มหาวิทยาลัยอัสสัมชัญ
ASSUMPTION UNIVERSITY
of THAILAND

VINCENT MARY
SCHOOL of SCIENCE and
TECHNOLOGY

Senior Project Report

Randevoo: Business Reservation Mobile Application

Mr. Sokvathara Lin 601-8002

Mr. Menh Keo 603-8302

A. Chayapol Moemeng (Advisor)

A. Suparwat Charoenvikrom (Committee)

Asst. Prof. Paitoon Porntrakoon (Committee)

Ms. Valerie Chirathivat (External Exposure)

CS 4200 Senior Project 2 (2/2020)

Vincent Mary School of Science and Technology

Senior Project Approval

Project Title:	Randevoo: Business Reservation Mobile Application (IOS)
Members:	Mr. Sokvathara Lin
	Mr. Menh Keo
Advisor:	A. Chayapol Moemeng
Academic Year:	2/2020

The Department of Computer Science, Vincent Mary School of Science and Technology of Assumption University has approved the final report of the three credits course; the course title is CS4200 Senior Project II.
This senior project report is submitted in partial fulfillment of requirements of the degree of Bachelor of Science in Computer Science/Information Technology.

Approval Committee:

_____

A. Chayapol Moemeng

(Project Advisor)

_____ _____

A. Suparwat Charoenvikrom
(Committee Member)

Asst. Prof. Paitoon Porntrakoon
(Committee Member)

ACKNOWLEDGEMENT

First and foremost, we cannot express enough gratitude to our advisor, A. Chayapol Moemeng for his continued reinforcement and encouragement. We offer our sincere appreciation for his recommendations and guidance. From the start of this project, A. Chayapol persistently assists us in many aspects until completing the project. We appreciate his patience, inspiration, and enthusiasm for our project.

We also would like to thank our project committee. Our completion of this project could not have been accomplished without the committee's suggestions, A. Suparwat Charoenvikrom, and Asst. Prof. Paitoon Porntrakoon. Their recommendations produce significant impacts on our project.

We would like to express our profound thanks to Ms Valarie Chirathivat for spending her valuable time evaluating this project. Her recommendation has supported the improvement of the project, especially the business logic for the application.

Finally, our commitments to this very project have brought us many pieces of knowledge and experiences. The encouragement from them when the times got rough is much appreciated and noted. It was a great comfort and relief to know that they were willing to manage our household activities while completing our work. It is our honor to be qualified for our works and evaluate us with what we have accomplished.

ABSTRACT

Nowadays, Mobile applications are tools for altering web applications in many cases. Its native platform runtime provides a seamless experience to access the Internet and facilitate many services and products as desire. These applications are currently a big trend in business solutions and disrupt other businesses that failed to adopt and accept the digital transformation. Many companies nowadays are demanding to have mobile applications to facilitate their services to end customers and clients to ease their usages. Not to mention, technologies are being advanced every single day to support every industry's demands.

Randevoo App is an online reservation platform via a mobile application that connects customers and business owners to do business together simultaneously.

Table of Contents

List of Figures

Chapter 1

Introduction

1.1. Motivation

These days, according to the pandemic that we have been through, people have come up with many regulations to prevent the spread of the virus. One of the regulations is to set a limit amount of people in a shopping mall and store. It becomes difficult for people who want to check out the product first, which they need to queue up for quite a long time to get into the store.

Many high-end luxury stores were using this regulation a long time ago before the pandemic happen. Because the amount of the product is limited and valuable, they use this queue up-regulation to serve as first come, first serve.

Therefore, the main goal of this project is to solve this kind of problem, which is physical queue up the system to digital queue up system that using our mobile phone to book a timeslot to check the product out without going to the store early and waste time to queue up and wait up until their turn.

1.2. Overview of Randevoo

This project aims to develop a mobile application based on the iOS operating system, which provides a convenient method for queue up systems for all the business stores and customers. The goals are further divided into the following objectives:

1. Customer side
    a. Browse the products which are listed by the business store
    b. Book favorite products with a specific schedule to reserve it without queue up
    c. Real time chat with stores
    d. Register their business accounts
2. Business side
    a. Listing products into the platform
    b. Approve or disapprove the customer reservation
    c. Real time chat with customers
    d. Update products
3. Admin side
    a. Banned users
    b. Banned products
    c. Inspect products that are listed by stores

1.3. Scope & Limitation

With the limited amount of time, we are able to complete it only on one platform, therefore our scope and limitation are shown below:

1. iOS Application only for end-users.
2. Web application for administration purpose.

Chapter 2

Background

2.1. Randevoo Application in Business

In business, the Fashion and Apparel industry is currently one of the biggest industries in the world. Once again, fashion and apparel items are represented over a third (35.9%) of online purchases, according to Statista, which is a German company specializing in market and consumer data[1]. Not to mention, this industry is valued at around $662 Billion in 2021 and expects to grow up to $713 Billion in Figure 2.1. Randevoo App will be focusing on Fashion consumers who love physical shopping for luxury products or labels.

First, physical shopping is still relevant when it comes to luxury products. Additionally, consumers who are willing to spend money on expensive products will need to inspect and observe products physically before purchasing them. Second, every consumer desires to have their products meet their expectation. As a matter of fact, in fashion eCommerce retailers, there are approximately 30% of online purchases returned, according to Shopify[2]. In addition, Randevoo App wants to reduce purchases' returns by focusing on instead of eCommerce platform to opt-in reservation platform in fashion retails. Third, the Randevoo platform provides multi-channel selling with reservation as a scheduling tool integrated solutions. Moreover, each consumer can have their exclusive timeslot for reserving their desires products via an app then enjoy inspecting their products at the physical store. Furthermore, this will reduce fewer purchases' returns to the store.

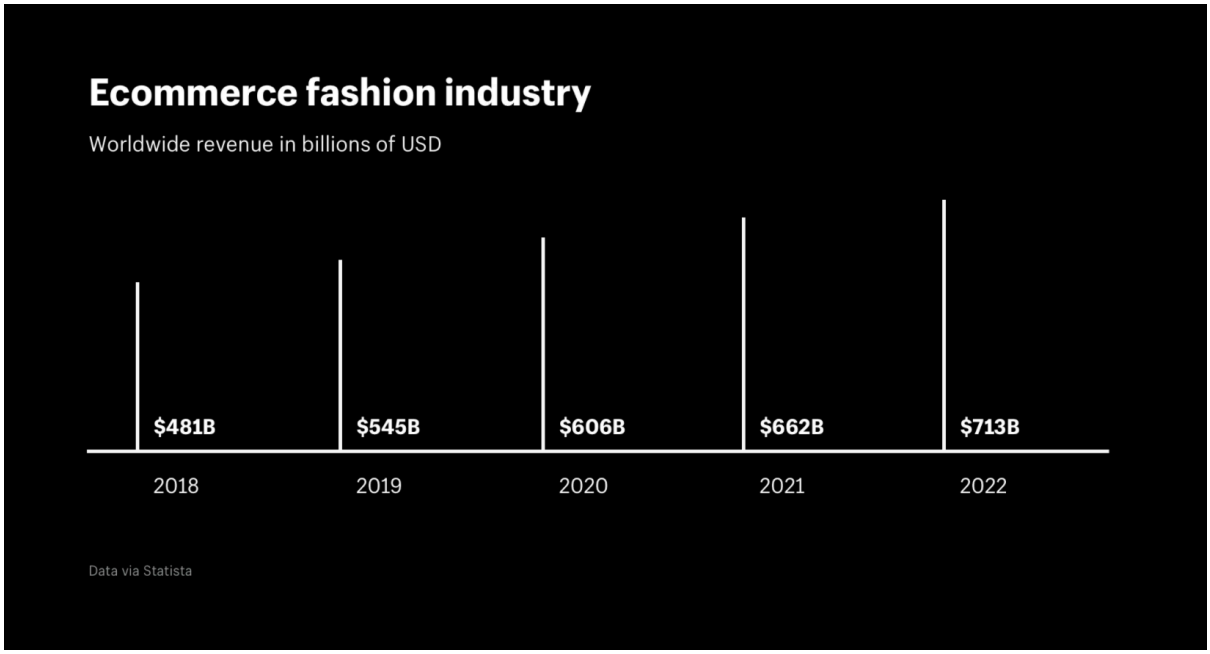

Figure 2.1: Statista Fashion & Apparel Report.

Chapter 3 Related Work

3.1. Microsoft Bookings

Microsoft Bookings is a meeting and appointment scheduling platform for businesses. This platform provides digital scheduling solutions to businesses' clients in Figure 2.2. And it also has an integration with a website or Facebook page that can automate scheduling to business. However, this Microsoft platform is focusing on business clients instead of mainstream customers. Additionally, this application will have only a business account side with a setup to assign business' staff to handle and cover appointments or setup prices for their business services. Additionally, to use the Microsoft Bookings platform, business users have to subscribe to a Microsft 365 account, starting from $5 user/month up to $20 user/month. And the features, tools, multi-platforms, and advanced supports will depend upon the subscription of a business user.

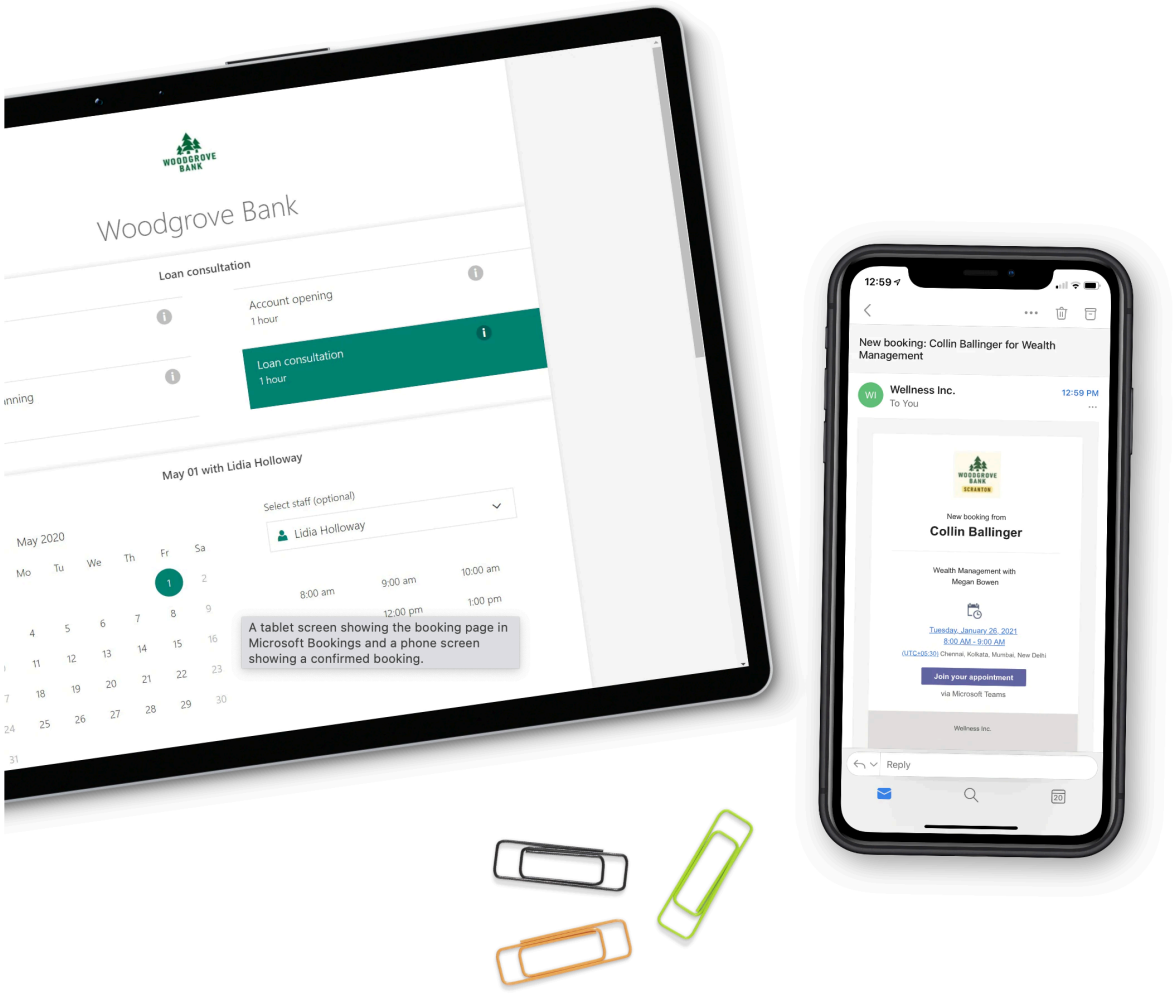

Figure 3.1: Microsoft Bookings Platform

Chapter 4

iOS Implementation

This chapter will focus on the implementation and processes of the Randevoo project in both the client-side app and its microservices' environments. It will also clarify the entire flow of the app from the customers' perception of user experience (UX) with this mobile application to their existed perceptions with other mobile applications. Moreover, the iOS UI designs are being followed by Apple iOS Design Themes and Design Principles[3].

4.1. System Diagram

The Randevoo Project is implemented in a microservices architecture. IOS App requires authenticating users' credentials by sign in or sign up to get access tokens from Firebase Authentication. After the app is received tokens, it can access information to the users' specific requirements. Moreover, each access will be determined upon their needs which will be routed to different services in Firebase services. And most services are under Firebase Console. However, most regulation of APIs' endpoints is required to access from GCP Console. The system diagram is shown in Figure 3.1.

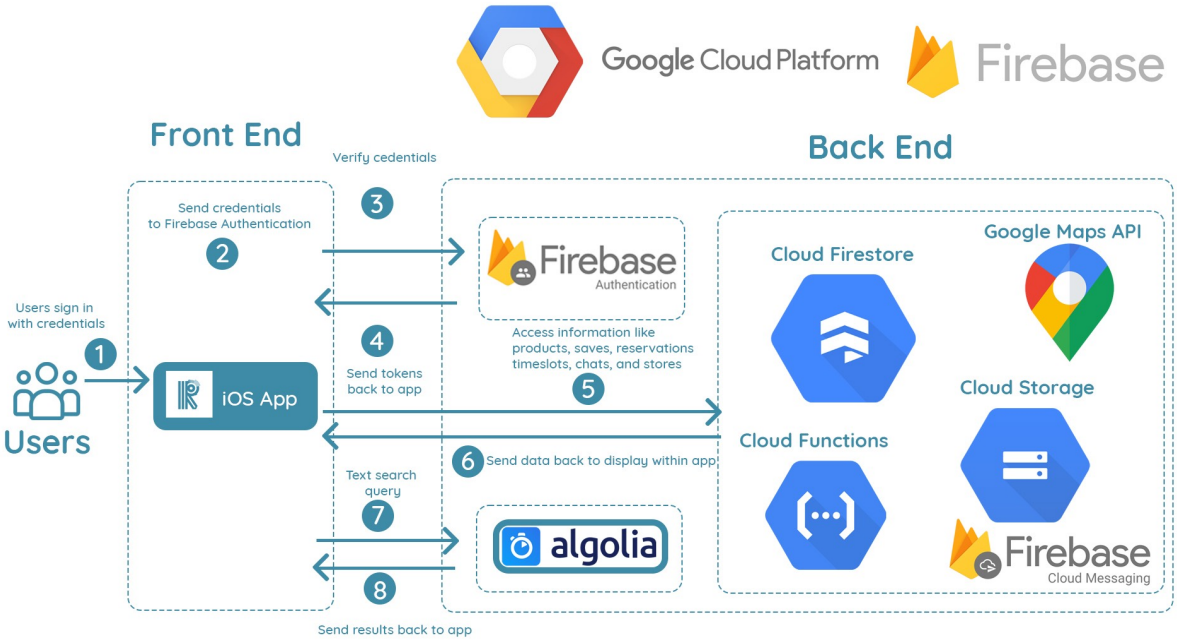

Figure 4.1: System Diagram.

4.2. Data Structure

The Randevoo App is leveraging NoSQL Database in the cloud. And most services are provided by Google Cloud Platform. This app opts for Cloud Firestore to store its data. And Cloud Firestore is a horizontally scaling document-model NoSQL database in GCP. One of the most challenging aspects of building an app is determining the optimal data structure. Primarily, developers want to maximize performance while minimizing costs. However, that is easier said than done and especially so with a NoSQL database like Cloud Firestore. Additionally, when it comes to Data Modeling in the NoSQL database is flexible and comprehensive, almost any complexity developers need.

In the NoSQL world, things are a little different. First, data are not stored in the tables. And data can be stored from a plain old key values store to a big nested tree-like in the Firebase Realtime Database and a collection of JSON Objects. Second, one thing that most of them have in common is that NoSQL databases are usually schema-less, which means there aren't any database-level restrictions around what kind of data developers can put at any point in the database. The other important aspect of NoSQL databases is that many big tech companies are adopting NoSQL databases. Third, when it comes to the data query, the database reads end up being really fast. The most significant advantage with a NoSQL database over traditional databases is that it can distribute its data across multiple machines efficiently and more sophisticated. With most relational databases, an app gets super popular and requires a database to scale up to a larger and larger data set. Developers need to put it on bigger and beefier machines. And this is known as scaling vertically. On the other hand, with many NoSQL databases like Cloud Firestore, if Randevoo app needs to scale up to a larger and larger data set, Randevoo's database can distribute that data across several servers and everything just kind of works, and this is known as scaling horizontally. Last but not least, the Randevoo app is highly scalable and can swiftly provide services to customers. With the environments like Google Cloud Platform or AWS, it is pretty easy for these systems to automatically add or remove servers to the Randevoo database as needs, with very little to no downtime.

4.3. Functionality

4.3.1. Frontend System Overview

The Randevoo App is a comprehensive iOS software solution for business stores with the old traditional queuing system. Client Side App's users are divided into two groups which are customers and business owners. Each group has different features and functions. For instance, the customer will book a timeslot with the business store, check available products, and register their business store in the application. Meanwhile, for a business owner, the application provides many functions and features such as listing the products, approving, or disapproving from the the customer. Both customer and business owner are able to contact via real-time chat within the application without using a third-party application.

4.3.2. Executive Highlights

There are several functionalities that the Randevoo application will provide. One of the main functions is making a reservation between customer and business owner. To prevent human error, the application must be designed with a good UX.

First, once the user creates a business store inside the application, the application will generate default information for the business. The information contains the open and close time of the business, available slots that the store can serve the customer at a time, the duration of each reservation, and the range of reservation date that can be booked.

Second, to prevent human error, the available reservation schedule will be generated by the application itself from the information that has been discussed earlier. The information of the store can be modified later. However, suppose the customer reservation was approved by the store owner with the old information before the modification. In such case, the application will not modify the booked timeslot for the customer or store owner to prevent the customer's confusion. Therefore, the business owner must contact the customer by themselves if the reserved hour is no longer available for the store.

As shown above, every reservation will guarantee that there is no confusion or human error, which leads to conflict between the business store and the customer. Again, the approved reservation will not be able to modify or change by any user.

4.4. Stakeholders

4.4.1. Customer

a) Login/Signup

Once the user opens up the application, the first screen that shows up will be a registration screen, which is shown in figure 2.1. In the registration screen, the application provides two choices which are login with Google and with Facebook. When the user chooses either choice, the application will check with the backend system to verify whether the user is a new or existing one. If it is the new user, it will obtain the information from either platform that the user chose and store it in our database. However, if the user is existing, the application will start to display the information that the user has been store in the account before.

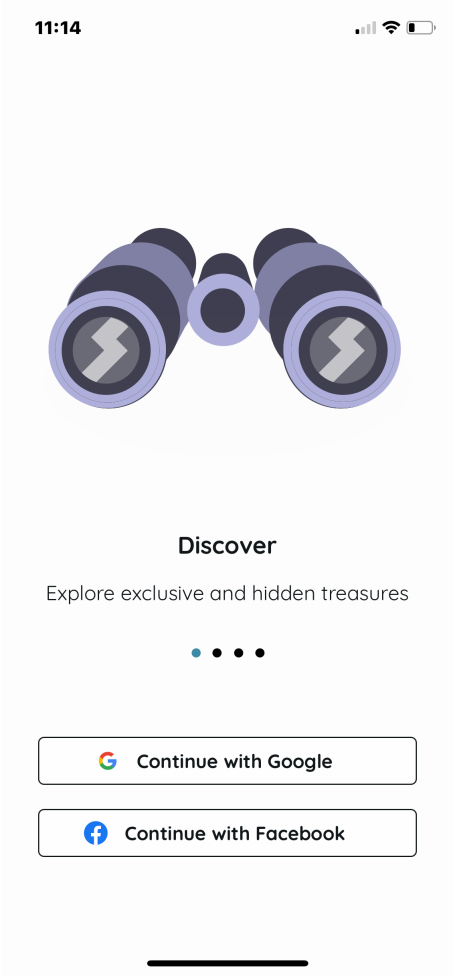

Figure 4.2: Sign-in/Sign-up with Google or Facebook.

b) Reservation

Customers can start making a reservation with the store once the login or sign-up is completed. The reservation starts off by adding the product which they want to check out into the shopping bag. Once they are ready, they need to choose the available schedule that has been generated by the application base on store information. In the case that the user chooses today as the current reservation date, the schedule gap is 1 hour, counting from the current hour. For instance, the current time is 15:00, so the earliest reservation that can be made is 16:00. However, the application will show the timeslot is already exceeded to prevent the customer from booking that timeslot which is shown as an example in figure 3.3.

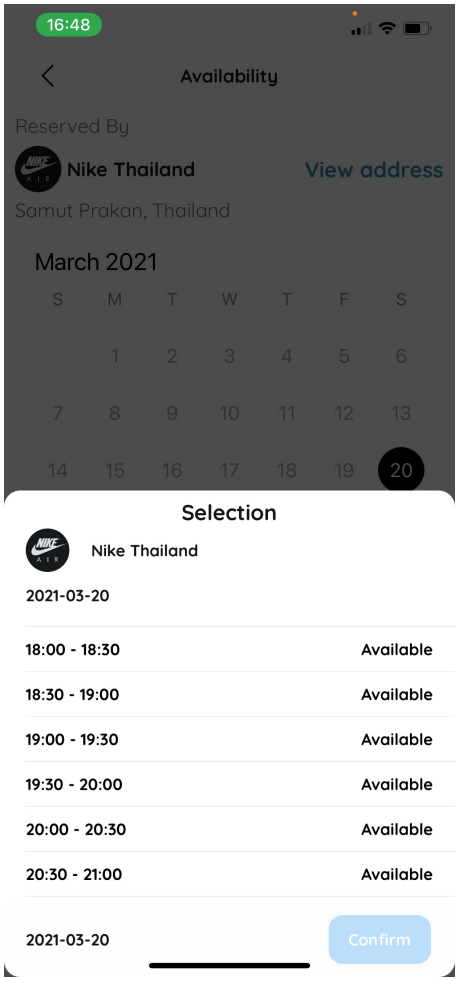

Figure 4.3: Available timeslots.

Although they choose any different date, the application will check whether the timeslot is occupied or not. For example, suppose the generated timeslot is a conflict with the booked timeslot. In that case, the application will show a status call Occupied, shown in Figure 3.4, which is to describe to the user that the timeslot is not able to be booked due to the conflict.

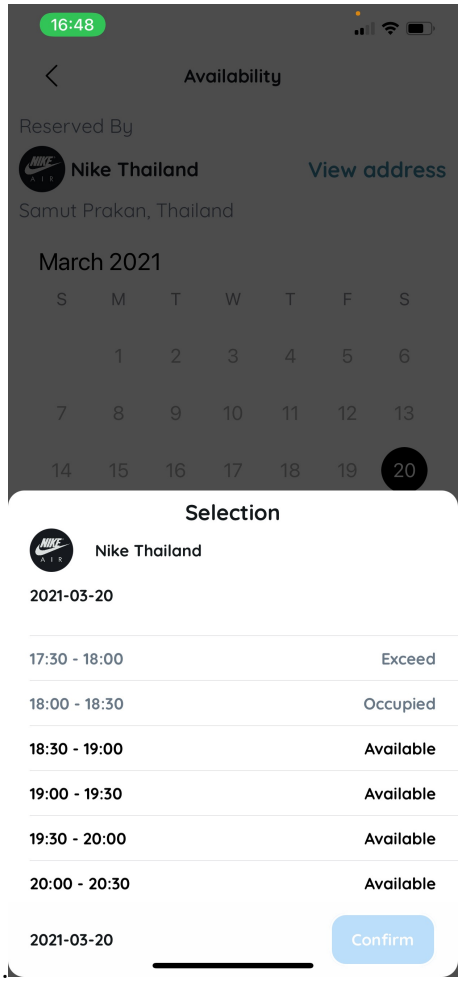

Figure 4.4: Timeslot with multiple indicate status.

Moreover, the application will also check with the availability per timeslot that provided by the store. If the availability is zero which mean that the current time is already full, the application will show a status as unavailable which shown in figure 3.5.

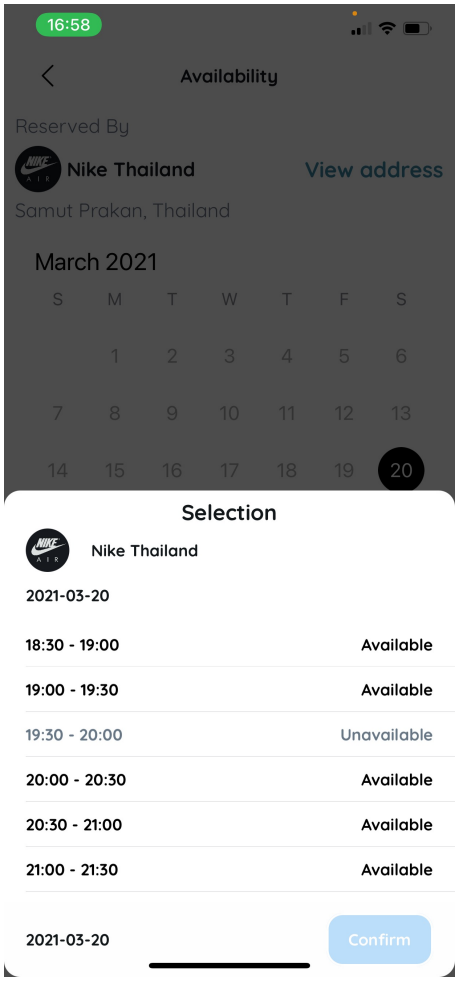

Figure 4.5: Unavailable timeslot

But if the status is available, the customer will able to booked the product with the timeslot that they choose and the next step is to waiting for the approval from the store.

After they got approval from the store, the application will generate a QR code which is used to verify the booking timeslot later when the customer arrived at the store.

c) Realtime chat

Every user want an application that implement with efficiency functionality without base on a third-partyt application, therefore our application have a build in chat. The purpose of build-in realtime chat is to avoid using a third part application to interact between customer and business owner. With the build-in chat, customer can contact the business owner if they have any question, and business owner can contact the customer in case of any update about the products.

4.4.2. Business store owner

a) Listing product

Listing product will allow business store owner to listing their product into the application with detail such as price, size, color, etc. After listing the product, it will display to all the customer. Moreover, it keep in under the store profile, so the customer can see all the product that the specific store available which shown in Figure 3.6. In addition, business store's owner can mange the product with feature that the application provide such as store them into a collection, update and delete the product.

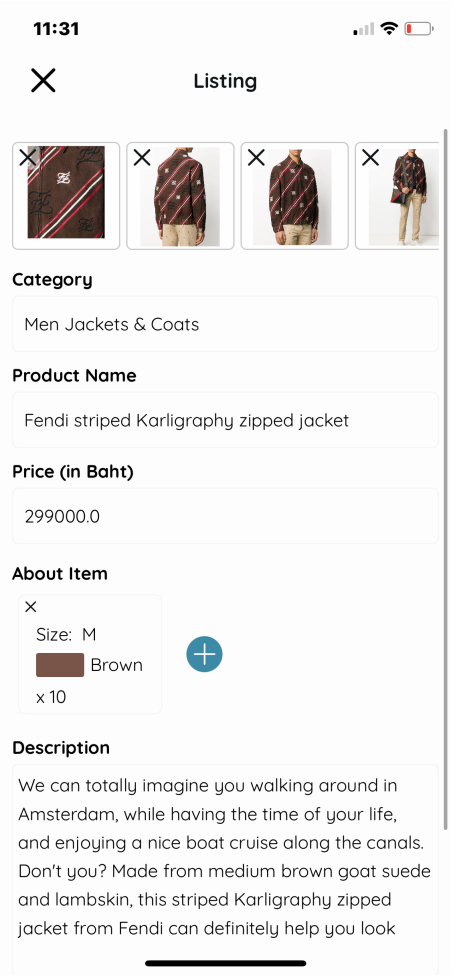

Figure 4.6: Listing a specific product

b) Update store information

Once user create a business store account, the information of store, for instance, open and close time, duration of reservation, duration of business day in a week, etc, will generate with default information. However, those informations can be updated later in the store setting in reservation hour tab which demonstrate in figure 3.7. Futheremore, if the store have some special policy, they can list those information as well which will display later in store profile and under the product description.

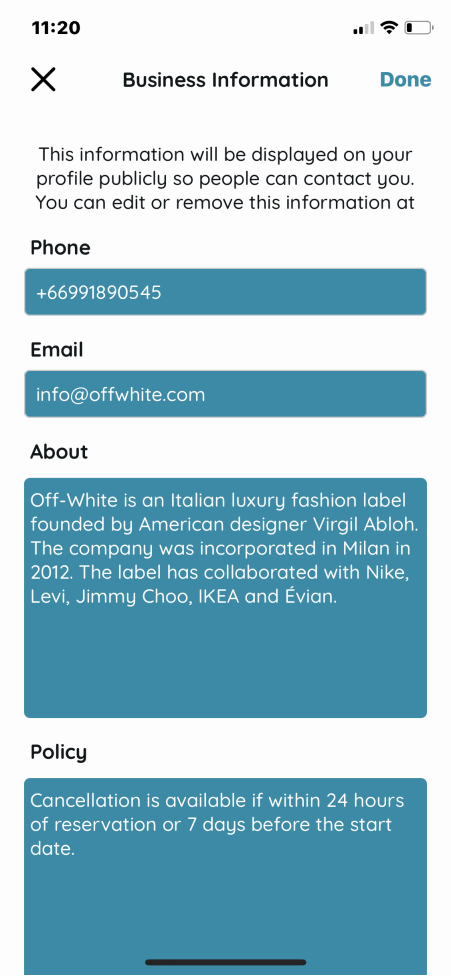

Figure 4.7: Update business information screen

c) Managing customer reservation

Business store owner will get a notification alert when there is a new reservation request are make from the customer. They can manage the reservation by approve or disapprove. Later on, when customer come to check out the products, store owner can just scan the Qr code of the reservation from customer account by using a built-in QR scanner and the information of the reservation will be clearly summarize for the store owner which shown in figure 3.8.

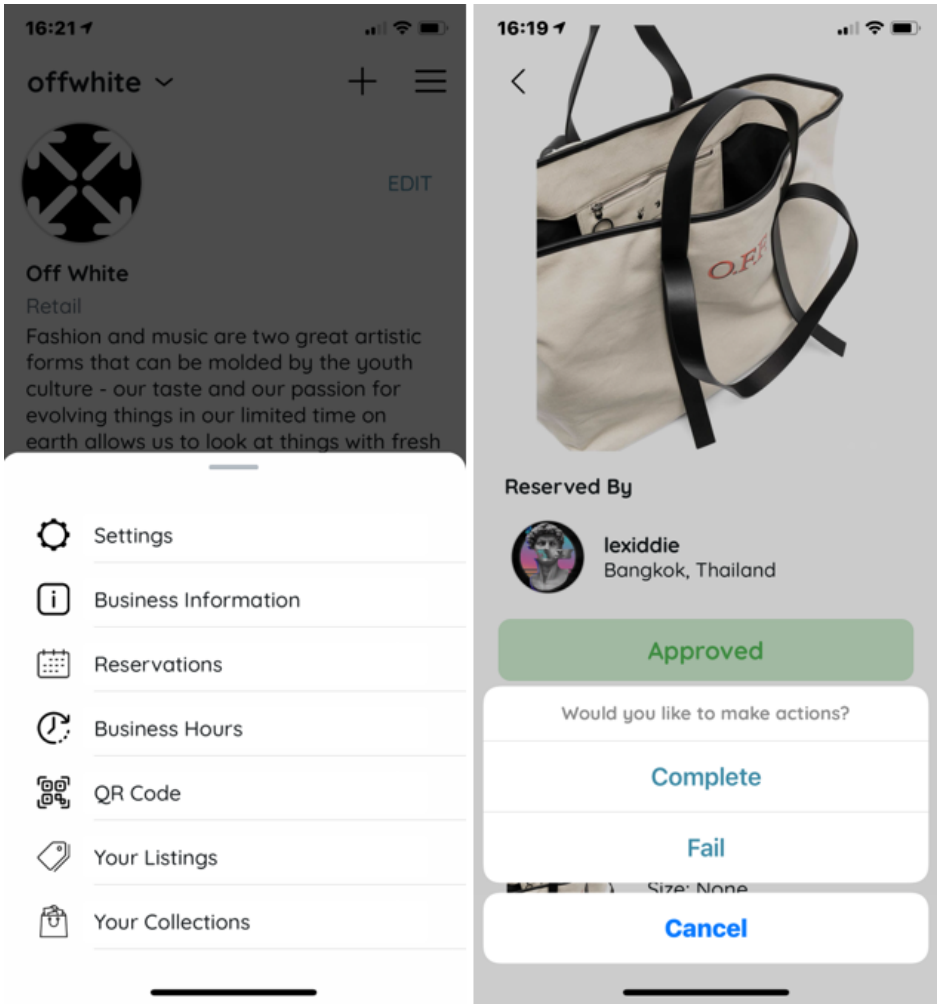

Figure 4.8: The action of complete or fail a reservation by a business store.

4.4.3. Administration

As mentioned in the scope and limitation, Administration is built on another platform which is web application. The administration will be monitor the user and product. If there is any product or user that violate the term and condition of the application. The administration can banned them or delete those product out from the database which will effect the application as well.

4.5. Notifications

4.5.1. Overview

When it comes to notifications in the mobile application, it is crucial to keep users informed with timely and relevant content. Moreover, even the mobile app is inactive or running in the background. With Apple Push Notification Service (APNS) [4], many features of notifications like display a message, play a distinctive sound or update a badge on an app icon which shown in figure 3.9.

Figure 4.9: Example of iOS notification.

4.5.2. Implementation

First of all, implementing this feature is a bit complicated. However, it is archivable to accomplish due to the many resources and services. First, Apple Push Notification Service, also known as (APNS), is an Apple platform service for developers to leverage its notification services. Moreover, to achieve this push notification into Apple devices, every developer has to use this service only, and it requires Development Certificates from Apple. Second, to generate certificates from Apple, the

developer has to have a paid Apple developer account. And only paid developer account has the feature to create Development and Production certificates.

In the Randevoo iOS app, the app is registered its Bundler Identifier with the Apple Developer Account. And after a certificate has been signed from the account, it will be used to upload to Cloud Messaging in Firebase in the Figure below. Additionally, there are server key and APNs auth key to be used in Cloud Messaging to enable authentication during dispatch payloads. In figure 3.10 is shown how the structure of the notification service was built.

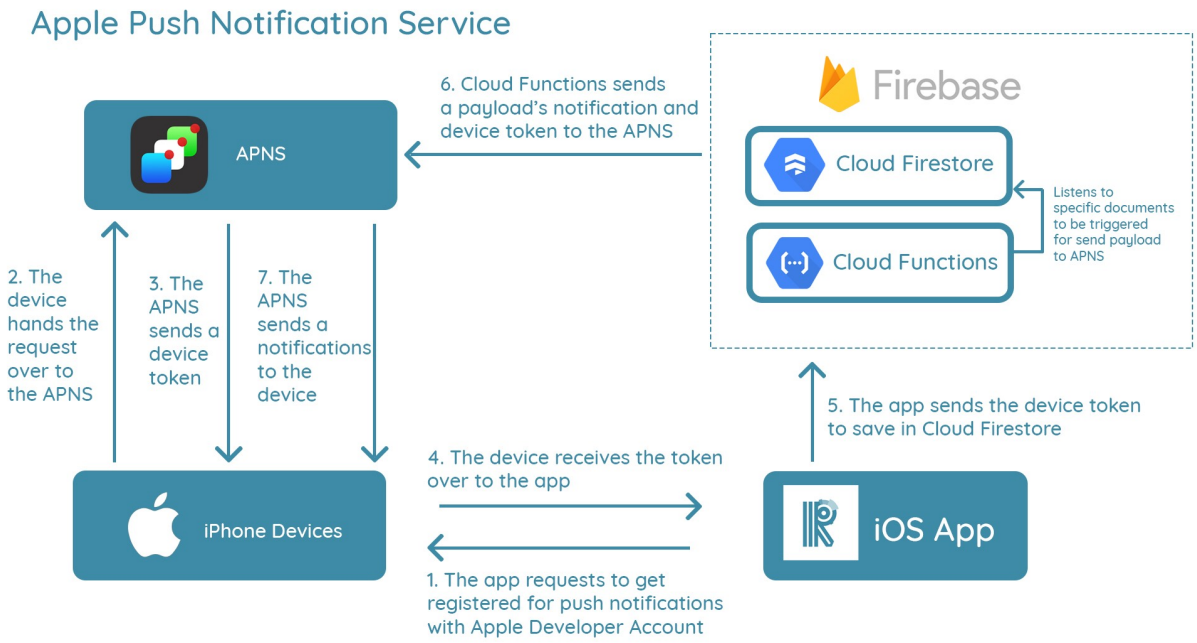

Figure 4.10: Implementation technique of Push Notifications

4.6. Microservices

4.6.1. Overview

Microservices[5] is a software development technique —a variant of the service-oriented architecture (SOA) structural style— that arranges an application as a collection of loosely coupled services. In a microservices architecture, services are fine-grained, and the protocols are lightweight.

On a topic like microservices, it is tough to come up with any firm definition. In this architecture, many different approaches depending on the organizations that define the implementations of each process. However, this project is being implemented with the following architecture, a technology that many enterprises worldwide have adopted. This kind of project fits the design pattern to follow up, and it has good practices.

4.6.2. Introduction

There are several reasons that this project has been opted for this Microservices Architecture. First, the development of this project can concentrate on small, independent units of working software, in which the team can work with partials independently. For instance, this project is leveraging GCP Cloud Functions in Firebase to facilitate backend functions.

4.6.3. Methodology

Cloud Functions is a serverless execution environment for building and connecting cloud services. Randevoo wouldn't be accomplished with GCP Cloud Functions. It plays an essential role in serving dynamic content and hosts other microservices. First, it handles Randevoo backend codes in push Notifications to APNs flawlessly with its specifications of triggering functions to handle the business logic to notify customers and stores. Second, it focuses on single-purpose JavaScript functions that are executed in a secure, managed Node.js environment. Additionally, some business functionalities require manipulating NoSQL collections dynamically. The client-side app cannot trigger some logics, and Cloud Functions provide the ability to handle server-side rendering. Third, developers can either opt for JavaScript or TypeScript with ESLint to write their functions. TypeScript gives developers more flexibility in maintaining and reduce errors with its static typing programming language. Fourth, Cloud Functions can host HTTP type functions and also a caveat to building RESTful API. For instance, in Randevoo, a Full-text search cannot be done from the Cloud Firestore platform. This means developers have to opt a software as a service like using a third-party search service Algolia. Furthermore, to leverage Agolia, developers have to write RESTful API functions to forward and update data from Cloud Firestore to Algolia. Not to mention only specific collections of data that will be updated to Aloglia for the index each time a document is written or deleted. Fifth, Cloud Functions provide future proof for scalability that helps developers migrate the NoSQL database comfortably from Cloud Firestore to other platforms like MongoDB Atlas. As a matter of fact, developers can forward a whole collection one by one with a single emitted. Moreover, developers do not need to shut the server to migrate data. Even data are being used by customers; developers can reroute new database endpoints to another end with existing previous data seamlessly. The structure of the microservice is shown further in figure 3.11.

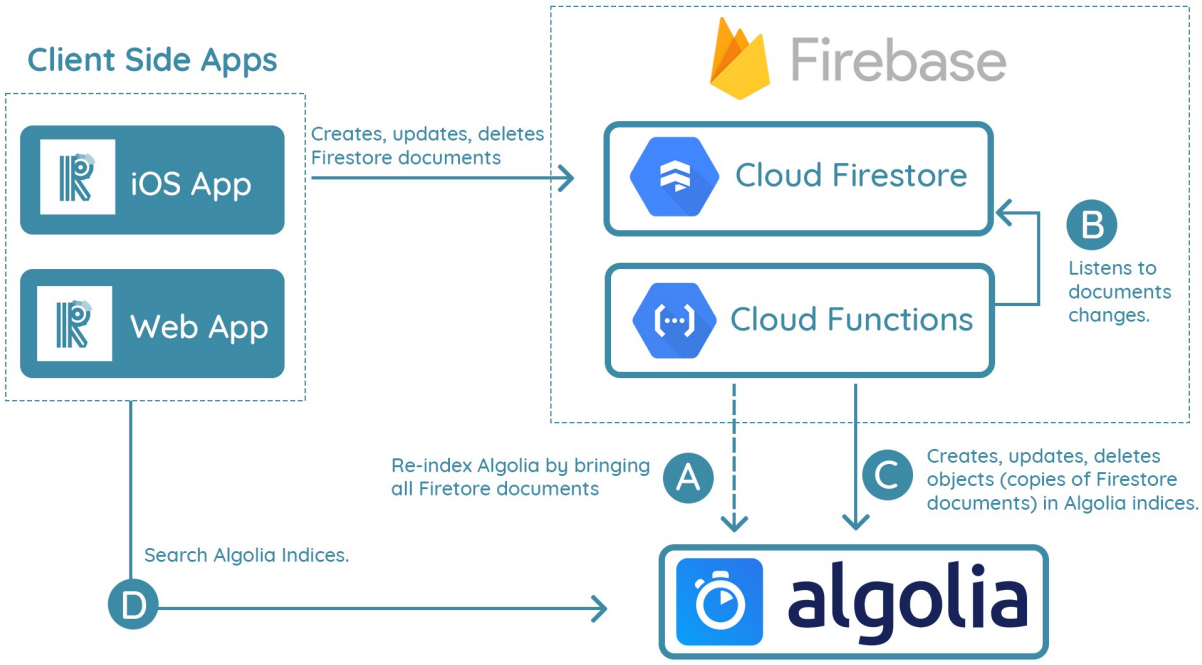

Figure 4.11: Algolia with Cloud Functions

4.7. UX/UI

The user experience (UX) is a significant and crucial design that usually uses this term to evaluate whether the application is convenient to interact with the business. As a result, Randevoo Application is built and designed with a good UX that guarantees to bring a beneficial and convenient experience to the user in terms of using as a reservation application.

4.7.1. Main Navigation

Main Navigation refers to the interaction that allows the user to navigate from one page to another page of the application's content, which is starting from a simple button click to a complex pattern such as a tab bar and navigation controller. According to Apple Human Interface Guidelines [6], navigation controllers divide into three main styles. In fact, Randevoo App has implemented the navigation bases on all these three main styles for different purposes. Randevoo Application begins with Flat Navigation. Once the user opens the application, there will be a tab bar at the bottom of the screen, which is used to switch between multiple content categories such as Home, Search, Reservation, Chat, and User Profile. The hierarchical Navigation structure is used to implement the Making Reservation controller where the user can navigate to the making reservation controller by starting with Home and checking the product. Once the user adds a product into Bag, the application will navigate to another controller, the Making Reservation controller. The last structure, which is Content-Driven or Experience-Driven Navigation, is used to navigate the customer to business owner profile account which customer can check the business owner profile from many controllers such as in Product controller or Searching the business owner directly from Search.

4.7.2. Seamless Transition

Every user wants seamless transitions when interacts with the software. Implementing a seamless transition is very crucial, which can improve the product to be more attractive. Since Randeveoo App has many functionalities, therefore, we add the transition to Navigation Bar. The transition will enhance the user interface of the application and the user's perception to alert the current landing page where they are processing.

4.7.3. Perception

Design an application with an uncertain flow can cause many effects on how the potential user perceives the system. Therefore Randevoo has to design base on a similar workflow and pattern of similar application which already existed to guarantee the user perception and understanding.

From the case study, Microsoft Booking Application is one of the applications that inspire the Randevoo application. However, most users in the present day are quite familiar with the workflow of social media applications, for instance, Instagram. Therefore, we design the workflow of the Randevoo app, which is the profile management with the familiar workflow of social media application, will reduce the confusion and increase the perception and experience. Moreover, the application not only uses the text label to indicate the status of the booking timeslot but also using a different color to indicate the status as well. For example, red color indicates that reservation is failed, or light grey to show that the amount is out of stock or date is not available to make a reservation.

4.7.4. UI

User Interface (UI) is referring to the graphical layout of an application, which consists of many elements and components within the layout or inside the whole application. For instance, the buttons for users to click on, the text that they read, the images, the text field to input, sliders, and all other elements and components that the user interacts with. Moreover, it includes screen layout, transitions, interface animations, every interface's interaction, and color that apply to each element to indicate different purposes.

4.7.5. Consistency

Consistency is a core principle in life and in design. It has abilities to enhance the works more persistent in a manner. Randevoo App has been implemented with mightiest principle like consistency to improve user experience (UX) and user interface (UI) as businesslike working system.

When it comes to design, consistency is one of the keys of the Design DNA. Randevoo app is being improved with intuitive design to make it highly useful and better. Moreover, usability and learnability have come from consistency which most users have learned to understand by this kind of perception.

There are various consistencies that the Randevoo app has been accomplished. First, the application has Visual Consistency that helps perceiving users to increase learn abilities to the system. Additionally, every element like font, button, label, text field, icon, color, selection, and other components are always be consistent across the product to enhance Visual Consistency to the users' perception. Second, the similarities of control are always functioning across the product that is Functional Consistency. Furthermore, it can increase the users' perception to predict the product's function, and predictability leads to users feeling safe and secure. For instance, Automate Generate Reservation Timeslot, after business owner fill in the store information, the application will automate generate the timeslot base on those information. Third, when the usability and learnability of the product have been improved by both various consistencies above, it will be convenient to users when new features/pages are introduced to the Randevoo app. With that mentioned, it leads users to keep along the path of how they use the system and that consistency is Internal Consistency. As showed above, users' knowledge and perception of the product that are archived by the consistency across multiple systems/products can be reused again with another. In addition, this good example of external consistency helps user to gain greater user experience and eliminate a lot of the friction.

Consistency is very important, and Randevoo app has been implemented with these four types of consistency above to help user gain better usability and experience.

Chapter 5

PROJECT METHODOLOGY & FRAMEWORKS

The Randevoo application has implemented with open-source frameworks to save time and boost production. The frameworks include Google Map, Object Mapper, MessageKit, and Horizon Calendar. Each framework use for different purposes, such as navigate the map, message, and calendar template.

5.1. Timeslot

The core system of the Randevoo application is the reservation. Therefore, the application must provide the best method to generate the schedule without any failure, such as duplicate timeslot or overlap timeslot. As a result, the timeslot must not be generated by humans, which is the business store owner.

5.1.1. Problem

People often encounter that timeslot they fill into the application always come with the mistake or misunderstood with how the timeslot work. Therefore, the timeslots sometimes generate with the wrong time or overlap each other. Moreover, some users are just tired and lazy to fill in the information repeatedly. They wish that the application is smart enough to generate the timeslot for them by the store's open and close time.

5.1.2. HorizonCalendar

HorizonCalendar is an opensource calendar framework with a declarative, performant, calendar UI component that support use cases ranging from simple date pickers all the way up to fully-featured calendar application [7]. HorizonCalendar is one of the best calendar UI that has been developed by a famous enterprise application company known as Airbnb. With the help of HorizonCalendar, the development process will significantly improve since the calendar view does not need to build from scratch. The example of HorizonCalendar is shown in figure 4.1

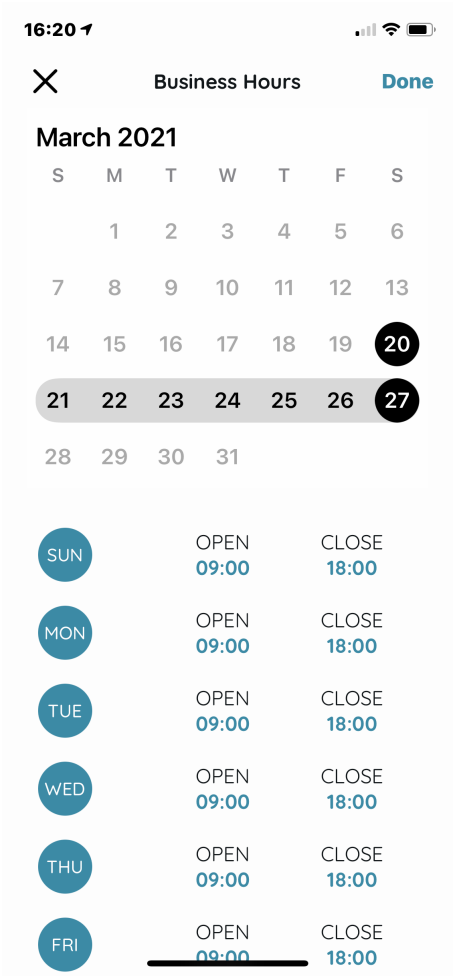

Figure 5.1: HorizonCalendar with day range indicator.

5.1.3. Methodology

The timeslot is the period that contains the date, started time, ended time, and availability per timeslot of each reservation. The timeslot is generated from the information of the store. First, the information begins with the open and close times of the store. Second, the day that business store opens, for instance, some business store open only weekday but not weekend. Third, the information of the duration per timeslot. For example, the store set that each customer has only 1 hour for checking into the store and review the product. If the customer does not show up during their time, their product will sell to other customers. The additional information is the range of the day that the business set to allow their customer to book their timeslot. For instance, the business store set the range one week. Therefore, the available timeslot to book will be only a week which counting from today. This day range will help the business store, which frequently updates their store information such as close time or open time. The last information that the store must input is availability per timeslot. Availability per timeslot is referring to the number of customers that can be booked per timeslot. However, store open and close time change be updated each day differently. For instance, the store owner usually opens the store at 8 am on the weekday but not on Sunday, open at 10 am.

From the information above, the application will generate a proper timeslot for the store and customer. When customer checkout the product, the system requires them the select a date from the calendar, which builds from the HorizonCalendar framework. Once they selected the date, a list of timeslot will show up. Each timeslot will indicate clearly that whether it is available or unavailable. Moreover, the application will not allow the customer to book the timeslot that is already happen during the booked timeslot which they have made before.

5.2. Realtime chat

Chatting is one of the essential features that the Randevoo application should have. As a shopping and reservation application, both customers and business store owners usually have questions to ask or update. For instance, the customer asks for more detail about product or policy, or sometimes business owners mistaken about information they have to inform their customer.

5.2.1. Problem

From time to time, most people encounter the common problem when there is no build-in chat inside the application that they are using is they will need to use a third-party chat application. However, suppose the customer does not have the chat application that the business owner provides as a contact. In that case, they will need to install the application and set it up just to chat with the store, which they found it is a byzantine process, or sometimes they just ignore and choose not to buy the product and find other stores instead.

5.2.2. MessageKit

MessageKit is a Swift framework that provides a safe environment for others to learn through open sour and create a more comfortable chat application with beautiful and customizable Chat UI [8]. Messagekit provides a standard cell structure, shown in Figure 4.2, with most of the valuable components to start a chat application. With the help of MessageKit, the implementation structure will go smoothly with the user interface; however, how the data are store and design is critical in this process.

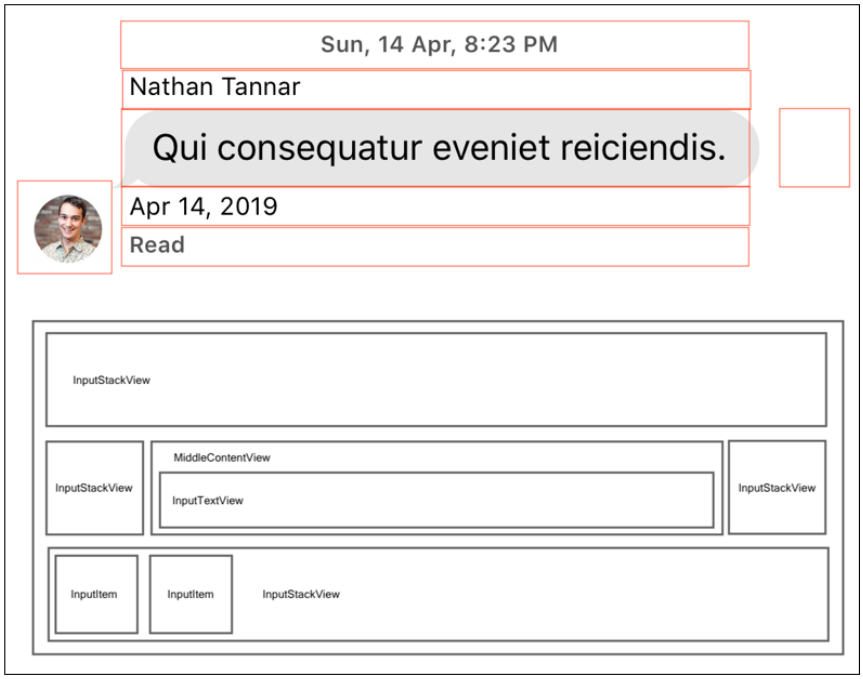

Figure 5.2: MessageKit cell structure.

5.2.3. Methodology

As mentioned earlier, the user interface of the build-in chat will be based on MessageKit. Therefore, the methodology will focus on the structure of the database in order to make the application more efficient as well as fast performance. Since this project uses Cloud Firstore (Firebase) as the database and Cloud Firestore is different from the usual database, it is a NoSQL data model. The data are store in a document that is separate by collection and contains field mapping to values. The data in a collection of firebase are store separately in a document by a unique id, and if the document's id exists, it will replace the old document with the new document.

To improve the performance speed of Cloud Firebase, the application must construct with multiple collections for a different type, for instance, a collection of a **groups** of people who is in the same group of chatting and a collection of **messages** where store the message content of user [9].

The collection of **groups** contains multiple documents of groups of users, but the application supports only two user chatting systems: customer and business store owner. Therefore, each document in the **groups** collection will contain two user ids, the content of the last message and created date. This data in the document of **groups** collection will be displayed on the title page of the chat page, as shown in Figure 3.5. In case there is a new message, the document in the collection with the same customer and business store owner's id will update with the lastest messages.

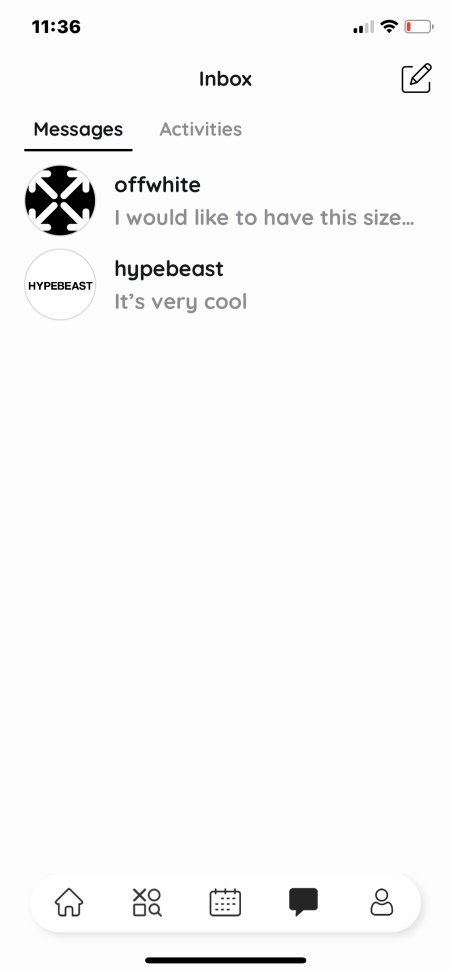

Figure 5.3: Chat home screen with lastest messages.

The other collection is **messages** that contain only the messages of the user. The structure in this collection is different from **groups.** However, the key structure is base on **groups** collection. Once the user starts messaging each other, the application will generate a unique id to create a document in the **groups** collection containing data described earlier. After generating the group, the application will be using the same id and create a document in the **messages** group. Still, inside the document, it contains information of the same people from the **groups** collection such as message content, createdAt and sender and receiver's id. The sender and receiver's id are using to identify who is the owner of each chat.

By creating these collections, it increases the speed of data retrieval from Cloud Firestore. Imagine the application retrieves the message to display in the application by checking each document in Cloud Firestore one by one whether the sender and receiver's id matches the condition, which will take a while to complete. In comparison, the grouping method will only retrieve the group of the message inside a document by just checking only one condition, whether the document id in **messages** collection is satisfied with the id of the document in **groups** collection of the two users.

5.3. Map

As a shopping and reservation application, a map to indicate the location of the business store is an essential feature. The map provides a clear indicator about the business store location compared to the description's text listing address. With the map, customers can quickly identify the area of the store and estimate the duration of the traveling.

5.3.1. Problem

Most of the time, when people do not become familiar with the place of the destination, they often use the help from internet to searching the area. However, people also found it disturbing that sometimes they have to copy the location address from the application and paste it into the internet to get more vision about where the location is, which means they keep switching back and forth between two applications. Therefore, the Randevoo application provides a build-in map that already points to the business store location without typing the address.

5.3.2. Google Maps

Google Maps is one of the most famous open-source frameworks provide by Google. Google Maps provide plenty of useful features such as Google Maps servers, map display, and response to the user gesture such as drag and click [10]. With the help of Google Maps, the build-in map in the Randevoo application will be more functional with many features such as a marker indicate current customer location and store location and a drawn route from current customer location to store location that shown in Figure 4.4.

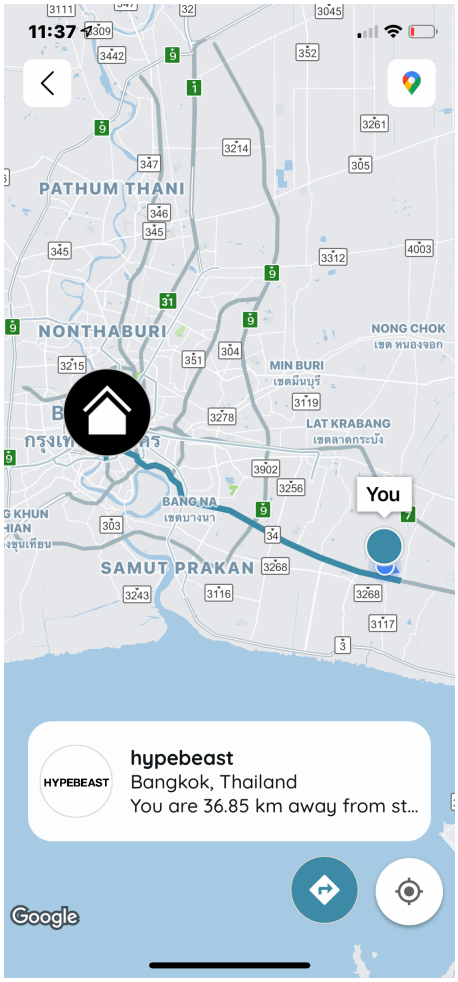

Figure 5.4: Map with route from customer to store location.

However, Apple also provides a built-in Apple Map for the developer to use in their project as well. But, the Apple Map does not offer the best coverage and detail on a worldwide map compared to Google Maps. Moreover, one of the most valuable features, which is a drawn route from source to the destination location, is not available on the Apple Map at the moment. Therefore, the Randevoo application chooses Google Map over Apple Map in order to provide more functionality to the end-user.

5.3.3. Methodology

In the Randevoo Application, the map is used for business store owners to mark their store location. Once the user signs up for a business account, the application will ask for the city and district that the store is currently located in. After selecting the location description, the map will show up with two options. The first option is users are allowed to select their current location as the store location. The second option is to drag and drop the marker point to the location of the store. When the marking location process is done, the application will grab the latitude and longitude of the store location from the map and store it in the database inside **BusinessInfo** collection.

Later on, the store location will show on the map base on the latitude and longitude that have been filled in before and display to the customer, shown in figure 3.7. Customers will be able to click and drag the map around to get more information about the store's area. Moreover, they can choose to open the map in the Google Map application with a button at the navigation bar. It will directly zoom in to the storage area in the google map.

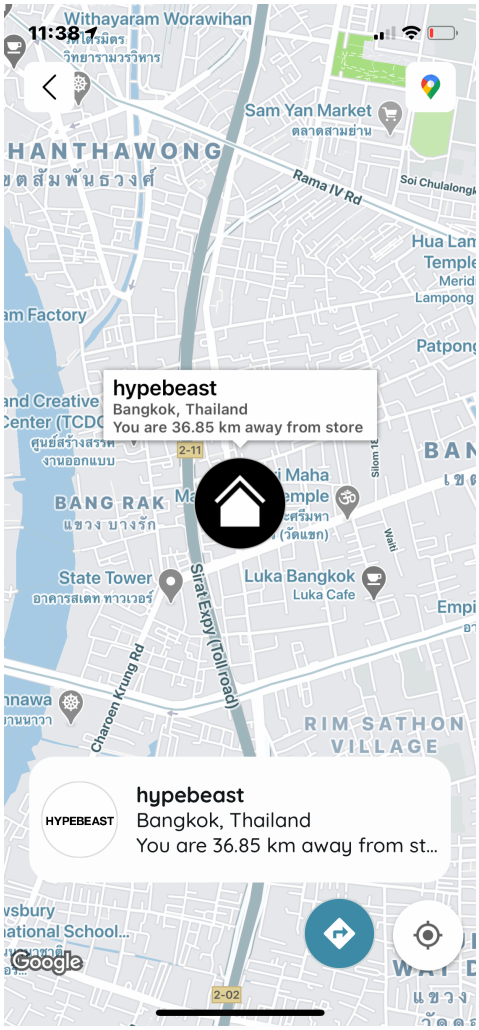

Figure 5.5: Built-in map in Randevoo Application

In addition, the application also offers a drawn route feature to the store location, as mentioned earlier. The drawn route is drawn from the information of routing detail step by step, which provide by Google Direction Service API [11]. To obtain the information from Directions Service, first, the application must collect information from the user. That information includes store location's latitude and longitude and customer current location's latitude and longitude. Once the application obtains all the information, it will start requesting the API by generating an API request HTTPS with the following format:

https://maps.googleapis.com/maps/api/directions/json?origin=\(sourceLocation)
&destination=\(destinationLocation)&mode=driving&key=YOUR_API_KEY"

`mode` can be set differently according to transportation method. But in this project, the transportation method has been set to driving, and it does not offer any option to change at the moment.
`YOUR_API_KEY` is referring to the Google API key, which obtains after registered the project into the Google Service & API.

5.4. Auto Layout

The Randevoo Application layout is design base on the guideline provide by Apple [3] which is Auto layout. Auto layout is a development tool for constructing an adaptive interface. Using Auto layout, the application will define rules that govern the content of the application [3]. Auto layout automatically readjusts the layout according to the specified constraint when certain environment variables are detected. For instance, a button has been set to be the center of the screen. Therefore, if the application is run on a different device with a different screen, the auto layout will keep the button at the same position as what has been set.
Apple provides many auto layout tools for a developer to construct the application's user interface, such as Live Preview, which is known as Storyboard, UIKit, and SwiftUI.
Live preview is one of the most convenient ways to design the user interface. The developer can just drag the component or element they want to use, drop it at the wanted position on the screen, and constraint it. It is sound convenient yet the most power consumption among all the tools. Since it is a Live Preview, it requires a high specification machine to render the result as soon as the element or component is dropped on the screen.
UIKit and SwiftUI are both the tools to construct a user interface, but the difference from Live Preview is both of them do not have a preview of how the user interface gonna look like. Therefore, the application must be run every time to check the constraint and layout. However, it does not require a high specification, unlike Live Preview. UIKit is an old tool that iOS release since the beginning of the first iPhone release, which is around ten years ago. Meanwhile, SwiftUI is a new tool that is released in September 2019. Therefore, there is a lack of SwiftUI documentation that can be used to support the development process. Moreover, there is a lack of components and elements compare to UIKit.
In this project, the user interface development tool that use to implement the application is UIKit. The reason that the project use UIKit is low power consumption and plenty of documentation to assist the development. However, to make the constraint layout design even smoother, the application using an auto layout constraint framework called SnapKit.

5.4.1. SnapKit

SnapKit is an open-source auto-layout framework which use to manipulate auto-layout constraints easily [12]. By using SnapKit, UI view’s layout constraint can be created, update, remove and manage faster compare with UIKit. If the project using UIKit alone as the user interface design tool,

it will take longer compared with the help of SnapKit. The developer needs to write plenty of codes to set a constraint, but using SnapKit, it offers a solution for this problem by providing a concise abstraction of Apple's Auto Layout Constraint system. In figure 4.6, it shows the difference and the number of code between how to set a constraint with UIKit and SnapKit.

```
//UIKIt
child.translatesAutoresizingMaskIntoConstraints = false

NSLayoutConstraint.activate([
  child.leadingAnchor.constraint(equalTo: parent.leadingAnchor),
  child.topAnchor.constraint(equalTo: parent.topAnchor),
  child.trailingAnchor.constraint(equalTo: parent.trailingAnchor),
  child.bottomAnchor.constraint(equalTo: parent.bottomAnchor),
])

//SnapKit
child.snp.makeConstraints { make in
  make.leading.equalToSuperview()
  make.top.equalToSuperview()
  make.trailing.equalToSuperview()
  make.bottom.equalToSuperview()
}
```

Figure 4.6: Comparison Code between UIKit and SnapKit

5.4.2. Methodology

As mentioned earlier, the user interface of the Randevoo application is builtbuilt from two tools: SnapKit and UIKit. Both tools work for a different purpose but in the same jobs, which is built the user interface of the application.

UIKit is being used to set up the behavior of the UI element. The behavior includes color, text, alignment, background color, and shape. Figure 4.7 showss the example code of how the UIKitsets up the element's behaviort. After completely setting up the behavior of the element, the next step will be to make the constraint of the element, which SnapKit handles.

```
//UIKit
var child: UIImageView = {
    let cornerRadius = (UIScreen.main.bounds.height / 40) - 1
    let imageView = UIImageView()
    imageView.layer.cornerRadius = cornerRadius
    imageView.image = UIImage(named: "ProfileIcon")!.withRenderingMode(.alwaysOriginal)
    imageView.clipsToBounds = true
    imageView.contentMode = .scaleAspectFill
    return imageView
}()
```

Figure 4.7: Example code of how the behavior of an element is set

SnapKit will set up the auto layout constraint with easier constraint code which is shown in figure 4.8. The constraint is quick and makes the code look neat and clean. SnapKit provides various constraint,

for instance, size.equalTo is used to set the size of element compared with other element or width.height.equalTo to set up a static width and height of the element. Moreover, the constraint can be set to a position of another element such as top.equalTo is used to set the position of the element whose top always equal to the parameter element that has been set, which is shown as an example in figure 3.10. Moreover, the element can be set to a fixed position such as center.equalTo, which sets the element to be center of the parameter element or using centerX.equalTo and centerY.equalTo to set the horizontal element center or vertical center of parameter element.

```
//SnapKit
view.addSubview(topView)
child.snp.makeConstraints { make in
  make.leading.equalToSuperview()
  make.top.equalToSuperview()
  make.trailing.equalToSuperview()
  make.bottom.equalToSuperview()
}
```

Figure 4.8: Example code of how an element constraint is set using SnapKit

Chapter 6

Evaluation

The project must be proven by the process of justifying the quality of works to evaluate the result and outcome. Evaluation should be crafted to address the specific goals and objectives. The following sections are going to discuss how this project is validated in different aspects.

6.1. Security

To secure the user's account that sign-in in the application, the application does not store user password anywhere on the database. Because every time a user signs in or sign-up, the application never require the user to fill in a password, but the user can sign in or sign up using a Google account or Facebook. The sign-in/sign-up with Facebook or Google is the process of using only the information such as First Name, Last Name from the chosen platform. Furthermore, once use done sign-up with any platform, the Firebase Authorization [13] will securely save that information in the cloud and provide the same personalized experience across all of the user's device. Next time, when user login with either platform that they use to sign-up before, Firebase Authorization will check whether the account exists in the server before and sending the information that saves securely in the cloud to the application. Up to this step, it is clear that the application does not store any information except the user's first name and last name. Therefore, the user account's security is based on how secure does the account from the chosen platform. We advised the user to use two-factor authentication that Google or Facebook provide to protect their account.

However, data encryption does not available in the application yet due to the lack of development time. Therefore, other information such as chat and business store information is stored as raw data in the cloud. But for future work, all the information will be safely encrypted and store in the cloud.

In the iOS operating system, when the application tries to access to user photo library, the operating system’s security and privacy will ask permission whether they allow access or not [14]. In this case, our application does ask the user to access their photo library when they choose to upload any photo into the application. However, the application will access only the photo that the user selected and upload to the database, secure by Google Firebase. The application does not have the right to access and get all the photo from the user library, which is a violence term of policy and privacy of the application. Moreover, the application will also ask to access location when the user uses the built-in map to check the route from their location to the business store location.

Nevertheless, with the new policy on the latest version of the iOS operating system, the application does not allow access to the location when the application is not running, so the user will not need to be aware of tracking their location. In addition, the location neither broadcast nor store anywhere in the

database but use only for the purpose of showing the route from source to destination location only. What's more, those allowances can be disabled in the user's device setting.

6.2. Data Quality

The data stored in Randevoo is the data built to be further integrated with the user information and stored securely in Cloud Firestore. So, the quality of the data used for the application is based on the Cloud Firestore performance. Nevertheless, these are the data quality's verifications criteria, which we considered based on the current integration with Assumption University data sources. Firstly, the application's database is Cloud Firestore, which is own by Google, is one of the giant tech companies with a good reputation in the world. Therefore, the quality of the database is outstanding and durable, with fast read and write speed in just milliseconds. Secondly, the data used to generate information in the application have been saved in the local device to increase the application's performance speed without retrieving data from the database every time any function is performed. Thirdly, the structure of data was designed to be a standard form of NoSQL data model and to perform scalable for some function, for instance, real-time chat, as discussed earlier, which will improve the performance of both application and database.

6.3. Business Logic

In Randevoo development, the main goal is focused on real-world scenario use cases. For instance, UX and UI are being designed with the best principles and existing users' experience to provide this very service to real-world customers. The requirements are gathered from actual users and feature every functionality and component to fit the service. Randevoo app ensures that customers will satisfy with the service and continue to use it without hesitation. Furthermore, all the constraints and requirements are to ensure that users have the best experience and satisfaction. Last but not least, with the completion of this current app, there are still many things to improve to provide a better result in the competitive market.

6.4. Architecture Quality

The prominent architecture of this project has been mentioned in Chapter 3.6, which is the Microservices Architecture. There are several characteristics of Microservices that make this project accomplish the standard. First, it is all about the componentization via service, which means the software should be broken into components and various pieces to work with. Moreover, it is independently replaceable and independently upgradeable. Second, to organize business capabilities, each team can always have some elements and focus on the specific workspace. Third, the unit project can be decentralized data management. This means every service should be responsible for its data and its persistence to maintain the unit. Again, the frontend unit will not be allowed to communicate through

a service data store without authenticating its credentials from Firebase Authentication; it can only communicate with another service through its services API and tokens. As shown above, this Microservices' architecture is the best practice of the enterprise organization's architecture to adopt.

6.5. Source Code Quality

There are various technologies and architecture that this project is being implemented. First, the main design pattern in each unit in this project is Model View Controller (MVC)[15]. Moreover, this architecture is widely being used in many enterprises around the world. Additionally, MVC is a design pattern that divides an application into three major aspects: Model, View, and Controller which explain future as diagram in figure 5.1.

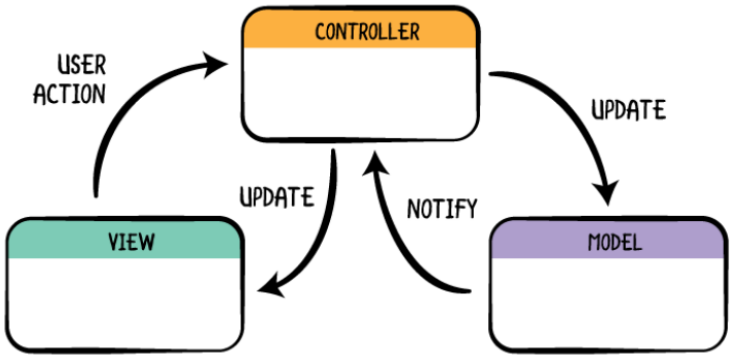

Figure 6.1: Diagram of MVC

In MVC, the input is directed at the Controller first, which is not the view. That specific input might be coming from a user interacting with a user interface. In either case, its controller is interfaced with to trigger the action then it send to view which is the face of the application. The Model is where the data reside. Persistence model object, parsers, managers, and networking code in live in Model.
The combining role of MVC play by an object, making an object, for example, fulfill both the controller and view roles, in which case, it would be called a view controller. In the same way, model which is controller object is a must have to combining the role to become an acceptable design [16].

6.6. Open-Sources Use

There are several open sources that are used by this project. First, MessageKit[8] is being used to ease the workload of designing a message user interface structure. Moreover, it is very flexible to work since every component is customizable. Second, Google Map[10][11] is a Google framework to generate map and route for the built-in application map. Further, to accomplish the latest updated information of the worldwide map, this project will use Google Map over the built-in Apple Map. Third, SnapKit[12] is another main option for the developer to opt for this popular framework to create auto layout constraints when it comes to the user interface. As can be seen, it is one of the fastest tools to

create user interfaces in terms of performance. Moreover, it provides a better way to write clean and neat code compare to the built-in tools.

Chapter 7

Conclusion

Randevoo will change the physical queue-up system of the business stores and deliver a new shopping experience known as Phygital. Additionally, Randevoo also provides a unique platform for the business stores to listing their luxury products to the platform. Not to mention, the application is being implemented with plenty of valuable features. First, real-time chat provides a convenient way for the user to interact without relying on a third-party application. Second, with the built-in Google Maps' help, customers can easily navigate the store's location. Third, the application is also providing comprehensive real-time notifications to alert the users to every action that has been happened in the application. Fourth, UX/UI design is the main objective of this platform to deliver standard perceptions to end-users in conducting business.

The application built base on a design pattern call Model View Controller or MVC. The MVC Pattern is standard for iOS development since it is not difficult to understand and easily update and manipulate the application. With the support of The MVC pattern, the application will ease creating new features in future work.

Also, the Randevoo application is implemented base on a good design user experience. The design aims to reduce the manual input by the user to avoid human error to the application. For instance, the automate generate reservation schedule. Last but not least, this platform will deliver uniqueness in the way of shopping business, while the "**Phygital**" consumer experience is evolving. Randevoo is providing a holistic approach to customer service is the way of the future.

7.1. Future Work

We are looking forward to implementing the application to other platforms such as Android and Web Application. The Randevoo application starts with the iOS development for the end-user; meanwhile, the web application is for the administration to manage the user. Therefore, the application will be expanded to other platforms to extract more user to join the application. First, more features will be implemented in the future, such as notifications to alert the business store's owner when the customer is nearby to prepare the product. This feature is an essential feature for the application. However, due to the new Apple location policy, which does not allow tracking the customer location while not opening the application, this feature will be implemented in the future after we get to understand more about the Apple policy. Second, the application will support multiple store branches. The multiple store branches will allow the user to linked any specific product with all the available branches. Therefore, when the customers check the product, they will have an option to choose the location which they are more convenient in term of travel. Third, the business store owner will also have an option to link their Inventory Management System with the platform at some point in the future. With the Inventory Management System, the business store owner will easily update their products and easily interact between their inventories and the platform. Fourth, cloud technologies, we might be opting on AWS due to the Randevoo platform requirements. And we will be comparing the usage outcome between GCP and AWS. Additionally, there are specific requirements that GCP cannot deliver. Moreover, some costs and benefits will be considered before a final decision to be made. Fifth, the invitation feature will

be implemented for the business owners to invite their customers to join the platform. Not to mention, the customers will be not publicly registered into the platform. They have to have the invitation to join the platform, and phone number verification will be added. Lastly, to improve the application's standard in the commercial, the platform will be implemented Terms of Use, Privacy Policy, Cookies Policy, Content Policy, and User Agreement. Additionally, these are the legal agreements between a service provider and a person who uses the service. And also to protect the platform in the business.

References

[1]. Ecommerce Fashion Industry by Shoptify. Retrieved 2021-03-15
https://www.shopify.com/enterprise/ecommerce-fashion-industry

[2]. Fashion industry report by Shoptify. Retrieved 2021-03-15
https://www.shopify.com/plus/industry-reports/fashion-and-apparel?itcat=plusblog&itterm=ecommerce-fashion-industry

[3] . Human Interface Guidelines by Apple Inc. Retrieved 2021-03-15
https://developer.apple.com/design/human-interface-guidelines/ios/

[4]. User Notifications by Apple Inc. Retrieved 2021-03-15
https://developer.apple.com/documentation/usernotifications

[5]. Microservices by Wikipedia. Retrieved 2021-03-15
https://en.wikipedia.org/wiki/Microservices

[6]. Apple Human Interface Guideline, App Architecture by Apple,
https://developer.apple.com/design/human-interface-guidelines/ios/app-architecture/navigation/

[7]. HorizonCalendar by Airbnb. Retrieved 2021-03-15
https://github.com/airbnb/HorizonCalendar

[8]. MessageKit, open sour Swift Chating application Framework. Retrieved 2021-03-15
https://messagekit.github.io/

[9]. Structure Firestore for scalable chat application by Hoang Minh Bach. Retrieved 2021-03-15
https://levelup.gitconnected.com/structure-firestore-firebase-for-scalable-chat-app-939c7a6cd0f5

[10]. Google Maps Platform Documentation for iOS Development. Retrieved 2021-03-15
https://developers.google.com/maps/documentation/ios-sdk/overview

[11]. Google Direction Service Documentation. Retrieved 2021-03-15
https://developers.google.com/maps/documentation/javascript/directions

[12]. Snapkit, An Auto Layout DSL for iOS and OSX. Retrieved 2021-03-15
https://snapkit.io/docs/

[13]. Firebase Authentication Documentation by Google. Retrieved 2021-03-15
https://firebase.google.com/docs/auth

[14]. Managing Devices & Corporate Data On iOS by Apple. Retrieved 2021-03-15
https://www.apple.com/business/docs/resources/Managing_Devices_and_Corporate_Data_on_iOS.pd

[15]. Model-View-Controller (MVC) in iOS – A Modern Approach by Felipe Laso Marsetti. Retrieved 2021-03-15
https://www.raywenderlich.com/1000705-model-view-controller-mvc-in-ios-a-modern-approach

[16]. Model-View-Controller Design Pattern by Apple. Retrieved 2021-03-15
https://developer.apple.com/library/archive/documentation/General/Conceptual/CocoaEncyclopedia/Model-View-Controller/Model-View-Controller.html