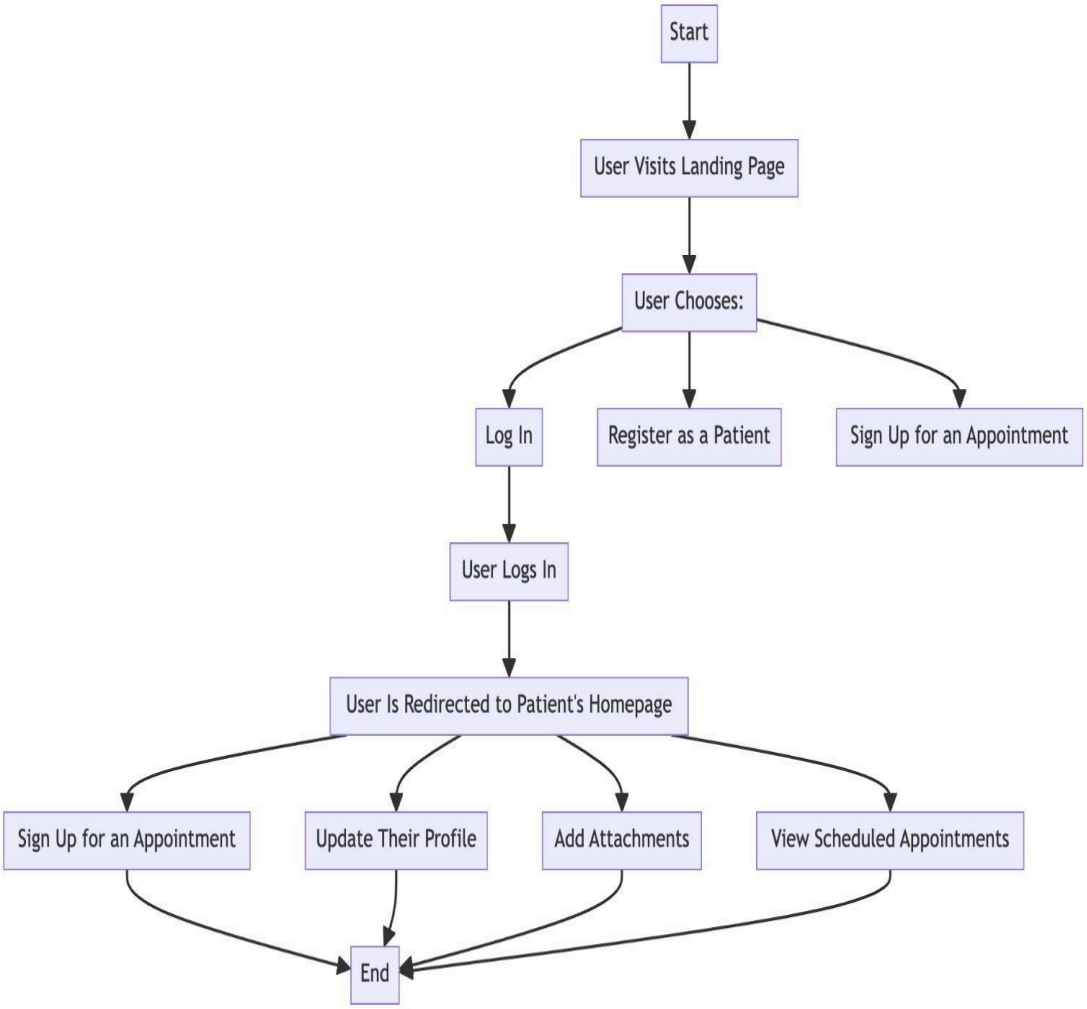


Fig 2. Flowchart of patient

2.2 Architecture Subsystems

Presentation Layer

The presentation layer should be designed to be user-friendly and accessible to users with disabilities. The presentation layer should also be designed to be flexible and extensible so that new features can be added easily in the future.

Business Logic Layer

The business logic layer should contain the core functionality of the app, such as scheduling appointments, managing patient data, and sending feedback. 19 The business logic layer should also be designed to be modular and reusable so that it can be used in other projects if needed.

Data Access Layer

The data access layer should be responsible for interacting with the app's data sources, such as the Telehelp Ukraine backend API. The data access layer should also be designed to be abstract so that it can be easily switched to a different data source in the future.

Communication between Layers. The presentation layer should communicate with the business logic layer using a well-defined interface. This will make the code more modular and easier to maintain. The business logic layer should communicate with the

data access layer using a well-defined interface. This will make the code more abstract and easier to test.

Security

All communication between the app and the backend server should be encrypted using TLS. The backend server should also use authentication and authorization mechanisms to protect user data. The app should also be designed to be resilient to common security attacks, such as SQL injection and cross-site scripting.

Scalability

The app should be designed to be scalable so that it can handle many users and transactions. The use of cloud-based services, such as App Engine and Cloud SQL, will help to make the app more scalable.

Availability

The app should be designed to be highly available so that it is always available to users. The use of cloud-based services, such as App Engine and Cloud Load Balancing, will help to make the app more highly available.

App Deployment

Prepares the app for deployment on the Google Play Store, ensuring it meets the platform's guidelines and requirements.

Chapter 3. Technology Descriptions

3.1 Client Technologies

In developing the THUAPP mobile application, we select client technologies that ensure robust performance, security, and user-friendly interfaces. The application is designed to bridge communication gaps in healthcare, facilitating efficient interactions between patients, healthcare providers, and administrative staff. This section outlines the key technologies employed on the client side of the THUAPP.

Frontend Framework

React Native: We chose React Native for its cross-platform capabilities, allowing us to build a mobile app that performs efficiently on both Android and iOS with a single codebase. React Native's component-based architecture makes it highly reusable, which accelerates development cycles and reduces the chances of bugs. Its vast community support and continuous updates make it an ideal choice for our project.

State Management

Redux: To manage the application’s state predictably, we integrate Redux with React Native. Redux provides a centralized store for all the state that needs to be shared within the app, enabling us to keep different parts of the app in sync. It helps in managing the

states of various forms and user sessions, making the user experience smooth and consistent.

Navigation

React Navigation: This is the standard for navigating between screens within a React Native app. React Navigation is flexible and fully customizable, supporting various types of navigation patterns, such as stack, tab, and drawer navigations. This is crucial for THUAPP, where users need to navigate through multiple layers of content efficiently.

API Integration

Axios: For network calls to our backend services, Axios is used due to its wide compatibility with modern web browsers and its promise-based architecture which makes handling asynchronous requests straightforward. Axios enhances the development of features like fetching doctor details, scheduling appointments, and managing user profiles.

User Interface

Material-UI: Leveraging Material-UI components provides a set of accessible and customizable components, that conforms to Google's Material Design guidelines. This not only speeds up the development process due to the availability of pre-built elements but also ensures that the app remains visually appealing and functionally rich.

Local Storage

AsyncStorage: React Native's AsyncStorage is utilized to store user preferences and session states locally. This is essential for maintaining a seamless user experience, particularly for functions such as auto-login and language settings.

Authentication

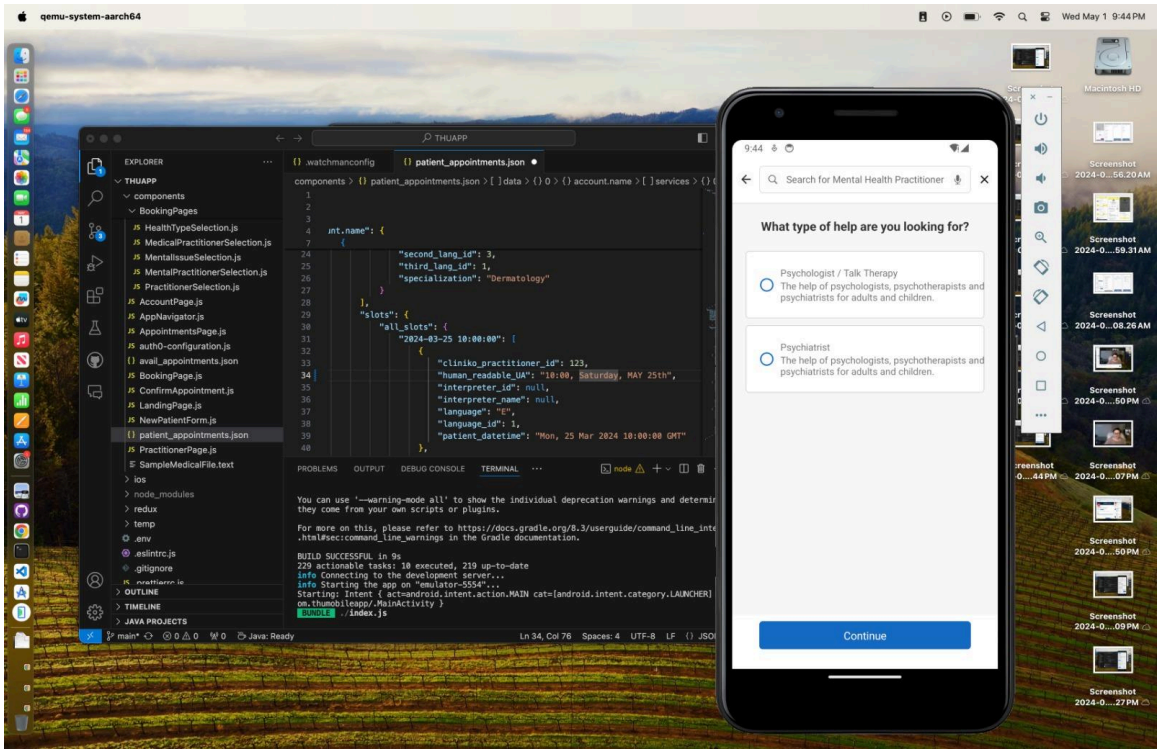
Auth0: To secure and simplify the authentication process, Auth0 is integrated into THUAPP. It supports advanced features like social login, multi-factor authentication, and user management, which are imperative for maintaining high security and data privacy standards in healthcare applications.

Testing Framework

Jest and Enzyme: For unit and integration testing, Jest is used in conjunction with Enzyme. Jest provides a robust testing framework with a focus on simplicity, supporting both JavaScript and TypeScript. Enzyme allows for more granular testing of React components, ensuring each component behaves as expected independently and within complex UIs.

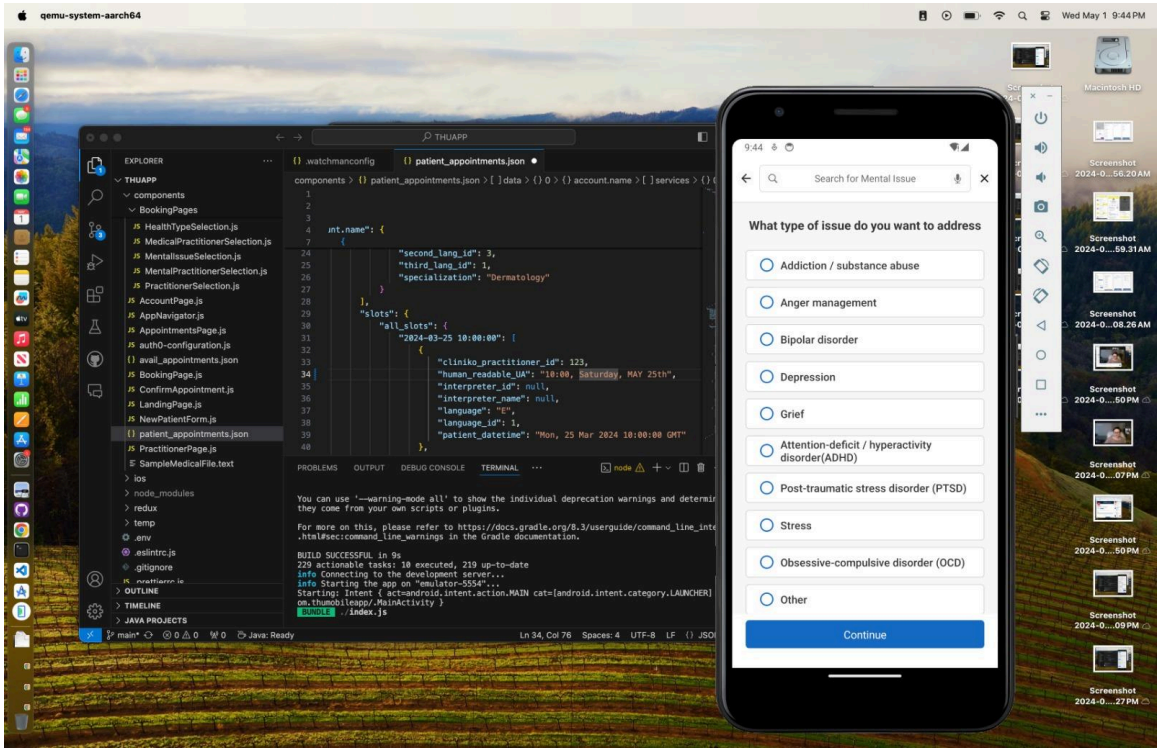
Chapter 4. Project Design

4.1 UI Design



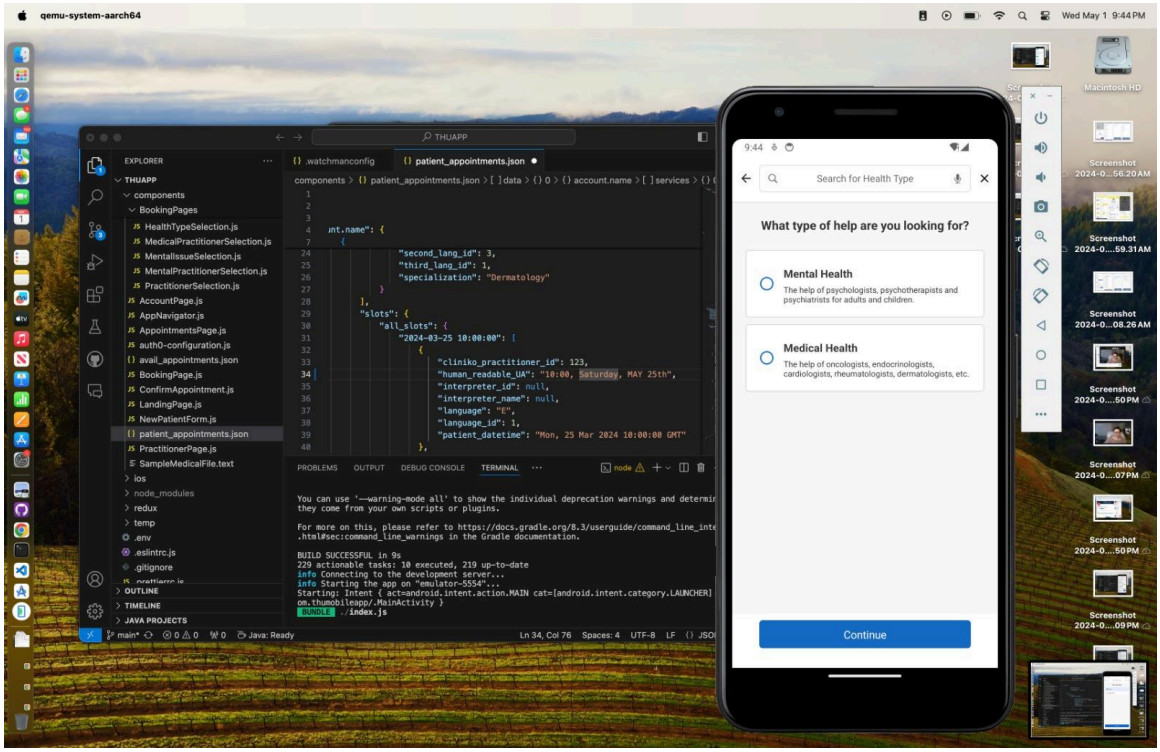
This image depicts a software development environment with a focus on mobile application development for healthcare services. On the left side, the screen shows the code editor with files related to the "THUAPP" project, indicating different components for the app like "HealthTypeSelection.js" and "MentalIssueSelection.js." The right side of the image displays a mobile phone emulator running a healthcare app interface where a user can select between different types of mental health professionals, such as

"Psychologist / Talk Therapy" and "Psychiatrist." The interface is user-friendly and designed to facilitate easy access to healthcare professionals for mental health issues.



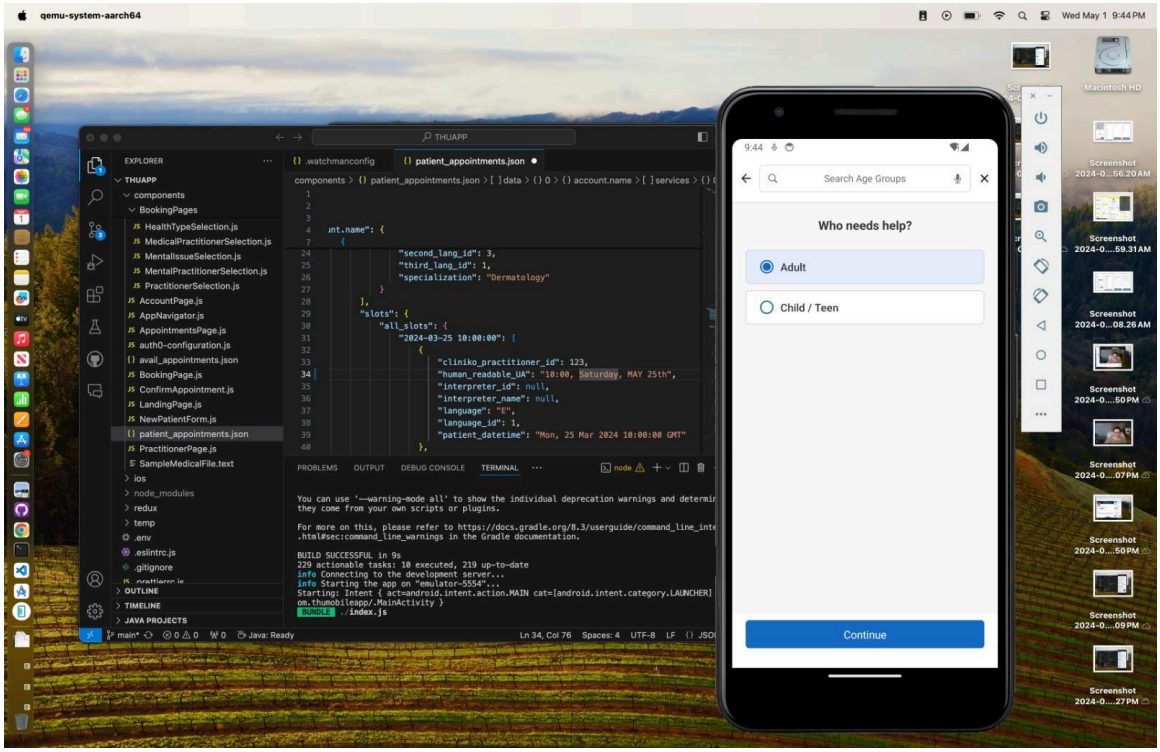
This image showcases a mobile application development environment aimed at addressing mental health issues. The left side of the screen displays a code editor with project files, highlighting components like "HealthTypeSelection.js" and "MentalIssueSelection.js" within the "THUAPP" project directory. On the right, a mobile phone emulator presents an interface from a mental health services app, where users can select specific mental health issues they wish to address, including options like "Addiction/substance abuse," "Depression," and "Post-traumatic stress disorder (PTSD)."

This interface helps users easily navigate to the appropriate services they need for their mental health concerns.

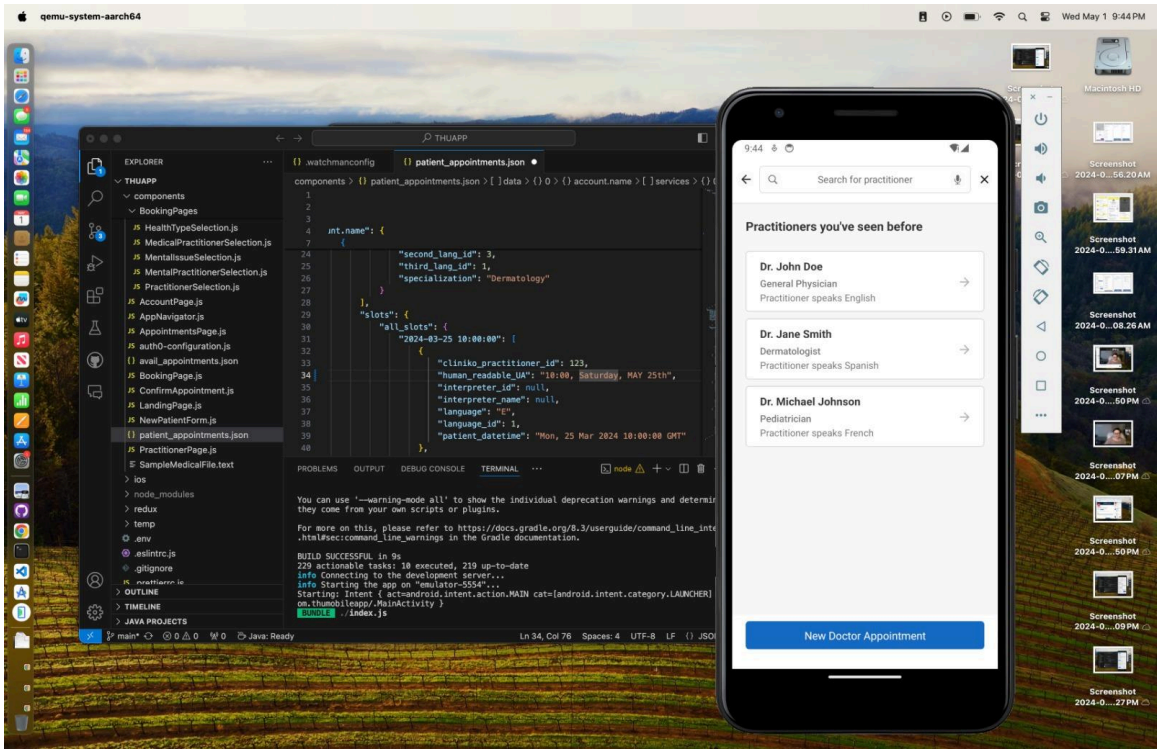


This image presents a development environment for a healthcare mobile application, featuring an emulator and source code editor. On the right, the emulator displays a user interface where individuals can choose between two main categories of health services: "Mental Health" and "Medical Health." The "Mental Health" option suggests services from psychologists, psychotherapists, and psychiatrists, while the "Medical Health" category covers a broad range of medical specialties such as oncologists, cardiologists,

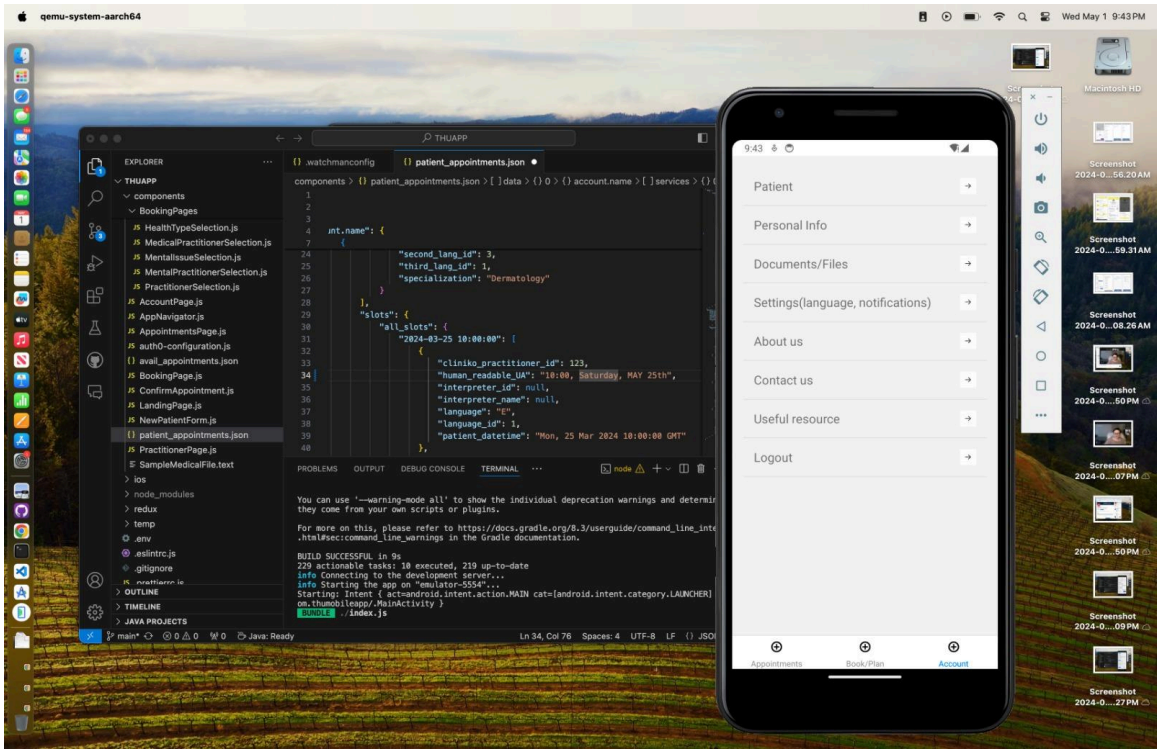
and dermatologists. This allows users to specify the type of healthcare professional they need, streamlining access to appropriate medical services.



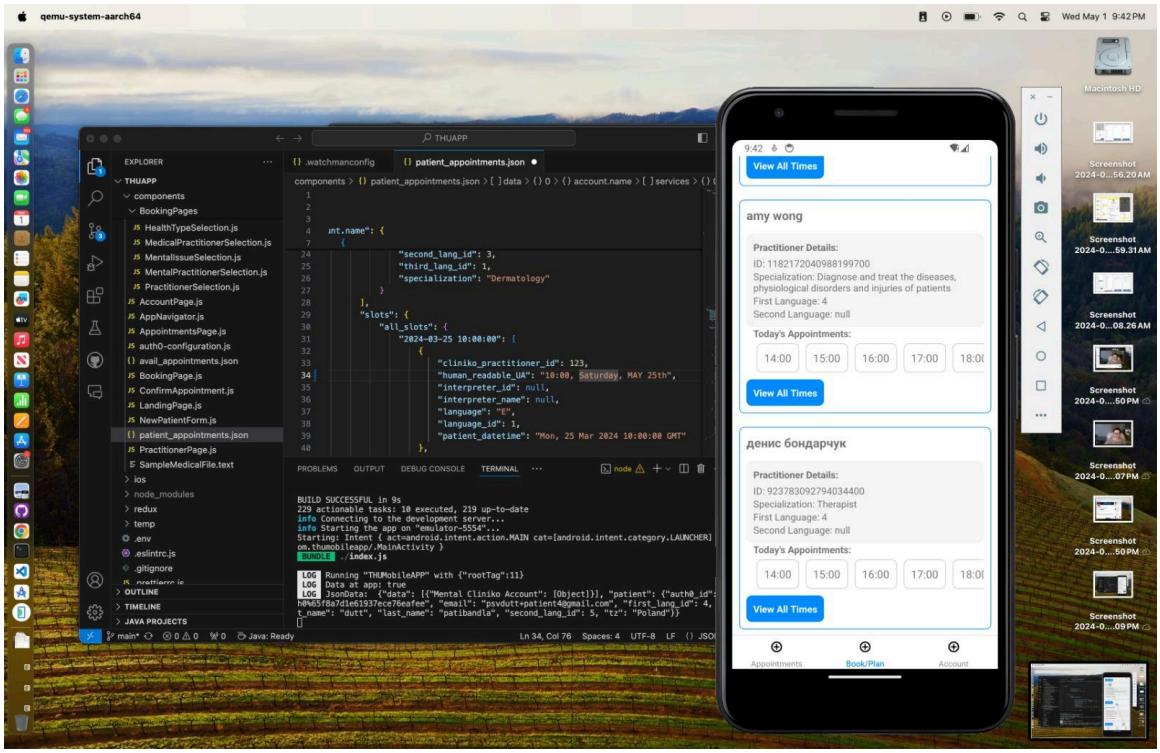
This image showcases a software development environment focusing on a mobile application designed to help users select healthcare services according to specific demographic needs. On the right side, the mobile phone emulator displays a user interface for choosing the age group of the individual needing help, with options like "Adult" and "Child / Teen." This selection helps to tailor the healthcare services and recommendations to the appropriate age group, ensuring that users receive age-appropriate care. On the left, the code editor is open to files within the "THUAPP" project, hinting at the development process for this functionality.



This image features a development environment for a mobile healthcare application, showing both a code editor and a mobile phone emulator. On the right side, the emulator displays a screen for selecting medical practitioners that a user has seen before, including doctors like "Dr. John Doe" (General Physician, speaks English), "Dr. Jane Smith" (Dermatologist, speaks Spanish), and "Dr. Michael Johnson" (Pediatrician, speaks French). This interface facilitates easy access and booking of follow-up appointments with healthcare providers known to the patient, enhancing user convenience and care continuity. The left side shows project files within the "THUAPP" project, indicating active development work on components like "PractitionerSelection.js," which likely manages the logic for this feature.

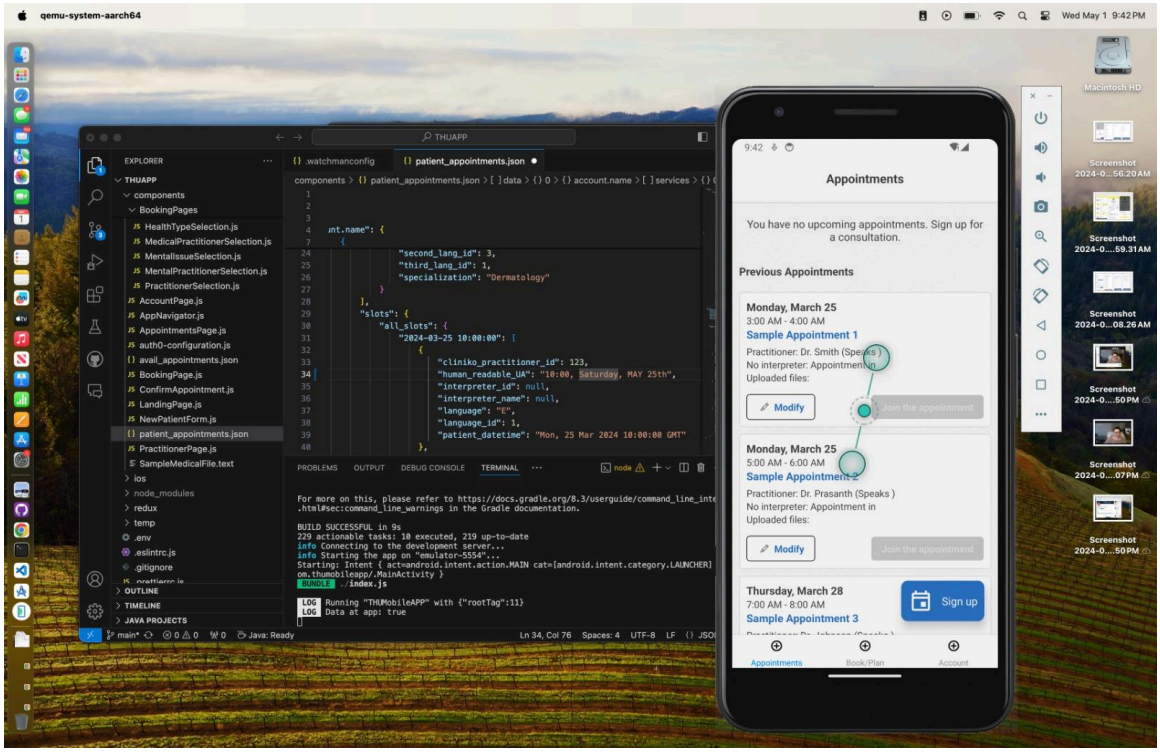


This image features a mobile application development setup for managing patient accounts within a healthcare app. On the right side, the mobile phone emulator displays an interface where patients can access various account-related options such as "Personal Info," "Documents/Files," "Settings (language, notifications)," "About us," "Contact us," "Useful resource," and "Logout." This menu structure helps patients easily navigate and manage their personal and medical information within the app. The left side shows the code editor, highlighting the active development of files like "AccountPage.js" within the "THUAPP" project, suggesting ongoing enhancements to the account management features of the application.



This image showcases a mobile application development environment focused on scheduling appointments within a healthcare app. The emulator on the right displays a user interface where patients can view available appointment times for selected practitioners, helping streamline the booking process. Each practitioner's details are listed, including specialization, languages spoken, and available times, providing comprehensive information at a glance. The interface features practitioner names such as "Amy Wong" and "Денис Бондарчук," reflecting a multilingual setup catering to diverse patient needs. The left side of the image shows the code editor with files related to the "THUAPP" project, suggesting active development on features like

"AppointmentsPage.js" to enhance the appointment scheduling functionality of the application.



This image features a software development environment for a healthcare mobile application focusing on appointment management. The emulator on the right side of the screen displays an interface for managing appointments, showing a section for "Previous Appointments" with options to modify or join the appointment. Each listed appointment provides details about the practitioner, time, and the option to upload files, enhancing patient interaction and record-keeping. This screen also offers a button to "Sign up" for new appointments, emphasizing user engagement and easy access to healthcare services. On the left, the code editor is open to files related to the "THUAPP" project, indicating

active development work on features like "AppointmentsPage.js" to manage the appointments functionality effectively.

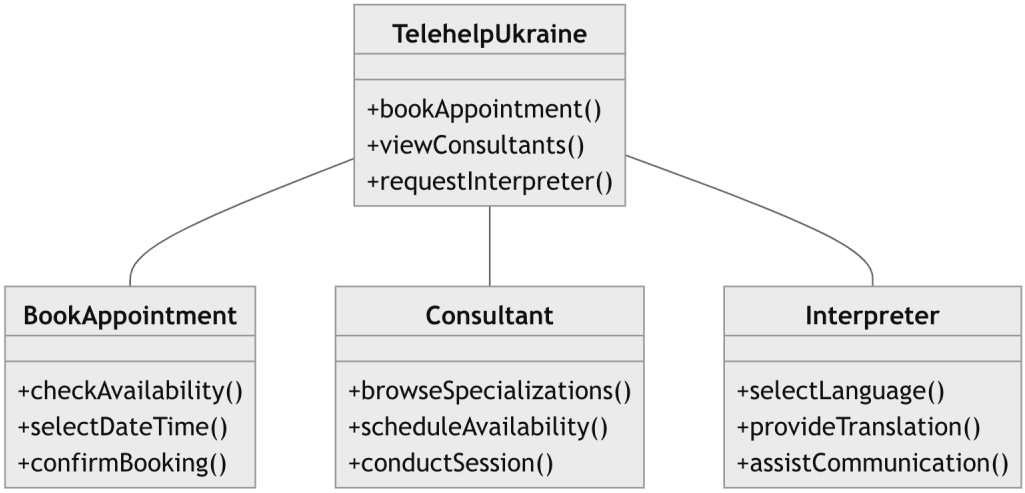


Fig.4 UML Diagram

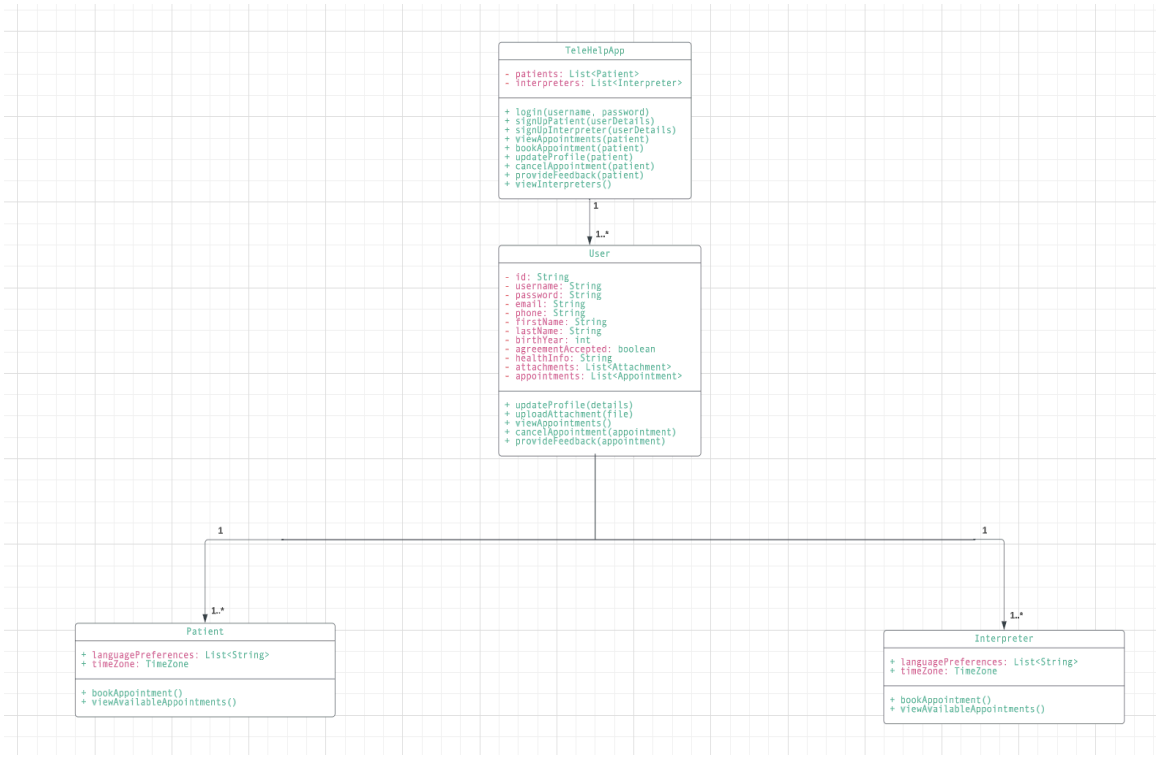


Fig.5 UML Diagram

Chapter 5. Project Implementation

5.1 Client Implementation

- Developed an application for patients to book appointments with specific doctors and specific times.
- patient log-in check using OAuthentication.
- Implemented async storage to store the credentials and authentication ID. This helps preserve the state of the app. Even if the app is closed and then reopened, the user will be on the main page, not in the login. He will be redirected to log in again if the sync is not implemented.
- Developed the navigation system with a clear option to revert to previous pages.
- Implemented a warning system for English-speaking patients who may not require a translator.
- Prepared the confirmation screen with all necessary appointment details.

Chapter 6. Testing and Verification

6.1 Functional Testing:

1. **User Registration**: Test the registration process to ensure that users can create their accounts correctly. Verify that they receive confirmation emails and can log in successfully.

2. Appointment Scheduling: Validate that patients can choose the desired service, language, and practitioner for their appointments. Ensure that interpreters can see and manage their appointments.

3. Profile Management: Test profile updates and verify that user information (e.g., name, language preferences) is saved accurately.

4. Feedback Submission: Confirm that users can submit feedback after appointments and that the feedback is stored correctly.

6.2 Compatibility Testing:

Test the app on a variety of Android devices (phones and tablets) to guarantee that it functions consistently on different screen sizes and resolutions.

6.3 Usability Testing:

1. Conduct user testing with both patients and interpreters to identify any usability issues.

2. Gather feedback on the app's user interface, including navigation, layout, and any pain points in the user experience.

6.4 Performance Testing:

1. **Load Times**: Measure how quickly the app launches and loads different screens.

2. Responsiveness: Ensure that the app responds promptly to user interactions without significant delays.

3. Resource Utilization: Monitor CPU and memory usage to guarantee that the app runs efficiently.

6.5 Security Testing:

1. Verify that user data is securely handled and transmitted.

2. Check for vulnerabilities such as data leakage, unauthorized access, or potential

security risks.

3. Test user authentication to ensure that only authorized users can access the app.

6.6 Localization Testing:

1. Test the app in different languages, particularly Ukrainian, Russian, and English.

2. Verify that language preferences, date formats, and translated content are accurate.

6.7 Regression Testing:

1. After each app update, perform regression testing to ensure that new features or changes do not break existing functionality.

2. Confirm that previously identified issues have been resolved without introducing new ones.

Chapter 7. Performance and Benchmarks

In this chapter, we outline the evaluation methodology for assessing the performance of the Telehelp Ukraine mobile application. This evaluation aims to measure the mobile app's effectiveness, efficiency, and user satisfaction. The methodology includes details about the testbed setup, test parameters, datasets used, baseline approaches, and performance measures.

7.1 Testbed Setup

To conduct the evaluation, we have set up a testbed environment that simulates real-world usage of the Telehelp Ukraine mobile application. The testbed environment consists of

1. A dedicated server environment to host the backend services and APIs required by the mobile app.
2. Simulated user profiles representing both patients and interpreters.
3. Test cases to mimic various user scenarios, including registration, appointment scheduling, interpreter onboarding, and feedback submission.
4. Test scripts and automation tools to execute test cases and gather data.

7.2 Test Parameters

For the evaluation, we consider a range of parameters to measure the mobile app's performance, including

1. User load: Simulated concurrent users to assess the app's scalability and response times.
2. Network conditions: Various network conditions to evaluate app reliability and responsiveness under different circumstances.
3. Device diversity: Simulated access from different mobile devices (smartphones and tablets) to ensure the app's compatibility.
4. Data load: Varying amounts of user data to test API performance.

7.3 Baseline Approaches

In this evaluation, we employ baseline approaches to compare the performance of the Telehelp Ukraine mobile app against existing or common industry standards. These baseline approaches include

1. Industry-standard telehealth mobile apps: We compare the mobile app's performance against established telehealth apps that are widely used in the industry.

2. Performance metrics of mobile applications: We consider common mobile application performance metrics such as response times, load times, and error rates.

7.4 Performance Measures

To assess the Telehelp Ukraine mobile app's performance, we use a range of performance measures, including

1. Response times: Measuring the time taken for the app to respond to user requests, such as appointment scheduling and profile updates, through API calls.
2. Throughput: Evaluating the number of concurrent users the app can support without performance degradation while making API calls.
3. Error rates: Assessing the occurrence of errors, such as failed API requests or connectivity issues.
4. User satisfaction: Collect feedback from simulated users to gauge their satisfaction with the app's usability and functionality.
5. Scalability: Determining the app's ability to handle increasing user loads while maintaining acceptable performance levels through API interactions.
6. These performance measures provide a comprehensive evaluation of the Telehelp Ukraine mobile app's performance and its ability to meet the needs of both patients and interpreters when interacting with the backend through API calls.

Chapter 8. Deployment, Operations, Maintenance

Google Play Store:

Publish the app on the Google Play Store, the primary platform for distributing Android applications. Follow Google's guidelines for submission.

Version Control:

Maintain version control to manage different app versions. Keep track of changes, bug fixes, and feature updates.

Automatic Updates:

Enable automatic updates on the Google Play Store to ensure users receive the latest features, bug fixes, and security patches.

Distribution to Interpreters:

If interpreters need a separate distribution method, consider an internal distribution system, or provide them with a secure download link.

User Support:

1. Set up a support system that allows users to report issues, ask questions, or seek assistance with the app.

2. Provide clear contact information for user support.

Analytics:

Implement app analytics to monitor user behavior. This will help us identify issues, understand how users interact with the app, and gather insights for future improvements.

Regular Maintenance:

Plan for regular app maintenance, addressing user feedback and bug reports. Regularly update the app to enhance features, fix issues, and maintain its overall quality.

By following this comprehensive plan, we can ensure the quality, performance, and successful deployment of the Telehelp Ukraine Android mobile app. It addresses various testing scenarios and deployment considerations, making the app a reliable and user-friendly tool for both patients and interpreters.

Chapter 9. Summary, Conclusions, and Recommendations

9.1 Summary:

The THU Bridge smartphone app aims to bridge language gaps in healthcare communication between patients, interpreters, and administrators. Patients can request interpreter services and make appointments, interpreters manage requests and timetables, and administrators supervise operations. The app's goal is to improve communication, make healthcare more accessible to people with language problems, minimize interpretation expenses, and increase patient satisfaction. It uses cutting-edge technologies such as Flask, PostgreSQL, and CI/CD.

Overall, THU Bridge is a novel strategy to solve healthcare communication challenges, leveraging technology and worldwide relationships to revolutionize healthcare delivery.

9.2 Conclusion:

A ground-breaking initiative to transform healthcare communication by removing linguistic barriers between patients, interpreters, and administrators is the THU Bridge mobile app. By utilizing state-of-the-art technologies like as Flask and PostgreSQL, the project's creative methodology produces a stable and intuitive platform that boosts accessibility, lowers expenses, increases communication, and increases patient

satisfaction. Its strategic partnership with Telehelp Ukraine sets it up for international growth, bridging national boundaries to link patients with volunteers and healthcare professionals anywhere in the world. This initiative sets a standard for future work in this area by demonstrating how technology and international cooperation may bring about revolutionary change in healthcare delivery. The THU Bridge smartphone application is proof of the ability of creative thinking to solve persistent problems in healthcare and opens the door to more accessible futures.

9.3 Recommendation:

The project's ambitious long-term goal is to connect patients globally with healthcare practitioners and volunteers through a relationship with Telehelp Ukraine, thereby expanding its reach and impact.

Glossary

1. Scalability - The ability of a system, network, or process to handle growing amounts of work or increase in throughput without adversely affecting performance or efficiency.
2. UML (Unified Modeling Language) - A standardized modeling language used in the field of software engineering to visualize the design of a system.
3. Patient Satisfaction - A measure of how well a healthcare service or provider meets the expectations and preferences of patients.

References

1. Muthamilselvan, Chinmaya Joshi, Ananthajith Tca, Anoushka Dutta, "Android Application for Emergency Helpline Services,". Proceedings of the International Conference on Communication and Electronics Systems (ICCES 2018).

2. Mostafa Langarizadeh1, Fatemeh Moghbeli1, Ali Aliabadi2, "Application of Ethics for Providing Telemedicine Services and Information Technology,". Department of Health Information Technology Paramedics School, Zahedan University of Medical Sciences, Zahedan, Iran,doi: 10.5455/medarh.2017.71.351-355, MED ARCH. 2017 OCT; 71(5): 351-355.

3. Bhavika Gaurkar, Deepa Amne, Ashwathy Manimala. "BIT Student Helpline Application Based on Android Platform", IRJET, Volume: 04 Issue: 04, April 2017.

4. Abhinav Bassi, Sumaiya Arfin, Oommen John & Vivekanand Jha, "An overview of mobile applications (apps) to support the coronavirus disease 2019 response in India", IJMR, May 2020.

5. P. Sharma, A. Gupta, "Mobile Internet-based TELEDOCTOR CONSULTATION APP," 2021 Fifth International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC) | 978-1-6654-2642-8/21/$31.00 ©2021 IEEE | DOI: 10.1109/I-SMAC52330.2021.9640626.

6. Luciano V. Araújo, Bianca C. Letti, Felipe T. Cantagalli, Gabriela S. Silva, Philippe P. Ehlert, Lara M. Q. Araújo, "A Health Mobile Application and Architecture to Support and Automate In-home Consultation,", 2015 IEEE 28th International Symposium on Computer-Based Medical Systems.

7. Ahmed Imteaj and Muhammad Kamrul Hossain, "A smartphone-based application to improve the health care system of Bangladesh," IEEE Explore.

8. Ajan Ahmed, Md. Talat Mahmud, and Mohammad Monirujjaman Khan, "Info Hospital Web Mobile Application based Health Care System,", Proceedings of the Sixth International Conference on Computing Methodologies and Communication (ICCMC 2022) IEEE Xplore Part Number: CFP22K25-ART; ISBN: 978-1-6654-1028-1

9. Hemavathi S, Sam Abhishek A C, Sai Koushik J, Sai Deepak D, Dr. K. Latha, Dr. K. Jayasakthi Velmurugan. "Medical Assistant App,", 2022 International Conference on Power, Energy, Control and Transmission Systems (ICPECTS) | 978-1-6654-6275-4 2/22/$31.00 ©2022 IEEE | DOI: 10.1109/ICPECTS56089.2022.10047617.

10. Priyanka Nathani, Monika Chakhne, Pooja Patil, Sandhya Pati, “An Application for customized helpline”, Int.J.Computer Technology & Applications, Vol 5 (6),1859-1862.

Appendices

Appendix A. Description of Implementation Repository

1. Setting up a Programming and Execution Environment

1. Objective: Establish the development environments and tools required for the THU Bridge Mobile Application. 2. Progress: ○ Setting up React Native and the necessary compilers, emulators, and debuggers to accommodate the application's cross-platform nature was probably required for the front end. ○ Choose appropriate development platforms, IDEs, and version control systems. ○ Ensured compatibility with selected technology stacks.

2 Acquiring Development Tools

1. Objective: Procure the necessary development tools, libraries, and frameworks. Successfully acquired all necessary development tools, libraries, and frameworks. Ensured the availability of licenses and subscriptions for development tools.

3 Setting Up a Software Prototyping and Simulation Environment

1. Objective: Set up a testing environment for simulating real-world scenarios. The testing environment setup is completed. To evaluate the mobile application's compatibility with various devices and outside services, software components will be used.

4 Understanding/Executing Example Programs or Simulations

1. Objective: Study and execute example mobile app programs and simulate healthcare communication scenarios.Analyzing the features and functionalities of these apps to get knowledge about interface design, user flow, and component integration. reviewing popular mobile apps' code in-depth to learn about best practices for architecture, coding, and design patterns using the React Native framework. Simulation of healthcare communication scenarios is in progress. Initial findings indicate feasibility.

5 Implementing a Prototype of the Proposed Method

1. Objective: Developed a functional prototype of the THU Bridge Mobile Application. Ensuring a seamless onboarding experience for new users is being accomplished through the coding of the sign-up registration process. Core features, including appointment scheduling and user profile information, are being implemented. Integration of selected technologies and APIs is ongoing. Internal testing and debugging are underway.