Pet Grooming Booking App
(Groomify)

Name: DAMIEN TAN LEK KHEE

Coventry Student ID: 12672844

Supervisor Name: KAVITHA THAMADHARAN

6000CEM INDIVIDUAL PROJECT PREPARATION

School of Engineering and Technology, INTI

International College Penang

10th November 2023

Table of Contents

List Of Figures

List of Tables

Chapter 1

Introduction

1.1 Introduction

In the current generation, the number of pet owners are increasing daily and it comes with a lot of responsibilities to take care of a pet such as pet grooming. Pet grooming industry has grown massively these years as more pet owners are seeking to maintain the health and appearance of their pets. However, the process of finding and booking a reliable and convenient pet grooming services is too traditional, thus inconvenient. This is because it is time-consuming and frustrating for pet owners to research and book for pet grooming services through phone calls. Fortunately, there is a potential solution to this issue which is the development of pet grooming apps. The pet grooming booking app, Groomify, offers a user-friendly platform that connects pet owners with professional pet groomers. The aim of this project is to design and develop a pet grooming app that provides a range of features and services which makes booking pet grooming services more convenient for pet owners and rolling in business easier for pet groomers. The range of features and services includes appointment scheduling, appointment reminders, service selection and payment integration. By providing a convenient and reliable platform for pet grooming services, this app could potentially improve the overall experience for pet owners and help pet grooming businesses streamline their operations. The content of this report includes the development process of the pet grooming app such as research, design, implementation, testing and evaluation stages, including insights of potential future directions of the app.

1.2 Problem Statement

In the current generation, technologies had been making daily task more convenient for its users such as make appointments. Since people are on their mobile phones these days, the statistics had shown that majority of the population make their appointments through their mobile devices. Based on the article from PhocusWire, Booking.com which is a company that hosts a hotel booking website mentioned that 60% of their users made bookings via mobile,

with about 40% of them coming from its mobile apps (Fox 2022). This has proven that the mobile app is really growing rapidly.

Compared to traditional methods for booking, it is much more reliable and time saving to make appointments for a pet grooming services. This is because the traditional method requires pet owners to research and look for a groomer who suits their demands by going through several internet directories and reviews with no assurance of the calibre of the services provided. Additionally, pet grooming businesses struggle to effectively manage their appointments and schedules, which causes scheduling conflicts and missed appointments. Hence, potentially result in the reduction of business revenue.

In addition, there is a lack of uniformity in the pet grooming sector, with groomers charging differently and having differing degrees of experience. These make it inconvenient as various pet owners may require or prefer pet groomers with different experience as it is time-consuming to conduct research to look for pet grooming services that suit their needs. It may also cause pet owners and pet grooming establishments to become frustrated and unsatisfied, which might lead to a lack of confidence in the dependability and calibre of pet grooming services. By offering a user-friendly platform that links pet owners with reputable and competent pet grooming services and streamlines the appointment scheduling process for pet grooming businesses, the creation of a pet grooming app seeks to address these issues (Benefits of Pet Grooming Software in the Pet Industry | Meetbrandwide n.d.).

1.3 Project Objectives

1. To create a mobile app which allows users to schedule appointments online.

2. To implement reminder feature which reminds users of their scheduled appointments.

1.4 Detailed Research Question

1. What are the current challenges faced by pet owners in finding and booking a pet grooming services that suit their needs?

2. What are the key features and functionalities that pet grooming booking app require?

1.5 Client, Audience, Motivation

1.5.1 Client

For pet grooming businesses, Groomify streamlines their appointment management and scheduling process, as well as expand their customer base and increase revenue. It is often inconvenient and challenging for pet grooming businesses to manage their appointments and schedules, which might lead to overbooking or under booking. Hence, not maximizing the profitability of the business. The pet grooming app provides a user-friendly and efficient appointment management system which could solve this problem. This is because pet grooming businesses would be able to manage their appointments and optimize their workload. Additionally, pet grooming businesses could reach a wider audience of pet owners who are actively searching for reliable pet grooming services through the app.

1.5.2 Audience

The primary audience for the pet grooming booking app includes pet owners who are seeking professional pet groomers. This audience will be motivated to use the app as they could easily find and book reliable pet grooming services for their pets such as, able to manage their pet grooming appointments and receive reminders for upcoming appointments. Pet owners might not have the time to research and book appointments for their pets as they might be busy with work or other commitments. The app provides a convenient and time-saving experience for pet owners as the app will be integrated with a comprehensive directory of trusted pet grooming businesses that can be easily searched and booked at any time of the day. Furthermore, pet owners could manage their pet grooming appointments in one place and a reminder notification will be popped to prevent missing any appointments.

1.5.3 Motivation

The goal to enhance the whole grooming experience for both pets and their owners served as the driving force behind the creation of the pet grooming app. The writer is aware that locating dependable groomers, making appointments, and keeping track of their pets' grooming histories can be difficult for pet owners. By offering a simple user interface that enables pet owners to quickly search for available grooming services, view groomer profiles and client ratings, and schedule appointments at their convenience, the app hopes to overcome these difficulties. The app aims to increase client pleasure, foster customer loyalty, and set Groomify apart from rivals by providing a practical and effective platform.

1.6 Scope

This pet grooming app aims to provide a user-friendly platform for pet owners and pet grooming businesses to connect and interact. A range of features will be implemented in the app which will fulfil the requirements of both parties, including a directory of trusted pet grooming businesses and appointment management system. Additionally, the app will provide features such as pet profiles, which allow pet owners to track their pet’s grooming history.
The scope of the pet grooming app will be a massive improvement compared to the traditional appointment booking system as it provides a platform for pet owners and pet grooming businesses to connect. Pet owners will be able to search for and book appointments with trusted pet grooming businesses in their area and receive reminders for upcoming appointments. Pet grooming businesses however will be able to keep track and manage their schedule and appointments more efficiently, as well as improve their customer base and increase revenue.

One thing is that this pet grooming booking app is made for small scale companies, as it still has a massive room for improvement. The pet grooming software is made to be flexible and adaptable, with the option to later broaden its use and capabilities. There may be chances to introduce additional features and services as the app grows in popularity and draws more users in order to suit the changing demands of the pet grooming sector. The goal of the pet grooming app is to ultimately offer a comprehensive and user-friendly platform for connecting, interacting, and streamlining the pet grooming process for pet owners and pet grooming businesses.

1.7 Conclusion

In conclusion, the pet grooming booking app is a user-friendly and time-saving platform designed for both pet owners and pet businesses. The app is able to simplify and improve the overall pet grooming experience for both parties. By offering a comprehensive directory of trusted pet grooming businesses and a user-friendly appointment management system, the app strives to solves all the inconveniences of both pet owners and pet groomers. The app is scalable and customizable, with the potential to expand its scope and functionality in the future as the pet grooming industry continues to grow. Lastly, the pet grooming booking app saves time and makes it convenient for pet owners, and streamline the management system for pet grooming businesses.

Chapter 2

Literature Review

2.1 Introduction

This section of the report includes a comprehensive examination of the existing research, studies and literature which relates to pet grooming booking app and the pet grooming industry. The data collected will be used to understand more on the current state of pet grooming services, the challenges faced by pet owners and grooming businesses and also the potential benefits that a pet grooming booking app can offer. Scholarly articles, industry reports and relevant literature will be reviewed and analysed, hence identifying key trends, best practices and gaps in the existing knowledge related to pet grooming booking app.

The literature review begins with researching on the challenges faced by pet owners regarding finding and booking a reliable pet grooming services. The writer will take a deep dive on the issues faced by pet owners, such as lack of information on available pet grooming services, inconvenience on scheduling appointments and concerns about the quality and reliability of pet grooming services provided. Furthermore, investigation on the challenges faced by pet grooming businesses will be conducted, such as issues on appointment management, marketing and promotion strategies, including the need for efficient communication with pet owners.

Researches on the potential benefits that a pet grooming booking app could offer to both pet owners and pet groomers includes, the key features and functionalities required for the app. For instance, user-friendly interface, reliable search and booking functionalities, appointment reminders and the ability to give comment and reviews of the pet grooming services. A review will also be conducted regarding the essential features that pet groomers required to manage their appointments efficiently.

Critical evaluation will be conducted on existing literature to thoroughly understand the current landscape of pet grooming booking apps and gain valuable insights on the development and implementation of pet grooming booking apps. The findings of this literature review will act as a guide for the subsequent stages of the project, including the design and development phases

for the app. This is to ensure that the app effectively addresses the identified challenges and meets the needs for both pet owners and pet groomers.

2.2 Terminologies

The pet grooming app, Groomify presents a comprehensive set of terminology that are tailored to the particular needs and specifications of the pet grooming sector. These terminologies have been thoughtfully designed to promote effective scheduling, transparent communication, and smooth interactions between pet owners and groomers. Groomify hopes to deliver a user-friendly and intuitive platform that improves the whole grooming experience for pets and their owners by adopting these terms.

"Grooming Services," which refers to a variety of treatments and activities offered to pets, is one of the key terms in Groomify. These services include, among others, washing, hair trimming, nail trimming, and ear cleaning. Groomify enables pet owners to choose the precise treatments their animals need by offering a comprehensive selection of grooming services, assuring customised grooming sessions catered to each pet's particular need.

"Pricing Details", refers to allowing pet owners a thorough awareness of the expenses related to various treatments by clearly and up-front displaying pricing information for each grooming service. This openness fosters trust and aids pet owners in making selections based on their tastes and budget.

“Appointment Scheduling", refers to the tool integrated into Groomify which makes scheduling grooming sessions easier. Based on the availability of grooming experts, pet owners may effortlessly choose the date and time they choose for the grooming session. This makes scheduling easier and does away with the need for manual coordination, making it hassle-free for both pet owners and grooming specialists.

"Ratings and Reviews", refers to the system incorporated into Groomify, allowing pet owners to express their opinions and comment their grooming experiences. Other pet owners who are looking for dependable and excellent grooming services will benefit greatly from this feature.

Pet owners may choose a grooming expert with confidence by taking into account ratings and reviews, ensuring their animals get the best care possible.

"Notifications and Reminders", refers to the feature of Groomify which ensures that pet owners never forget their scheduled grooming appointments. Pet owners are promptly informed of upcoming grooming appointments and given reminders, which keeps them on schedule and guarantees that their animals receive regular grooming services. The ease and dependability of the app as a whole are enhanced by these notifications.

Last but not least, "Secure Payment", refers to the techniques incorporated by Groomify to ensure the privacy and security of financial transactions. Pet owners can securely pay for grooming services within the app by integrating dependable and secure payment gateways. By doing away with cash transactions, this feature fosters a quick and secure payment process.

2.3 Summary of Existing System

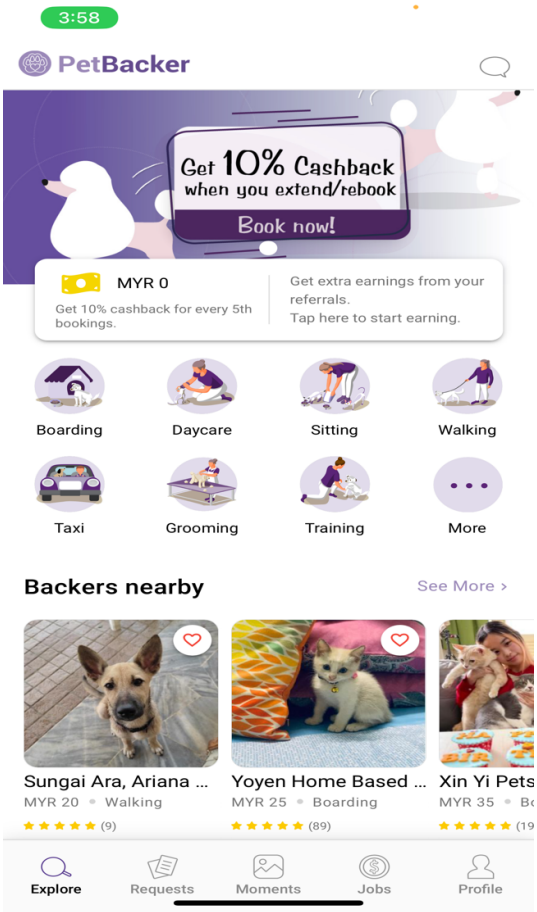


Figure 1: PetBacker App

An new and comprehensive mobile application called PetBacker was created to meet the various demands of pet owners. It functions as a virtual hub that links pet owners with a huge network of dependable and knowledgeable pet service providers. PetBacker provides a wide range of services to make sure that pets receive the care they require when their owners are abroad or unable to take care of them directly, including pet grooming, boarding, pet sitting, and dog walking.

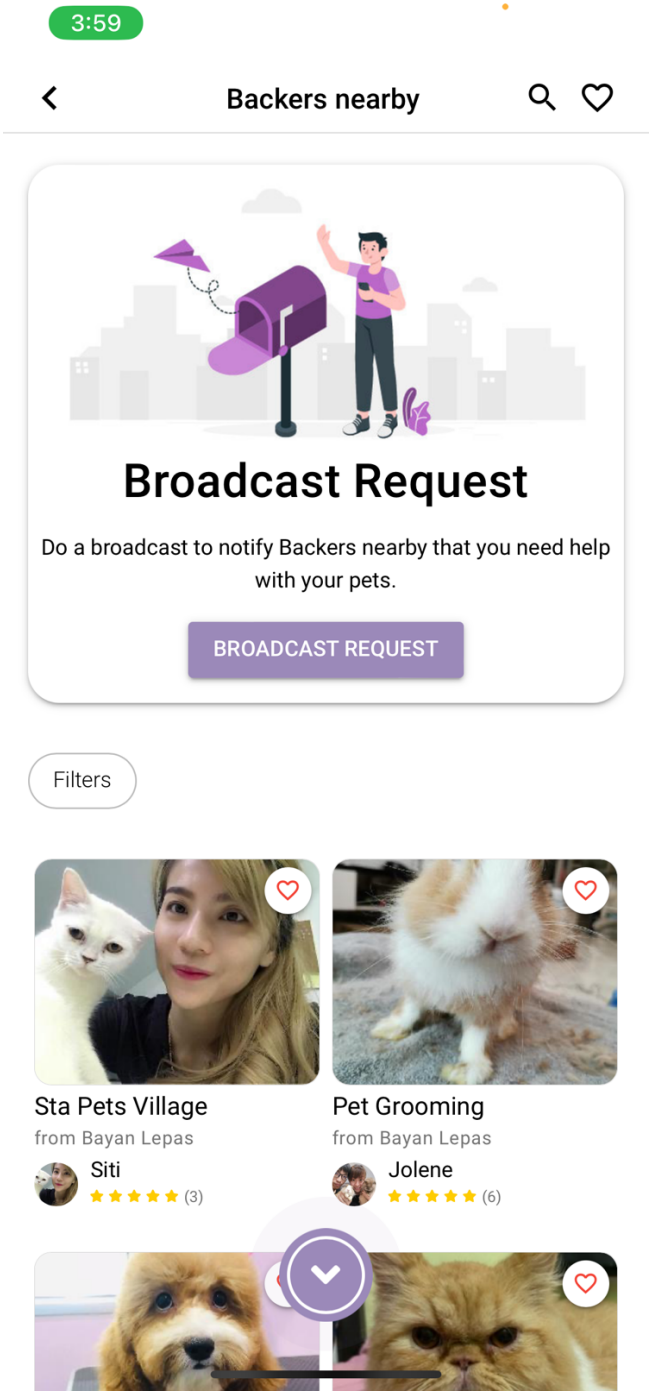


Figure 2: Service Providers

The vast network of pet service providers that PetBacker has is one of its unique advantages. The app connects a community of trained pet care professionals, including groomers, sitters, walkers, and boarders, who have all been screened and confirmed to meet the highest standards. This means that pet owners can rest easy knowing that their furry friends are in the care of competent, trustworthy people who are enthusiastic about their welfare.

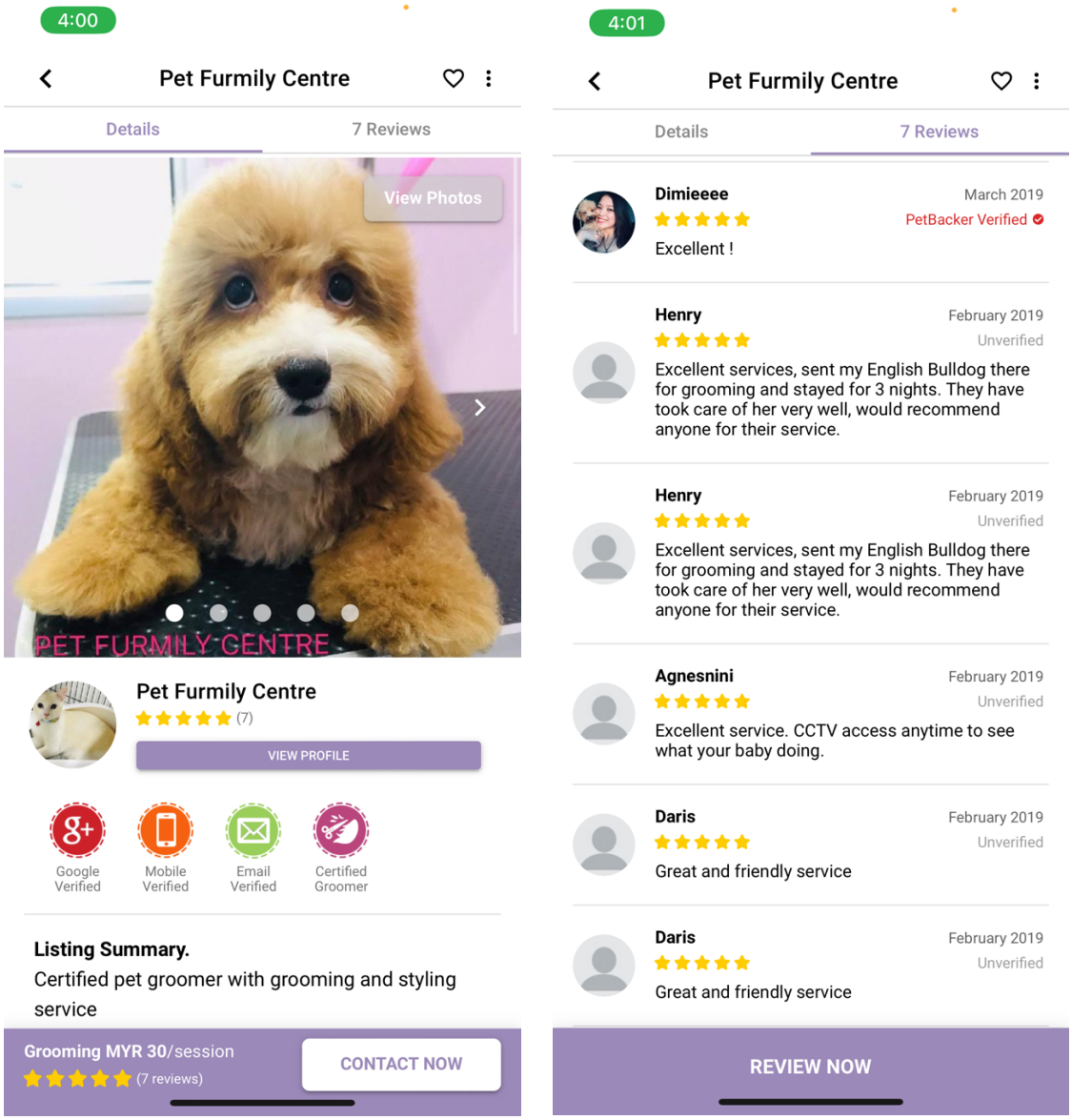


Figure 3: Groomer Details & Comments

The basic principles of PetBacker's design centre on convenience and use. Pet owners may easily search for and connect with service providers thanks to the app's user-friendly layout. Pet owners can gain a thorough idea of the experience and level of care provided by each

provider through thorough biographies and client testimonials. This gives them the ability to choose the expert who is best suited for their dogs' particular needs and to make informed decisions.

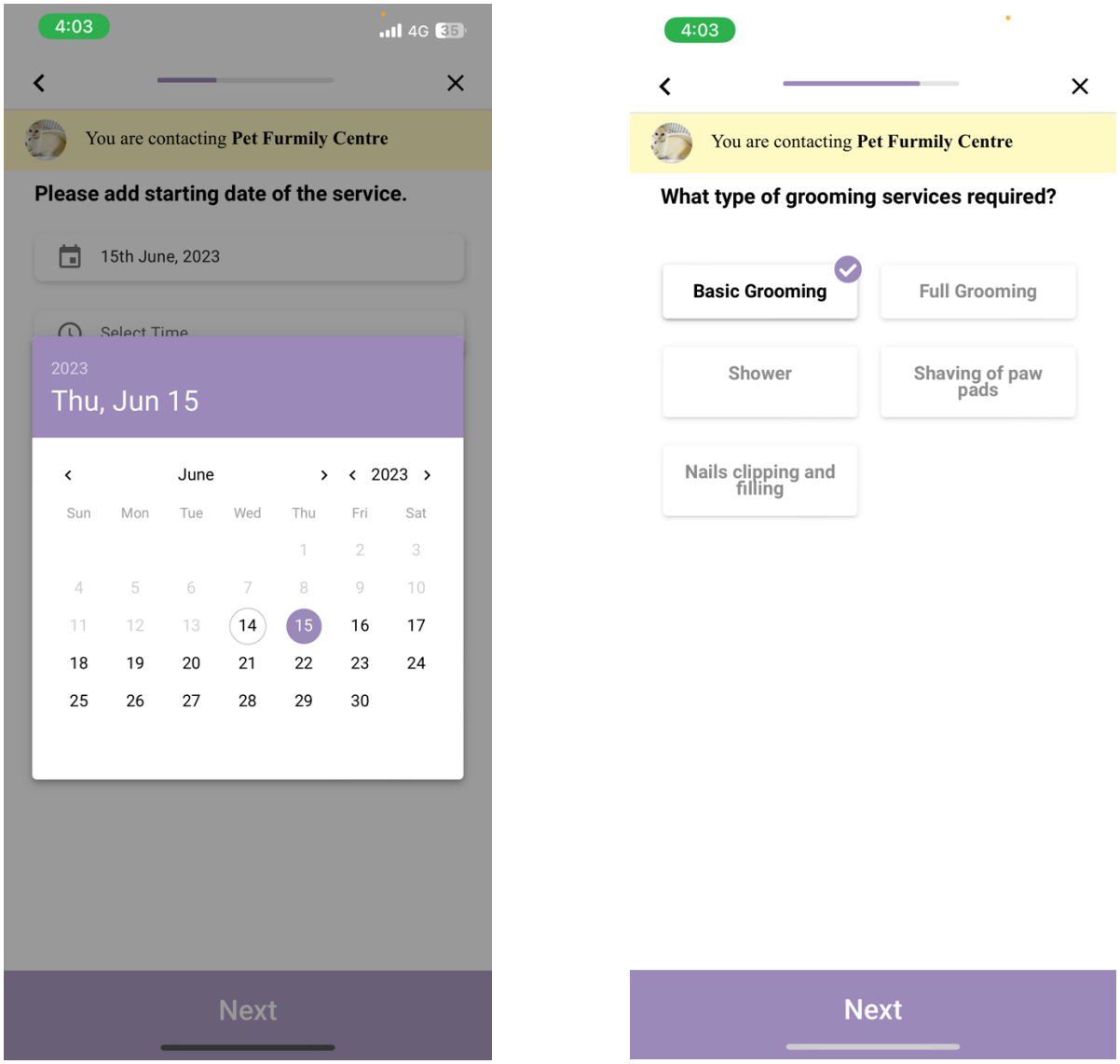


Figure 4: Scheduling System

Pet owners can quickly plan appointments and manage their pets' care needs with just a few taps on their mobile devices thanks to the integrated booking and scheduling system. PetBacker also has a handy reminder tool that notifies pet owners at the right times so they never forget key events or activities for their dogs. This function is especially helpful for time-pressed pet owners who may have a number of obligations and require a dependable system to monitor their pets' schedules.

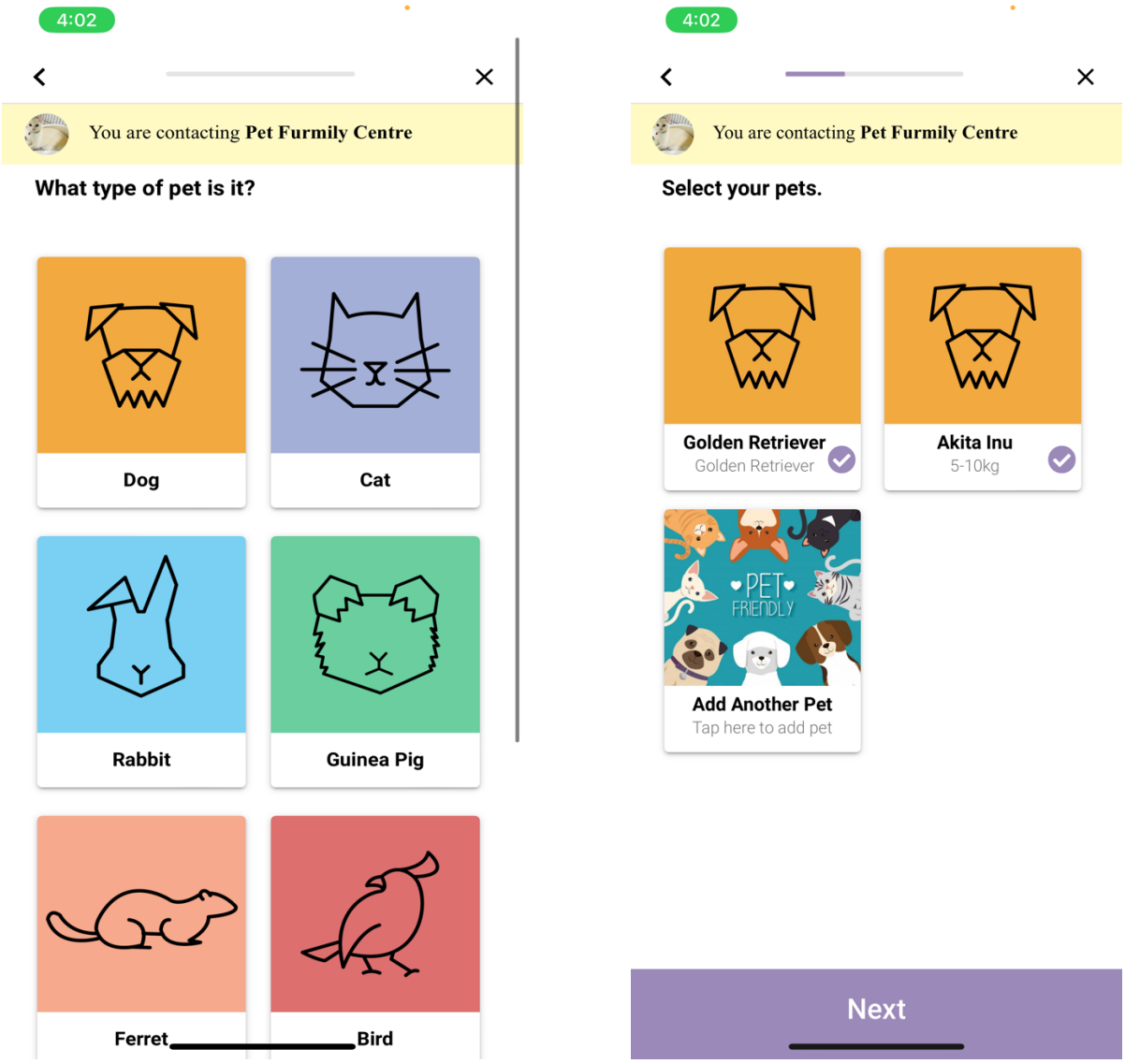


Figure 5: Multiple Pets Feature

PetBacker contains a feature where it allows pet owners to add multiple pets and keep track of them without hassle-free. They will be able to schedule multiple appointments for each pets at the same time, making it convenient to groom multiple pets and saves time.

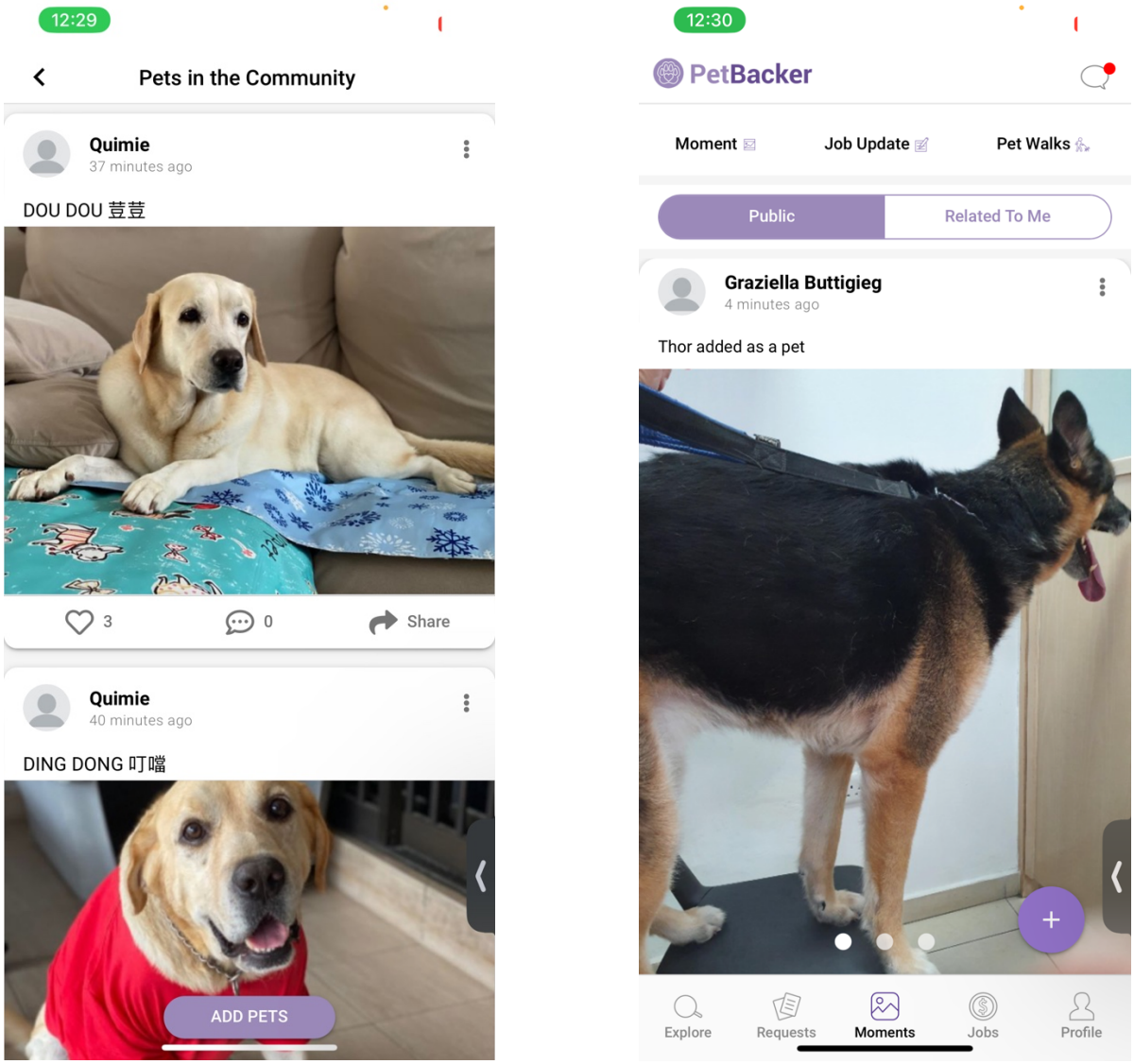


Figure 6: Pets Community

PetBacker does more than just link pet owners and service providers. It also provides a forum for pet owners to interact with one another and build communities of like-minded people who can exchange experiences, suggestions, and guidance. Pet owners can connect with people who share their love and passion for animals thanks to this community aspect, which provides an extra layer of support and involvement.

2.4 Features Derived to Groomify App

The pet grooming booking app, Groomify will be implemented with the key features inspired by the PetBacker app. Groomify intends to offer pet owners and grooming specialists a seamless, effective, and personalised platform for accessing and managing grooming services by utilising these technologies. Let's look more closely at the main technological advantages of Groomify that will revolutionise the pet grooming sector.

By providing a powerful and intuitive mobile application, Groomify makes the most of mobile technology. The app's visually appealing and simple layout makes it easy for pet owners to browse through the several grooming options, examine price information, and simply book appointments. Pet owners can access Groomify's features anytime, anywhere with the help of the mobile app, offering optimum accessibility and convenience.

The app can offer a wide variety of pet services to meet the varied needs of pet owners. This involves, among other things, dog walking, pet sitting, and grooming. Every service should be accompanied by thorough explanations, pricing details, and availability so that pet owners can make educated selections based on the particular needs of their pets. The app turns into a one-stop shop for pet owners by providing a wide range of services, removing the need for them to use several platforms for various pet care requirements.

The pet service app's method for scheduling and booking appointments is an important feature. Pet owners should be able to conveniently arrange appointments for their pets thanks to the system's user-friendly and effective design. Pet owners should be able to tailor their pet care appointments to fit their schedules by having flexible options for choosing dates, times, and preferred service providers. The app should also provide flexibility and adaptation by making it simple for pet owners to monitor and adjust their bookings.

Pet owners can submit comments and discuss their experiences with service providers through the app's review and comment system. This function encourages openness and fosters trust among pet owners. Other pet owners may choose service providers wisely by allowing pet owners to express their opinions and score the services they received. By fostering a sense of accountability and community, the review and comment system guarantees the calibre and dependability of the pet care services made available through the app.

To ensure that pet owners are always aware of their forthcoming grooming appointments, the app includes digital reminders and notifications. Users get timely alerts and reminders, which makes it easier for them to efficiently manage the grooming schedule for their pets. By utilising this function, Groomify encourages routine grooming and helps to improve the general health and welfare of pets.

In conclusion, a PetBacker-inspired pet service app may include a simple user interface, a wide range of service options, a practical booking and scheduling system, reminder notifications, and a review and comment system. By enhancing the user experience as a whole, these improvements give pet owners a solid and effective foundation for managing the care needs of their animals. The app turns into a crucial resource for pet owners looking for reliable and high-quality pet care services because to its simple navigation, extensive service selection, easy booking and scheduling, timely reminders, and transparent feedback.

2.5 Comparison of Development Tools

2.5.1 Database

a. Firebase

Figure 7: Firebase Logo

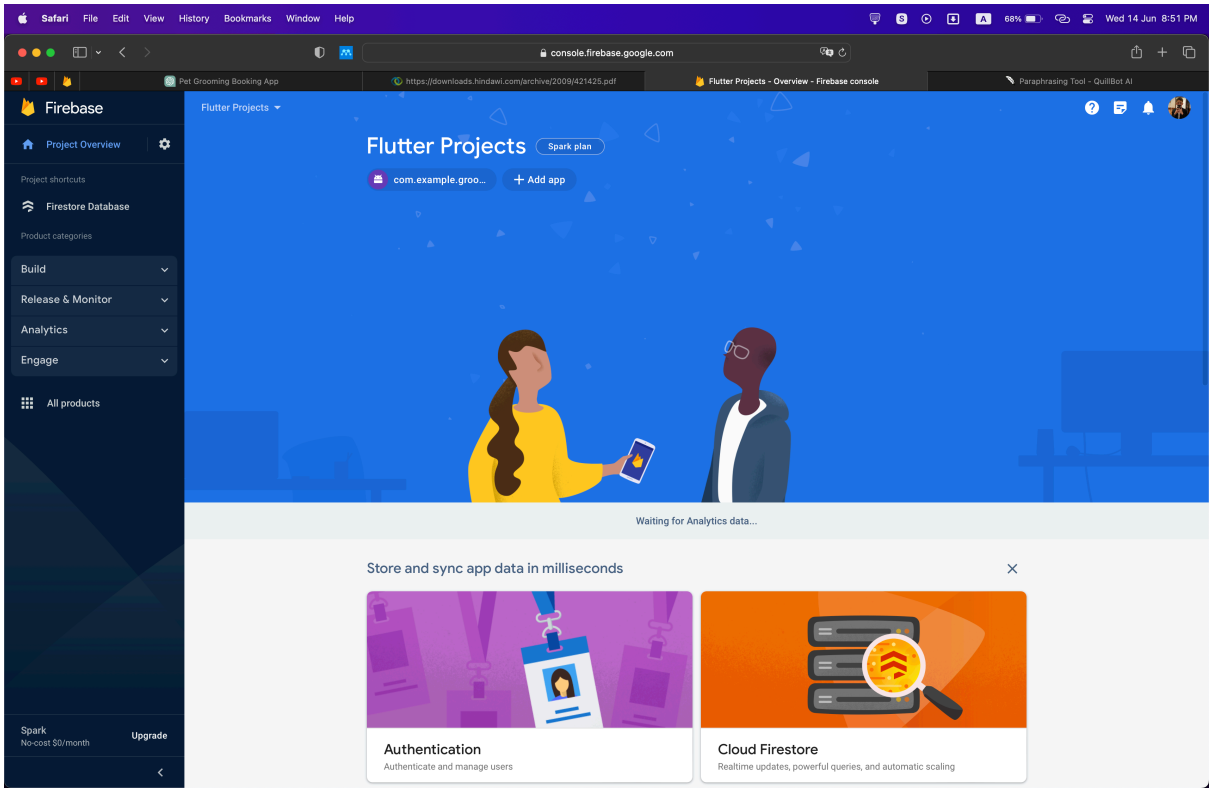


Figure 8: Firebase Interface

According to the journal (The Definitive Guide to Firebase 2017), Google offers Firebase, a comprehensive platform that provides diverse web and mobile application services. Among its features is the Firebase Realtime Database, a NoSQL cloud-hosted database intended for real-time data synchronization. It makes an outstanding choice for your pet grooming app development needs.

The Firebase Realtime Database offers a valuable advantage such as real-time synchronization. All connected clients receive instant updates whenever changes are made, keeping everyone in the loop at all times. This means if a pet groomer adjusts their schedule or a user books an appointment, the information is immediately available to everyone on any device. This seamless and synchronized experience ensures all users stay updated with current information.

Firebase Realtime Database's strong points include scalability and performance. It is designed to handle large amounts of data and concurrent connections, automatically scaling to meet the demands of an application. This ensures optimal app performance, even under high loads, providing a smooth and responsive user experience.

Firebase Realtime Database's offline support is a valuable feature for pet grooming app users. This feature ensures uninterrupted usage even when network connectivity is absent. The changes made by the users while offline saved locally and synchronized with the server automatically once there is network availability. This service guarantees that app users can continue to make updates in an offline environment without any interruption or data loss.

Additionally, Firebase Authentication, Cloud Storage, and Cloud Functions are all seamlessly integrated with Firebase Realtime Database. This enables you to improve your pet grooming app by utilising extra Firebase capabilities and services. For instance, you can utilise Cloud Functions to create server-side logic, Cloud Storage to store media files and photos, and Firebase Authentication to authenticate users.

b. MySQL

Figure 9: MySQL Logo

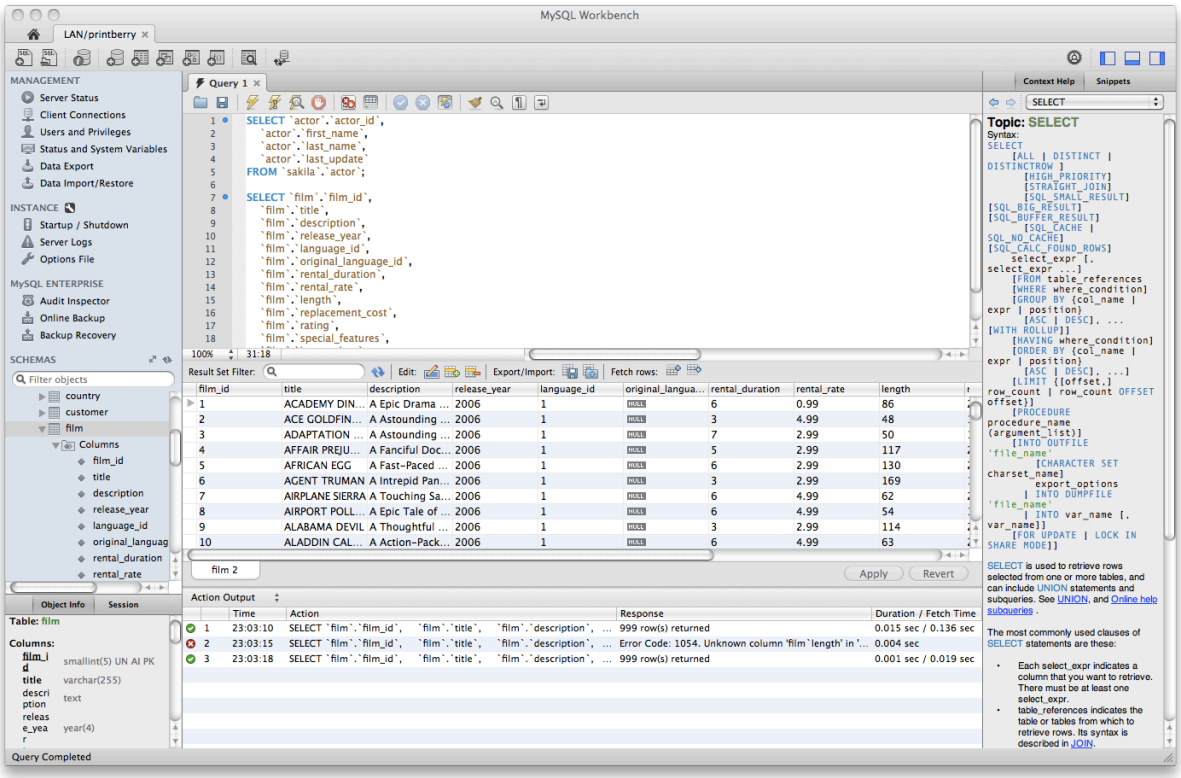


Figure 10: MySQL Interface

Based on the journal (MySQL - Paul DuBois - Google Books n.d.), An crucial tool in many contexts, including traditional uses in business, research, and educational settings as well as applications like those driving search engines on the Internet, a relational database management system (RDBMS) is used in many settings. Nevertheless, many organisations have discovered that, despite the value of a strong database system for maintaining and accessing information resources, it is out of their budgetary grasp. Database systems have always been expensive investments, with suppliers demanding steep prices for both software and service. Additionally, the cost was higher because database engines frequently needed a lot of hardware to operate at all reasonable performance levels. MySQL, which is one of a well-liked open-source relational database management system (RDBMS), as its database. It has several features and advantages that make it a good option for managing data in your application.

The adaptability and scalability of MySQL are two benefits. It has the capacity to manage enormous volumes of data and can be scaled to meet expanding data needs. In order to properly organise and arrange your data, MySQL supports a wide variety of data types and lets you build tables, relationships, and constraints.

The performance of MySQL is an additional advantage. It is designed to be quick and effective, enabling quick data retrieval and processing. Additionally, MySQL has indexing and query optimisation tools that can enhance the speed of your database searches and make sure that your pet grooming application can effectively handle a large number of requests.

Because MySQL is well-supported and has a sizable developer community, it is simpler to access materials, documentation, and assistance. You may seamlessly incorporate it with your pet grooming application because to its broad interoperability with a variety of programming languages, frameworks, and platforms.

Additionally, MySQL offers utilities and tools for managing and administrating databases. It provides command-line tools and graphical user interfaces (GUIs) that let you manage your database, execute backups and restores, and enhance database performance.

c. MongoDB

Figure 11: MongoDB Logo

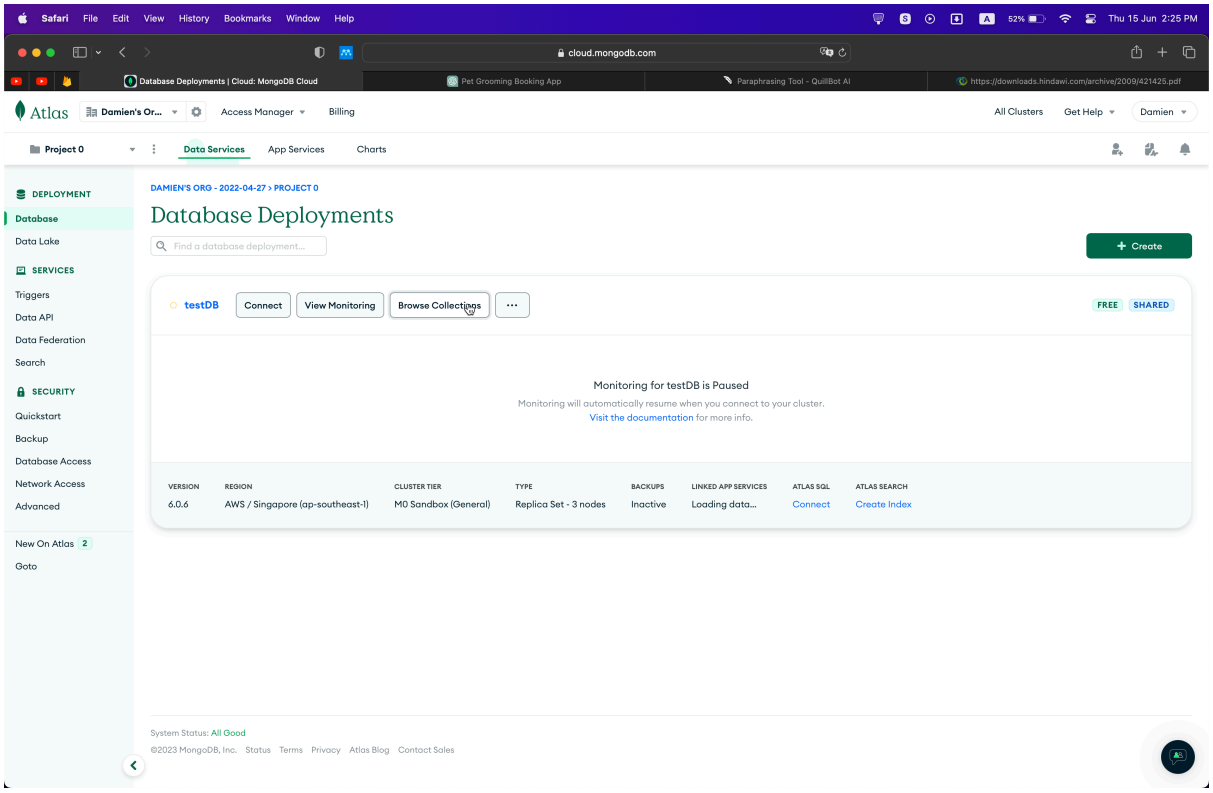


Figure 12: MongoDB Interface

Based on the journal (MongoDB in Action: Covers MongoDB Version 3.0 - Kyle Banker, Douglas Garrett, Peter Bakkum, Shaun Verch - Google Books n.d.), with a document-oriented data model, MongoDB is a well-known NoSQL database that offers various benefits for programmers and applications. MongoDB's versatility is one of its main advantages. You can store documents with various structures in the same collection because it supports dynamic schemas. Because of this flexibility, developers may simply modify and update their data models as the needs of their applications change over time.

The capacity to scale is yet another benefit of MongoDB. Because of its horizontal scaling capabilities, you can spread your data across a number of servers or clusters. With the help of built-in sharding and its horizontal scaling feature, MongoDB is able to efficiently handle enormous datasets and heavy traffic loads. MongoDB can handle rising read/write workloads and support expanding data volumes by splitting the data over several shards.

MongoDB has a robust query language that supports complicated operations for querying and indexing. It makes it simpler to retrieve the desired information from your data by enabling customizable filtering, sorting, and aggregating procedures. Additionally, MongoDB provides indexing, which greatly improves query efficiency by making it possible to quickly retrieve data based on particular fields.

MongoDB's high availability is an important feature. It has built-in replication capabilities that let you spread out numerous copies of your data across many servers or data centres. In the event of a server failure, automatic failover procedures take over to maintain data availability and reduce downtime. This replication ensures data redundancy and fault tolerance.

MongoDB is well-suited for situations where schemas may change often or are not predefined since it excels at handling unstructured or changing data. With MongoDB, you can run ad hoc queries without having to first build a schema, giving you more flexibility when handling dynamic data.

One of MongoDB's other noteworthy features is document validation, which enables you to enforce data consistency and integrity by defining validation rules for your documents. Additionally, MongoDB offers geographic features that make it simple to store and query location-based data.

With a wide ecosystem of tools, libraries, and resources accessible, the MongoDB community is alive and thriving. It provides comprehensive training, community assistance, and documentation, which makes it simpler for developers to understand and use the database successfully.

Features	Firebase	MongoDB	MySQL
NoSQL Database	Yes	Yes	No
Real-time Updates	Yes	No	No
Scalability	Yes	Yes	Yes
JSON Data Model	Yes	Yes	No
ACID Transactions	No	No	Yes
Schema Flexibility	No	Yes	No
Data Replication	Yes	Yes	No
Data Validation	Yes	No	Yes
SQL Support	No	No	Yes
Indexing	Limited	Yes	Yes

Table 1: Comparison of Firebase, MongoDB & MySQL

Because of a number of important factors, Firebase is an appealing option for application development. First off, because of its real-time database's ability to synchronise updates across different clients, it is perfect for applications that need real-time data updates. Second, Firebase provides automatic load balancing and scalability, guaranteeing that your application can manage increasing traffic and data quantities. Thirdly, access to extra functionalities is made possible by the smooth integration of other Google Cloud services. In addition, Firebase offers integrated access control and encryption as well as authentication and security measures. Its NoSQL database, Firestore, additionally provides adaptability, scalability, and potent querying features. Finally, Firebase Hosting makes it easier to deploy online applications. Firebase provides a compelling platform for creating dynamic and secure applications because to its extensive capabilities and solid infrastructure.

2.5.2 Programming Language

a. Flutter & Dart

Figure 13: Flutter Logo

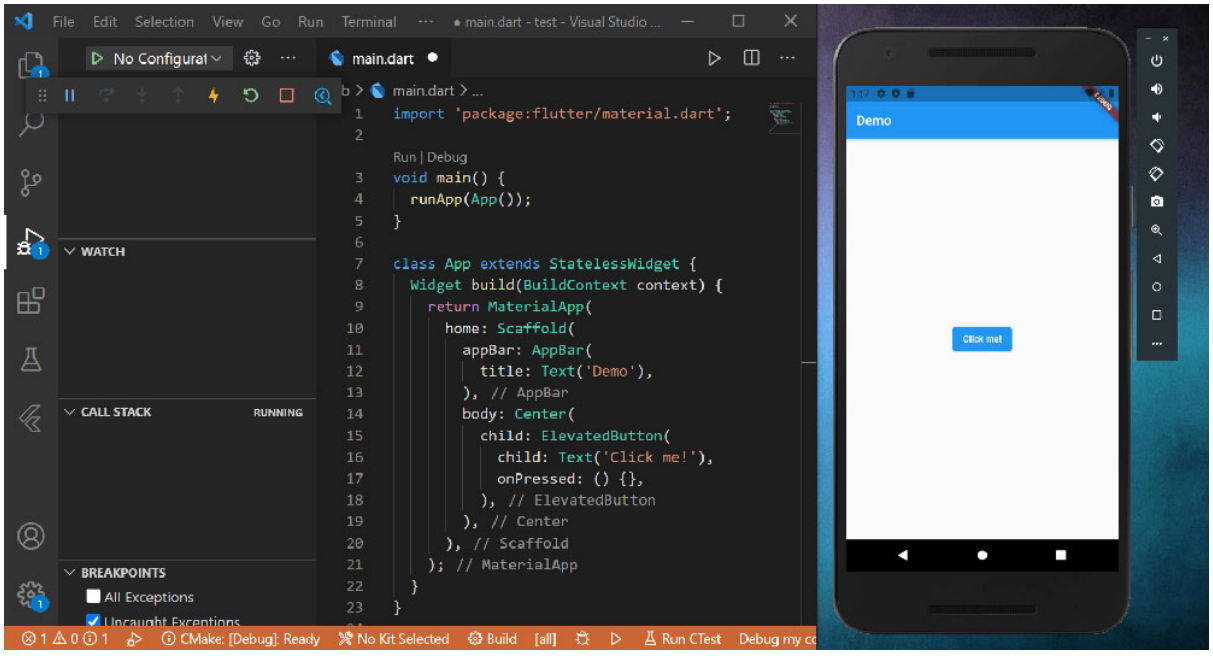


Figure 14: Flutter Interface

Figure 15: Dart Logo

```dart
void main() async{
  Injector.configure(Environment.MOCK);
  runApp(new MyApp());
}

class MyApp extends StatelessWidget {

  // This widget is the root of your application.
  @override
  Widget build(BuildContext context) {
    return new MaterialApp(
      theme: new ThemeData(
        primarySwatch: Colors.blue,
        primaryColor: defaultTargetPlatform == TargetPlatform.iOS
              ? Colors.grey[100]
              : null
      ), // ThemeData
      home: new HomePage(),
    ); // MaterialApp
  }
}
```

Figure 16: Dart Language

From the journal (Sci-Hub | Beginning App Development with Flutter | 10.1007/978-1-4842-5181-2 n.d.), Google created the mobile SDK Flutter, which is open-sourced and focused on enabling everyone to create stunning mobile applications. Flutter makes it simpler than ever to

design mobile apps in a familiar, streamlined manner, regardless of whether you have experience with web programming or native mobile development. Flutter is unique in that it makes it possible to "write once, and deploy everywhere." Flutter apps will roll out to Android, iOS, and ChromeOS as of this writing. Flutter apps will soon be available as desktop and web applications for all of the major operating systems. Flutter is a fully comprehensive SDK for building applications, to put it briefly. It's a platform that offers a rendering engine, UI components, testing frameworks, tooling, router, and many other capabilities you'll need to build applications. You can now concentrate on the intriguing issues in your software as a result. The domain functioning can be your exclusive area of attention while everything else is taken care of. The benefit that Flutter offers is astounding.

Based on the journal (Flutter for Beginners: An Introductory Guide to Building Cross-Platform ... - Alessandro Biessek - Google Books n.d.), Flutter framework's foundation is made on the Dart programming language. For the optimal developer experience and to enable the development of amazing mobile applications, a modern framework like Flutter needs a high-level modern language. Google created the Dart programming language, which may be used to create desktop, server-side, web, and mobile apps. Flutter apps are created using the programming language Dart, which gives developers the finest experience possible while building sophisticated mobile applications.

Dart's object-oriented structure, which enables programmers to organise code using classes, objects, and inheritance, is one of its main features. This improves code reuse and makes it simpler to cooperate on projects. It also makes it easier to structure and manage code.

Static typing is another feature of Dart that enables compile-time type verification and aids in the early detection of mistakes. This feature can improve code dependability and lower the frequency of runtime errors, resulting in programmes that are more reliable and stable.

Another crucial component of Dart is asynchronous programming. Using constructs like async/await and Futures, Dart's native support for asynchronous operations enables programmers to create non-blocking code that can effectively handle tasks like network requests or file operations. This makes it possible to develop programmes that are quick and efficient and can manage several activities at once.

Dart gains much greater capability when used with the Flutter framework. Flutter makes use of Dart's characteristics to create stunning and engaging user interfaces for a variety of platforms, such as iOS, Android, the web, and desktop. Dart and Flutter enable code sharing between platforms, cutting down on the time and effort required for development.

Dart gains from a thriving ecosystem and a growing developer community. Through forums, documentation, and libraries, the Dart community actively contributes to the language and provides support. The ecosystem provides a wide selection of packages and tools that can enhance Dart's functionality and make routine development activities easier.

The Hot Reload functionality of Dart and Flutter is one of its best qualities. Developers can make changes to their code and immediately see the improvements reflected in the active application thanks to Hot Reload. As a result, the development process is greatly sped up, and speedy experimentation and iteration are made possible.

b. Java

Figure 17: Java Logo

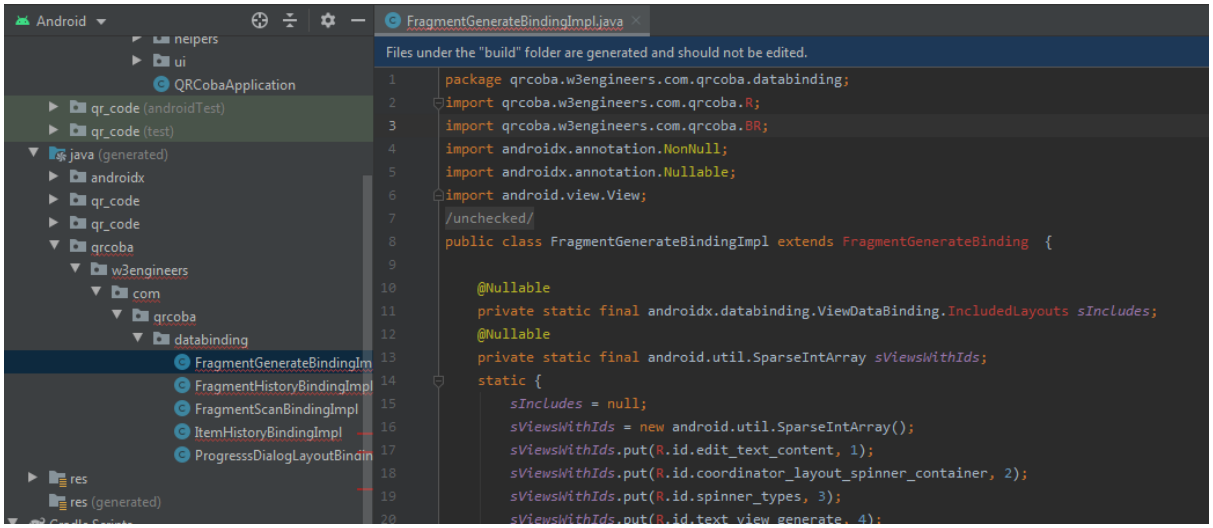


Figure 18: Java Language

Based on the journal (Programming Android - Zigurd Mednieks - Google Books n.d.), as an object-oriented language, Java places more of an emphasis on objects—combinations of data—and procedures for manipulating those objects. The fields (data) and methods (procedures) that make up an object are specified by a class. A Class is a specific type of object that serves as the template from which other objects are built. Classes serve as the cornerstone of Java's type system, which enables programmers to specify arbitrary levels of complexity in terms of an object's state and behaviour. Toes may inherit from other types in Java, as they can in the majority of object-oriented languages. A class is said to subtype or be a subclass of its parent if it inherits from another class. In turn, the Parent class may be referred to as the Subtype or Superclass. The base type for those subclasses is a class that has a variety of various subclasses. Within the class, both fields and methods have a global scope and may be accessed from outside the object by using a reference to a class instance.

Java provides a number of essential characteristics that make it suited for creating mobile applications. First and foremost, it is platform-independent, allowing Java code to execute on a variety of operating systems, including desktop environments, iOS, Android, and RoboVM. This saves time and effort by allowing developers to write code once and deploy it across various platforms.

Java's object-oriented programming (OOP) paradigm is one of its advantages. Java enables programmers to write reusable, modular code through the use of classes, objects, and

inheritance. Complex mobile app projects can be more easily managed thanks to OOP concepts, which encourage code organisation, maintainability, and extensibility.

Java offers a wide variety of libraries and frameworks that make it easier to create mobile applications. For instance, since the Android platform is based on Java, developers may access a wide range of APIs, libraries, and tools through the comprehensive Android SDK (Software Development Kit) to create feature-rich and engaging Android apps.

Java also enables multithreading, enabling programmers to manage several tasks at once. This is especially helpful for mobile apps that need real-time updates or background tasks, such messaging apps or apps that download data from distant sites.

c. Kotlin

Figure 19: Kotlin Logo

```
buildscript {
    ext.kotlin_version = '1.1.3-2'
    ext.jUnitVersion = '4.12'
    ext.supportVersion = '26.0.0-beta2'
    ext.espressoVersion = '2.2.2'

    repositories {
        google()
        jcenter()
    }
    dependencies {
        classpath 'com.android.tools.build:gradle:3.0.0-alpha6'
        classpath "org.jetbrains.kotlin:kotlin-gradle-plugin:$kotlin_version"
        classpath "org.jetbrains.kotlin:kotlin-android-extensions:$kotlin_version"

        // NOTE: Do not place your application dependencies here; they belong
        // in the individual module build.gradle files
    }
}
```

Figure 20: Kotlin Language

Based on the journal (Android Development with Kotlin - Marcin Moskala, Igor Wojda - Google Books n.d.), Kotlin is a contemporary, strictly typed, Android-compatible language called Kotlin resolves several Java issues like null pointer exceptions and too verbose code. Languages like Swift, Scala, Groovy, C#, and many others served as inspiration for Kotlin. Kotlin was created by JetBrains experts based on a study of both developers' experiences, the best usage recommendations (the most crucial being clean code and effective Java), and usage statistics for this language. Other programming languages have been thoroughly investigated. Kotlin makes a concerted effort to avoid the flaws of other languages and capitalise on their best traits. Working with Kotlin makes it clear that this is a sophisticated, well-designed language.

By enhancing code quality and security and enhancing developer productivity, Kotlin elevates application development to a whole new level. Google announced official Kotlin support for the Android operating system in 2017, however the Kotlin language has existed for a while. It has a thriving community, and the use of Kotlin on the Android platform is already accelerating. We can characterise Kotlin as a secure, expressive, succinct, adaptable, and tool-friendly language with excellent Java and JavaScript interoperability.

Based on the journal (Kotlin in Action - Dmitry Jemerov, Svetlana Isakova - Google Books n.d.), the null safety feature of Kotlin is one important benefit. Null pointer exceptions, a typical cause of defects in Java, can be avoided with Kotlin's built-in null safety support. Kotlin ensures safer handling of null values and lowers the danger of runtime crashes brought on by null references by differentiating between nullable and non-nullable types.

The principles of functional programming are also used in Kotlin. It has substantial support for immutability, lambda expressions, and higher-order functions. Developers can use functional programming principles, write more expressive and succinct code, and write code that is easier to comprehend and maintain as a result.

The coroutines in Kotlin are one of its distinguishing qualities. Programming asynchronously and concurrently is made easier by the use of lightweight constructs called coroutines. They

make asynchronous operations simpler to handle and reason about by enabling developers to create asynchronous code in a sequential and synchronous form. Coroutines are particularly helpful for managing asynchronous tasks more effectively and logically, including network requests, database operations, and other tasks.

Excellent tooling support is available for Kotlin, including a potent IDE plugin for IntelliJ IDEA, Android Studio, and other well-known IDEs. Code completion, refactoring tools, and error highlighting are just a few of the IDE support features that improve developer experience and productivity. Additionally, Kotlin is a language that is officially supported for creating Android apps, allowing for seamless integration into the Android developer community.

Kotlin encourages compatibility with current Java code. It is simple to connect with Java, enabling developers to reuse Java libraries and frameworks and progressively switch current applications over to Kotlin. Because of its interoperability, Kotlin is a sensible option for programmers who wish to use their existing Java codebase while taking use of a cutting-edge programming language.

Last but not least, Kotlin benefits from a vibrant community. The community actively contributes libraries, frameworks, and tools to improve the Kotlin ecosystem as a result of its rising popularity. It is simpler for developers to acquire and use Kotlin because there are learning tools, documentation, and community assistance available.

Feature	**Dart**	**Java**	**Kotlin**
Easy to learn	Yes	Yes	Yes
Interoperability	Yes	Yes	Yes
Concise syntax	Yes	No	Yes
Null safety	Yes	No	Yes
Functional programming	Yes	No	Yes
Coroutines	No	No	Yes
Excellent tooling support	Yes	Yes	Yes

Android development support	Yes	Yes	Yes
Performance	Yes	Yes	Yes
Community support	Yes	Yes	Yes
Large codebase	No	Yes	Yes
Popularity	No	Yes	Yes

Table 2: Comparison of Dart, Java & Kotlin

Dart is a strong and flexible programming language that is especially well suited for creating mobile apps with the Flutter framework. It is a desirable option due to its cross-platform capabilities, robust performance, static typing, concise syntax, solid tools support, and active community. Developers may produce apps that work flawlessly on the iOS and Android platforms by using the Dart programming language. The straightforwardness, abundance of tooling, and welcoming community of the language facilitate a more efficient development cycle and quicker iteration. Additionally, the writer is experience in Flutter and Dart thus making it a better choice. Overall, the experience in Flutter and Dart allows the writer to create top-notch mobile apps with a compelling collection of capabilities and advantages.

2.5.3 Operating System

According to the journal (Wukkadada et al. 2015), An operating system utilised in mobile devices is called a mobile operating system. This operating system is essentially a lightweight operating system that uses less memory and storage. This operating system, sometimes referred to as mobile OS or mobile OS, can be used in PDAs, tablets, smartphones, and other devices. All the features of a personal computer are combined into mobile operating systems. A touchscreen, cellular Bluetooth, WI-FI, GPS mobile navigation, camera, video camera, speech recognition, music player, voice recorder, near field communication, and infrared blaster are all included in mobile operating systems. Multiuser, multiprocessing, and multitasking are aspects of mobile operating systems. Mobile communication is a feature of mobile operating systems. Currently, the most popular mobile OS are android and ios.

a. Android

Figure 21: Android Logo

Figure 22: Android Interface

Android is a collection of software that consists of a mobile operating system, middleware, and important applications. On September 23, 2008, Android made its debut. Both an operating system and a software platform, Android is. Google created the Android mobile operating system, which is based on the lightweight Linux operating system. The Android operating system enables programmers to create applications in a language similar to Java that use Java libraries created by Google. The Android architecture is made up of the Linux Kernel, Android Runtime, and Core Libraries. Android stores data using SQLite. All types of network connectivity, including GSM, EDGE, 3G, LTE, and WiFi, are supported by Android. Due to its Linux OS foundation, it comes preinstalled with all necessary drivers and supports OTG (one the go) for any external devices. Voice recognition and multiple users are also supported. Alpha was the first Android version, while lollipop is the most recent.

Being an open-source mobile operating system, Android has a number of benefits that make it a popular option for both users and developers. First off, Android has a far bigger market share than other operating systems, giving developers access to a huge user base and more opportunities to connect with their target market. Additionally, Android runs on a variety of devices made by different companies, enabling developers to support a range of hardware requirements and user preferences.

Support for numerous programming languages, particularly Java and Kotlin, is one of Android's advantages. With this flexibility, developers are able to select the language that best suits their needs or preferences for their project, resulting in quick and easy development cycles. Additionally, Android has a wide range of customisation choices, allowing designers to create user interfaces that are aesthetically pleasing and extremely interactive. The option for users to customise their Android smartphones results in a more interesting and customised user experience.

The seamless integration of Android with Google services is another benefit. Developers can improve the functionality of their applications and give customers simple access to well-liked and widely-used services by utilising services like Google Maps, Gmail, and Drive. The whole user experience is improved as a result of this integration, which also simplifies the development process.

Developers also have access to the most recent tools, technologies, and features thanks to Android's dedication to constant innovation and frequent updates. This enables them to maintain their leadership position in mobile app development and take use of new tools to produce cutting-edge applications.

b. IOS

Figure 23: IOS Logo

Figure 24: IOS Interface

IOS is a mobile operating system that Apple Inc. created for the iPhone. Both the iPad and the iPod Touch were used to deploy it. IOS originally became available on June 29, 2007. IOS was created particularly for Apple hardware and is descended from Mac OS X, with which it shares the Darwin foundation. As a result, IOS is a Unix-based operating system by nature. The Media layer, the Cocoa Touch layer, the Core OS layer, and the Core Services layer are IOS' four abstraction layers. IOS also supports every network connectivity type, including GSM, EDGE, 3G, LTE, and WiFi. A distinctive platform for sensor applications is offered by IOS hardware. Using Apple IOS to create sensor applications has a number of benefits over conventional methods. It is challenging for designers to provide dependable communication and useful user interfaces while using an IOS-based accessory. By not using the system's communication and

processing components, the problem can be avoided. Here, the term "accessory" essentially refers to a connection to the IOS hardware. Both the hardware and the sensor components would be included in this accessory. We will also examine the automatic collection of failure data from the IOS platform in this research. Due to its closed-source nature, gathering failure data is not simple. IOS 1.0 was the initial version, while the most recent version, IOS 8.0, has more functions.

The operating system that Apple created for its mobile devices, iOS, has many benefits that make it a popular option for users and developers. First of all, iOS is renowned for integrating seamlessly with Apple hardware, creating a highly optimised and effective user experience. Fast performance, fluid animations, and long battery life are all features that iOS devices are able to offer because to the close integration of hardware and software.

iOS places a lot of emphasis on privacy and security, which is one of its main benefits. User data is safeguarded and applications are reliable because to Apple's tight app review policies, app sandboxing, and other strong security features. Because of its focus on security and privacy, iOS is a top option for organisations and individuals who value data security.

Additionally, iOS delivers a standardised user interface that is consistent across all devices, giving users a comfortable and simple experience. Apple's high design standards and quality control lead to applications that are aesthetically pleasing and easy to use. Because developers can only build and test programmes for a finite range of device models and screen sizes, this uniformity also streamlines the development process.

Advantages	**Android**	**IOS**
Open-source	Yes	No
Customizability	Yes	No

Hardware variety	Yes	No
App distribution	Multiple app stores	App Store only
Fragmentation	Yes	No
Development tools	Java, Kotlin, C++	Swift, Objective-C
Market share	Higher globally	Lower globally
Price range	Wide range of devices at different price points	Limited range of premium devices
Ecosystem	Integrated with Google services and third-party platforms	Integrated with Apple services and limited third-party platforms
User base	Diverse user base across different demographics	More affluent and tech-savvy user base

Table 3: Comparison of Android & IOS

In this project's environment, Android has a number of advantages over iOS. First off, because it is open-source, Android gives developers more freedom and customization choices to adapt the programme to certain needs. Furthermore, Android offers a variety of hardware alternatives, making it compatible with a variety of devices and price points. Android gives more freedom in terms of app distribution thanks to its availability in several app stores. The diversified user base of Android and its connection with a number of Google services and platforms make it suited for reaching a wider audience, despite the difficulties that fragmentation might cause. Android would ultimately offer greater customization, interoperability, and market reach for this project.

2.5.4 Frameworks

a. Figma

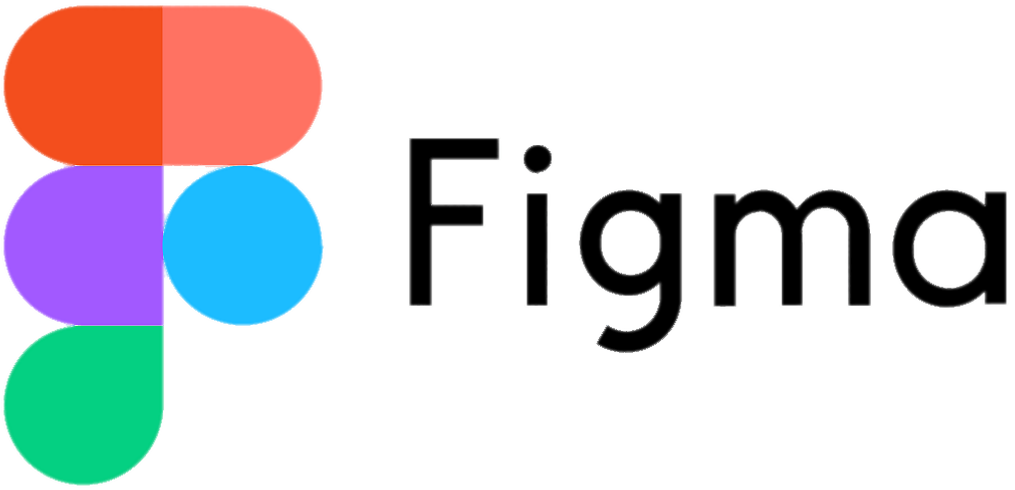


Figure 25: Figma Logo

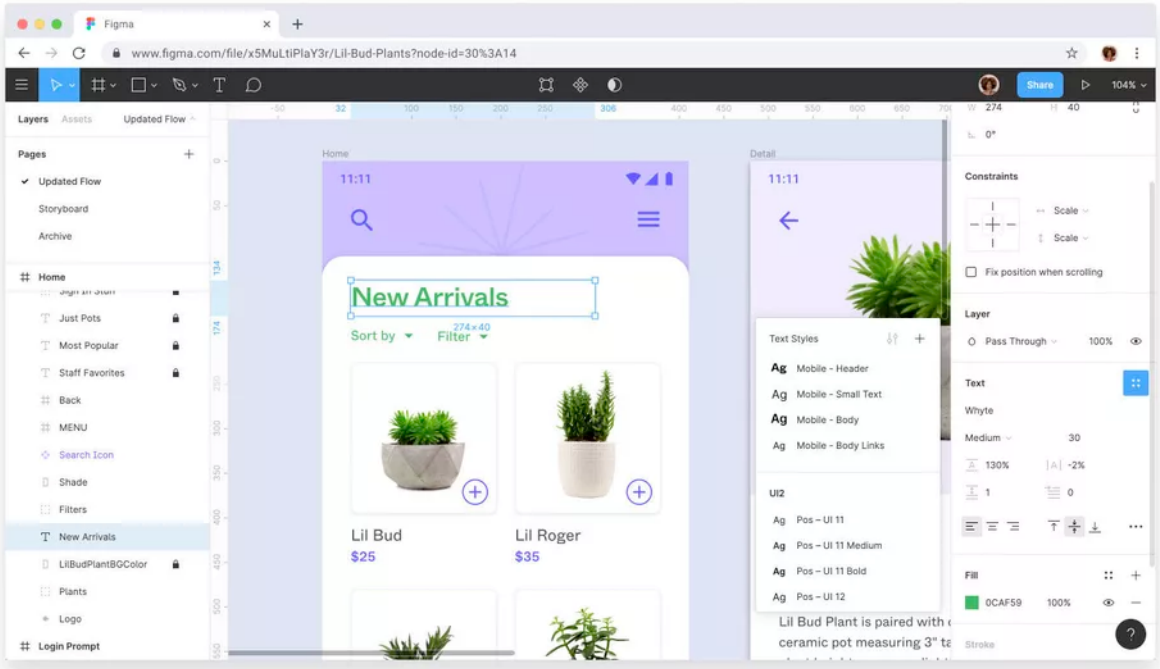


Figure 26: Figma Interface

Based on the journal (Designing and Prototyping Interfaces with Figma: Learn Essential UX/UI ... - Fabio Staiano - Google Books n.d.), Figma is a powerful real-time collaborative tool that can easily replace several design applications. In other words, it is a well-liked web-based design and prototyping tool that designers and teams use to make interactive prototypes, wireframes, and user interfaces. It has a number of benefits that make it a popular option for many design projects.

First of all, Figma is a teamwork solution that enables seamless teamwork and real-time collaboration. Because multiple users can work on a project at once, it is perfect for design teams collaborating on challenging tasks. The option to provide comments and input while viewing changes in real-time improves collaboration and speeds up the design process.

The fact that Figma is cloud-based is another benefit. There is no requirement for software installation or ongoing updates because Figma is web-based. Design files are conveniently available from any location with an internet connection thanks to cloud storage. Version control difficulties are eliminated, and designers may work on their designs using various devices thanks to this.

A complete range of design and prototyping features are also available in Figma. It offers a variety of capabilities, such as vector editing, text style, and shape manipulation, for generating and modifying design elements. Designers may more easily communicate and evaluate design concepts thanks to interactive prototyping tools that allow them to construct clickable prototypes and replicate user interactions.

Figma also has a sizable and vibrant design community. Users have access to a sizable library of design tools, templates, and UI kits that have been made available by the community, which helps to streamline the design process and provide opportunities for inspiration and knowledge sharing.

Finally, it's important to note Figma's platform-independent strategy. It operates without a hitch on Windows and Mac operating systems, guaranteeing interoperability with various environments and letting designers select their favourite operating system without any restrictions.

b. Adobe XD

Figure 27: XD Logo

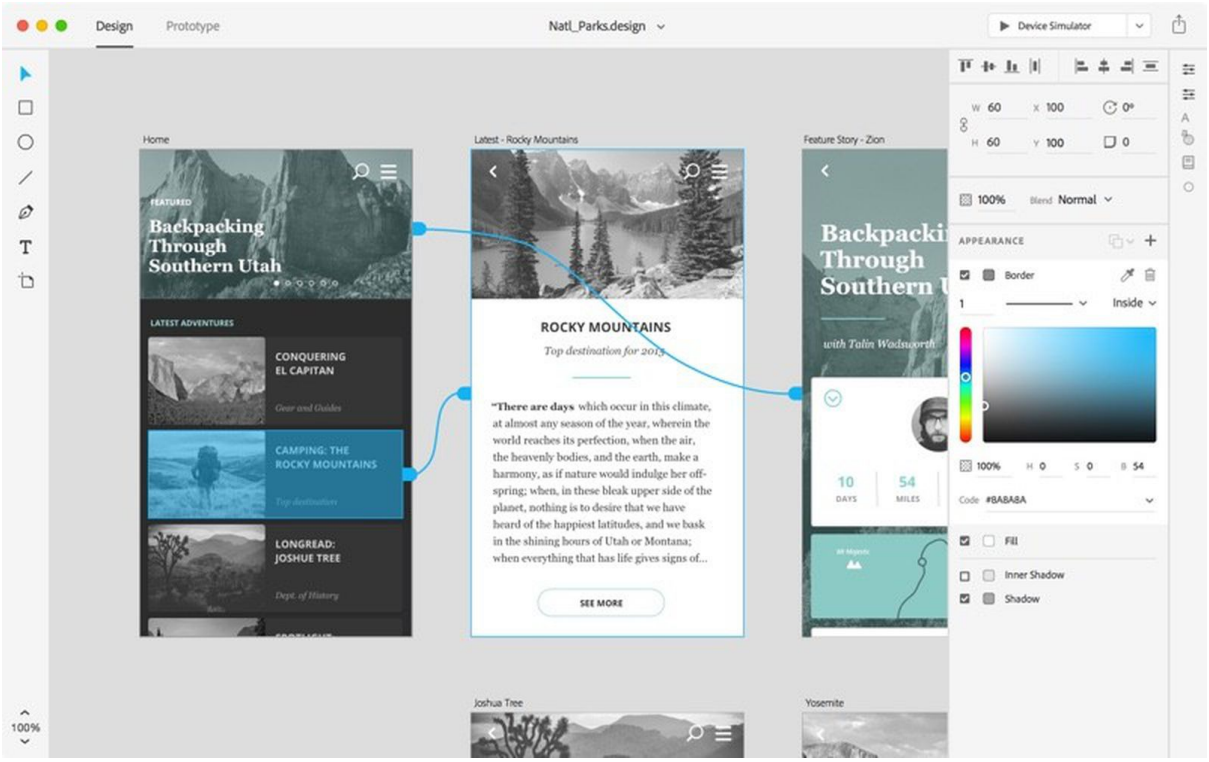


Figure 28: XD Interface

Based on the website (What Is Adobe XD and What Is It Used For? n.d.), a robust and user-friendly vector-based experience design platform, Adobe XD provides teams with the tools they need to jointly create the best experiences in the world. It provides designers with a well-liked design and prototyping tool, to make user interfaces, interactive prototypes, and user experiences. It has a number of benefits that make it a popular option for many design projects.

According to (Adobe XD Review: Expanding Your Design Toolbox | Toptal® n.d.), the strong prototyping features of Adobe XD are one of its main benefits. Designers can easily define interactions, transitions, and animations while creating interactive prototypes. Prior to moving on to the development phase, designers can gather feedback and make the required iterations by testing and validating their concepts using the built-in preview tool. By streamlining the design process, a more user-centric strategy is ensured.

A comprehensive collection of design features and tools are also available through Adobe XD. To maintain consistency throughout the project, designers can develop and alter UI elements, use different styling options, and manage design components. It supports various third-party plugins like Runner, Color Accessibility Check, Magic Mirror, Auto Animate, etc. Designers

can also create adaptive designs that can scale across various screen sizes and devices thanks to the responsive resize function.

Adobe XD also excels in the area of collaboration. Designers can collaborate in real time and receive comments and feedback by sharing their design files with team members or stakeholders. Co-editing design files at the same time improves teamwork and speeds up the review and approval process.

Lastly, designers may access a variety of resources, including UI kits, icons, and plugins, through the active design community that Adobe XD offers. The design process is further improved by this community-driven environment, which fosters inspiration, knowledge sharing, and the exchange of design assets.

c. Sketch

Figure 29: Sketch Logo

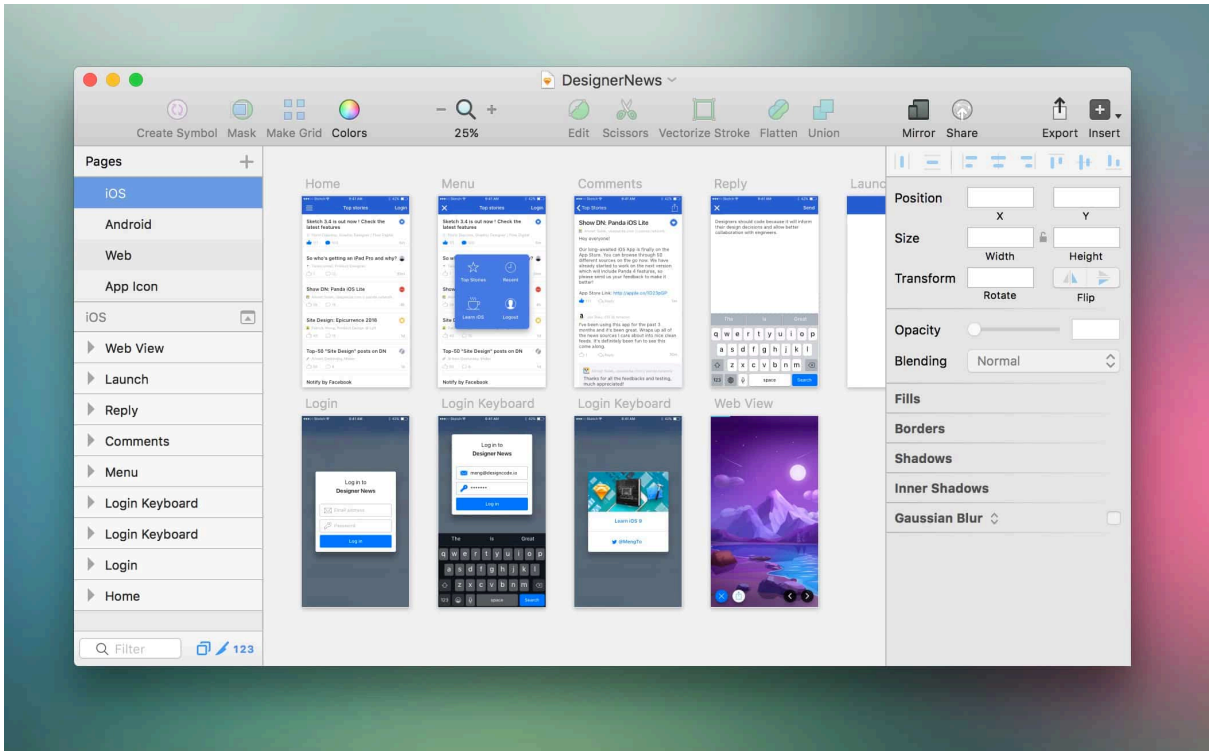


Figure 30: Sketch Interface

According to (What Is Sketch and What Can You Do With It? n.d.), Sketch is a vector graphics programme is called Sketch. It is a tool for product design that web designers frequently use to make idea pages, icons, and other online components. In addition, UI and UX designers love it a lot. Sketch offers a wide range of effective vector editing capabilities, as well as a variety of boolean operations, making it suitable for both amateur and professional artists. Non-Mac users can utilise the web app, and it has a native macOS app. The initial macOS version was launched in 2010, and a number of improvements have been made since then.

Referring to (The Benefits of Using Sketch Designs in Software Development - Lightflows n.d.), one of the key advantages of Sketch is its focus on user interface design. It offers a comprehensive collection of features and tools made especially for creating aesthetically pleasing and useful interfaces. Pixel-perfect designs are simple to make and may be precisely customised by designers.

The ability to edit vectors is another benefit of Sketch. With Sketch, designers can simply create scalable designs that can be easily resized without sacrificing quality. Sketch is based on a vector editing concept. This makes it perfect for creating flexible interfaces that can adjust to

various screen sizes and resolutions while delivering a consistent user experience across devices.

A growing ecosystem of plugins and extensions is another advantage of Sketch. By incorporating different plugins, designers can improve their work processes and increase the capability of Sketch. These plugins give designers extra tools, templates, and automation features so they may speed up their design workflow, work more efficiently, and access a greater variety of design materials.

The collaborative capabilities of Sketch make collaboration simpler. Several designers can collaborate on the same design file at once, making changes in real time and exchanging feedback. This encourages teamwork and makes the design review process run more smoothly, resulting in effective communication and swift project advancement.

Despite being primarily a design tool, Sketch also has some basic prototyping features. To show user flow and interactions, designers might construct interactive prototypes with basic transitions and animations. Although Sketch's prototyping capabilities are less robust than those of specialised prototyping tools, it nonetheless offers designers a handy way to present their work and test interactions.

Last but not least, Sketch benefits from a vibrant design community that actively supports its development. By using UI kits, icon libraries, and other resources, designers may take advantage of the ecosystem that is generated by the community. They can take part in debates, learn from tutorials, and keep up with the most recent design trends, encouraging continual inspiration and learning.

Features	Figma	Adobe XD	Sketch
Multi-platform support	Yes	Yes	No
Real-time collaboration	Yes	No	No
Prototyping	Yes	Yes	No

Robust vector editing	Yes	Yes	Yes
Plugins and integrations	Yes	Yes	Yes
Design components	Yes	Yes	Yes
Handoff and sharing	Yes	Yes	Yes
Ease of learning	Yes	Yes	No
Comprehensive vector editing tools	No	Yes	Yes
Prototyping built-in	No	Yes	No

Table 4: Comparison of Figma, XD & Sketch

For this project, Figma is the recommended design tool because of its outstanding features and capabilities. Figma provides a complete solution for designing and refining project visuals with its multi-platform compatibility, real-time collaboration, and strong prototyping capabilities. Productivity and efficiency are increased by having the ability to work smoothly across various devices and collaborate with team members in real-time. The vast selection of plugins and connectors available for Figma further increases its usefulness and customization possibilities. Both novice and seasoned designers can utilise it thanks to its simple interface and user-friendly layout. A streamlined workflow is also made possible by Figma's vector editing tools, design elements, and ease of sharing and handoff of designs. In conclusion, Figma is the best option for this project due to its adaptability, collaborative tools, and user-friendly interface.

2.5.5 Comparison of SDLCs

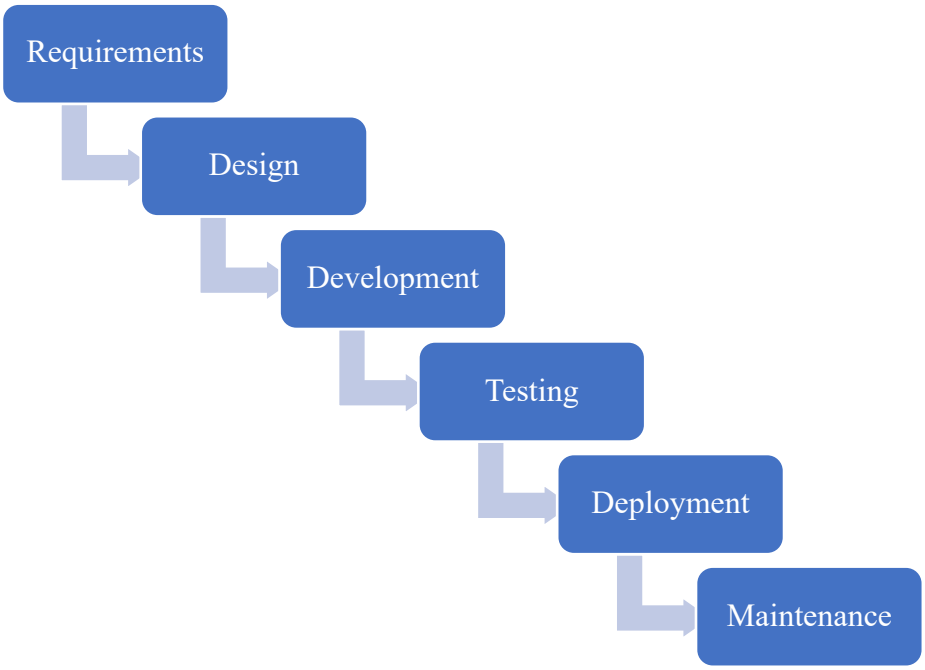


Figure 31: Waterfall Model

By studying the journal (Alshamrani and Bahattab n.d.), three types of waterfall models will be compared, waterfall, spiral and incremental models. The waterfall model is a linear and sequential approach to software development. Each phase, including requirements gathering, design, development, testing, deployment, and maintenance, is finished before moving on to the next in an organised process. The paradigm emphasises meticulous documentation, up-front planning, and tight adherence to predetermined protocols. This method functions effectively for projects with clear end goals from the beginning and requirements that are stable and well-defined. The Waterfall model, however, lacks flexibility and might make it difficult to take into account modifications or input after a phase is finished.

Figure 32: Spiral Model

On the other side, the Spiral model employs an iterative and risk-driven methodology. It incorporates aspects of both iterative development and the waterfall methodology. Through a series of rounds, the Spiral model focuses on early risk detection and mitigation. The planning, risk analysis, development, and assessment phases make up each iteration. The concept promotes regular consumer input and participation throughout the development process. Large, complicated projects with shifting requirements and a need for efficient risk management are particularly well-suited to this strategy. The Spiral model, however, necessitates ongoing monitoring and risk assessment, which could make projects more difficult.

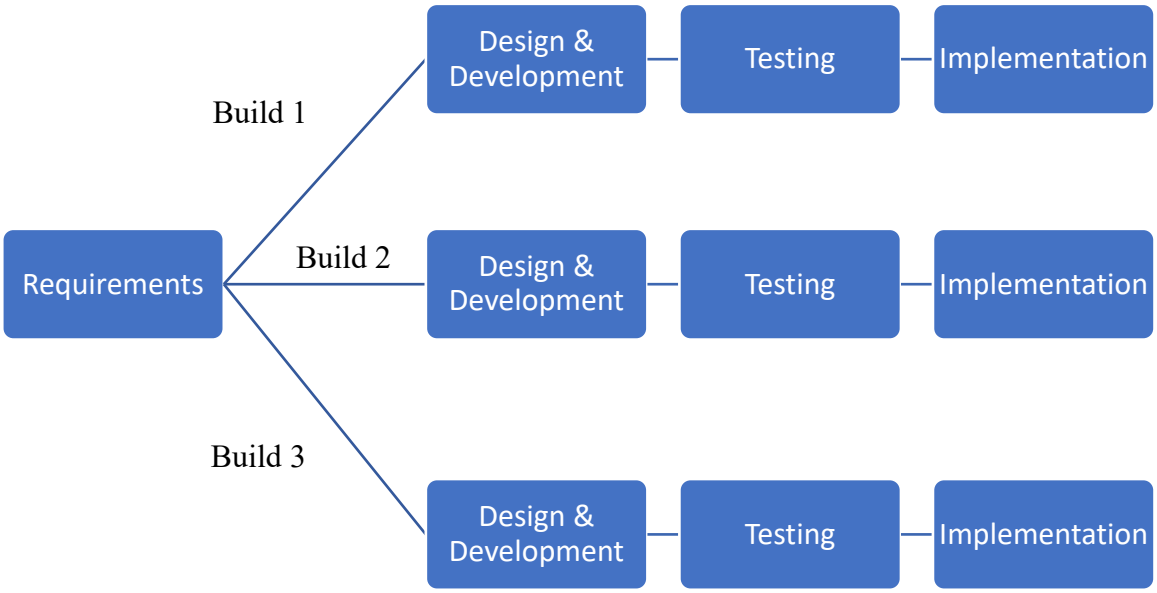


Figure 33: Incremental Model

The incremental model breaks the software development project up into smaller pieces, or modules, each of which offers a certain set of functions. This allows for the early delivery of viable software because the increments are produced and provided incrementally. It is possible to test and analyse each increment independently, giving room for immediate feedback and customization. The incremental model offers flexibility and allows for adjustments in response to client input and changing requirements. It is especially helpful for projects where user input is required and needs are expected to change frequently. To ensure integration and compatibility between increments, though, careful planning and collaboration are required.

Models	**Waterfall**	**Spiral**	**Incremental**
Approach	Linear and sequential	Iterative and risk-driven	Iterative and incremental
Phases	Sequential phases (requirements, design, development, testing, deployment, maintenance)	Iterative cycles (planning, risk analysis, development, evaluation)	Iterative development in smaller increments

Requirements	Well-defined and stable	Evolving and may change over time	Evolving and may change over time
Flexibility	Limited flexibility for changes once a phase is completed	Accommodates changes throughout the process	Accommodates changes based on feedback
Risk Management	Not explicitly built-in	Emphasizes early risk identification and mitigation	Implicitly manages risk through iterations
Customer Involvement	Less involvement during development process	Frequent customer involvement and feedback	Frequent customer involvement and feedback
Delivery Time	Longer delivery time due to sequential nature	Iterative cycles may reduce overall delivery time	Early delivery of usable software
Complexity	Lower complexity due to predefined processes	Moderate complexity due to continuous risk assessment	Moderate complexity with coordination needs
Suitable For	Well-defined and stable requirements	Projects with changing requirements and risk management needs	Projects with evolving requirements

Table 5: Comparisons of Waterfall, Spiral & Incremental Models

Due to its compliance with well-defined requirements, structured development methodology, emphasis on documentation, milestone-based progress monitoring, and obvious project deadline, the Waterfall model is quite ideal for the Groomify project. The Waterfall methodology offers a methodical and structured development process with Groomify's unique grooming and personal care functions. Complete documentation acts as a reference for upcoming maintenance and aids in sustaining project discipline. The writer can effectively create the app while concentrating on the specified goals, lowering the risk of scope creep, and delivering a high-quality final product by using the Waterfall approach.

2.6 Conclusion

The pet grooming software will be created using Flutter, which enables cross-platform development for both Android and iOS devices, in accordance with the decisions made. Because of its scalability, real-time data synchronisation, and simple connection with other Firebase services, Firebase will be used as the database. Dart, a programming language renowned for its effectiveness and cutting-edge capabilities, was selected. Because of its bigger market share and simple app distribution, the software will primarily target users of the Android operating system.

Flutter, a framework, offers a thorough and effective development environment that makes it possible to design a visually beautiful and responsive user experience. The development process is sped up by its hot reload functionality, which allows developers to make real-time changes and immediately view the results. The construction of unique and interactive components is also possible thanks to the vast widget library offered by Flutter, which improves the user experience of the app.

The project will move through consecutive phases, such as requirements gathering, design, implementation, testing, and deployment, by employing a waterfall software development approach. This strategy offers a well-organized and structured process that makes sure each stage is finished before going on to the next. It enables extensive planning and documentation, lowering the risk of scope creep and guaranteeing a stable and trustworthy end result.

In conclusion, there are various benefits to using Flutter, Firebase, Dart, and the Android operating system for the pet grooming app project. Scalability, real-time data synchronisation, effective cross-platform development, and simple Firebase service integration are all made possible. Programming in Dart offers a contemporary and effective development environment. The larger user base and simple app distribution of the Android platform make it worthwhile to target it. A high-quality and user-friendly pet grooming app can be produced by using an agile SDLC because iterative development, regular feedback, and a shorter time to market are all advantages of the approach.

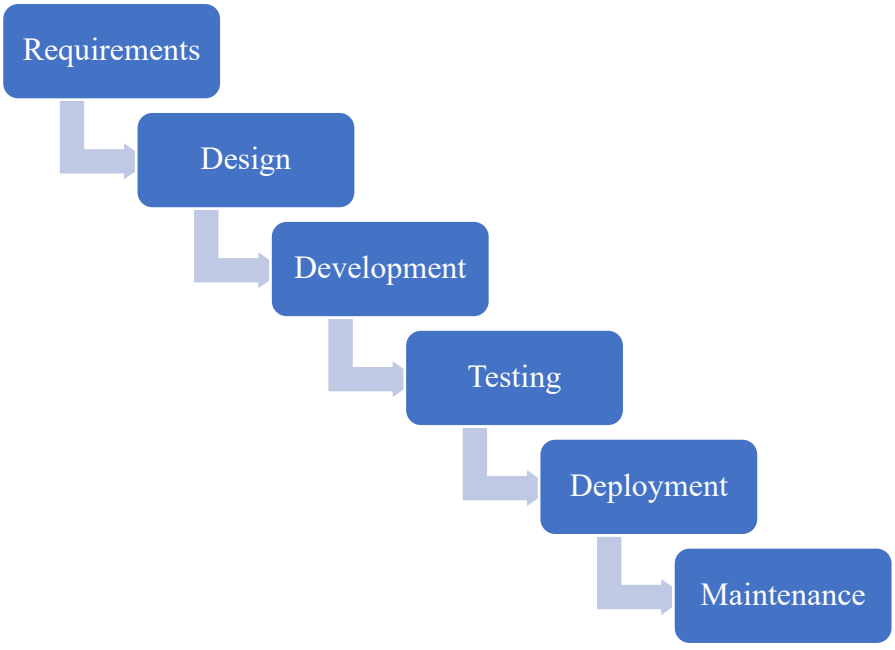
Chapter 3

Research Methodology

3.1 Introduction

The Waterfall model is a well-known software development technique that includes requirements gathering, design, implementation, testing, and deployment among its sequential phases. Every stage builds on what came before it, ensuring a disciplined and orderly approach to development. Delivering the produced application to end users successfully is the main goal of the deployment phase. This calls for thorough preparation, coordination, and the use of suitable deployment technologies. The Flutter and Dart frameworks will be used for this project to support cross-platform development, enabling effective code sharing and quick development cycles. The usage of Figma as a collaborative design tool will make it possible to create interfaces that are both aesthetically pleasing and user-friendly. As a strong backend platform, Firebase will also enable smooth deployment, scalability, and performance monitoring by providing hosting, database, and analytics services. The project intends to provide a top-notch pet grooming application that meets the needs and expectations of its consumers by embracing the Waterfall paradigm and utilising these deployment methods.

3.2 SDLC



Referring to Chapter 2 of the report, the Waterfall Model has been chosen as the Software Development Life Cycle (SDLC) approach. The Waterfall model, is a popular software development methodology renowned for its sequential and linear approach, was employed in the pet grooming app deployment process. The Waterfall model's phases all contributed significantly to the rigorous and organized development process that led to the app's successful release.

The requirements' gathering phase is the first stage of the waterfall model. This stage involves extensive investigation and analysis to pinpoint and record the precise needs of the pet grooming software. This entails being aware of the requirements of pet owners, the intended features, and functionalities of the app, and any technical or regulatory limitations that must be taken into account. The requirements collecting phase ensures a comprehensive grasp of the project scope and objectives and lays the groundwork for the succeeding phases.

The design phase follows the collection and documentation of the requirements. The pet grooming app's overall architecture and organization are established during this phase. This comprises setting the database structure, developing the user interface, and producing wireframes and prototypes. The design phase is concerned with turning the specified requirements into a practical and aesthetically pleasing app that satisfies the expectations of the intended consumers.

The development phase starts after the design phase. Here is where the pet grooming app's actual development and execution happen. The Flutter framework and the selected programming language, Dart, are both used to create the app's functionality and features. To provide a reliable and effective app, the development phase adheres to design standards and best coding practices.

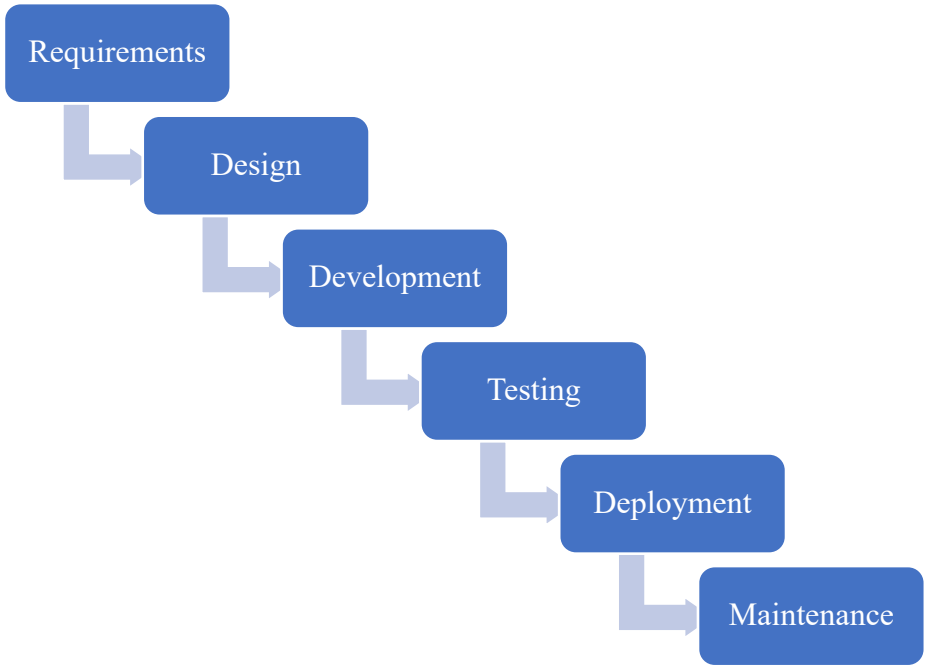
The testing phase starts once the development phase is over. To find and fix any flaws or difficulties with the software, this step entails a variety of testing tasks. In order to verify the app's functionality, performance, and user experience, many testing techniques, including unit testing, integration testing, and user acceptability testing, are used. Before being made available to users, the testing step makes sure the pet grooming software satisfies quality requirements and performs as planned.

The pet grooming app is finally made available to the target consumers during the distribution phase. This include putting together the app's packaging, gathering the necessary materials, and submitting it to the appropriate app stores for release. To ensure the app's ongoing functionality and user happiness, the deployment phase also entails managing updates and maintenance.

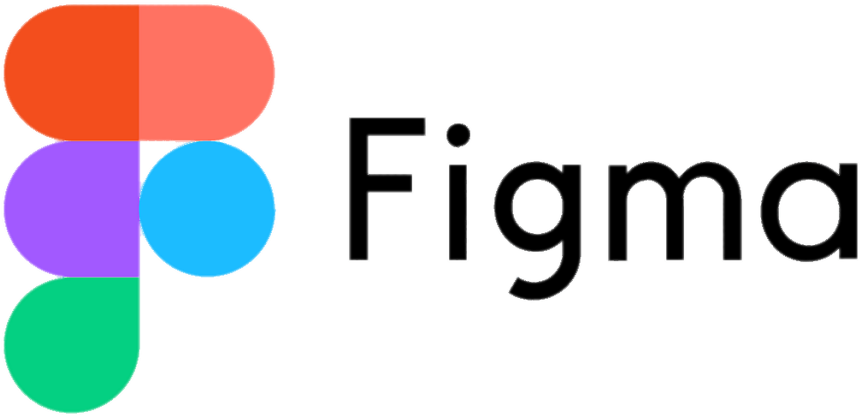
The pet grooming app project benefits from a well-structured and systematic approach to development by using the Waterfall model in the research technique. The Waterfall model's sequential structure allows for careful planning, documenting, and quality assurance at every stage. It encourages a clear knowledge of the project requirements, lowers the chance of scope creep, and makes sure the finished software satisfies the desired objectives and user expectations.

The Waterfall approach may, however, be constrained in terms of adaptability and flexibility. Making major changes without returning to prior phases might be difficult once a phase is over and the project goes on. To reduce the need for large changes later in the development process, it is necessary to undertake a thorough analysis and acquire correct requirements up front.

3.3 Deployment Tools



The requirements gathering phase was the first stage of the waterfall model. To fully grasp the needs and specifications of the pet grooming software, a significant amount of research and analysis were done during this phase. A thorough list of functional and non-functional criteria was established through interviews, surveys, and interactions with stakeholders like pet owners and service providers. The following stages of the development process were built on top of these criteria.



The system design process started when the requirements were precisely outlined. In this step, the specified requirements were turned into a thorough system design. It involved building wireframes or interactive prototypes to visualize the user interface as well as designing the app's architecture and database structure. Figma will be utilized, as described in Chapter 2, to streamline the design process, enabling collaborative design reviews and iterative improvements based on the writer's experience in UI/UX designing.

The Implementation phase started after the Design phase. During this stage, the pet grooming app was coded and developed. The Dart framework and the selected programming language,

Flutter, gave the app's features and functionalities a strong base. To guarantee the dependability and maintainability of the code, the development team adhered to industry standards, used version control systems, and followed best practises for coding.

The Testing phase gained prominence as the Implementation phase moved forward. To find and fix any flaws or problems with the app's operation, performance, and usability, thorough testing was done. To ensure that the software complied with the requirements, several testing methodologies, including unit testing, integration testing, and user acceptability testing, will be used. To ensure a high-quality and stable program, bugs and glitches must be quickly fixed through several rounds of testing.

Once the testing phase was successfully completed, the app proceeds to the Deployment phase. The app must be released and made accessible to end users at this phase. With its scalable and secure backend infrastructure for hosting the app's data and controlling user authentication, Firebase played a significant part in this phase. Additionally, the seamless deployment of the app to Android users through the Google Play Store was made possible using Firebase's app distribution capabilities. However, the deployment phase is not included in the project's objectives because development is its exclusive focus.

The discrete phases and linear progression of the waterfall model enabled a methodical and regulated approach to the pet grooming app's implementation. The development team managed project timeframes, scope, and deliverables well by adhering to this paradigm. The distinct division of the processes enabled thorough planning, careful design, meticulous development, thorough testing, and successful app release. Overall, the use of the Waterfall model helped the pet grooming software be successfully implemented and meet the needs of pet owners and pet groomers in a dependable and effective way.

3.4 Use Case Diagram

In the Unified Modelling Language (UML), a use case diagram is a type of visual representation that shows how users, also referred to as actors, interact with a system to accomplish particular objectives or functionalities. It is a high-level view that concentrates on how a system functions from the standpoint of its users. Use cases are the particular functions or features that the system offers, and actors are the external entities that interact with the system in a use case diagram. The relationships or interactions between actors and use cases are represented by arrows, which show how users use the system to complete tasks. In the early phases of software development, use case diagrams are especially helpful for comprehending and conveying the functionalities, requirements, and ways in which users will interact with the system.

3.4.1 Admin UML

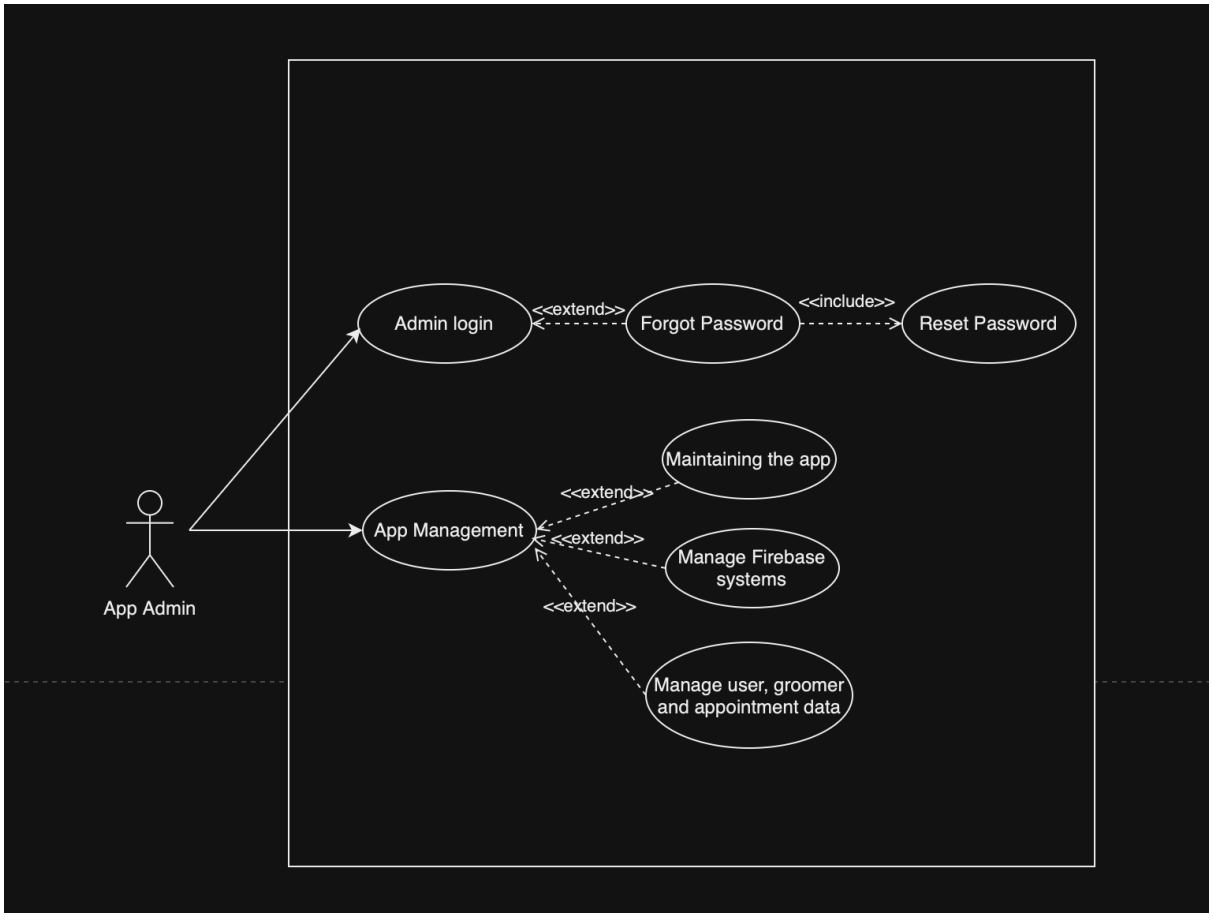


Figure 34: Admin UML

Based on figure 34, the "Admin UML" represents the administrative user's functionality. Important interactions are included here, like "Admin Login," which expands on the "Forgot Password" use case and offers the ability to "Reset Password." Furthermore, the use case "App Management" encompasses a range of administrative responsibilities, such as upkeep of the application, supervision of user, groomer, and appointment data, and management of Firebase systems. These use cases offer a thorough rundown of the administrative features available in the system, guaranteeing safe access and effective handling of important information.

3.4.2 Users UML

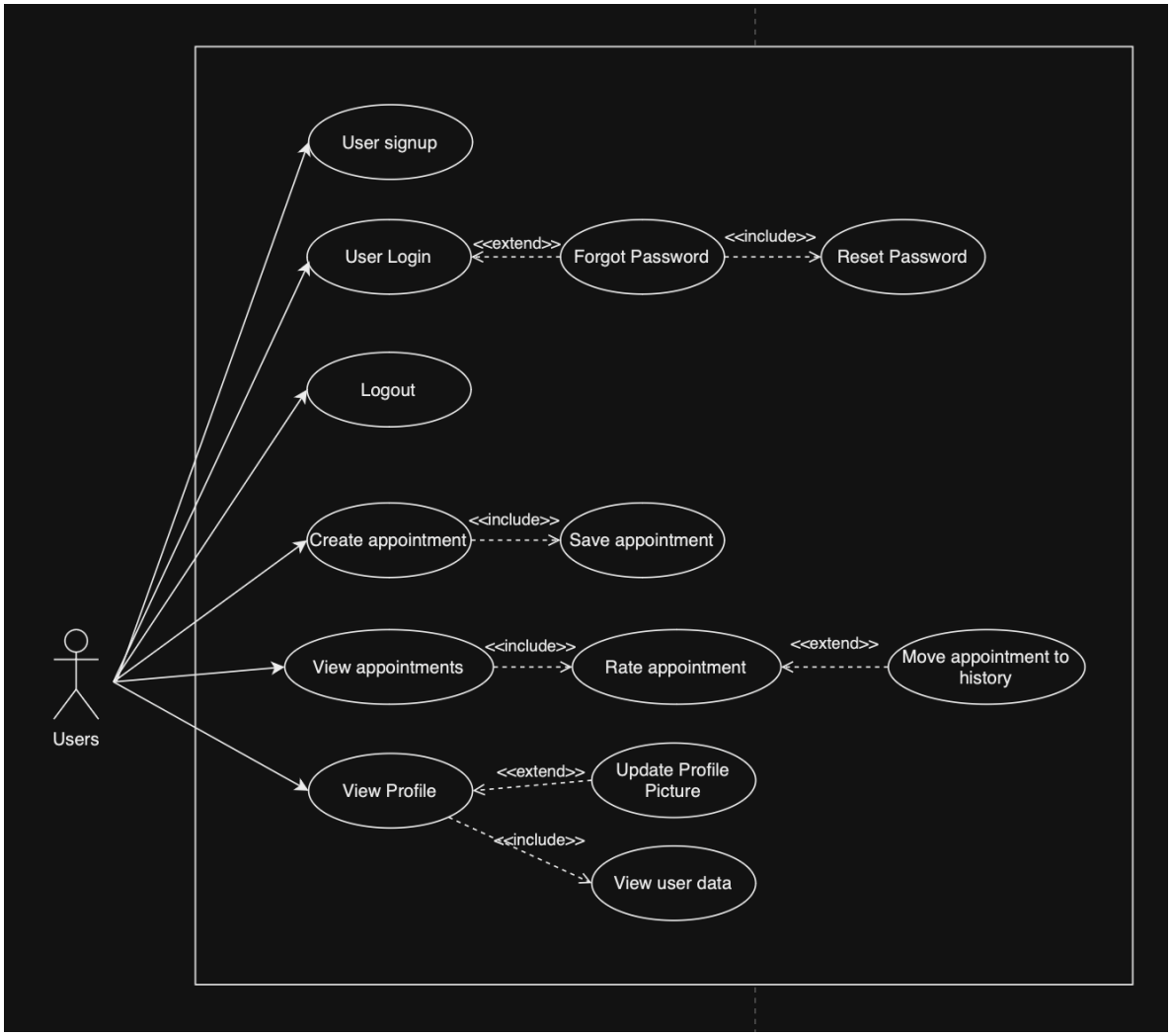


Figure 35: Users UML

The "Users URL" section describes a number of use cases for users interacting with the application. While the "User Login" use case expands the "Forgot Password" functionality and

adds the ability to reset the password, the "Signup" use case permits new users to register. The "Logout" use case makes sure that user sessions end securely. The "Create Appointment" use case, which incorporates the "Save Appointment" feature, serves as an example of creating an appointment. Users can view their scheduled appointments with the "View Appointments" use case. It also allows users to rate appointments and has an extension that allows users to move appointments into the past. The "View Profile" use case encompasses viewing user data in addition to features like changing the profile picture. Together, these use cases offer a thorough depiction of the user-centric functionalities, guaranteeing a smooth and feature-rich application experience.The "View Profile" use case encompasses viewing user data in addition to features like changing the profile picture.

3.4.3 Groomers UML

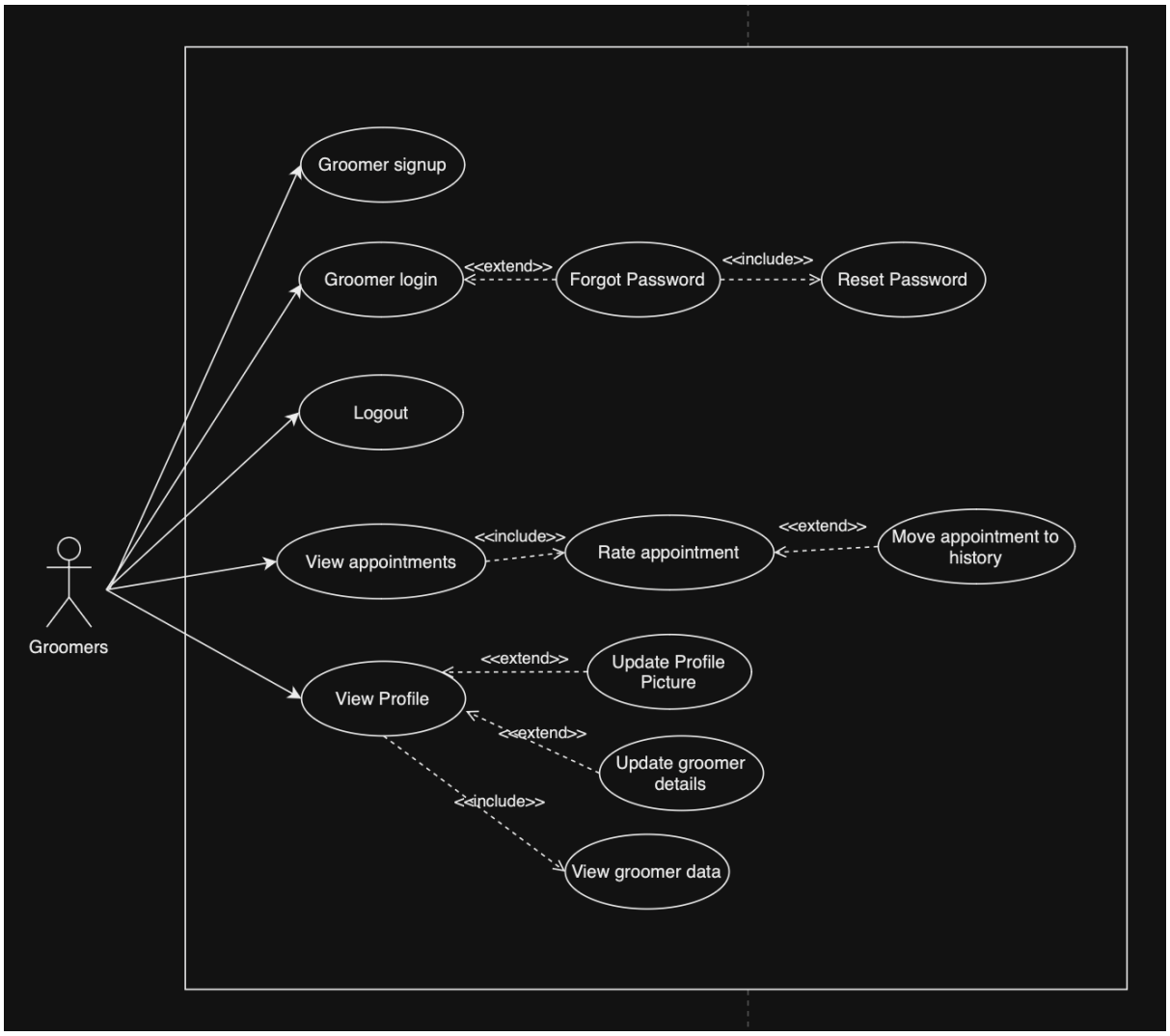


Figure 36: Groomers UML

The "Groomers URL" describes important features in the use case diagram section devoted to groomers' interactions within the application. Groomers can use the "Signup" use case to start the registration process. Then, they can safely access their accounts using the "Groomer Login" feature, which expands on the "Forgot Password" feature and adds a secure "Reset Password" option. For increased security, the "Logout" use case makes sure that groomer sessions end. The "View Appointments" use case, which includes functions like rating appointments and bringing them into a historical context, allows groomers to manage their schedules. Groomers can access detailed groomer data, change professional details, and update their profile picture with the "View Profile" use case, which offers a comprehensive view. When combined, these use cases provide grooming professionals with an extensive feature set for organising their schedules, profiles, and overall application experience. Groomers can access detailed groomer data, change professional details, and update their profile picture with the "View Profile" use case, which offers a comprehensive view.

3.5 Entity Relationship Diagram

A database's structure and relationships are defined by the data model, which is represented visually in an entity-relationship diagram (ERD). To depict entities (like tables in a database), attributes (properties of entities), and the relationships between entities, ERDs employ a set of symbols. Relationships show the connections between entities, which are essentially nouns that represent concepts or objects. Extra details about entities are provided by attributes. Database designers frequently utilise Entity Relationship Diagrams (ERDs) to aid stakeholders and developers in visualising how data is organised and connected. They are an essential step in the database design process and act as a guide for creating relational databases.

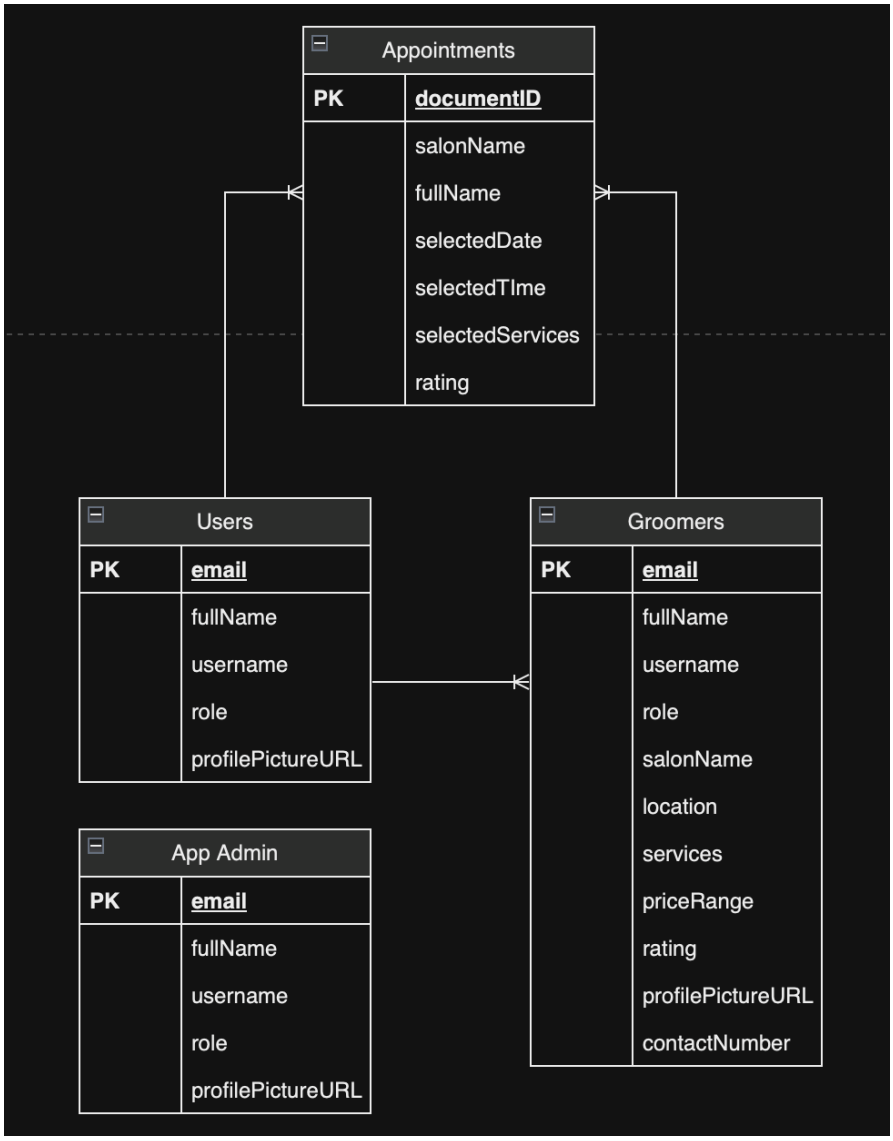


Figure 37: ERD

Based on Figure 34, a thorough representation of the database structure is provided by carefully identifying the main entities and the attributes that go along with them. One of the cornerstones, the "Users" entity, is identified by a primary key called "email." This object contains relevant properties such as "fullName," which provides information about the user, "role," which describes the role or title of the user, and "profilePictureUrl," which is a URL pointing to the user's profile picture.

Similarly, a primary key of "email" distinguishes the "Groomers" entity, a crucial part of the database architecture. There are several attributes that this entity has to offer, including "fullName," "username," "role," "salonName," "location," "services," "priceRange," "rating," "profilePictureurl," and "contact number." Together, these characteristics offer a complex picture of the groomer's identity, credentials, and contact information.

The central component of interactions between users and groomers is the "Appointments" entity, which is identified by the primary key "documentID." Important attributes like "salonName," "fullName," "selectedDate," "selected time," "selectedServices," and "rating" are encapsulated in this entity. Through the capture of these aspects, the "Appointments" entity serves as a pivot, providing extensive insights into the scheduling and evaluation of interactions between users and groomers.

The application's administrative component is contained in the "App Admin" entity, with "email" serving as the primary key. This entity includes attributes like "fullName," "username," "role," and "profilePictureURL," which denote the identity, function, and visual representation of the administrative staff in the system.

Additionally, the ERD precisely defines one-to-many relationships that highlight the dynamic relationships among entities. Of particular note are the connections made between "users" and "appointments" as well as "groomers," which demonstrate that a user is able to participate in multiple appointments and communicate with different groomers. In addition, the connection between "groomers" and "appointments" clarifies that a groomer is capable of serving several appointments at once. These subtle relational insights contribute to our understanding of the system's interconnections and provide a strong basis for designing a robust, linked, and efficiently organised database.

3.6 Flowchart

A flowchart is a graphic representation of a system or process that shows the steps and decision points involved using arrows and standard symbols. It is an effective tool for simplifying and streamlining the understanding and analysis of complex systems by depicting the movement of data, resources, or actions within them. Flowcharts are extensively used for process documentation, analysis, and optimisation in a variety of fields, including business, engineering, programming, and project management. Flowcharts aid in effective communication, problem-solving, and decision-making by offering a clear and organised representation of the logical connections between various system elements. This, in turn, contributes to process improvement and efficiency.

3.6.1 Login Page

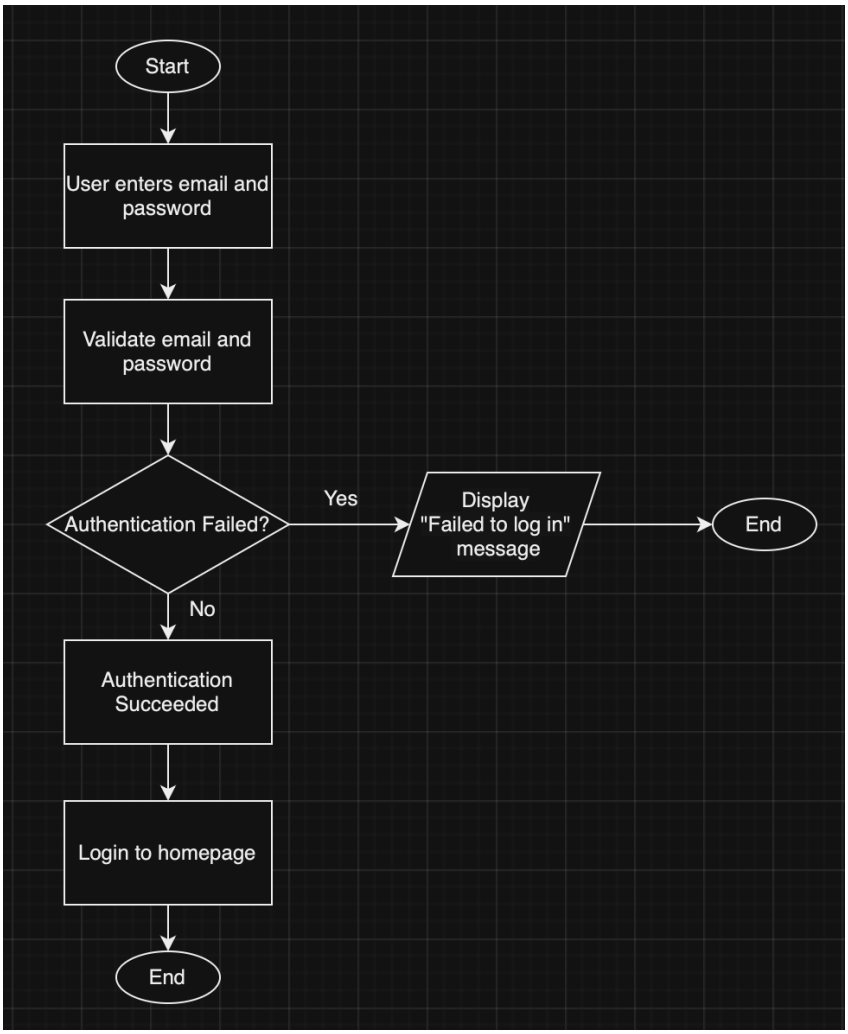


Figure 38: Login Flowchart

This flowchart shows the process of user authentication in login page. Users are required enter email and password, the system then validates the login credentials. If authentication fails, the app prompts an error message, else it navigates user into the homepage after successful login.

3.6.2 Signup Page

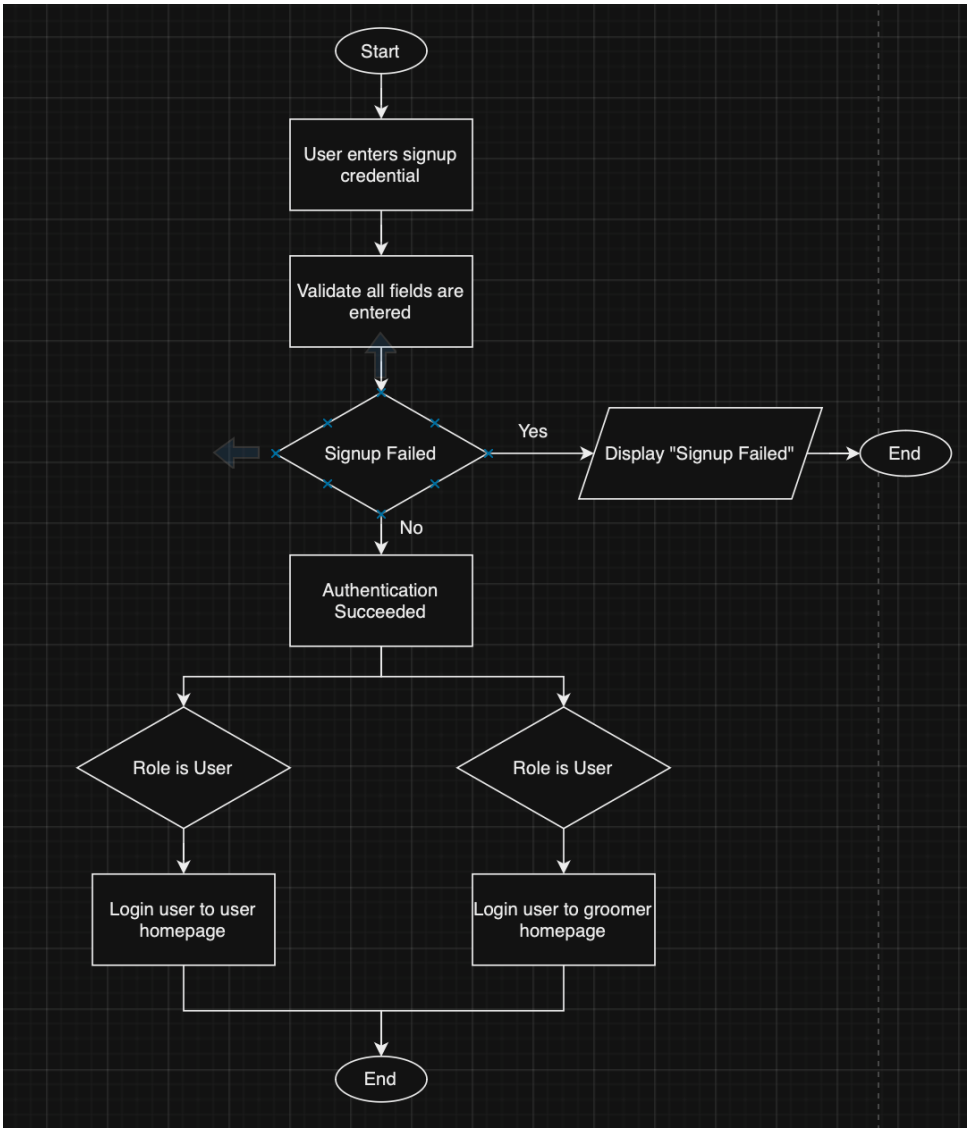


Figure 39: Signup Flowchart

This flowchart shows the process of signup. Users are required to enter signup credentials, the system then validates if all fields are entered. The signup fails if there are missing credentials,

which will display error message. If signup is successful, it will based on the role selected, then navigates the user to the respective homepages.

3.6.3 Groomers

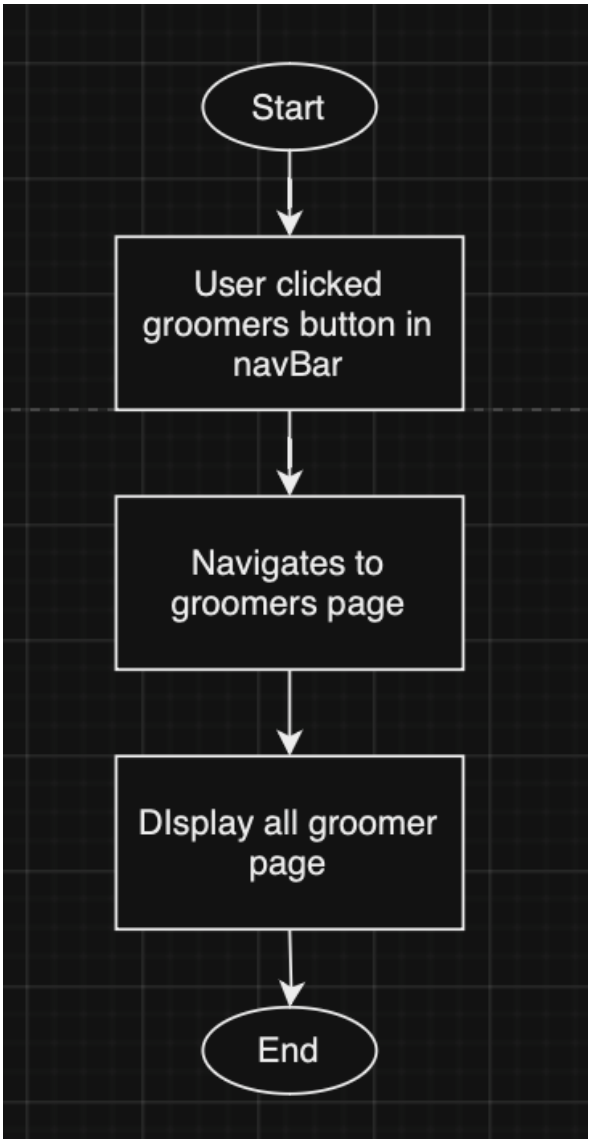


Figure 40: Groomer Page Flowchart

This shows the groomer page flowchart. When user clicked the groomers button in the navigation bar, they will be directed to the groomers page. The page displays all available groomers in the database.

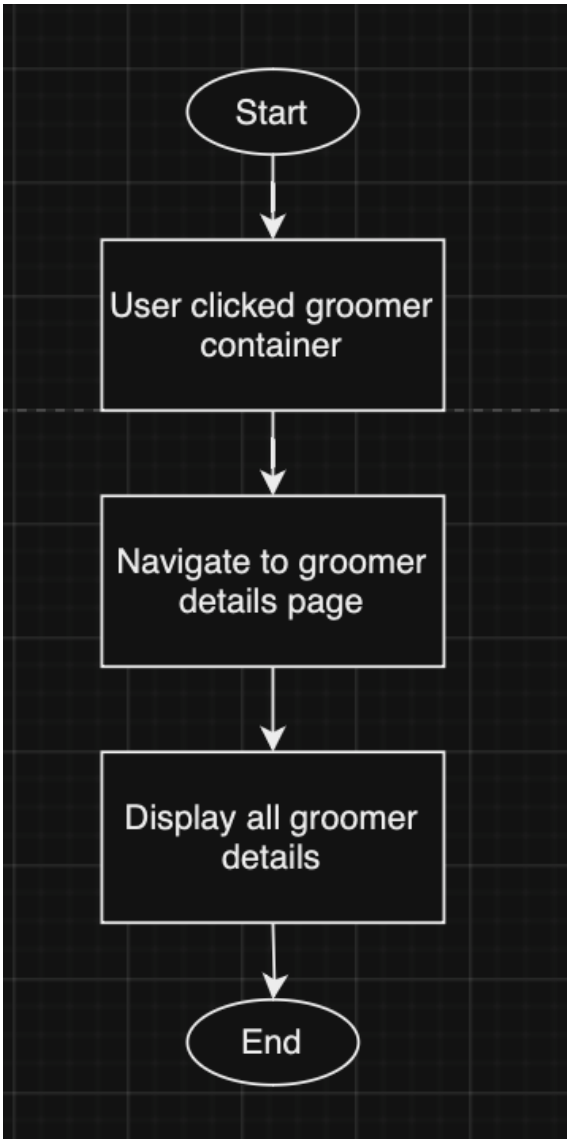


Figure 41: Groomer Details Flowchart

This shows the groomer details flowchart. When user clicked a groomer in the groomers page, they will be directed to this page, which displays all groomer details.

3.6.4 Appointments

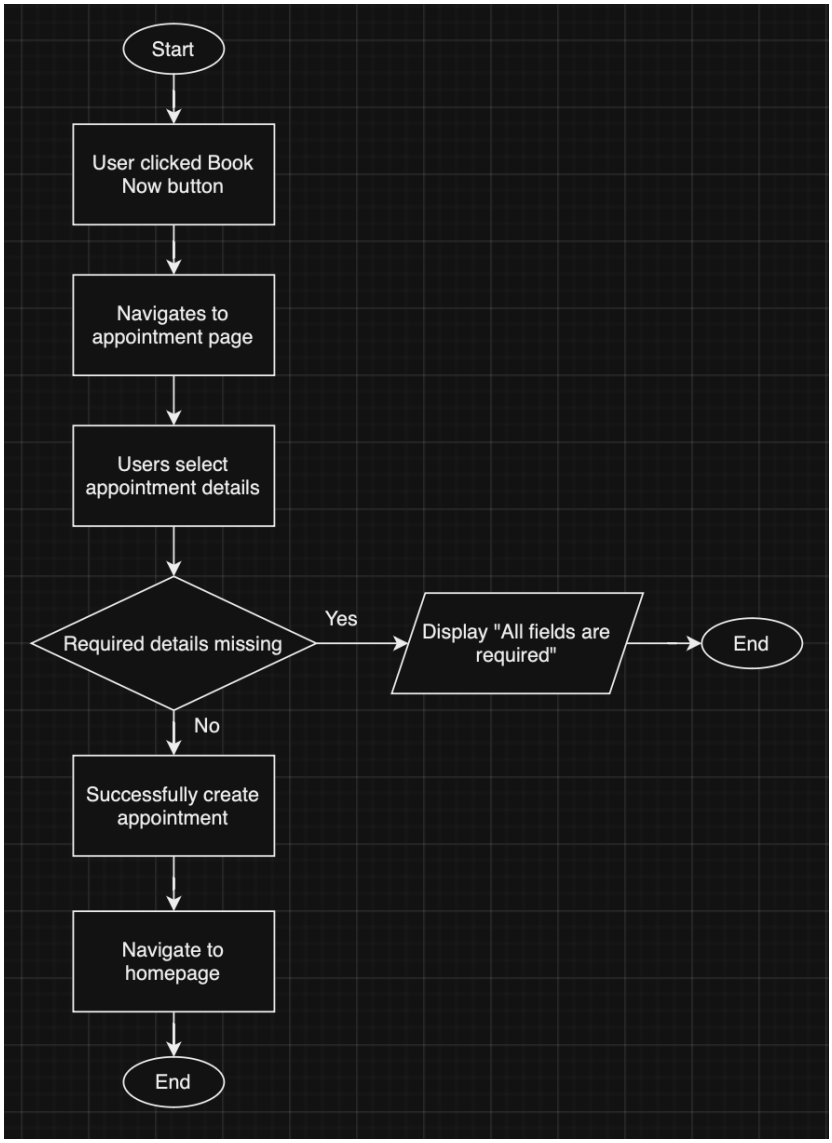


Figure 42: Appointment Flowchart

This shows the appointment flowchart. When user clicked the Book Now button in the groomer details page, they will be directed to the appointment page. User are required to select all fields such as date, time and services. If required appointment details are missing, the system will prompt an error message. The user will be directed to the homepage after successful appointment creation.

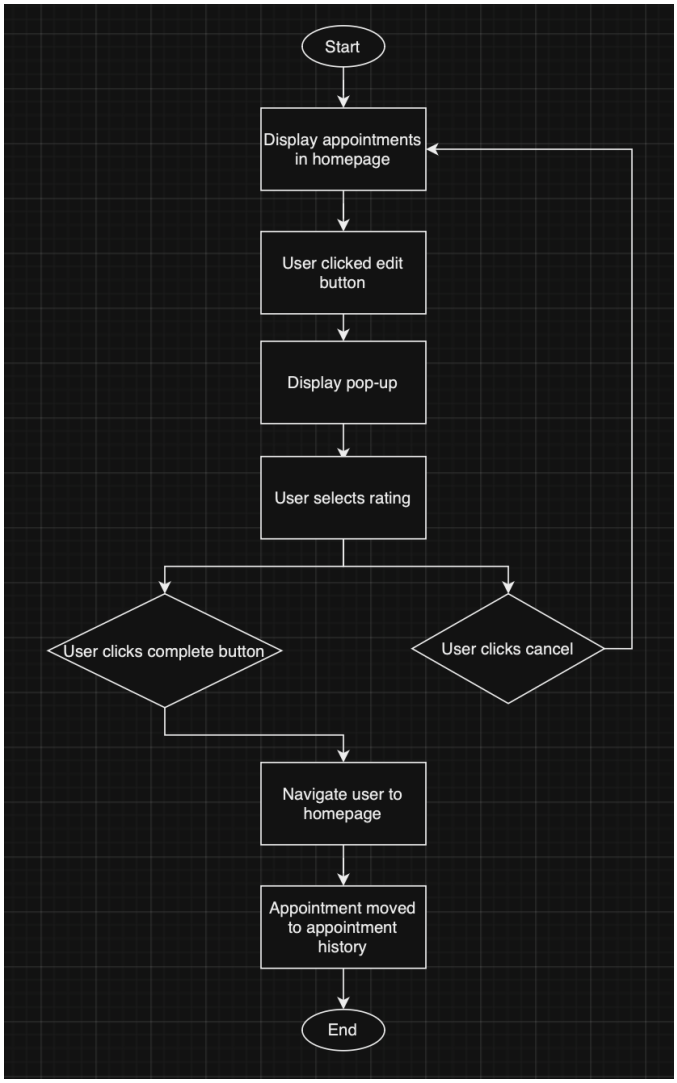


Figure 43: Make Appointment Flowchart

This shows the make appointment flowchart. Appointments will be displayed in the homepage, when user clicked the edit button, it displays a pop-up. User can then rate the service, and click the complete button which will navigate the user back to homepage. However, if the user clicks cancel, they will be directed to the homepage, where the appointment is still present. The appointment will be moved to appointment history when user clicks complete.

3.6.5 Profile Page

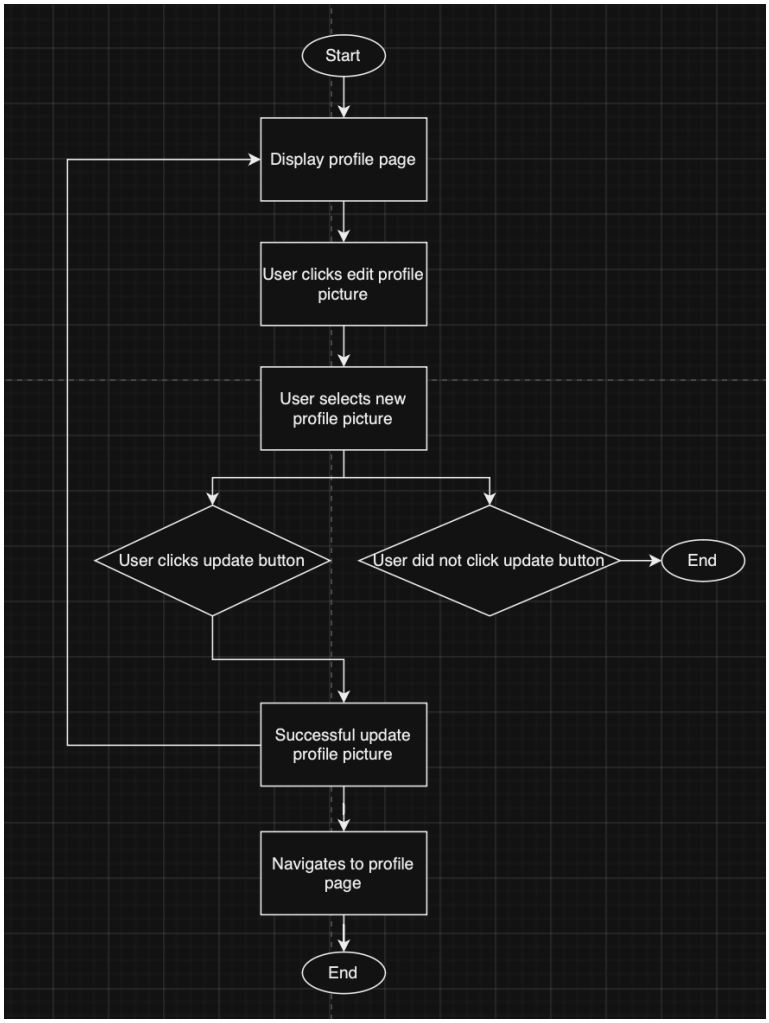


Figure 44: Update Profile Picture Flowchart

This shows the update profile picture flowchart. Profile picture is displayed in the profile page. When user clicked edit profile picture button, they will be directed to the gallery to let user select profile picture. If the user clicks update button after selecting a new picture, the app updates the current profile picture using the new picture, then navigates back to the profile page. If user did not click update button, the profile picture is not updated.

3.6.6 Groomer Profile Page

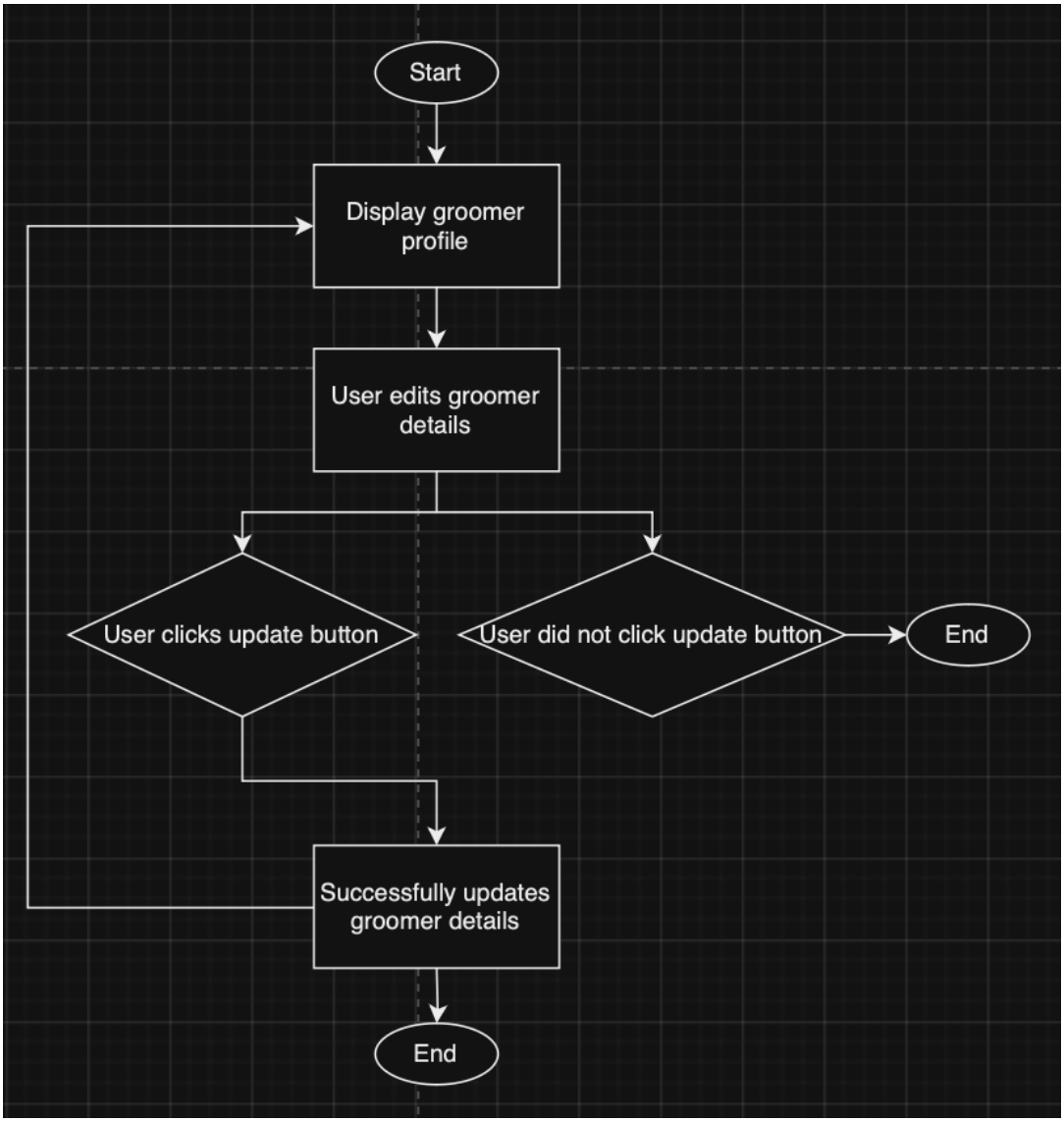


Figure 45: Update Groomer Details Flowchart

This shows the update groomer details flowchart. After user make changes to their groomer profile such as salon name, location, services and price range, they could click the update button which successfully updates the groomer details and navigate back to the groomer profile page. However, if the user did not click update button, the groomer details are not updated.

3.6.7 Logout

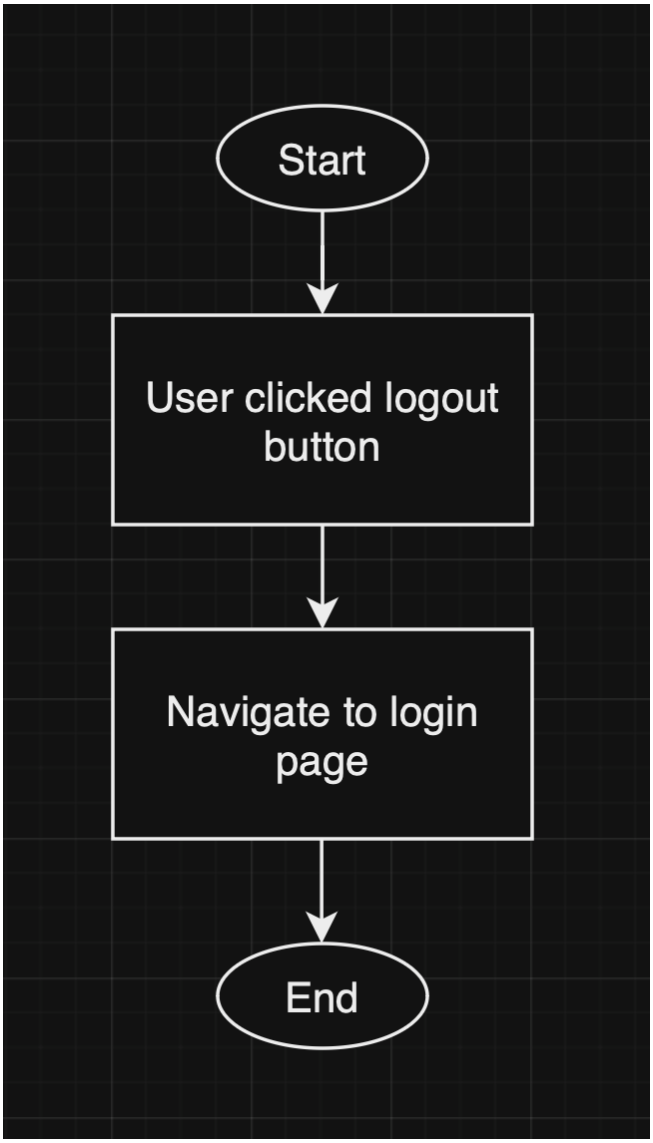


Figure 46: Logout Flowchart

This shows the logout flowchart. When user click the logout button, they will be directed back to the login page.

3.7 Gantt Chart

A Gantt chart is a visual aid for project management that shows tasks or activities as they progress over time, giving a thorough timeline of the timeline for a project. Every task is represented by a horizontal bar, which is arranged along a timeline and whose length is equal to the task's duration. Task dependencies are commonly represented by connecting the bars. Gantt charts provide a simple and easy method for comprehending project schedules, spotting gaps or overlaps in tasks, allocating resources effectively, and tracking advancement. They help project managers and team members keep track of and manage the various components of a project. They are widely used in a variety of industries to plan, coordinate, and communicate project schedules.

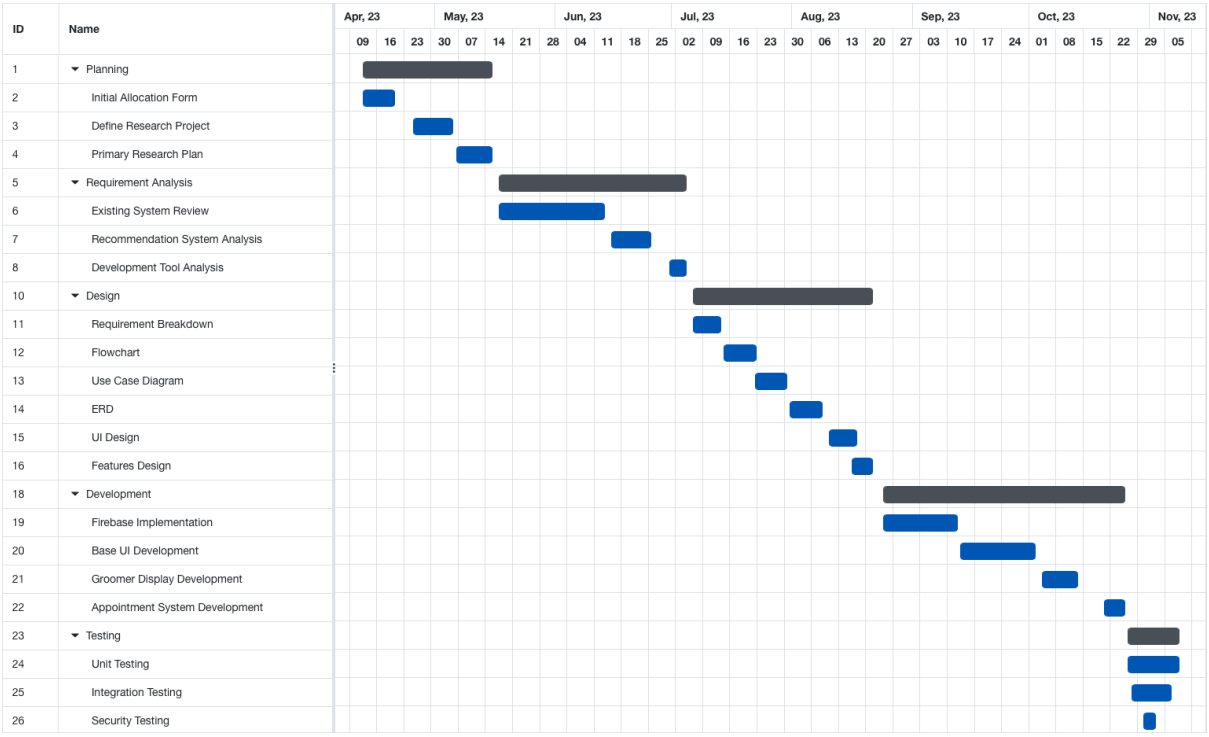


Figure 47: Gantt Chart

3.8 Conclusion

In conclusion, the pet grooming software was created using an organised process based on the Waterfall model, including stages like requirement gathering, system design, implementation, testing, and delivery. The programming language and framework used were Flutter and Dart, ensuring quick development and cross-platform interoperability. The user interface of the app was significantly influenced by Figma, and Firebase supplied a solid backend architecture. The software was created specifically for the Android operating system with a broad user base in mind. The development portion of the app was successfully finished thanks to thorough planning and adherence to the Waterfall paradigm throughout the process.

Chapter 4: Project Implementation

Authentication

4.1 Login

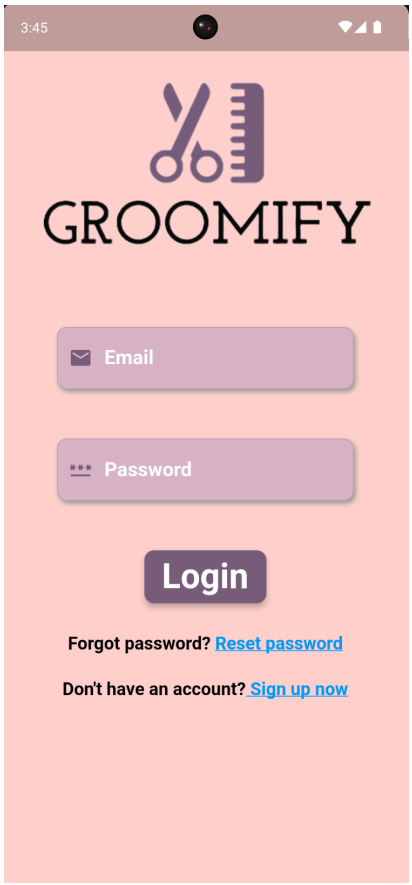


Figure 48: Login

This is the login page of the Groomify app. When the user starts the app, they are greeted with this page, which they could enter their login credentials (email and password). There is also a forgot password button for users to reset their password. However, if the user does not have an account with Groomify, there is a sign-up button which will navigate user to the signup page.

```
onPressed: () async {
  final emailValidationResult = AuthController.instance.validateEmail(emailController.text);
  final passwordValidationResult = AuthController.instance.validatePassword(passwordController.text);

  if (emailValidationResult == '' && passwordValidationResult == '') {
    // Both email and password are valid, proceed with login
    AuthController.instance.login(
      emailController.text.trim(),
      passwordController.text.trim(),
    );
  } else {
    // Show error popups for invalid input
    if (emailValidationResult.isNotEmpty) {
      AuthController.instance.showErrorPopup(context, 'Email Error', emailValidationResult);
    }
    if (passwordValidationResult.isNotEmpty) {
      AuthController.instance.showErrorPopup(context, 'Password Error', passwordValidationResult);
    }
  }
},
child: const Text('Login'),
```

Figure 49: Login Code Snippet

When login button is clicked, it validates the email and password with the firebase authentication data. It then executes the login function when both email and password are valid. The app will also prompt error message when email or password is invalid.

4.2 Password Reset

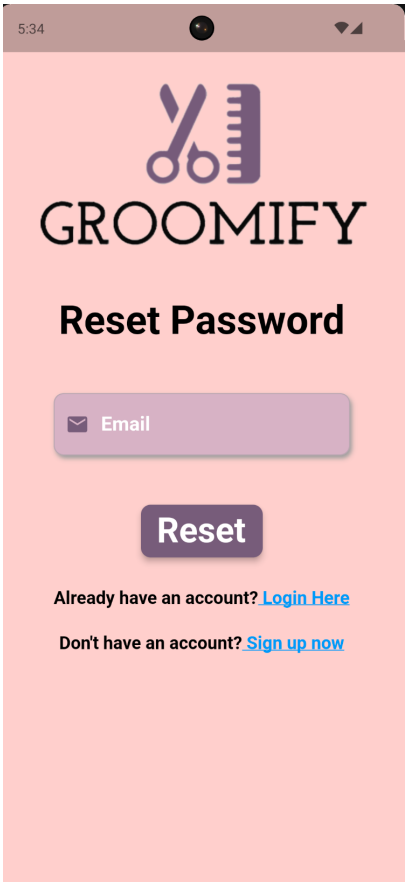


Figure 50: Password Reset

After the user clicked the reset password button, they will directed to this page. Users are required to enter the registered email in the email textfield. Login and signup options are also available which will direct user to the respective pages.

```
//Button
SizedBox(
  width: w * 0.3,
  height: h * 0.06,
  child: ElevatedButton(
    style: ElevatedButton.styleFrom(
      elevation: 5,
      backgroundColor: const Color(0xff735D78),
      shape: RoundedRectangleBorder(
        borderRadius: BorderRadius.circular(10),
      ),  // RoundedRectangleBorder
      textStyle: const TextStyle(
        fontSize: 35,
        fontWeight: FontWeight.bold,
        color: Colors.white,
      ),  // TextStyle
    ),
    onPressed: () {
      String email = emailController.text;
      ResetPassModel().resetPassword(email, context);
    },
    child: const Text('Reset'),
  ),  // ElevatedButton
),  // SizedBox
```

Figure 51: Password Reset Code Snippet

The reset button takes the emailController which is to retrieve the input from the textfield. Then, the app executes the resetpassword function, which will send an email to the user to reset their password.

4.3 Sign Up

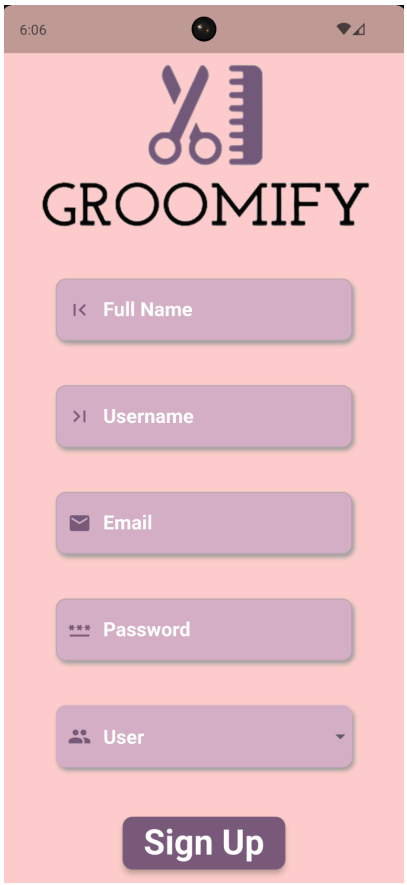


Figure 52: Sign Up

This is the sign up page where users could register as user or groomer. Users are required to enter their credentials such as fullname, username email and password to complete the signup process.

```
onPressed: () {
  final emailValidationResult = AuthController.instance.validateEmail(emailController.text);
  final passwordValidationResult = AuthController.instance.validatePassword(passwordController.text);
  final fullNameValidationResult = AuthController.instance.validateFullName(fullNameController.text);
  final usernameValidationResult = AuthController.instance.validateUserName(usernameController.text);

  if (emailValidationResult == '' && passwordValidationResult == '') {
    // Based on the selected role, navigate to the appropriate page
    if (selectedRole == 'User') {
      // Both email and password are valid, proceed with registration
      AuthController.instance.register(
        emailController.text.trim(),
        passwordController.text.trim(),
        fullNameController.text.trim(), // Provide full name here
        usernameController.text.trim(),
        selectedRole,
      );

      Get.to(const HomePage());
    } else if (selectedRole == 'Groomer') {
      AuthController.instance.groomerRegister(
        emailController.text.trim(),
        passwordController.text.trim(),
        fullNameController.text.trim(), // Provide full name here
        usernameController.text.trim(),
        selectedRole,
      );

      Get.to(const GroomerHome());
    }
```

Figure 53: Sign Up Code Snippet

This part of the code shows that the signup button takes emailController, passwordController, fullNameController, usernameController and selectedRole that will retrieve the entire signed up data from the textfield and role selection box. Similar to the login page, the function validates the credentials to ensure that it is valid before successful registration.

4.4 Homepage

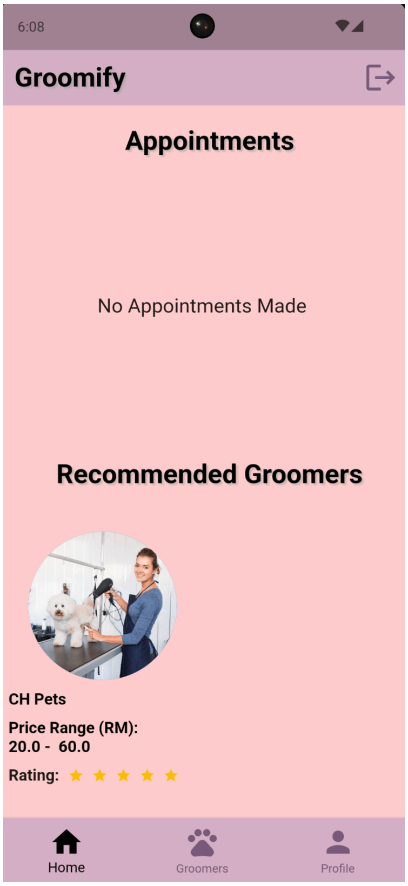


Figure 54: Home Page

Users with ‘User’ role will be directed to the homepage after successful login and signup. In the Homepage, the appointment made by the logged in user is shown in a listview. In the recommended groomers section, only groomers with 5 star rating will be shown.

```
// Appointments ListView
Container(
  padding: const EdgeInsets.only(top: 2, bottom: 2),
  height: appointmentData.isEmpty ? 280 : 450,
  child: appointmentData.isEmpty
      ? const Center(
    child: Text(
      "No Appointments Made",
      style: TextStyle(
        fontSize: 20,
      ),  // TextStyle
    ),  // Text
  )  // Center
      : ListView.builder(
    shrinkWrap: true,
    physics: const AlwaysScrollableScrollPhysics(),
    itemCount: appointmentData.length, // Show a maximum of 2 appointments
    itemBuilder: (context, index) {
      final appointment = appointmentData[index];
      final selectedDate = appointment['selectedDate'];

      // Format the date to "day, month, year" format
      final formattedDate = DateFormat('d MMMM y').format(selectedDate.toDate());

      return Container(
        margin: const EdgeInsets.all(10),
        padding: const EdgeInsets.all(10),
        decoration: BoxDecoration(
          border: Border.all(color: Colors.black),
          borderRadius: BorderRadius.circular(15),
        ),  // BoxDecoration
```

Figure 55: Home Page Code Snippet

This part of the code shows the listview function to display the appointments. There is also an exception where if there are no appointments made, it display ‘No appointments made’ text.

4.5 Groomer List

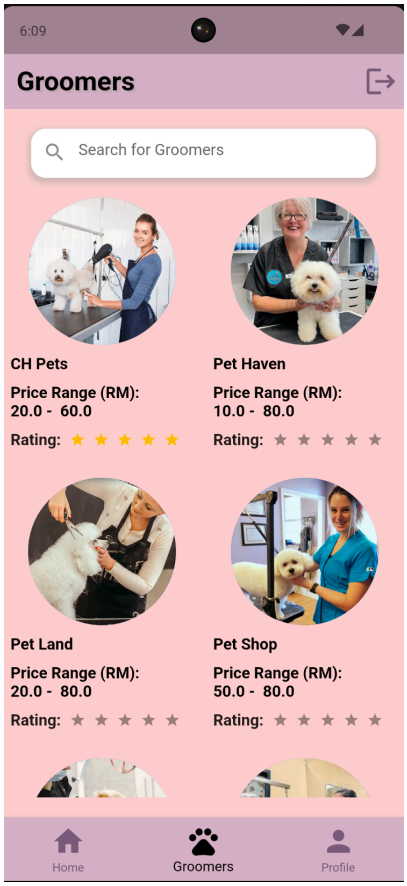


Figure 56: Groomer List

This page displays all the groomers registered to Groomify along with its summarize details (salonName, price range and rating). The groomers container are interactable which will direct user to groomer details for more information and make an appointment.

Figure 57: Groomer List Code Snippet

This part of the code shows that GridView is used to display the groomers with all adjustments necessary for better visuals. It takes filteredGroomerEmails which are groomer emails except that they are filtered to only display 5 star groomers. As for the groomer details, it is retrieved through the function getGroomerDatabyEmail.

4.6 Groomer Details

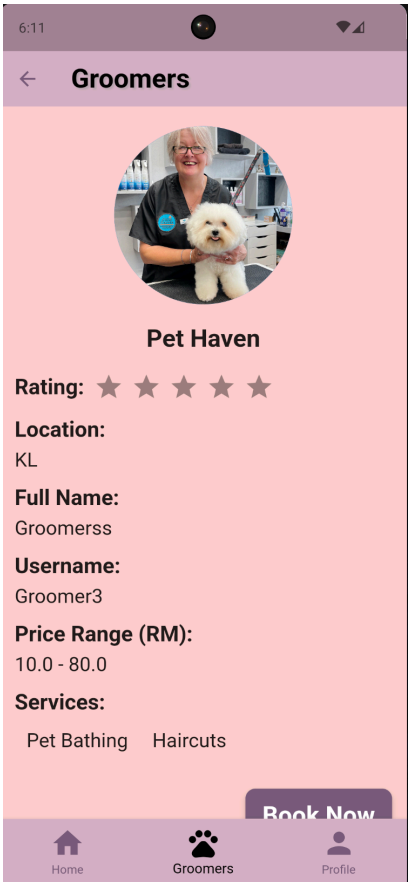
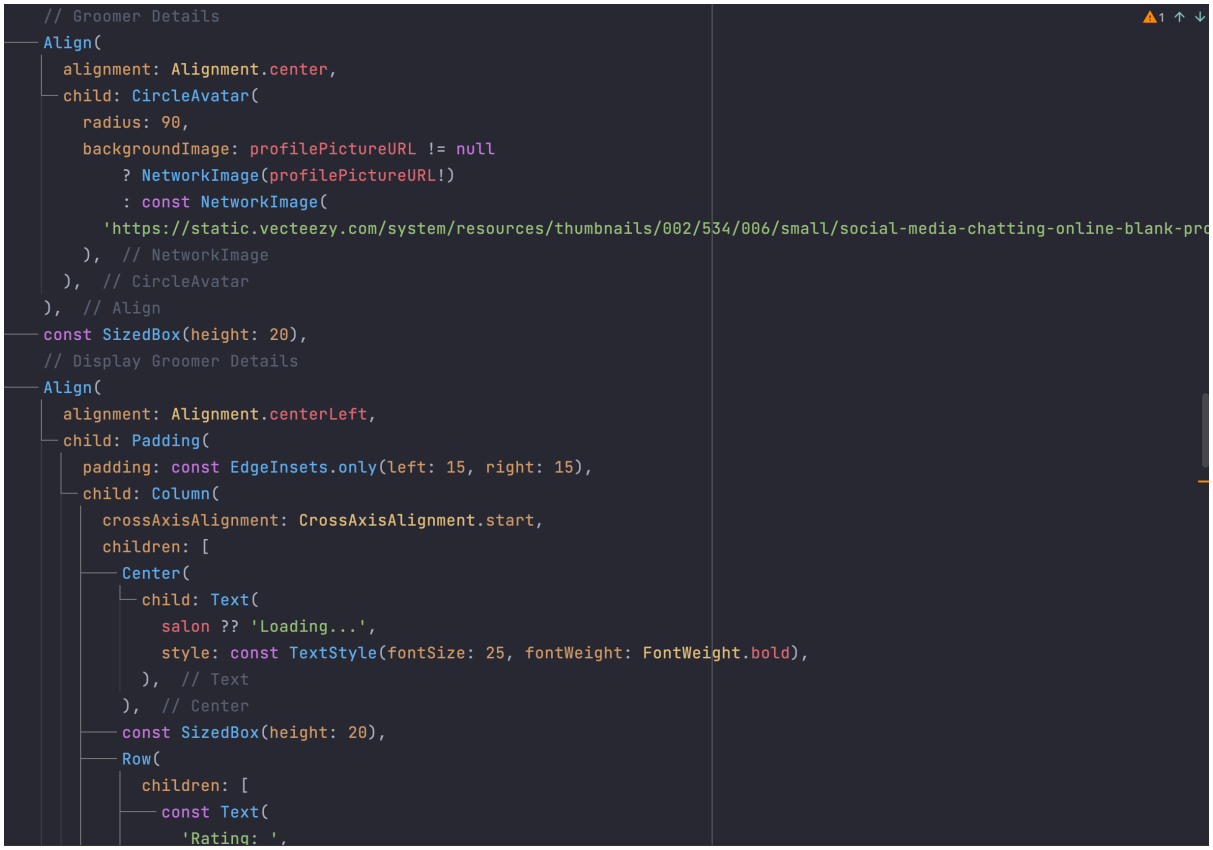


Figure 58: Groomer Details

After the user clicked the groomer containers, they will be directed to this page. This page displays detailed information of a groomer such as rating, location, fullName, username, price range and services. There is also a 'Book Now' button which direct users to appointment page.



```
// Groomer Details
Align(
  alignment: Alignment.center,
  child: CircleAvatar(
    radius: 90,
    backgroundImage: profilePictureURL != null
        ? NetworkImage(profilePictureURL!)
        : const NetworkImage(
      'https://static.vecteezy.com/system/resources/thumbnails/002/534/006/small/social-media-chatting-online-blank-pro
    ),  // NetworkImage
  ),  // CircleAvatar
),  // Align
const SizedBox(height: 20),
// Display Groomer Details
Align(
  alignment: Alignment.centerLeft,
  child: Padding(
    padding: const EdgeInsets.only(left: 15, right: 15),
    child: Column(
      crossAxisAlignment: CrossAxisAlignment.start,
      children: [
        Center(
          child: Text(
            salon ?? 'Loading...',
            style: const TextStyle(fontSize: 25, fontWeight: FontWeight.bold),
          ),  // Text
        ),  // Center
        const SizedBox(height: 20),
        Row(
          children: [
            const Text(
              'Rating: ',
```

Figure 59: Groomer Details Code Snippet

The code shows how groomer details are being displayed, including the profile picture. Similar to the variables such as salon, profile picture is display using the profilepictureurl retrieved from firestore.

4.7 Appointment Page

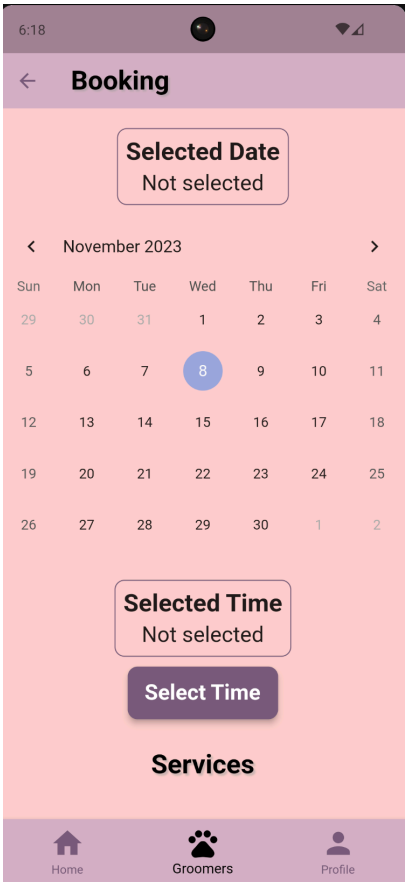


Figure 60: Appointment

This page displays a calendar which allows user to select date, a button to schedule the time and checkbox to select services. The user are required to enter all values to successfully make an appointment.

```
body: SingleChildScrollView(
  child: Column(
    children: [
      const SizedBox(height: 20),
      DateBox(
        title: 'Selected Date',
        date: _selectedDate,
      ),  // DateBox
      const SizedBox(height: 10),
      Column(
        crossAxisAlignment: CrossAxisAlignment.center,
        children: [
          TableCalendar(
            firstDay: DateTime.utc(2023, 1, 1),
            lastDay: DateTime.utc(2023, 12, 31),
            focusedDay: _focusedDay,
            calendarFormat: CalendarFormat.month,
            selectedDayPredicate: (day) {
              return isSameDay(_selectedDate, day);
            },
            onDaySelected: _onDaySelected,
          ),  // TableCalendar
          const SizedBox(height: 30),
          // Display selected time in one box
          GestureDetector(
            onTap: _selectTime,
            child: TimeBox(
              title: 'Selected Time',
              time: _selectedTime,
            ),  // TimeBox
          ),  // GestureDetector
          const SizedBox(height: 10),
```

Figure 61: Appointment Code Snippet

This part of the code shows the tablecalendar function which is used to display a calendar in the appointment page. The calendar is shown in a monthly format, and it calls the _onDaySelected function to retrieve the date that user selected. Similarly, _selectedTime variable is used to display the time selected in the box.

4.8 Complete Appointment

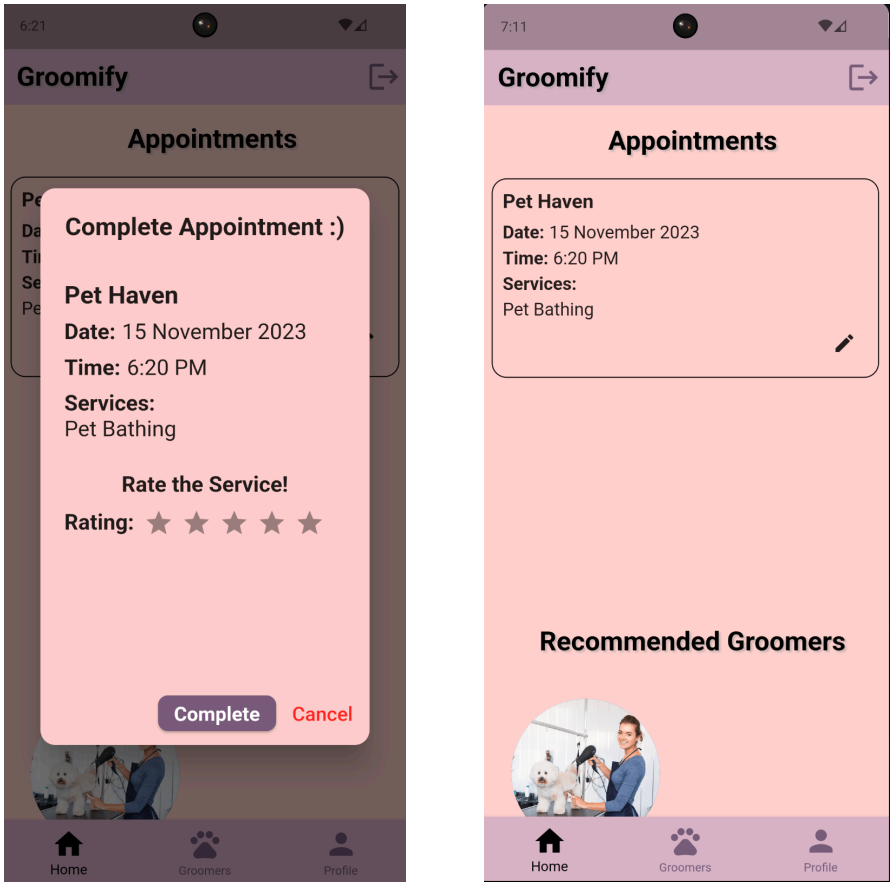


Figure 62: Complete Appointment

After the user made an appointment, they will be directed to the homepage where the appointments will be displayed. There is an edit icon which pop up a dialog for users to complete the appointment. This function is to allow users to confirm that the services has been received and they could rate the grooming services.

```
void _showEditDialog(Map<String, dynamic> appointment) {
  showDialog(
    context: context,
    builder: (context) {
      final groomerEmail = appointment['email'];
      final selectedDate = appointment['selectedDate'];
      final formattedDate =
      DateFormat('d MMMM y').format(selectedDate.toDate());
      final salonName = appointment['salonName'];
      final selectedTime = appointment['selectedTime'];
      final selectedServices = appointment['selectedServices'];
      final docID = appointment['documentID'];

      return AlertDialog(
        shape: RoundedRectangleBorder(
          borderRadius: BorderRadius.circular(15),
        ),  // RoundedRectangleBorder
        backgroundColor: const Color(0xffF7D1CD),
        title: const Center(
          child: Text(
            'Complete Appointment :)',
            style: TextStyle(
              fontSize: 25,
              fontWeight: FontWeight.bold,
            ),  // TextStyle
          ),  // Text
        ),  // Center
        content: SizedBox(
          width: 500,
          height: 400,
          child: Column(
            mainAxisSize: MainAxisSize.min,
```

Figure 63: Complete Appointment Code Snippet

This part of the code shows the function to display the pop up dialog. It retrieves the appointment details from firestore and saved into variables which is then used to display the data.

4.9 User Profile

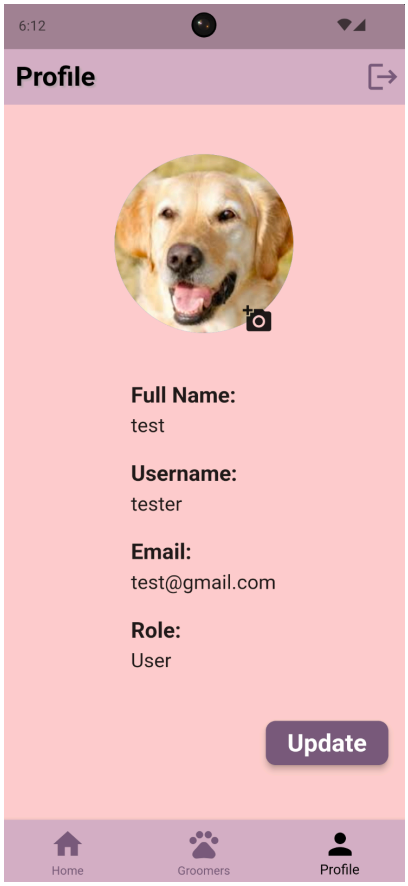


Figure 64: User Profile

The profile page displays the credentials that the user entered when signed up to the app. User can update their profile picture by clicking the camera icon.

```
body: Column(
  crossAxisAlignment: CrossAxisAlignment.center,
  children: [
    const SizedBox(height: 50),
    // Profile Picture
    Stack(
      children: [
        CircleAvatar(
          radius: 90,
          backgroundImage: profilePictureURL != null
              ? NetworkImage(profilePictureURL!)
              : const NetworkImage(
            'https://static.vecteezy.com/system/resources/thumbnails/002/534/006/small/social-media-chatting-online-blank-pro
          ),  // NetworkImage
        ),  // CircleAvatar
        Positioned(
          bottom: -10,
          left: 120,
          child: IconButton(
            iconSize: 30,
            onPressed: () {
              firestoreController.uploadGroomerProfilePicture(email!);
            },
            icon: const Icon(Icons.add_a_photo),
          ),  // IconButton
        )  // Positioned
      ],
    ),  // Stack
    const SizedBox(height: 50),
    //User Data
    Center(
```

Figure 65: User Profile Code Snippet

This part of the code shows how the user details is displayed in the page. The camera icon calls the uploadGroomerProfilePicture function which will save the profilePictureURL to firestore by overwriting it with the current url.

4.10 Groomer Home Page

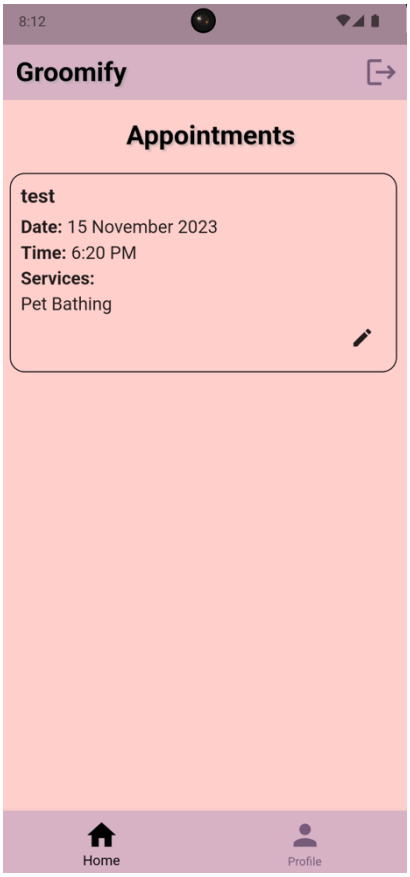


Figure 66: Groomer Home Page

When the user with 'Groomer' role successfully signed up or logged in to Groomify, they are greeted with this page. It displays appointments scheduled with them in details such as fullName, date, time and services required. Similar to the user homepage, groomers are able to click the icon button to mark the services has been provided.

```
body: SingleChildScrollView (
  child: Column(
    children: [
      const SizedBox(height: 20),
      //Text
      Container(
        alignment: Alignment.center,
        padding: const EdgeInsets.only(left: 15),
        child: Text(
          'Appointments',
          style: TextStyle(
            fontSize: 27,
            fontWeight: FontWeight.bold,
            color: Colors.black,
            shadows: [
              Shadow(
                offset: const Offset(2, 2),
                blurRadius: 3.0,
                color: Colors.grey.withOpacity(0.5),
              ),  // Shadow
            ],
          ),  // TextStyle
        ),  // Text
      ),  // Container
      const SizedBox(height: 10),
      // Appointments ListView
      Container(
        padding: const EdgeInsets.only(top: 2, bottom: 2),
        child: appointmentData.isEmpty
            ? const Center(
          child: Text(
            "No Appointments Made",
```

Figure 67: Groomer Home Page Code Snippet

This part of the code shows how the appointments are being displayed in groomer homepage. Unlike user homepage, 5 star groomers are not recommended to the groomers, thus using the entire homepage to display appointments.

4.11 Groomer Profile

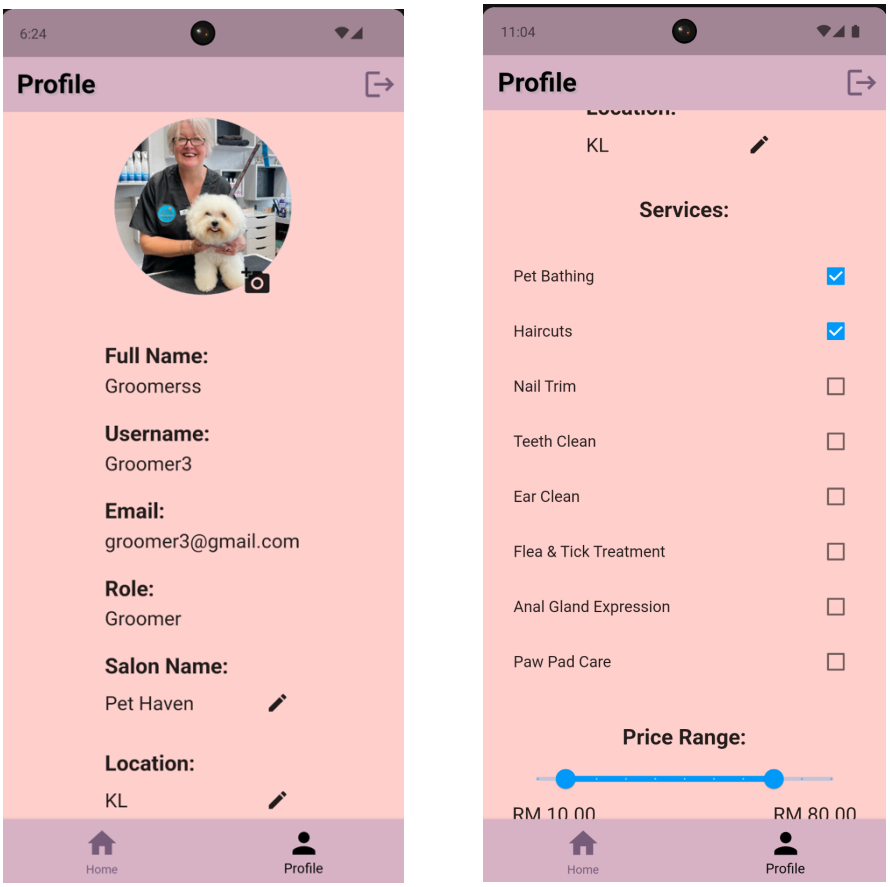


Figure 68: Groomer Profile

This is the profile page for groomers. Unlike user profile, groomer profile is more versatile as groomers are able to edit salonName, location, services and price range, hence managing their business.

```
// Salon Name
SizedBox(
  width: 200,
  child: Column(
    crossAxisAlignment: CrossAxisAlignment.start,
    children: [
      const Text(
        'Salon Name:',
        style: TextStyle(
          fontSize: 22,
          fontWeight: FontWeight.bold,
        ),  // TextStyle
      ),  // Text
      Row(
        mainAxisAlignment: MainAxisAlignment.spaceBetween,
        children: [
          if (!isEditingSalon)
            Text(
              isEditingSalon ? salonController.text : (salon ?? ''), // Use the controller text if editing, or t
              style: const TextStyle(fontSize: 20),
            )  // Text
          else
            Expanded(
              child: TextFormField(
                controller: salonController,
                style: const TextStyle(
                  fontSize: 20,
                ),  // TextStyle
                decoration: const InputDecoration(
                  hintText: 'Enter Name',
                ),  // InputDecoration
```

Figure 69: Groomer Profile Code Snippet

This part of the code shows the editing functionality of the groomer profile. isEditingSalon is a function that enables the groomer to edit the salon name, when salon name textfield is in editing status, the new salon name will replace the existing salon name placeholder. It is then uploaded to firestore when the user clicked update button.

4.12 Firebase Authentication

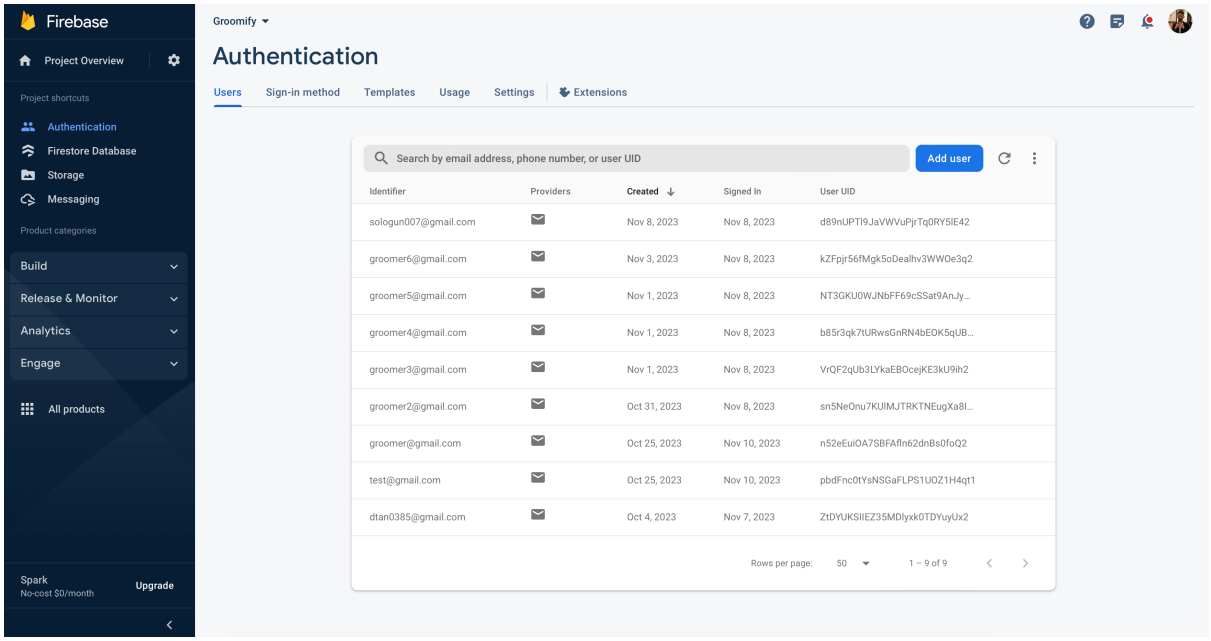


Figure 70: Firebase Authentication Interface

This shows the Firestore authentication interface. These are the emails of the users and groomers who have registered to Groomify. The password of the account are all hashed by the Firebase default parameters.

4.13 Firebase Firestore

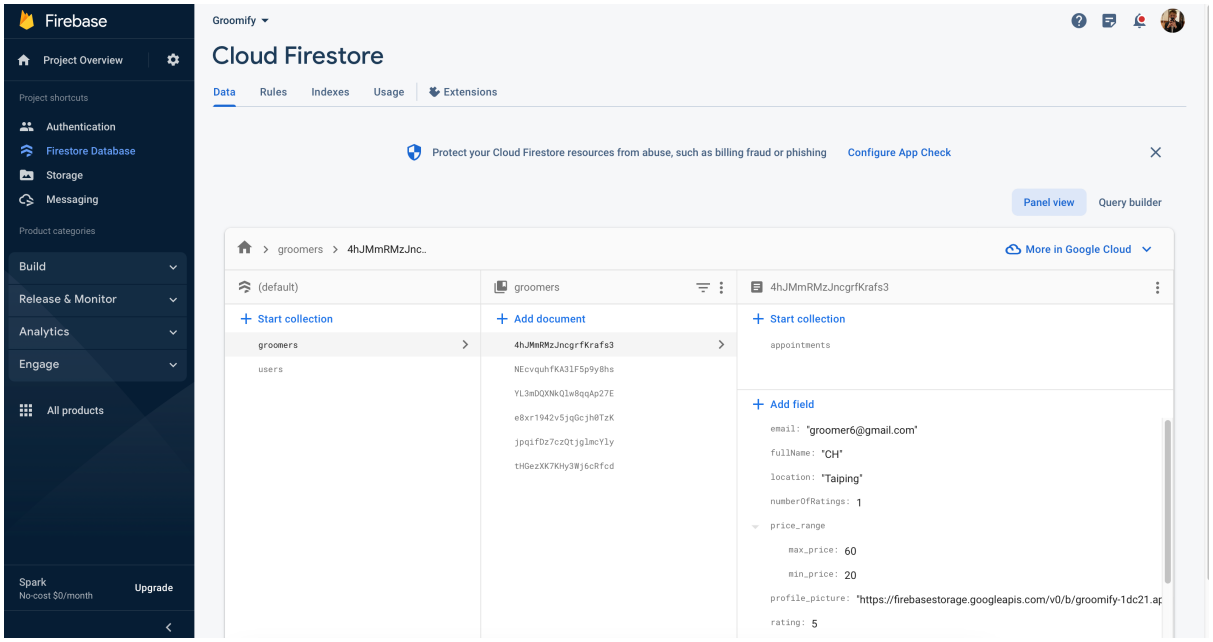


Figure 71: Firestore Interface

This shows the Firestore interface. This is where all user and groomers data are stored at, in different collections, ‘Users’ and ‘Groomers’.

4.14 Firebase Storage

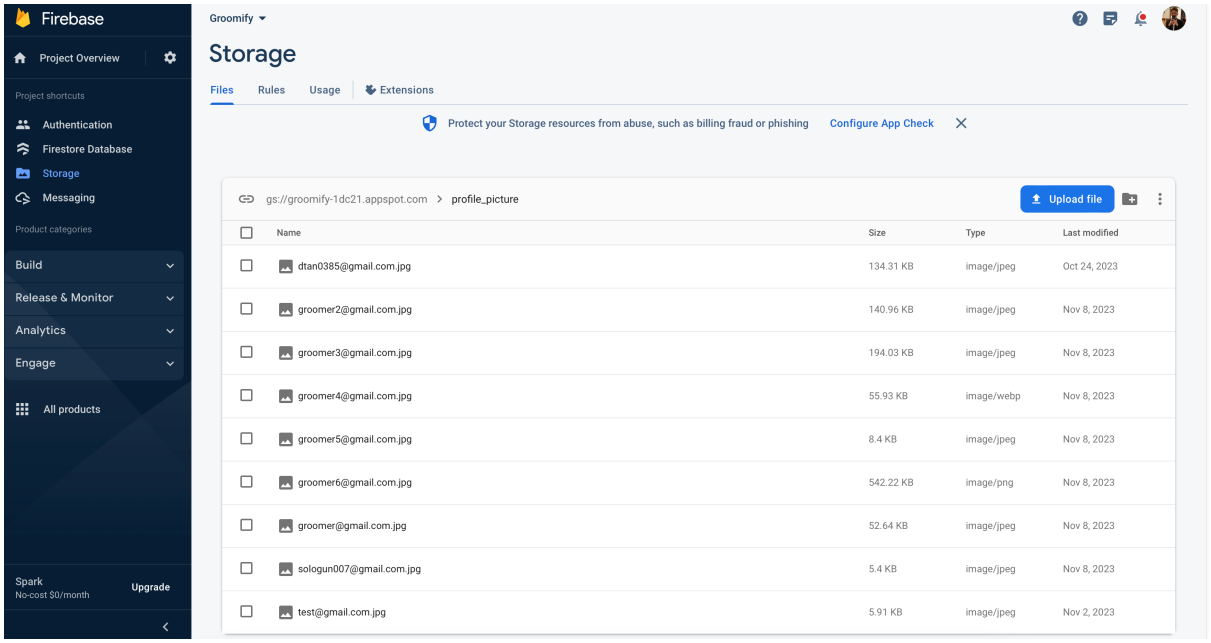


Figure 72: Firebase Storage Interface

This shows the Firebase storage interface. This is where all the profile pictures of users and groomers stored at.

Chapter 5: Testing and Evaluation

5.1 Functional Testing

Unit Testing

Unit testing is a software development methodology that entails a methodical analysis of discrete code units or components to verify their proper operation when isolated. Through the creation of brief, automated test cases, it focuses on confirming that each isolated unit of code, such as a function or method, accurately performs its intended task. Unit testing is essential for early bug detection and correction during development, improving the quality of the code, and making code maintenance easier by acting as a safety net to stop regressions during changes. It assists developers in finding and resolving problems more quickly by isolating and closely examining individual code units, resulting in software systems that are more dependable and resilient.

5.1.1 User Authentication

No.	Test Scenario	Expected Result	Evidence	Result
1	Log in without any credentials	Error Message will be displayed when email or password was empty	Figure 59	Pass
2	Log in using unregistered account	Error Message will be displayed if user uses unregistered email	Figure 60	Pass
3	Reset password using invalid email	Error Message will be displayed if user uses invalid email	Figure 61	Pass
4	Sign up without any credentials	Error Message will be displayed when required credentials are empty	Figure 62	Pass

Table 6: Unit Test (User Authentication)

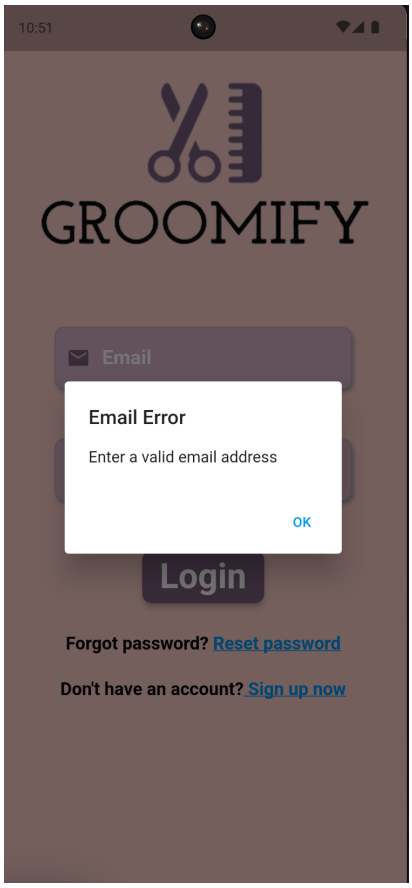


Figure 73: Log in without any credentials

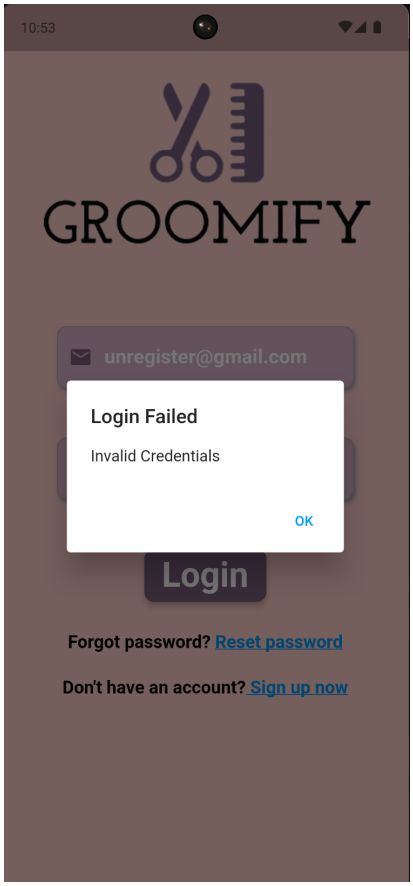


Figure 74: Log in using unregistered account

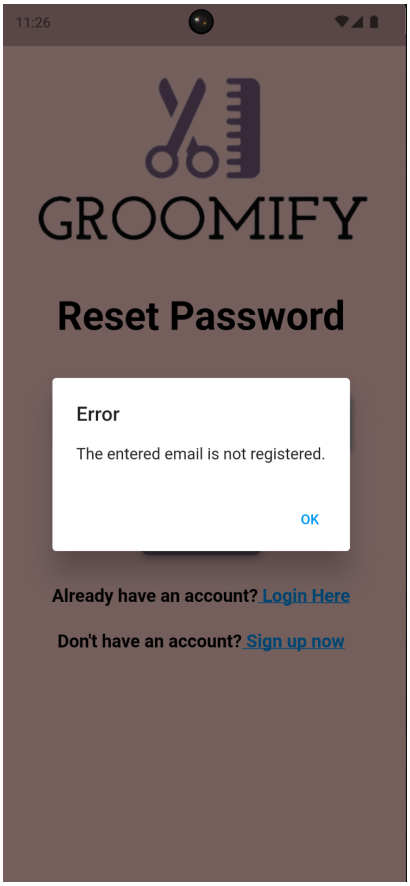


Figure 75: Reset password using invalid email

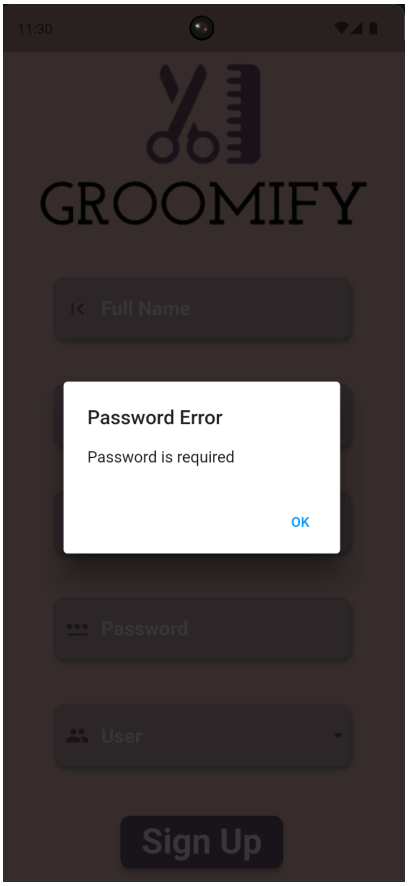


Figure 76: Sign up without any credentials

5.1.2 Appointment Creation

No.	Test Scenario	Expected Result	Evidence	Result
1	Create appointment for past date	Error Message will be displayed when date is before the current date	Figure 63	Pass
2	Create appointment for past time	Error Message will be displayed when time is before the current time	Figure 64	Pass
3	Create appointment without services	Error Message will be displayed when no services are selected	Figure 65	Pass

Table 7: Unit Test (Appointment Creation)

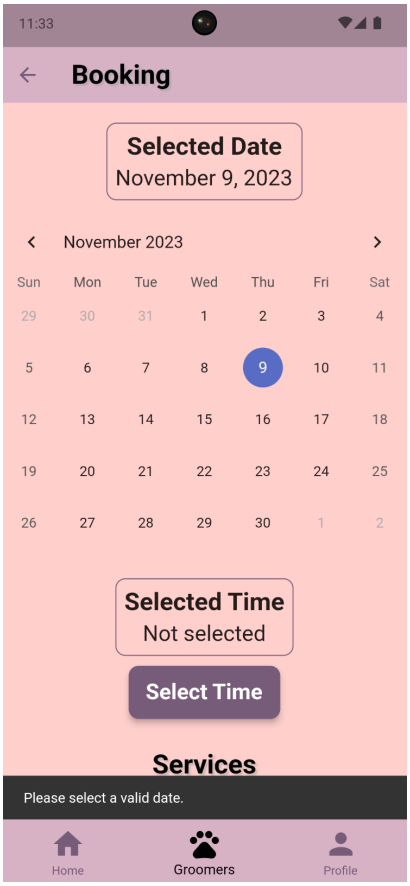


Figure 77:Create appointment for past date

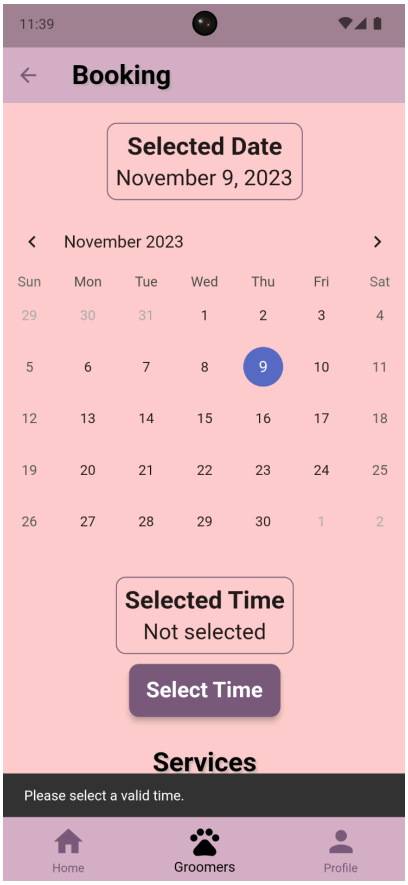


Figure 78: Create appointment for past time

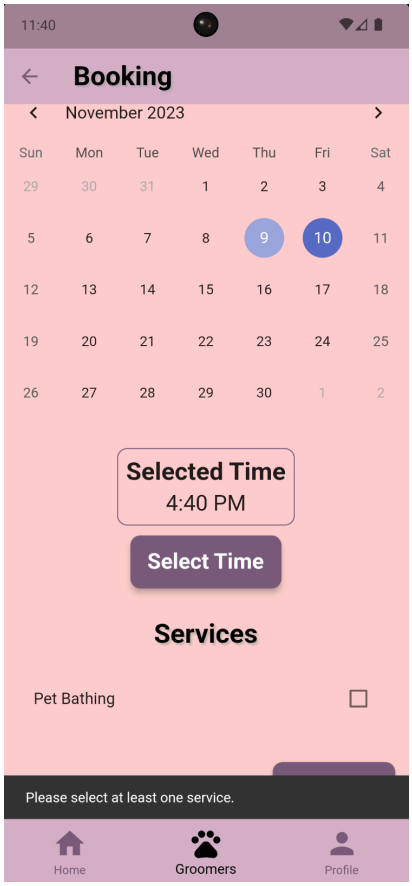


Figure 79: Create appointment without service

5.2 Integration Testing

Integration testing is a software testing methodology that evaluates the interactions and interfaces between various components or modules within a software system. It focuses on assessing how these components work together as a unified system and aims to uncover potential issues related to data flow, communication, and compatibility. Integration testing ensures that the integrated parts of the software collaborate seamlessly, identifying and resolving any inconsistencies or defects that may arise when these components are combined. By doing so, it guarantees that the software functions as intended in a real-world context and helps detect problems that may not be evident during unit testing, ultimately ensuring the reliability and stability of the entire system.

5.2.1 User Authentication

No.	Test Scenario	Expected Result	Evidence	Result
1	Log in and out	User able to login to homepage, and able to logout from homepage	Figure 66	Pass
2	Reset Password	User able to receive email to reset their password	Figure 67, 68	Pass
3	Sign up (User)	User able to signup to homepage	Figure 69	Pass
4	Sign up (Groomer)	Groomer able to signup to homepage	Figure 70	Pass

Table 8: Integration Test (User Authentication)

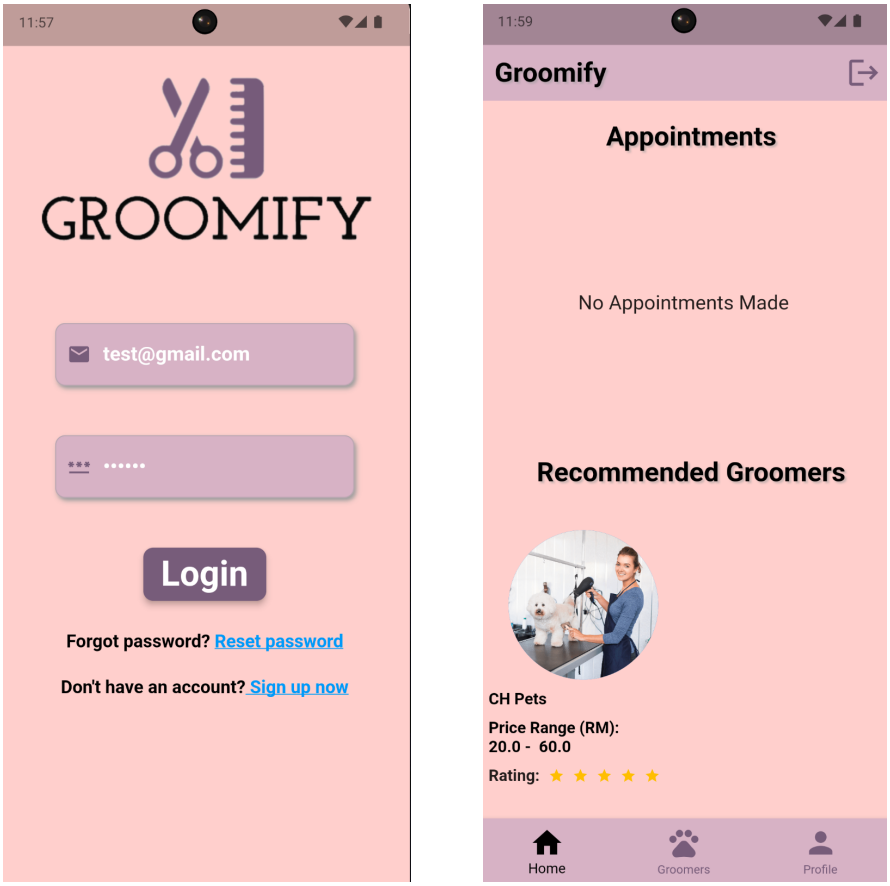


Figure 80: Log in and out

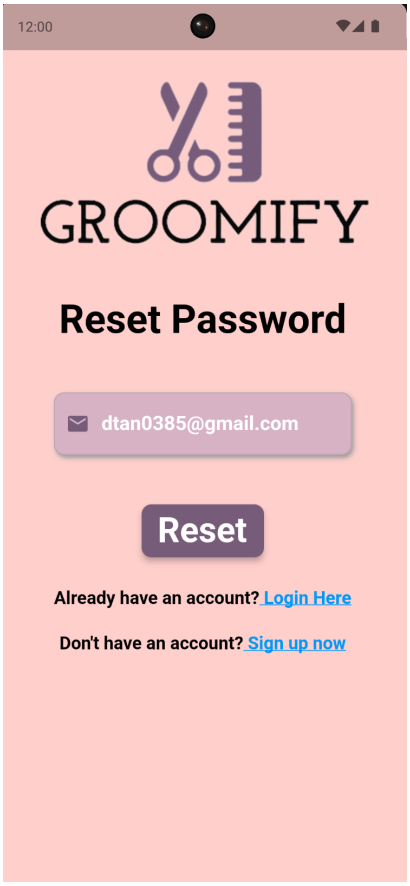


Figure 81: Reset password

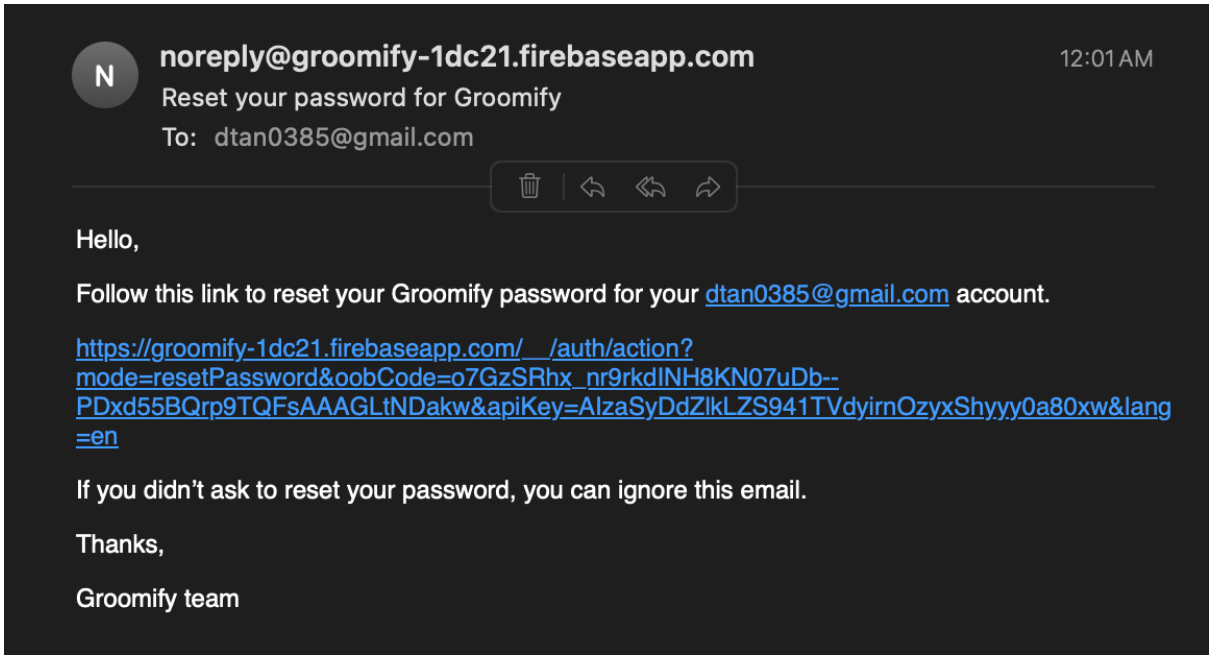


Figure 82: Reset password email

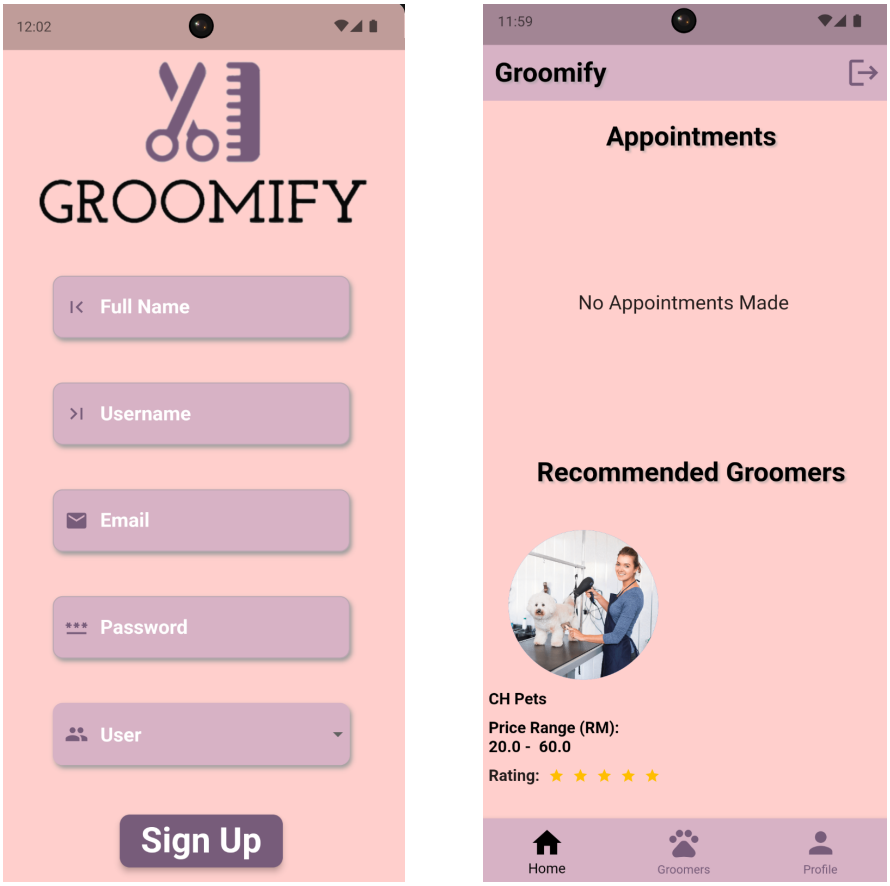


Figure 83: Sign up (User)

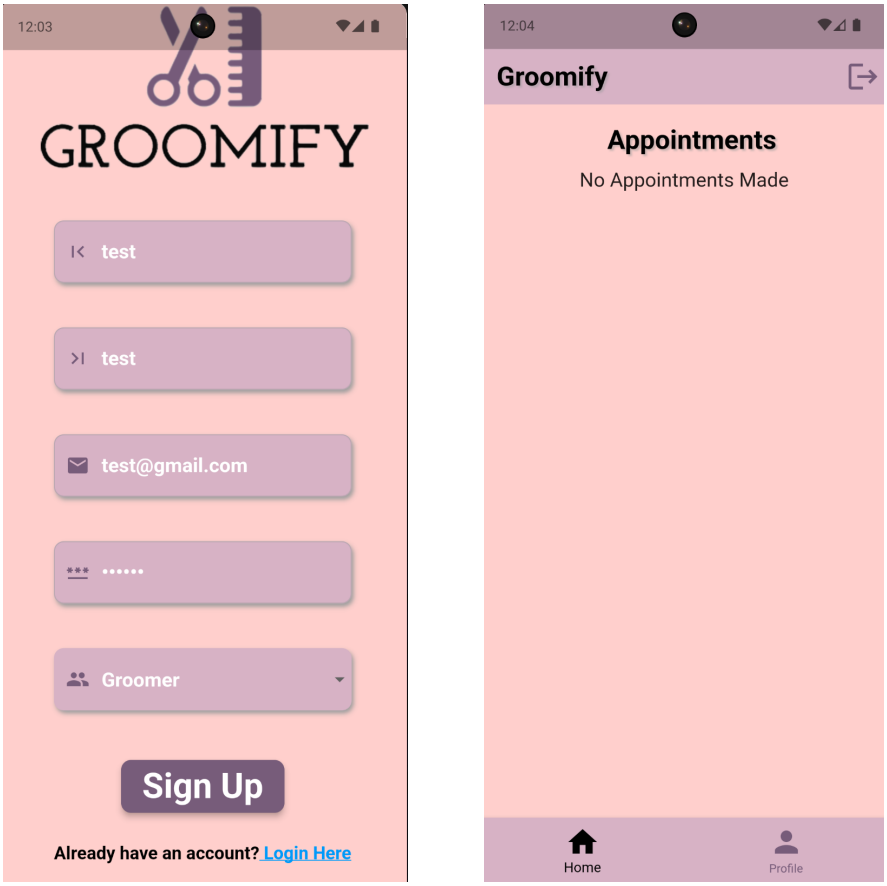


Figure 84: Sign up (Groomer)

5.2.2 Groomers

No.	Test Scenario	Expected Result	Evidence	Result
1	Display groomers	Able to display groomers	Figure 71	Pass
2	Display groomer details	Able to display more details when groomer is clicked	Figure 72	Pass
3	Search groomers	Able to search groomers	Figure 73	Pass

Table 9: Integration Test (Groomers)

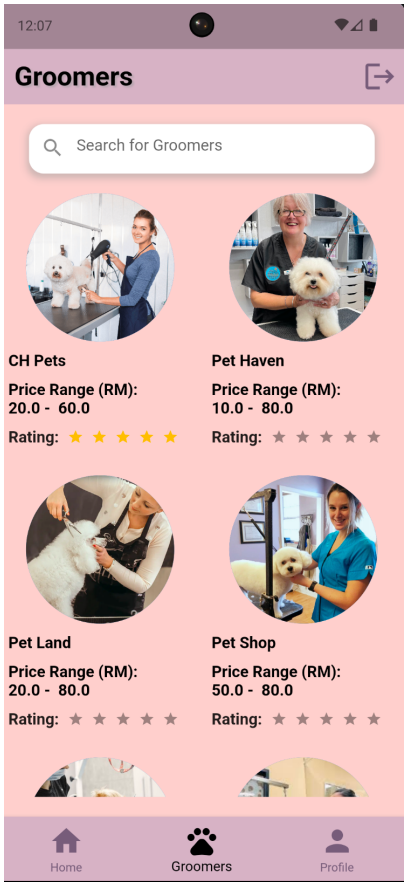


Figure 85: Display groomers

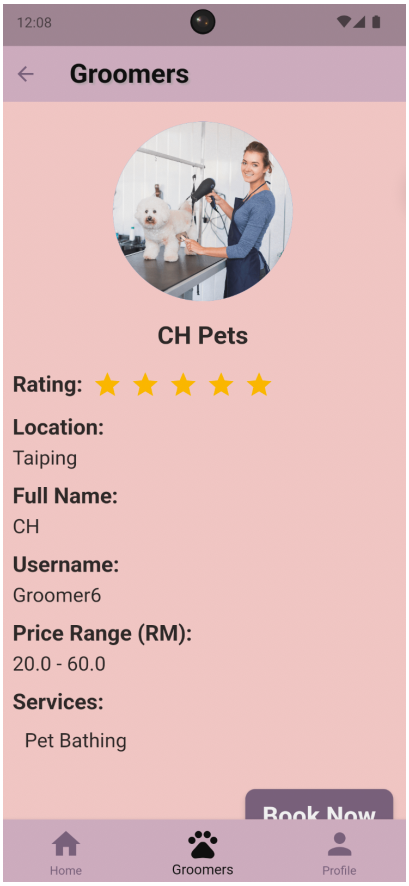


Figure 86: Display groomer details

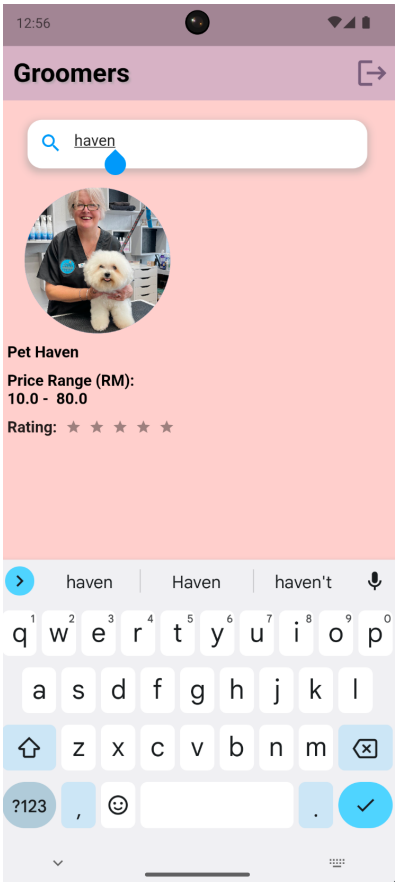


Figure 87: Search groomers

5.2.3 Appointments

No.	Test Scenario	Expected Result	Evidence	Result
1	Make appointment	Able to make appointment	Figure 74, 75	Pass
2	Display appointments	Able to display appointments in homepage	Figure 76	Pass
3	Rate service	Able to rate groomer service	Figure 77	Pass
4	Complete appointment	Able to complete an appointment by clicking the edit button and complete button	Figure 78	Pass
5	Remove appointment	Remove appointment after completion	Figure 79, 80	Pass

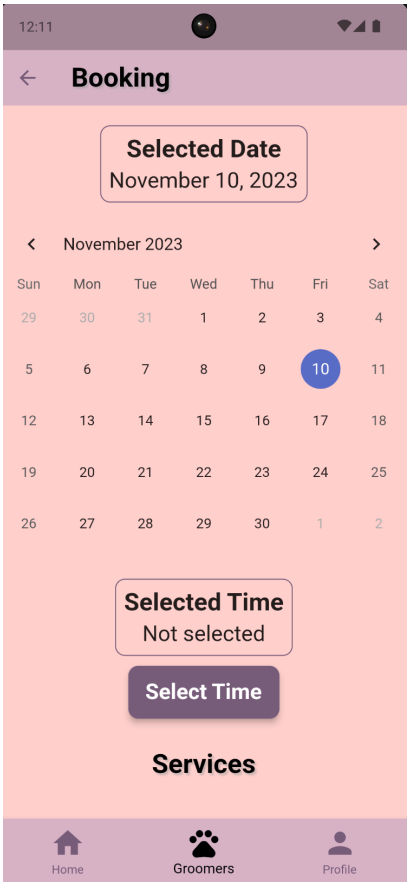


Figure 88: Make appointment

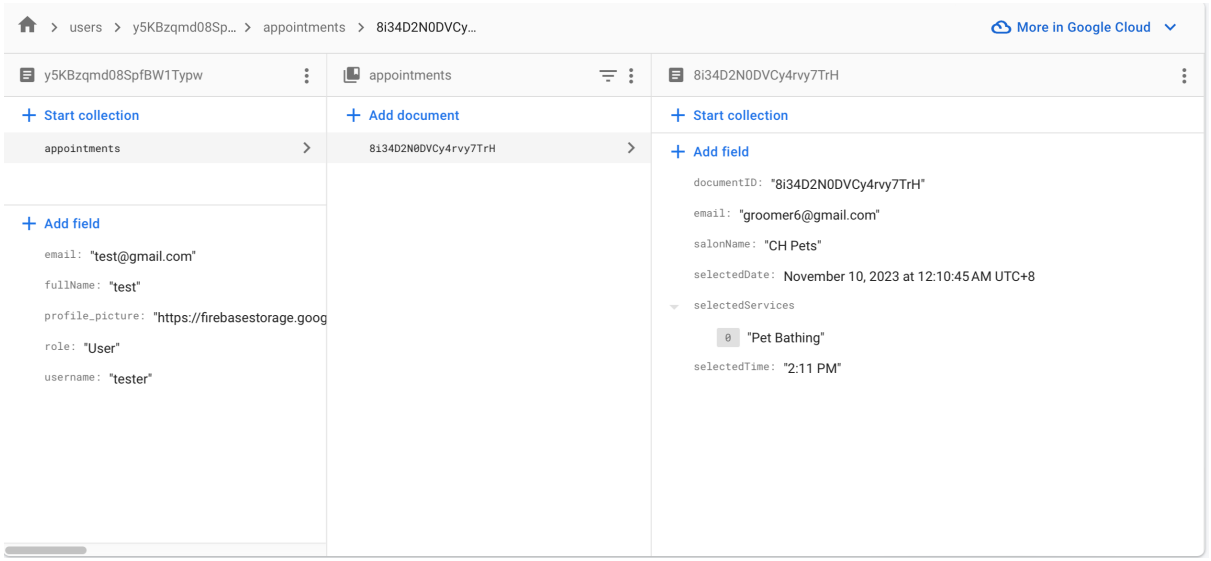


Figure 89: Appointment Firestore

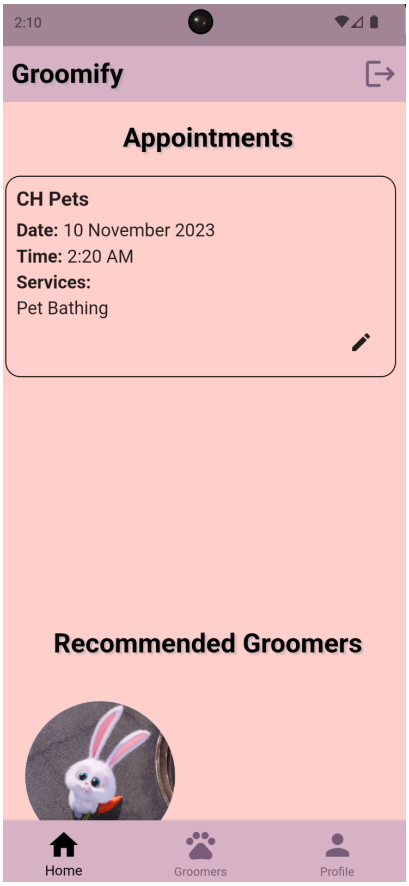


Figure 90: Display appointments

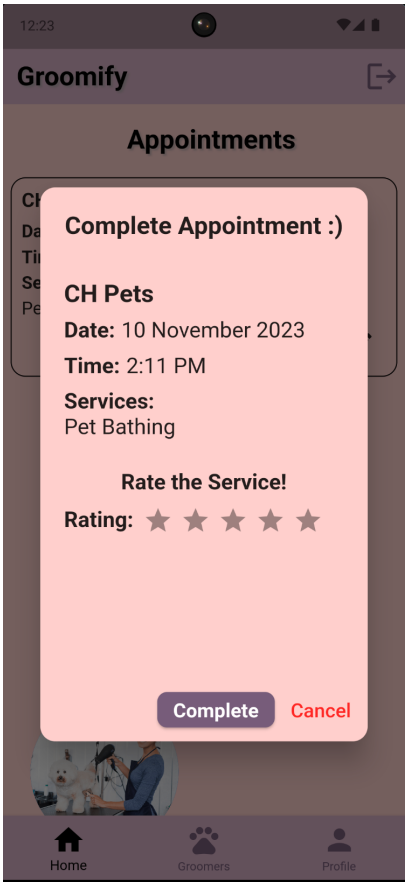


Figure 91: Complete appointments

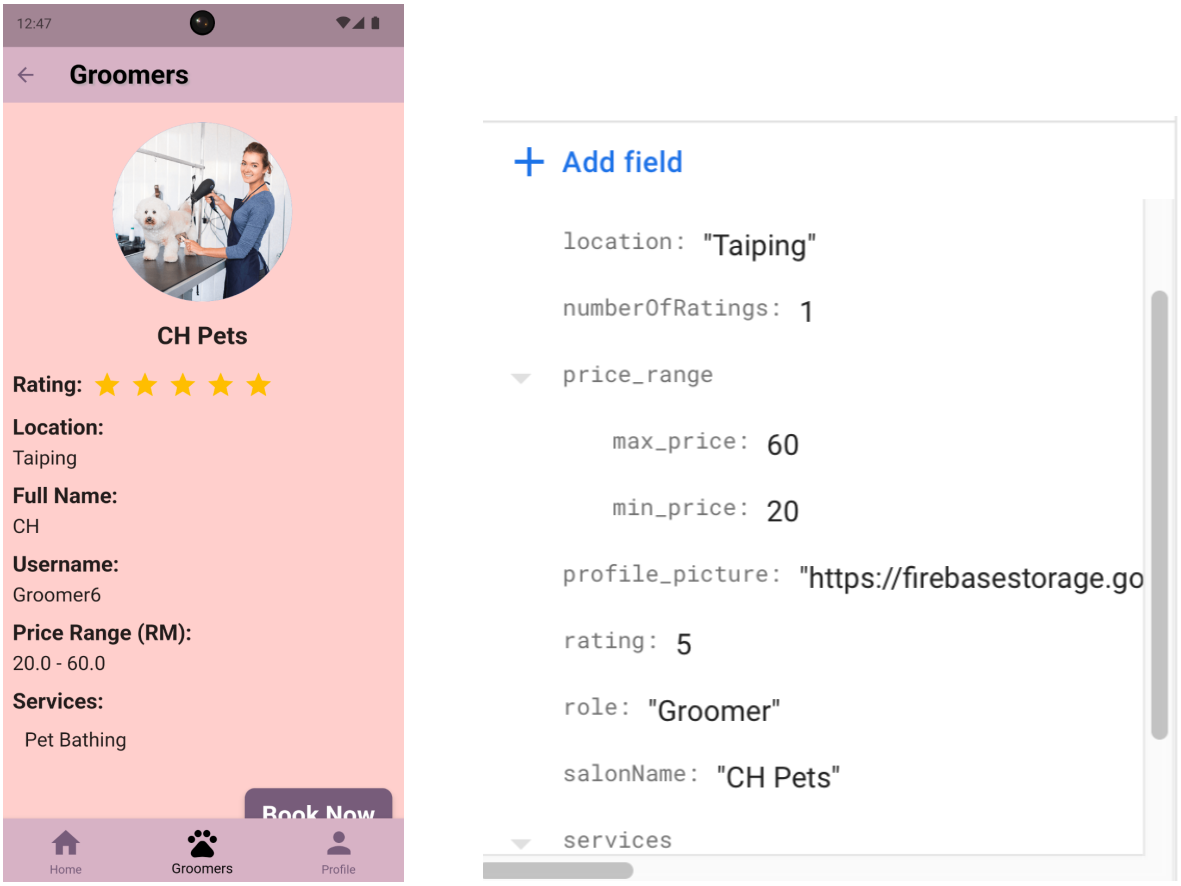


Figure 92: Rate service

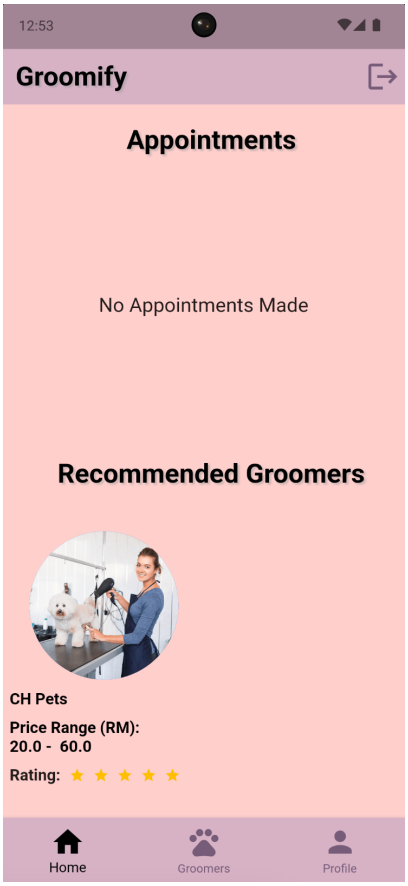


Figure 93: Complete appointments

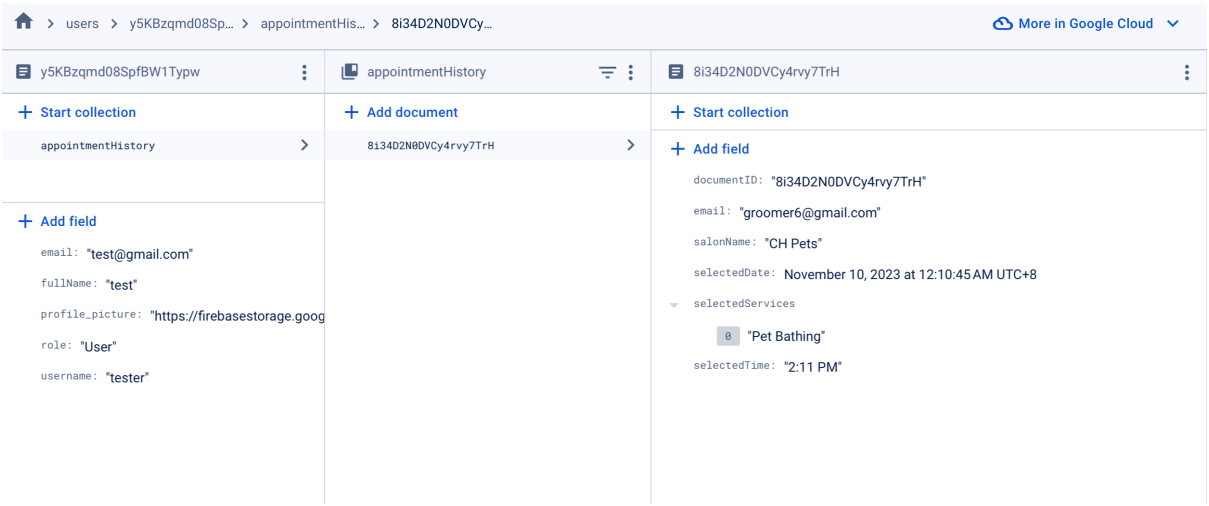


Figure 94: Complete appointments Firebase

5.2.4 Profile

No.	Test Scenario	Expected Result	Evidence	Result
1	Display user profile	Able to display user profile	Figure 81	Pass
2	Update user profile picture	Able to update profile picture	Figure 82	Pass
3	Display groomer profile	Able to display groomer profile	Figure 83	Pass
4	Update groomer profile picture	Able to update groomer profile picture	Figure 84	Pass
5	Update groomer details	Able to update groomer details such as salon name, location, services and price range	Figure 85	Pass

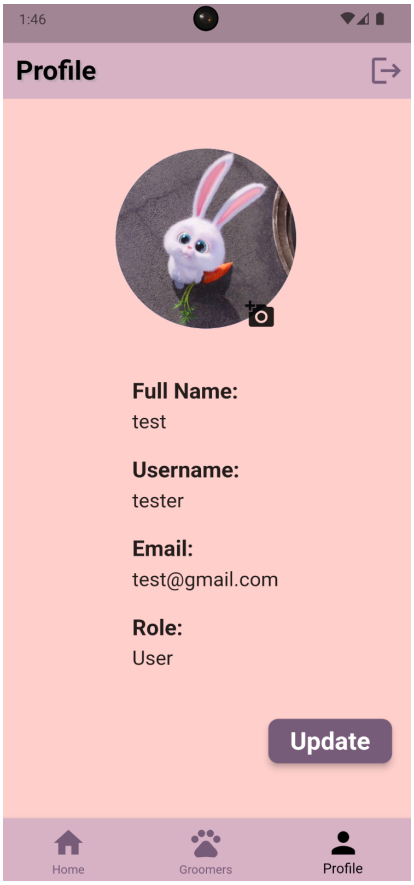


Figure 95: Display user profile

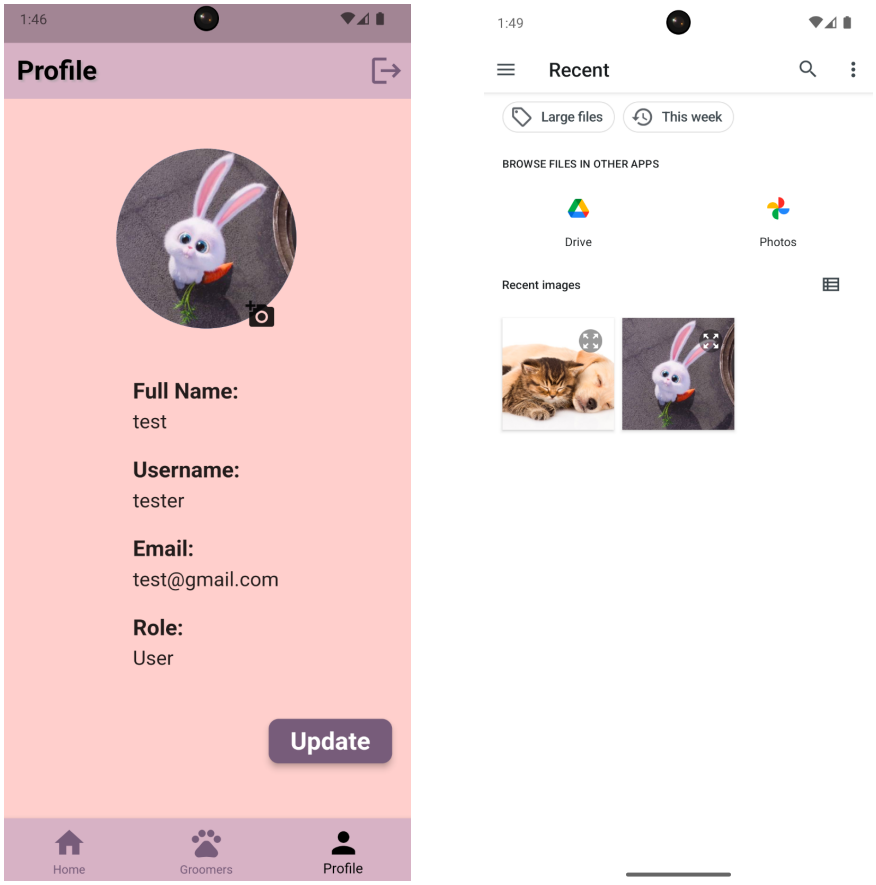


Figure 96: Update user profile picture

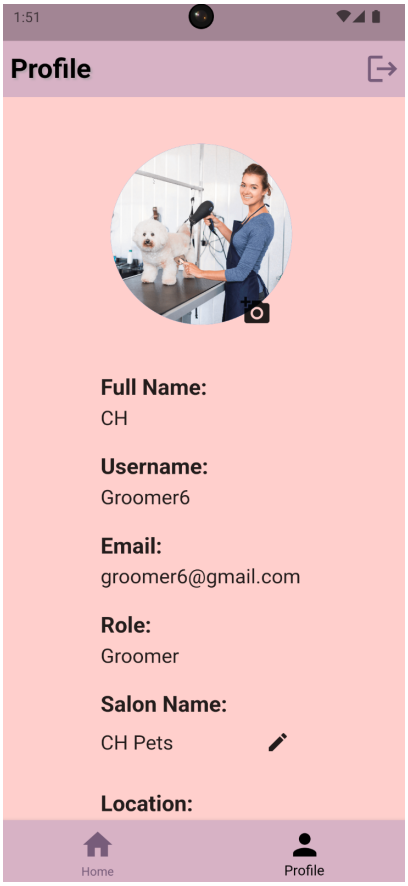


Figure 97: Display groomer profile picture

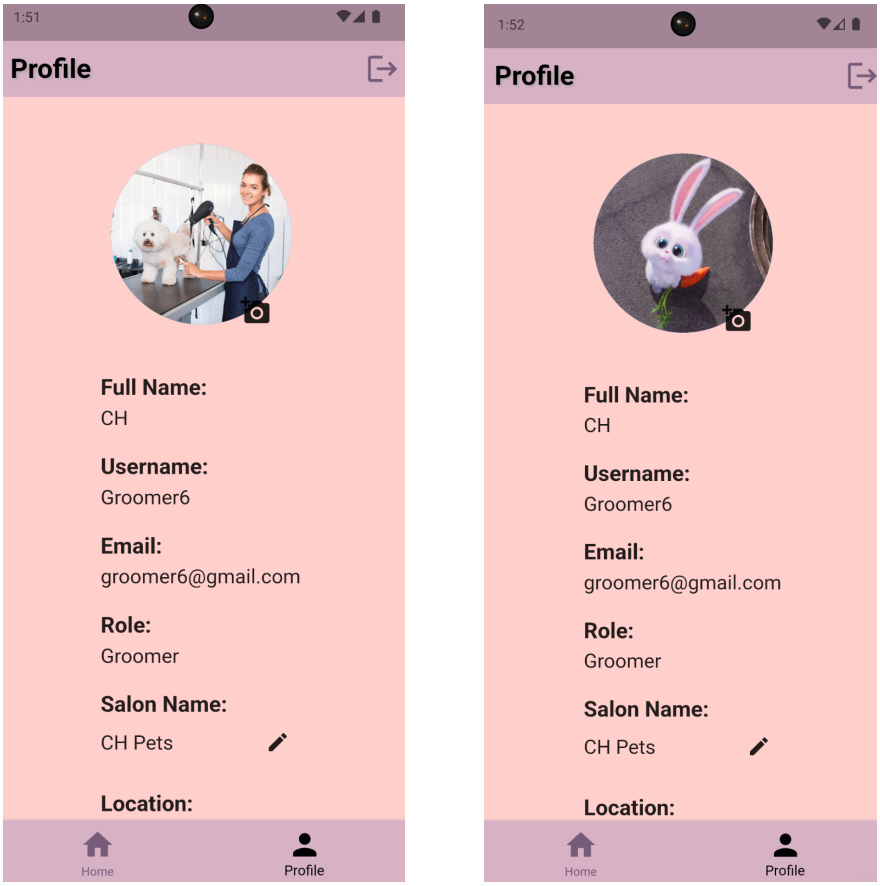


Figure 98: Update groomer profile picture

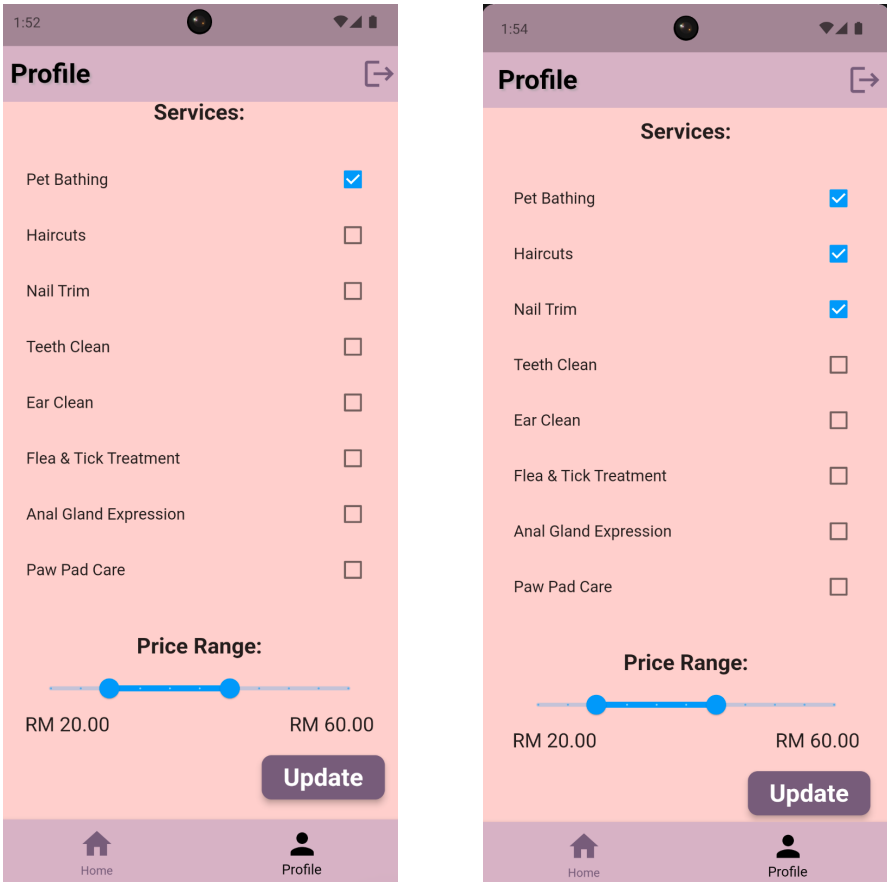


Figure 99: Update groomer details

5.3 Security Testing

```dart
//AuthController is accessible globally
class AuthController extends GetxController {
  //Allow AuthController to be accessible from any pages
  static AuthController instance = Get.find();
  //Declare variable of firebase users (email, pass, name...)
  late Rx<User?> _user;
  //Firebase auth module
  FirebaseAuth auth = FirebaseAuth.instance;
  //Firebase firestore module
  FirebaseFirestore firestore = FirebaseFirestore.instance;
```

Figure 100: Firebase Authentication Method

An instance of the AuthController is created for global accessibility in the provided Dart code for an AuthController in a Flutter app using GetX and Firebase. Using Firebase Authentication and Firestore, the controller maintains the status of user authentication. In order to enable dynamic UI updates, such as navigation to login or home screens based on the user's authentication status, initialization involves initialising a variable to hold the current user information, binding a stream to track user state changes, and using the ever method to react to user state changes by invoking the _initialScreen method. The Flutter application's management of user authentication is streamlined by this controller structure.

```dart
void login(String email, String password) async {
  try {
    final UserCredential userCredential = await auth.signInWithEmailAndPassword(email: email, password: password);

    if (userCredential.user != null) {
      navigateBasedOnRole(email);
    } else {
      showErrorPopup(Get.context!, 'Login Failed', 'Invalid Credentials');
    }
  } catch (e) {
    showErrorPopup(Get.context!, 'Login Failed', 'Invalid Credentials');
  }
}
```

Figure 101: Login Code Snippet

The login function in the given code takes care of user authentication by accepting an email address and password as parameters. For the authentication procedure, Firebase Authentication is used. The try block uses auth.signInWithEmailAndPassword(email: email, password: password) to try and log the user in with the supplied email and password. The user is presumably authenticated if the authentication process is successful and userCredential.user is not null. At this point, the user is navigated according to their role. However, in the event that authentication is unsuccessful, an error popup stating that the login attempt was unsuccessful due to invalid credentials is displayed once the catch block is executed.

For enhanced password security, Firebase Authentication employs a robust security practice by not storing actual plaintext passwords in the database. Instead, it securely stores hashed passwords. When a user registers or logs in, Firebase hashes their password and compares it to the hashed version stored in the database. Hashing is a one-way cryptographic function that transforms the password into an irreversible string of characters, making it computationally infeasible for anyone, including administrators, to retrieve the original password from the hash. This approach adds an extra layer of protection to users' sensitive information, ensuring that even if the database were compromised, attackers would not have direct access to user passwords. Hashed passwords are a fundamental component of modern security practices, safeguarding user data from potential breaches and minimizing the risks associated with storing plaintext passwords.

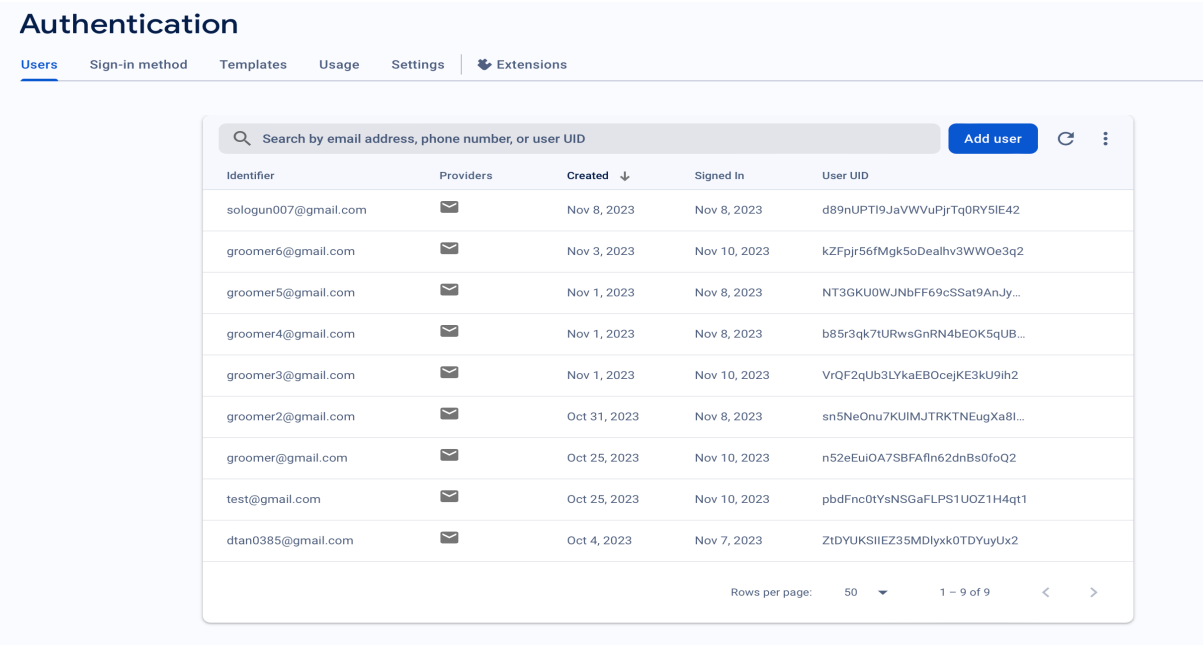


Figure 102: Firebase Authentication Interface

```
hash_config {
  algorithm: SCRYPT,
  base64_signer_key: m3Q0rg9qXphYnaufqESDbryt8cXnIHhjX2akn6YHSj3GZMzPW6AR7NRB8+KBf9LDlrZ7485PPafnc3LK40FGkg==,
  base64_salt_separator: Bw==,
  rounds: 8,
  mem_cost: 14,
}
```

Figure 103: Password Hash Parameters

No	Test Case	Observation	Evidence	Result
1	Password is hashed and hidden in Firebase Authentication	Password is hashed and hidden	Figure 88, 89	Pass
2	Account creation for Firebase Authentication	Registered user account is saved in Firebase Authentication	Figure 90	Pass

Table 10: Security Test (Firebase Authentication)

```
void login(String email, String password) async {
  try {
    final UserCredential userCredential = await auth.signInWithEmailAndPassword(email: email, password: password);

    if (userCredential.user != null) {
      navigateBasedOnRole(email);
    } else {
      showErrorPopup(Get.context!, 'Login Failed', 'Invalid Credentials');
    }
  } catch (e) {
    showErrorPopup(Get.context!, 'Login Failed', 'Invalid Credentials');
  }
}
```

Figure 104: Login Code Snippet

The login function uses Firebase Authentication to safely authenticate users. The `signInWithEmailAndPassword method of Firebase Authentication receives the user's email address and password when they try to log in. Google offers Firebase Authentication, a powerful and extremely secure service that uses industry-standard security procedures to protect user credentials. It guarantees that passwords are salted and hashed before being stored, preventing exposure. A `UserCredential` is obtained if the login process is successful, and if the credential's `user` property is not null, it signifies a successful authentication. This

indicates that Firebase's securely stored credentials have been used to validate the user's email address and password. By reducing common security risks related to managing user authentication, like password hashing and salting, Firebase Authentication offers a strong and dependable authentication system for the application. The code ensures a seamless user experience by handling errors gracefully, such as displaying an error message in the event of an unsuccessful login attempt.

No	Test Case	Observation	Evidence	Result
1	Login using unregistered email	Failed to Login	Figure 91	Pass
2	SQL Injection	Failed to Login	Figure 92	Pass
3	Brute Force Attack	Request Block from Device	Figure 93	Pass

Table 11: Security Test (User Authentication)

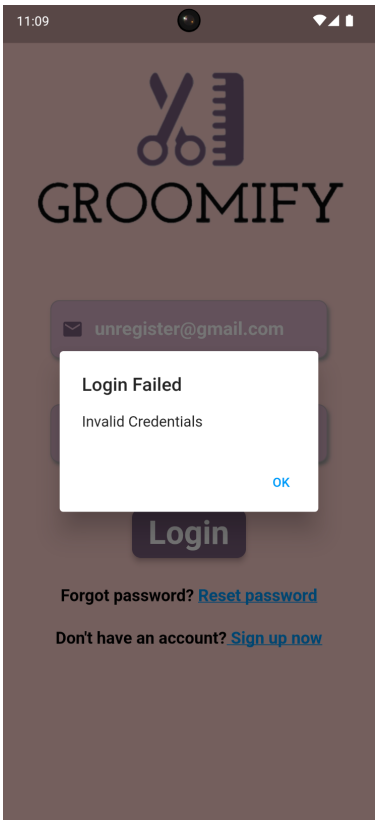


Figure 105: Unregistered Login Test

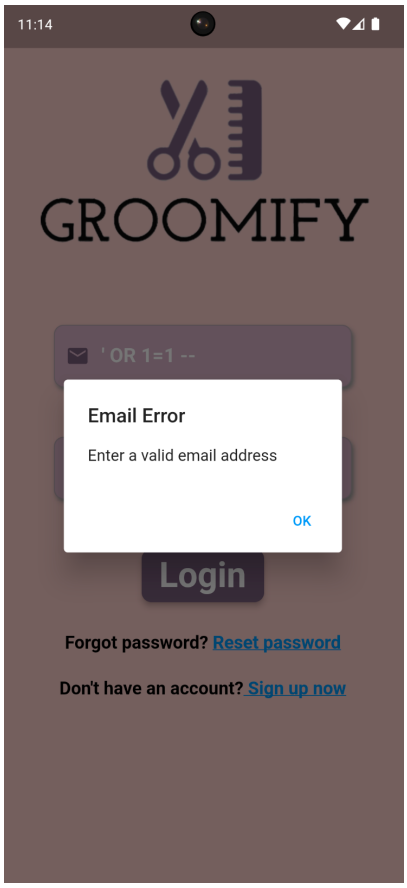


Figure 106: SQL Injection Test

```
E/RecaptchaCallWrapper( 4326): Initial task failed for action RecaptchaAction(action=signInWithPassword)with exception - We have blocked all requests from
this device due to unusual activity. Try again later. [ Access to this account has been temporarily disabled due to many failed login attempts. You can
immediately restore it by resetting your password or you can try again later. ]
```

Figure 107: Brute Force Test

The error message indicates that the Recaptcha system has identified unusual activity, specifically, six consecutive failed login attempts, triggering a temporary block on all requests from the affected device. This precautionary measure aims to protect the account from potential brute force attacks, where an attacker systematically attempts various password combinations to gain unauthorized access. To resolve this issue, the user is advised to reset their password immediately to restore access. Alternatively, they can wait for a specified period before attempting to log in again. This security feature is crucial for safeguarding accounts and preventing unauthorized access, providing a proactive response to potential threats and ensuring the user's account remains secure.

```dart
class ResetPassModel {
  Future<String?> resetPassword(String email, BuildContext context) async {
    final localContext = context;

    if (email.isEmpty) {
      // Display an error message when the email field is empty
      showDialog(
        context: localContext,
        builder: (BuildContext context) {
          return AlertDialog(
            title: const Text('Password Reset'),
            content: const Text('Please enter an email address.'),
            actions: [
              TextButton(
                onPressed: () {
                  Navigator.pop(context); // Close the dialog
                },
                child: const Text('OK'),
              ),  // TextButton
            ],
          );  // AlertDialog
        },
      );
    }

    String? validateEmail(String email) {
      if (!RegExp(r'^[\w-]+(\.[\w-]+)*@[\w-]+(\.[\w-]+)+$').hasMatch(email)) {
        return 'Enter a valid email address';
      }
      return null;
    }
```

```
    final emailValidationMessage = validateEmail(email);

    if (emailValidationMessage != null) {
      // Email is not valid, return the validation message
      return emailValidationMessage;
    }

    // Validate if the entered email is registered as a groomer or user
    final isGroomerEmail = await isEmailRegistered(email, 'groomers');
    final isUserEmail = await isEmailRegistered(email, 'users');

    if (isGroomerEmail || isUserEmail) {
      // Email is registered, proceed with password reset
      try {
        await FirebaseAuth.instance.sendPasswordResetEmail(email: email);

        // Show success dialog
        showDialog(
          context: localContext,
          builder: (BuildContext context) {
            return AlertDialog(
              title: const Text('Password Reset'),
              content: const Text('A Password Reset email has been sent to your inbox. '),
              actions: [
                TextButton(
                  onPressed: () {
                    Navigator.pop(context); // Close the dialog
                    Navigator.pop(context); // Navigate back to the previous page (login page)
                  },
                  child: const Text('OK'),
                ),  // TextButton
              ],
            );  // AlertDialog
          },
        );
        return null; // Return null to indicate success
      } catch (error) {
        return 'Password reset failed'; // Return an error message if password reset fails
      }
    } else {
      // Email is not registered, return an error message
      return 'The entered email is not registered.';
    }
  }

  Future<bool> isEmailRegistered(String email, String collectionName) async {
    try {
      final collection = _firestore.collection(collectionName);
      final querySnapshot = await collection.where('email', isEqualTo: email).get();

      return querySnapshot.docs.isNotEmpty;
    } catch (e) {
      // Handle any errors, such as Firestore query errors
      return false;
    }
  }
}
```

Figure 108: Password Reset Code Snippet

The ResetPassModel class that is provided takes care of the Firebase Authentication-based password reset functionality in a Flutter application. The resetPassword method checks that the email address entered is valid and does not contain any spaces before proceeding. It then calls the isEmailRegistered method to see if the email is registered as a user or as a groomer. The 'groomers' and 'users' Firestore collections are queried by this method to see if the email exists. The code uses FirebaseAuth.instance.sendPasswordResetEmail(email: email) to send a password reset email if the email is registered. Because Firebase Authentication manages email delivery and verification, this operation is secure. The code also has error handling to handle possible malfunctions and give the user the proper feedback. By guaranteeing that only registered emails initiate the password reset functionality and preventing unauthorised access to the reset feature, the use of Firebase Authentication in this context improves security.

No	Test Case	Observation	Evidence	Result
1	Reset using unregistered email	Failed to reset password	Figure 95	Pass
2	Reset Using Authenticated Email	Able to receive reset password link	Figure 96, 97	Pass

Table 12: Security Test (Reset Password)

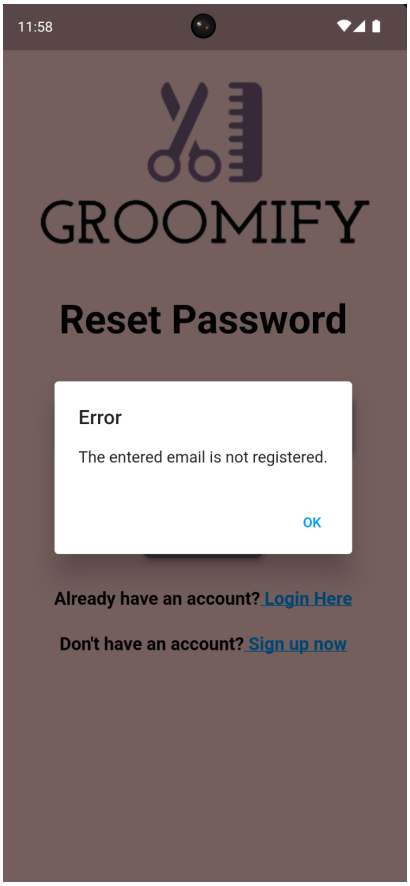


Figure 109: Reset Password Error

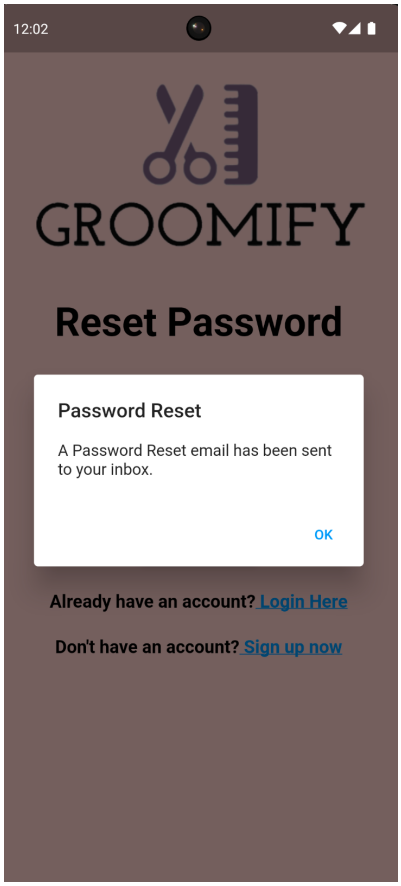


Figure 110: Successful Password Reset

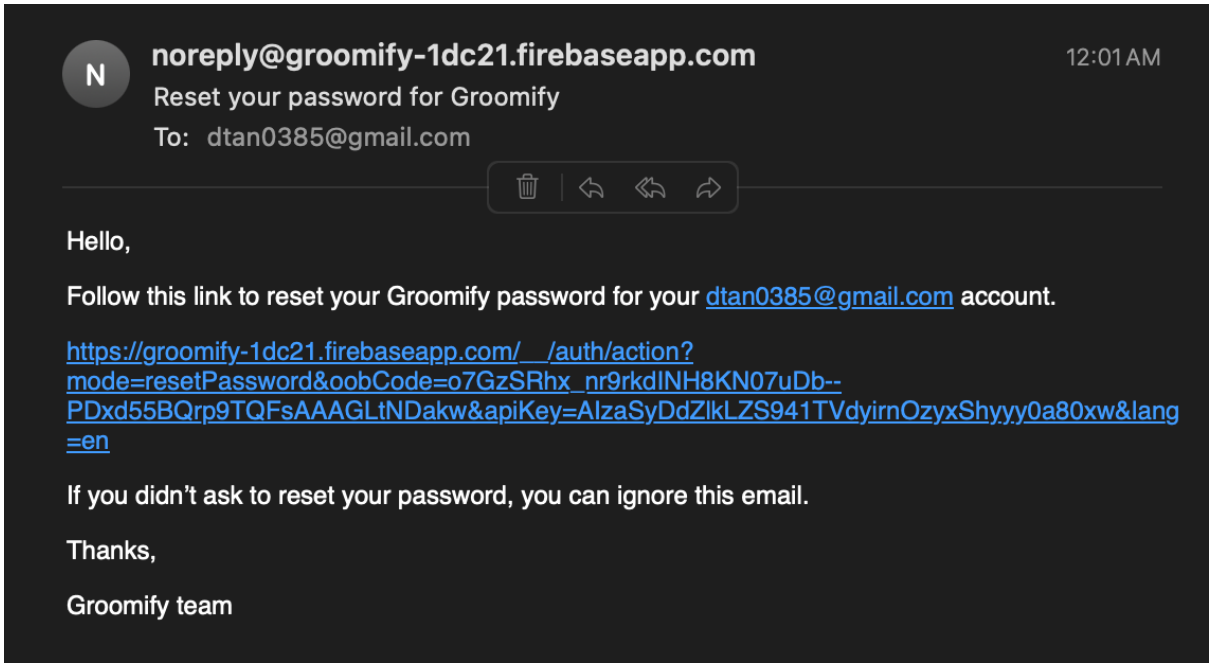


Figure 111: Reset Password Email

Chapter 6: Discussion

6.1 Achievement

A major milestone in the quest to offer a comprehensive solution for pet owners has been reached with the successful realisation of all the set objectives during the long process of developing Groomify, the pet grooming booking app. The principal achievement consists of developing a mobile application that is not only user-friendly but also effectively enables our consumers to schedule online pet grooming appointments. This accomplishment honours the commitment to offering pet owners an easy-to-use and effective platform so they can conveniently and easily manage their pets' grooming routines.

A notable accomplishment of the Groomify project is the effective integration of a strong reminder function. When it comes to their pet's grooming requirements, clients will never be caught off guard thanks to this feature, which has shown to be a useful addition to the app. Reminders are an essential tool that improves the user experience in general and raises customer satisfaction levels. It has decreased the possibility of missed visits and guaranteed that pets receive the care and attention they require, which has ultimately improved their well-being, by notifying pet owners of their scheduled appointments.

Beyond the core goals, Groomify has produced a plethora of extra accomplishments that have catapulted the project into the stratosphere. The booking procedure has been simplified by the use of an intuitive interface and effective back-end administration. The platform also includes features that have improved the user experience even further, like user reviews, secure payment processing, and real-time appointment tracking. Together, these achievements strengthen the dedication to offering pet owners a comprehensive solution for pet grooming services, easing the management of their pet's grooming requirements.

The successful realization of these objectives demonstrates the unwavering dedication to simplifying the lives of pet owners and ensuring that pets receive the care they deserve. Groomify's completion not only represents the culmination of extensive development efforts but also the commitment to continuous improvement and innovation. As the project moves forward, the aim is to further enhance the platform by exploring avenues such as integration with grooming salons and expanding the services to include additional pet care features. The future of Groomify holds the promise of even greater value for both pet owners and their beloved animals.

6.2 Limitations

Throughout its development, Groomify ran into a number of issues that affected both the user experience and the app's capacity to satisfy pet owners' expectations. These limitations include network-related problems that cause sluggish data loads, the absence of a comments section, an online payment mechanism, the inability to edit and cancel appointments, the lack of appointment history viewing, and the inability to communicate directly with groomers. Each of these restrictions has had a different impact on the Groomify app's usability and functioning, calling for attention and possible improvements to fix these issues and offer a more thorough and fulfilling user experience.

The Groomify app's vulnerability to network limitations is one of its ongoing issues. Users may encounter significant delays and interruptions in data load times in areas with erratic or slow network connectivity, especially when attempting to access and use the application to book pet grooming appointments. This limitation may irritate users and make them less likely to interact with the platform. Resolving this network limitation is critical because a lot of pet owners depend on the app to get necessary pet grooming services. Regardless of the user's location or network quality, Groomify can provide a more dependable and seamless user experience by strengthening the app's resistance to network problems and optimising data load times.

One glaring flaw in the current Groomify app is that there isn't a comments section. This deletion prevents users from offering insightful criticism, discussing their experiences, or asking questions answered directly within the app. An essential element of user interaction and engagement that promotes communication and a sense of community within the Groomify

platform is the comments section. Improving the user experience as a whole greatly depends on the user's capacity to interact, ask questions, and share experiences with other users and service providers. Groomify can foster a more dynamic and interactive environment by incorporating a comments section, thereby enabling users to actively participate and communicate within the platform.

The lack of a location API in the Groomify app is another significant drawback that makes it more difficult for users to find the closest groomer in their area. Users who are looking for grooming services right away may not have the best experience possible and may even miss out on opportunities. To increase the app's usefulness, a location-based service must be implemented. The app can help users make better decisions when booking appointments by giving them information about grooming salons in the area. An important potential upgrade would be the inclusion of a location API, which would enhance user experience and add a level of efficiency and convenience to the app's functionality.

Although the Groomify app makes scheduling pet grooming appointments easy, it is missing a crucial feature: the ability to make payments online. Users are currently forced to look for alternate payment methods, which might not be as easy or safe. For the purpose of increasing user convenience and expediting the booking and payment process for pet owners and service providers, it is imperative to incorporate an online payment function. In addition to the obvious advantages of allowing payments within the app, this feature improves the efficiency and security of transactions overall. It not only streamlines the user experience but also gives the Groomify platform an additional degree of dependability and trust.

The inability of Groomify to let users amend or cancel their scheduled appointments is another area in which it fails. Unexpected events frequently call for alterations to appointment schedules or, in certain situations, cancellations. The lack of an easy-to-use method for implementing such changes may cause users to become frustrated and experience inconvenience. Enabling users to change or cancel appointments in a timely manner is essential to meeting their dynamic scheduling demands. By guaranteeing flexibility in scheduling pet grooming appointments, this addition enhances user satisfaction and the overall user experience.

Another essential feature missing from the Groomify app is an appointment history section. Users would be able to view their pet's past appointment records and grooming history in this section. For pet owners who want to guarantee their pet's wellbeing and maintain an exhaustive record of their grooming requirements, this information is essential. The app improves the user experience by supporting responsible pet care and giving access to vital information through the implementation of an appointment history section. By providing users with an extra layer of value, this feature offers the chance to boost their confidence and trust in the Groomify platform.

Finally, there isn't a chat feature on the Groomify app at the moment that would allow users to speak with groomers face-to-face. This limitation prevents users from asking questions, giving detailed instructions, or talking to experts about issues related to grooming. Lack of such a channel for communication may cause misconceptions and misinterpretations, which may lower the calibre of the grooming services. For users and service providers to communicate and understand each other better, the app must have a chat feature. It directly contributes to higher service quality and greater customer satisfaction in addition to improving the user experience overall.

6.3 Future Improvements

While Groomify is a testament to its dedication to making pet owners' lives easier by offering a convenient way to schedule grooming appointments, it's also important to understand that the journey doesn't end here. A number of prospective future enhancements are envisaged in order to further improve the platform and solve the previously mentioned limitations.

Taking care of the problem of network limitations is still very important. Future plans call for the installation of a more effective data handling system that can lessen the negative effects of sluggish network connections. To do this, it will be necessary to improve offline functionality, use local caching to speed up load times, and optimise data transfer protocols. These enhancements will guarantee that users, irrespective of the quality of their network, can effortlessly access and utilise Groomify.

The inclusion of a comments section is being thought about as a way to promote a stronger sense of community and engagement. Within the app, users will be able to directly share their experiences, opinions, and questions. In addition to facilitating user interaction, this feature will offer a useful forum for constructive criticism, allowing services to be continuously improved based on user insights.

An API for location is soon to be integrated. This addition will make it easier for users to find the closest grooming salons, which will streamline the process of choosing the most convenient choice. By providing users with up-to-date information about grooming establishments in their immediate vicinity, users can expedite the process of scheduling an appointment.

The implementation of an online payment feature will further the goal of improving convenience. The goal is to give users a one-stop shop, simplifying the entire booking and payment process while guaranteeing the highest levels of security for financial transactions, by enabling safe and easy transactions within the app.

The app will fulfil users' needs for dynamic scheduling by allowing them to edit or cancel appointments. Users will be able to adjust their grooming schedules as needed with this upcoming feature, which will improve their overall Groomify experience and satisfaction.

A section dedicated to appointment histories will be added in order to give users a thorough overview of their pet's grooming history. A more planned and knowledgeable approach to pet care will be made possible by users' access to records of previous appointments and grooming services.

Finally, the implementation of a chat function aims to enable direct communication between grooming professionals and users. This feature will promote better understanding and satisfaction on both sides of the platform by allowing users to address grooming-related concerns in real-time, clarify questions, and give specific instructions.

Chapter 7: Reflection

Personal Reflection

7.1.1 Strengths

Reflecting on my role in the Groomify project, I am pleased to have had the opportunity to contribute to the development of a pet grooming booking app that addresses the evolving needs of pet owners. My ability to work under time constraints has been a big asset throughout the project. To make sure that project milestones were reached, it was imperative to adjust to constrained timetables and quickly shifting priorities. I've discovered that a critical skill in project management is the capacity to change course and maintain adaptability in the face of time-related obstacles. We were able to handle unforeseen setbacks and changing needs with resilience and effectiveness thanks to our flexibility.

Throughout the project, I have also utilised my strength of effective time management. A strong sense of time management was necessary to oversee numerous project tasks, communicate with team members, and fulfil deadlines. It became clear that the project's progress depended on a well-organized approach to time allocation and task prioritisation. My expertise in this field enabled me to optimise processes and guarantee that the project proceeded as planned.

My determination to see the project through to completion was probably the most important quality I brought to the table. Throughout the development process, a driving force was the resolve to overcome obstacles, learn from failures, and persistently work towards project objectives. This constant dedication to the project's success was a source of motivation for me as well as the rest of the team, encouraging a feeling of enthusiasm and shared dedication. This tenacity is what drove us to accomplish our goals and produce an app that we think will really benefit pet owners.

With the benefit of hindsight, I can say that the Groomify project has sharpened my flexibility, determination, and time management skills. It has reaffirmed how important these skills are for managing projects and solving problems. The project has also brought attention to how crucial it is to keep an open mind towards innovation, constant improvement, and user-centered

adaptation. Since learning and progress are continuous processes, I am eager to apply these insights and strengths to my upcoming endeavours. In addition to producing a successful application, the Groomify project has improved my knowledge and experience, positioning me for future chances and challenges in the rapidly changing app development and technology industries.

7.1.2 Challenges Faced

While the development of Groomify was a rewarding experience, it was not without its fair share of challenges. Overseeing additional academic courses in addition to the project proved to be a significant challenge. Effective time management and prioritisation were necessary to balance the demands of the project with the coursework. It required constant juggling to find the time and energy to devote to both project work and academic obligations. This difficulty brought home how crucial it is to keep a disciplined schedule and ask for help when needed in order to properly manage several responsibilities.

A recurrent obstacle that surfaced throughout the project was the inherent nature of software development. It frequently reminded me of the game of whack-a-mole, where every time one bug was fixed, another would eventually show up. This task illustrated how software development is dynamic and iterative. It underlined the necessity of rigorous testing, ongoing watchfulness, and the capacity for real-time troubleshooting and debugging. Notwithstanding the difficulties this task presented, it served as a useful exercise in adaptability and resilience, highlighting the significance of constant development and close attention to detail.

Testing presented yet another difficulty, as it is an essential part of the development process. It became clear that thorough testing requires time, and hurrying through this stage can result in unfavourable outcomes like undiscovered bugs and disgruntled users. Thorough testing and validation were necessary to guarantee the application's functionality, security, and user experience. This resulted in an extension of the project timeline, but it also made clear how important comprehensive testing is to producing a dependable and user-friendly application.

7.1.3 Weakness & Self Improvement

Reflecting on my role in the Groomify project, I've identified certain weaknesses that I view as opportunities for self-improvement. Writing reports was one of these flaws. Even though I handled the technical aspects of the project well, I realised that I needed to improve my report writing abilities. Effective project management requires the ability to communicate project progress, accomplishments, and challenges through thorough, organised reports. I've improved myself in this area by being proactive in response. I have taken additional courses and asked peers and mentors for advice in order to improve my communication abilities. The goal of this effort is to make sure that future project progress and results are accurately and thoroughly documented.

The project also revealed another area of personal weakness: my sporadic struggles with laziness and procrastination. There were times when I struggled to stay disciplined and motivated throughout the project. Nevertheless, in spite of these difficulties, I'm happy to report that the essential work was eventually finished. This experience has strengthened the idea that work ethic and diligence can triumph over lazy tendencies. I've put in place time management strategies and an organised schedule to address this shortcoming. This strategy has shown to be helpful in sustaining steady productivity and focusing on project tasks, especially when motivation is low. Understanding the value of self-control and persistent work, I've also looked into techniques like establishing daily objectives that are doable and sticking to a set work schedule. By reducing the consequences of indolence and procrastination, these self-improvement techniques hope to increase productivity on both a personal and project level.

7.2 Technical Discussion

One of the main technical aspects of the Groomify project was creating the mobile application with Flutter and Dart. I have a better understanding of these technologies as a result of this experience, which was a worthwhile learning opportunity.

The project's most notable technical accomplishment was the significant increase in Flutter and Dart knowledge and expertise. The Groomify app was built on top of Flutter, which is renowned for its adaptability and capacity to create natively compiled applications for desktop,

web, and mobile platforms from a single codebase. Throughout the project, I gained more knowledge about the Flutter ecosystem and improved my abilities to manage app navigation, create user interfaces that are easy to use, and handle state management. As I got more comfortable with Dart, the programming language for Flutter, I was able to write dependable, clear, and effective code. The Groomify app's success was largely due to the developer's increased proficiency with Flutter and Dart, which made it possible to integrate different features seamlessly and create a responsive, user-friendly application.

Apart from augmenting my proficiency in Flutter and Dart, the project provided a singular occasion to enhance my comprehension of Firebase functionalities, particularly Firebase Authentication (Fireauth) and Cloud Firestore. Ensuring a smooth sign-in and registration process while safeguarding user data was made possible by Firebase Authentication. I could create strong user authentication processes and guarantee data security and integrity if I learned how to use Fireauth. Additionally, Cloud Firestore was essential for managing and storing data like appointment records and user profiles because it was a NoSQL database solution. The project enhanced the app's dependability and speed by providing a deeper investigation of Firestore's real-time data synchronisation and seamless data retrieval capabilities. The project enhanced the app's dependability and speed by providing a deeper investigation of Firestore's real-time data synchronisation and seamless data retrieval capabilities.

All things considered, the Groomify project was a remarkable technical learning opportunity that helped me gain a deeper understanding of Flutter and Dart as well as Firebase features like Cloud Firestore and Firebase Authentication. In order to create a responsive, safe, and user-friendly platform that meets the needs of pet owners, these technological advancements proved essential in the development of the Groomify app. The creation of creative and effective applications in the constantly changing field of mobile app development will surely be made possible by this strengthened technical foundation, which will prove to be a valuable asset in upcoming projects.

Chapter 8: Conclusion

Conclusion

While creating Groomify, we set out to develop an app for pet grooming that would not only satisfy the needs of pet owners but also improve their experience in booking appointments for their cherished animals. As this project comes to an end, we can say with great satisfaction that Groomify has accomplished the goals that were set forth. With the app's user-friendly interface and effective backend systems, users can now easily schedule appointments for pet grooming online. Moreover, Groomify has demonstrated its flexibility by overcoming network limitations and changing to meet user demands.

The project's obstacles, which have included time management, complex software development, and extensive testing, have offered priceless teaching moments. These difficulties presented us with chances to hone our abilities and methods as well as obstacles to overcome. They have emphasised how crucial flexibility, self-control, and meticulousness are to project management and app development.

Groomify is expected to revolutionise the grooming industry for both pet owners and stylists in the future. Through the resolution of the noted shortcomings and the introduction of prospective enhancements, we see Groomify evolving into a vital instrument for pet owners, providing not only easy appointment scheduling but also a thriving community for feedback and experience exchange. Groomify is positioned to produce long-term advantages for pet owners, grooming facilities, and the larger pet care ecosystem now that it is more widely available in the market.

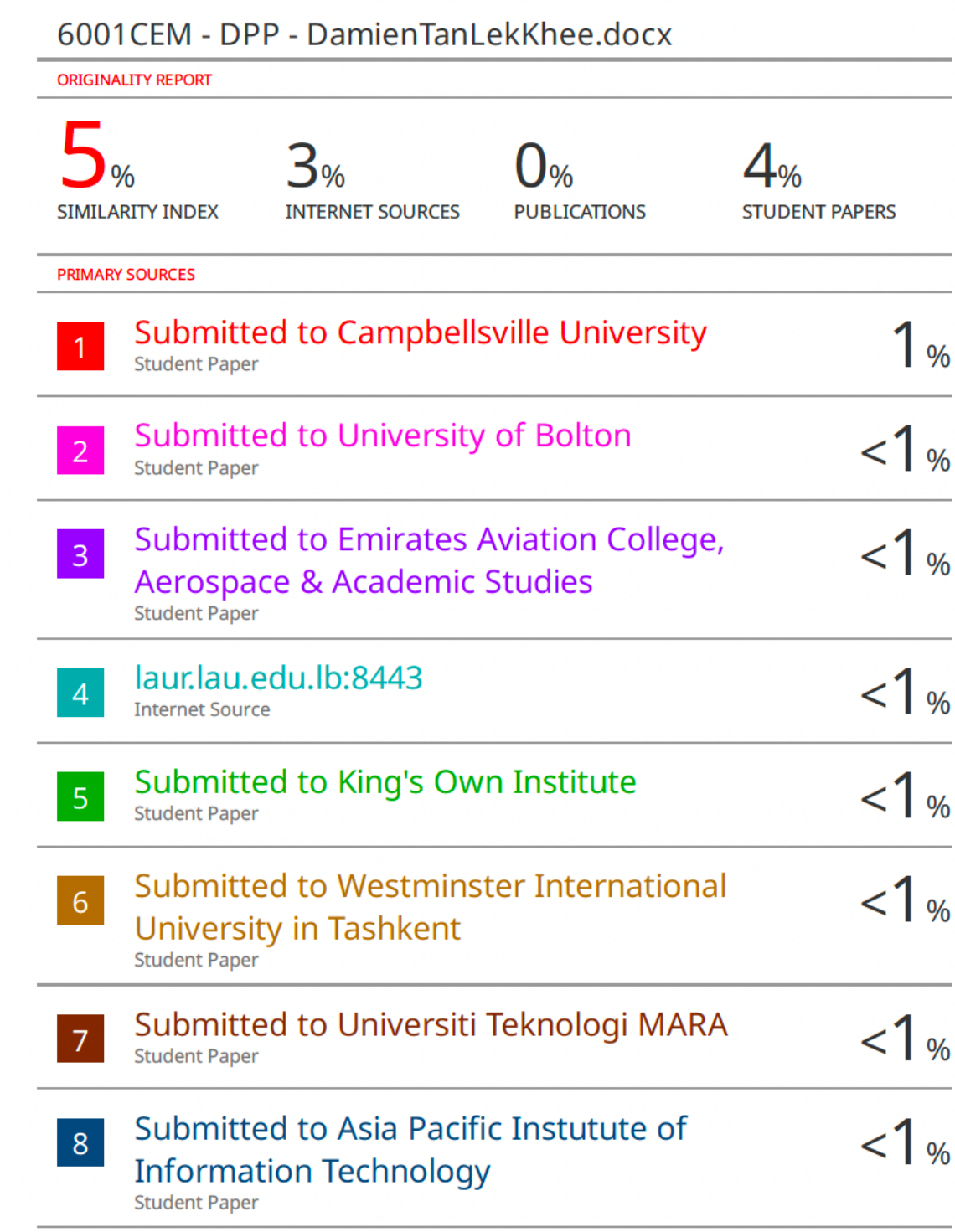
To sum up, the Groomify project has been an adventure in learning and success. It has improved our technical knowledge, abilities, and personal growth in addition to delivering a successful application. We are enthusiastic about Groomify's future and are looking forward to developing and improving the app even more to meet the changing needs of pet owners. Groomify is more than just an app; it's a guarantee of quality, convenience, and community in the pet grooming industry.

References

1. Adobe XD Review: Expanding Your Design Toolbox | Toptal® (n.d.) available from <https://www.toptal.com/designers/adobe/adobe-xd-review> [21 June 2023]
2. Alshamrani, A. and Bahattab, A. (n.d.) A Comparison Between Three SDLC Models Waterfall Model, Spiral Model, and Incremental/Iterative Model. [online] available from <www.IJCSI.org> [11 June 2023]
3. Android Development with Kotlin - Marcin Moskala, Igor Wojda - Google Books (n.d.) available from <https://books.google.com.my/books?hl=en&lr=&id=PJZGDwAAQBAJ&oi=fnd&pg=PP1&dq=kotlin&ots=3KkhfzPYyD&sig=3KiKFsX_rRfGH9fzfTB0bUb-H24&redir_esc=y#v=onepage&q=kotlin&f=false> [15 June 2023]
4. Benefits of Pet Grooming Software in the Pet Industry | Meetbrandwide (n.d.) available from <https://meetbrandwide.com/blog/2022/06/22/benefits-of-pet-grooming-software-in-the-pet-industry/> [9 May 2023]
5. Designing and Prototyping Interfaces with Figma: Learn Essential UX/UI ... - Fabio Staiano - Google Books (n.d.) available from <https://books.google.com.my/books?hl=en&lr=&id=GOBeEAAAQBAJ&oi=fnd&pg=PP1&dq=comparison+of+figma,+xd+and+sketch&ots=ekfwnM8O5M&sig=IGOn9ydUAFVYkIT72AOkenO7UaI&redir_esc=y#v=onepage&q&f=false> [21 June 2023]
6. Flutter for Beginners: An Introductory Guide to Building Cross-Platform ... - Alessandro Biessek - Google Books (n.d.) available from <https://books.google.com.my/books?hl=en&lr=&id=pF6vDwAAQBAJ&oi=fnd&pg=PP1&dq=dart+flutter&ots=dYOVVE0v9s&sig=_FVlEO07Rlqt7-mH_JJXqixwgHc&redir_esc=y#v=onepage&q=dart%20flutter&f=false> [15 June 2023]
7. Fox, L. (2022) Booking.Com on Mobile App Usage, Consumer Trends | PhocusWire [online] available from <https://www.phocuswire.com/booking-CMO-Arjan-Dijk-mobile> [5 May 2023]
8. Kotlin in Action - Dmitry Jemerov, Svetlana Isakova - Google Books (n.d.) available from <https://books.google.com.my/books?hl=en&lr=&id=OzkzEAAAQBAJ&oi=fnd&pg=PT16&dq=advantages+of+kotlin&ots=F10DqvmkMk&sig=T003i2RSK3C7UYSeyASEJbrwArY&redir_esc=y#v=onepage&q=advantages%20of%20kotlin&f=false> [15 June 2023]

9. MongoDB in Action: Covers MongoDB Version 3.0 - Kyle Banker, Douglas Garrett, Peter Bakkum, Shaun Verch - Google Books (n.d.) available from <https://books.google.com.my/books?hl=en&lr=&id=kzkzEAAAQBAJ&oi=fnd&pg=PT21&dq=mongodb&ots=8U8ZuU3166&sig=_-p2wY2w82-CV2dq0RY61Hdl6rQ&redir_esc=y#v=onepage&q=mongodb&f=false> [15 June 2023]
10. MySQL - Paul DuBois - Google Books (n.d.) available from <https://books.google.com.my/books?hl=en&lr=&id=cCiA8HsQhGUC&oi=fnd&pg=PT36&dq=mysql+database&ots=Tv225T2-Wf&sig=muM2VQ83Iyw_ln3lZOBlWw7phMU&redir_esc=y#v=onepage&q=mysql%20database&f=false> [15 June 2023]
11. Programming Android - Zigurd Mednieks - Google Books (n.d.) available from <https://books.google.com.my/books?hl=en&lr=&id=QP7VvnhDOOsC&oi=fnd&pg=PR3&dq=java+for+android+development&ots=Z0x1Q_RlqH&sig=nr1mYmX4l-_7QemBTQclj3aT3-M&redir_esc=y#v=onepage&q=java%20for%20android%20development&f=false> [15 June 2023]
12. Sci-Hub | Beginning App Development with Flutter | 10.1007/978-1-4842-5181-2 (n.d.) available from <https://sci-hub.se/https://link.springer.com/book/10.1007/978-1-4842-5181-2> [15 June 2023]
13. The Benefits of Using Sketch Designs in Software Development - Lightflows (n.d.) available from <https://www.lightflows.co.uk/blog/the-benefits-of-using-sketch-designs-in-software-development/> [21 June 2023]
14. The Definitive Guide to Firebase (2017) [online] available from <https://doi.org/10.1007/978-1-4842-2943-9> [15 June 2023]
15. What Is Adobe XD and What Is It Used For? (n.d.) available from <https://www.adobe.com/products/xd/learn/get-started/what-is-adobe-xd-used-for.html> [21 June 2023]
16. What Is Sketch and What Can You Do With It? (n.d.) available from <https://www.makeuseof.com/what-is-sketch/> [21 June 2023]
17. Wukkadada, B., Nambiar, R., Nair, A., Professor, A., and Somaiya, K.J. (2015) 'Mobile Operating System: Analysis and Comparison of Android and IOS'. IJCAT-International Journal of Computing and Technology [online] 2 (7). available from <www.IJCAT.org> [21 June 2023]

Appendix A



6001CEM - DPP - DamienTanLekKhee.docx

ORIGINALITY REPORT

5%	3%	0%	4%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	Submitted to Campbellsville University Student Paper	1%
2	Submitted to University of Bolton Student Paper	<1%
3	Submitted to Emirates Aviation College, Aerospace & Academic Studies Student Paper	<1%
4	laur.lau.edu.lb:8443 Internet Source	<1%
5	Submitted to King's Own Institute Student Paper	<1%
6	Submitted to Westminster International University in Tashkent Student Paper	<1%
7	Submitted to Universiti Teknologi MARA Student Paper	<1%
8	Submitted to Asia Pacific Instutute of Information Technology Student Paper	<1%

A.1

9	Submitted to University of Birmingham Student Paper	<1 %
10	Submitted to Bradford College, West Yorkshire Student Paper	<1 %
11	Submitted to National Institute of Business Management Sri Lanka Student Paper	<1 %
12	Submitted to Gloucestershire College Student Paper	<1 %
13	citeseerx.ist.psu.edu Internet Source	<1 %
14	www.mobileappdaily.com Internet Source	<1 %
15	Submitted to Purdue University Student Paper	<1 %
16	books.google.ro Internet Source	<1 %
17	norr.numl.edu.pk Internet Source	<1 %
18	www.browserstack.com Internet Source	<1 %
19	Submitted to Open University of Mauritius Student Paper	<1 %

A.2

20	dspace.lboro.ac.uk Internet Source	<1 %
21	Submitted to Middlesex University Student Paper	<1 %
22	Submitted to King's College Student Paper	<1 %
23	Submitted to Toronto Business College Student Paper	<1 %
24	books.google.com Internet Source	<1 %
25	books.google.com.my Internet Source	<1 %
26	etda.libraries.psu.edu Internet Source	<1 %
27	sci-hub.se Internet Source	<1 %
28	Submitted to Bahrain Polytechnic Student Paper	<1 %
29	Submitted to University of Maryland, Global Campus Student Paper	<1 %
30	www.toptal.com Internet Source	<1 %
31	Submitted to Liberty University	

A.3

32	Submitted to Universiti Tunku Abdul Rahman Student Paper	<1 %
33	Submitted to University of Glamorgan Student Paper	<1 %
34	Submitted to University of Hertfordshire Student Paper	<1 %
35	Submitted to Wilmington University Student Paper	<1 %
36	repository.uph.edu Internet Source	<1 %
37	Submitted to University of Liverpool Student Paper	<1 %
38	www.coursehero.com Internet Source	<1 %
39	www.phocuswire.com Internet Source	<1 %
40	jultika.oulu.fi Internet Source	<1 %
41	www.softwaretestingnet.com Internet Source	<1 %
42	Submitted to Ateneo de Naga University Student Paper	<1 %

A.4

43	Submitted to University of Wales Institute, Cardiff Student Paper	<1 %
44	books.google.com.au Internet Source	<1 %
45	kipdf.com Internet Source	<1 %
46	scholar.sun.ac.za Internet Source	<1 %
47	umpir.ump.edu.my Internet Source	<1 %
48	www.appypie.com Internet Source	<1 %
49	ir.juit.ac.in:8080 Internet Source	<1 %
50	link.springer.com Internet Source	<1 %
51	m.moam.info Internet Source	<1 %

A.5

Appendix B

GitHub Link: https://github.com/DamienTan01/Groomify