FIND MY PET – CROSS PLATFORM MOBILE APPLICATION

Final Report

Supervisor:	Paul Barry
Author:	Martin Walsh
ID:	C00170339

Contents

Table of Figures

Abstract

The purpose of this project is to create a cross platform mobile application (iOS & Android) where the user can post details of lost or found animals in their area. This mobile application will be written using Flutter Technology, an open-source UI software development kit created by Google. The application will present the user with the option to post or view lost and found animals in a selected area. There will be a range of features, such as push notifications if an animal is reported lost in a user's area, searching for posts based on location and communicating with other users via comments.

1. Introduction

This final report will document the *FindMyPet* mobile application in terms of what it is and each screen that can be seen when using the application. Additionally, it will include a high-level overview of the database structure. If more information is sought, the technical manual goes in depth into what is happening in each screen. It also details what the database is comprised of and how each section is structured.

The next section will cover the testing carried out in order to ensure each of the features are working as expected across multiple devices. From there the learning outcomes of working on a project this size, both technically and personally, will be explored.

The report will conclude with a project review. It will discuss what was and was not achieved, along with problems encountered and what could have been done differently. It will also explore what potential work can be done in the future.

2. Project Description

FindMyPet is a cross-platform mobile application built using Googles Flutter technology. It also utilises the Dart programming language to help build expressive and flexible UI that gives full native performance on either mobile platform. All development was carried out in Visual Studio Code, a source code editor developed by Microsoft. Visual Studio Code was the editor of choice as it is lightweight, easy to navigate and provided many extensions from the start to help with Flutter development. For example, syntax highlighting for Dart, code prediction and formatting.

The backend of this project was built using a multitude of Firebase tools. Firebase being a Backend-as-a-Service (Baas), used for mobile and web app development. Firebase was connected directly with the mobile application by registering the applications package name and the Debug signing certificate SHA-1. This allowed for more security as the application did not need any API keys or other potentially exploitable items to communicate with the Firebase tools.

The Firebase authentication service was used to create and authenticate credentials for users. Cloud Firestore was used as the projects database of choice. It is a NoSQL database that allows data to be stored and synced with mobile devices at scale. Arguably the projects goal becomes more effective as the userbase grows, making a scalable but low-cost database an

important factor. Firebase Storage tool is also used within this project in order to store media data, namely images. Firebase Functions were also used in order to automatically run backend code in response to predefined triggers. For the purposes of this project the functions, when triggered, send push notifications related to new posts in the user's area or when someone comments on their post.

The technical manual further details the purposes of each screen and how they function.

2.1 Application Screens

2.1.1 Authentication

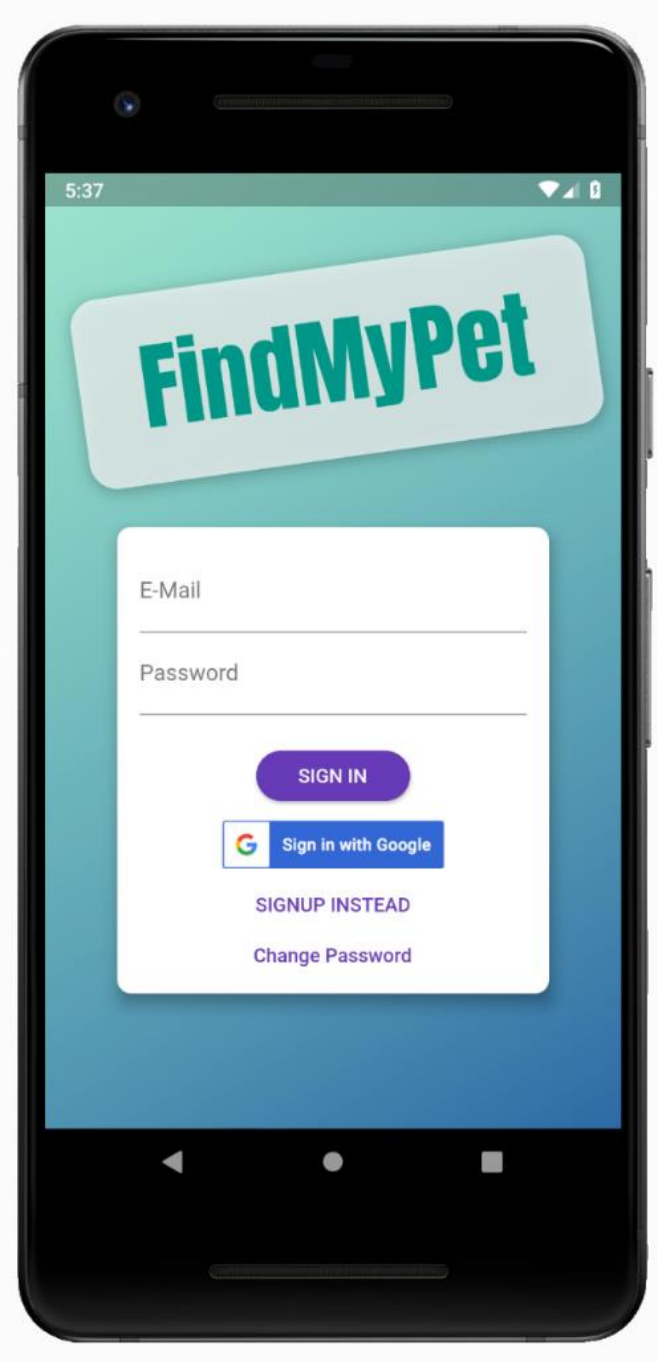
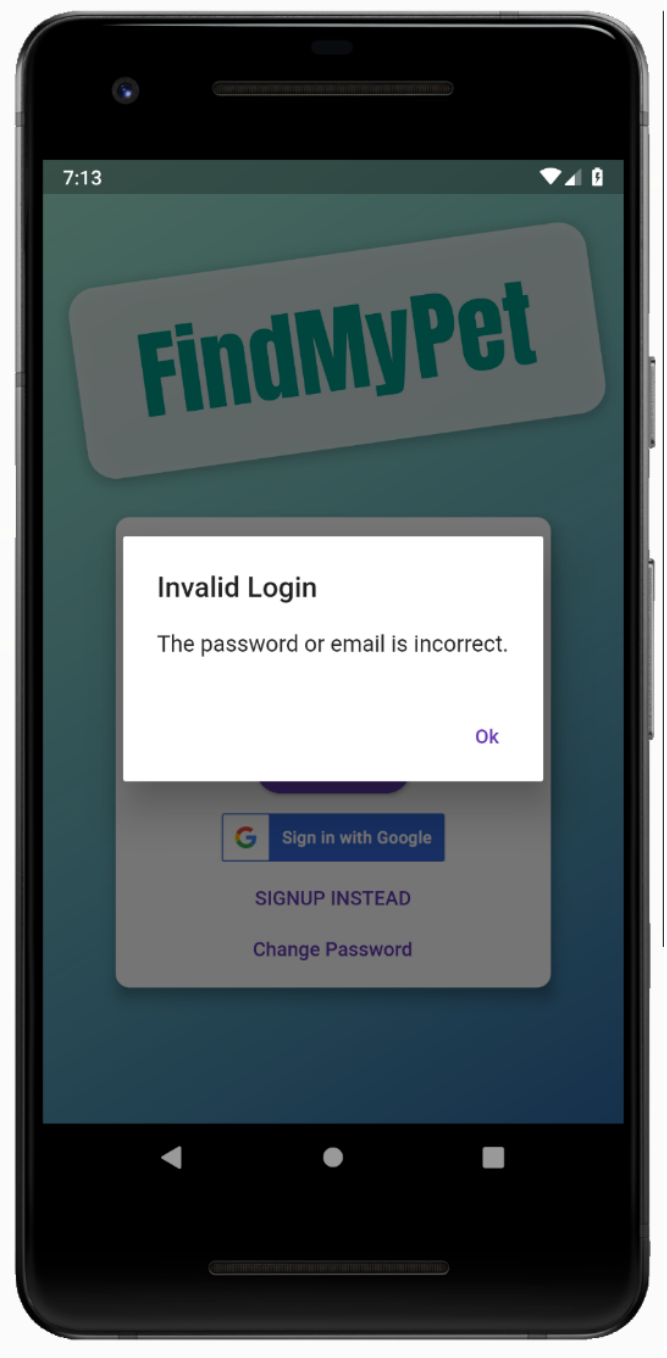


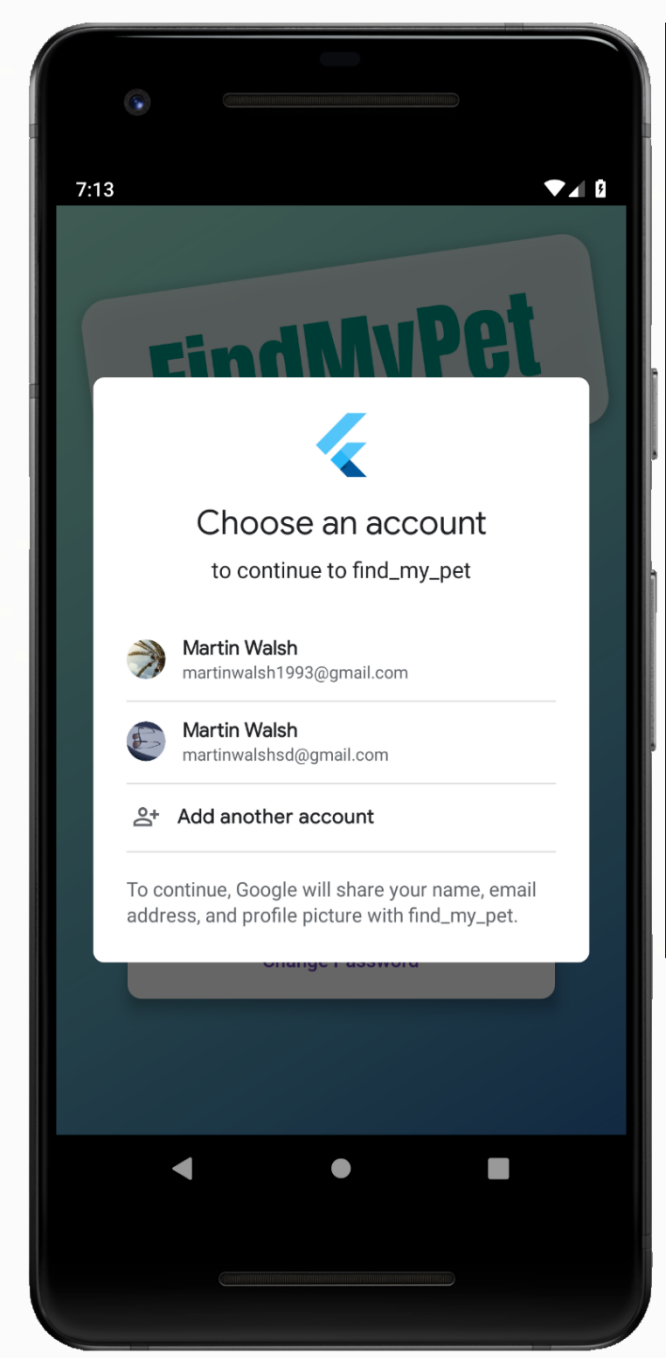
Figure 1: Login Screen

The user is presented with an authentication screen upon first opening the application. The user can login via the email and password combination. When this is done the application connects with Firebase authentication to check whether their credentials are correct.



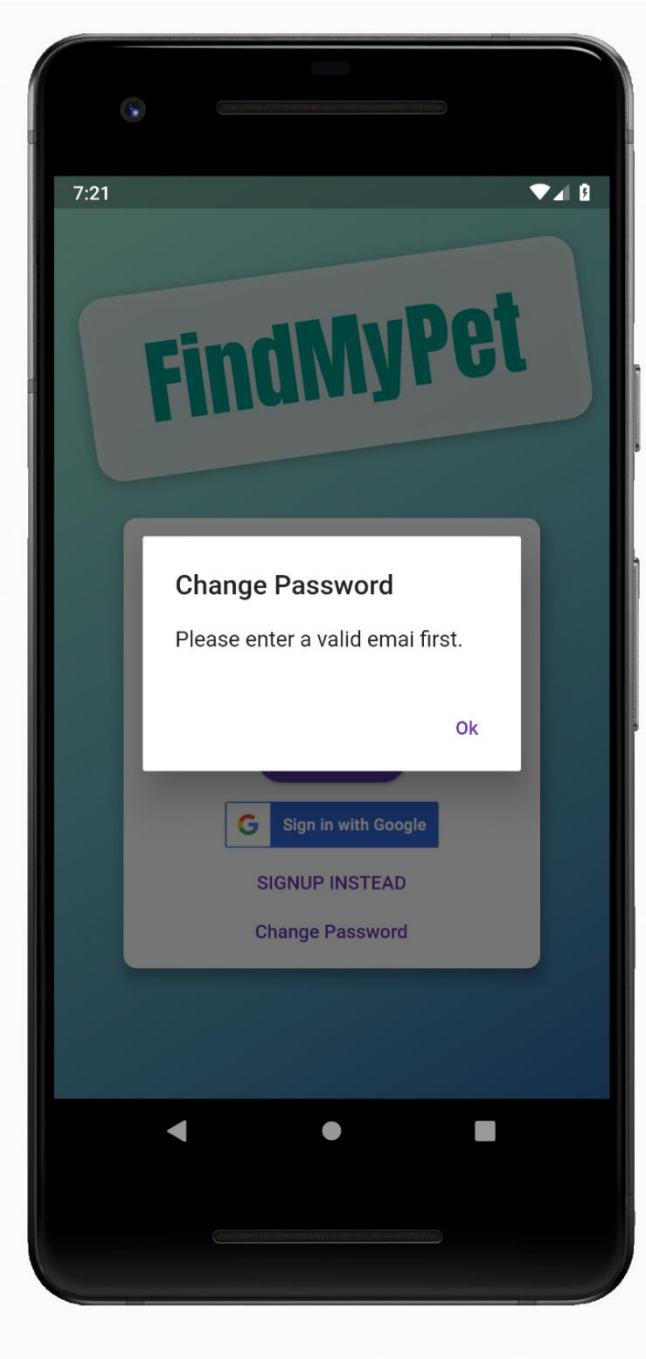
This screen will be shown if the authentication fails. The user will be shown an alert box, stating that the password or email is incorrect. The error message being general was a conscious choice as to not give away too much information to a potentially malicious user.

Figure 2: Invalid Login



The screen shows if the user chooses to login with Google. If the user chooses this option, the application must authenticate with Firebase. However, the backend is different as a Google sign-in package must be imported and different functions carried out.

Figure 3: Login with Google



Screen shown if the user chooses to change the password without entering their email first. To keep the UI simple, the user simply must enter their email and tap the change password button.

Figure 4: Change Password Error

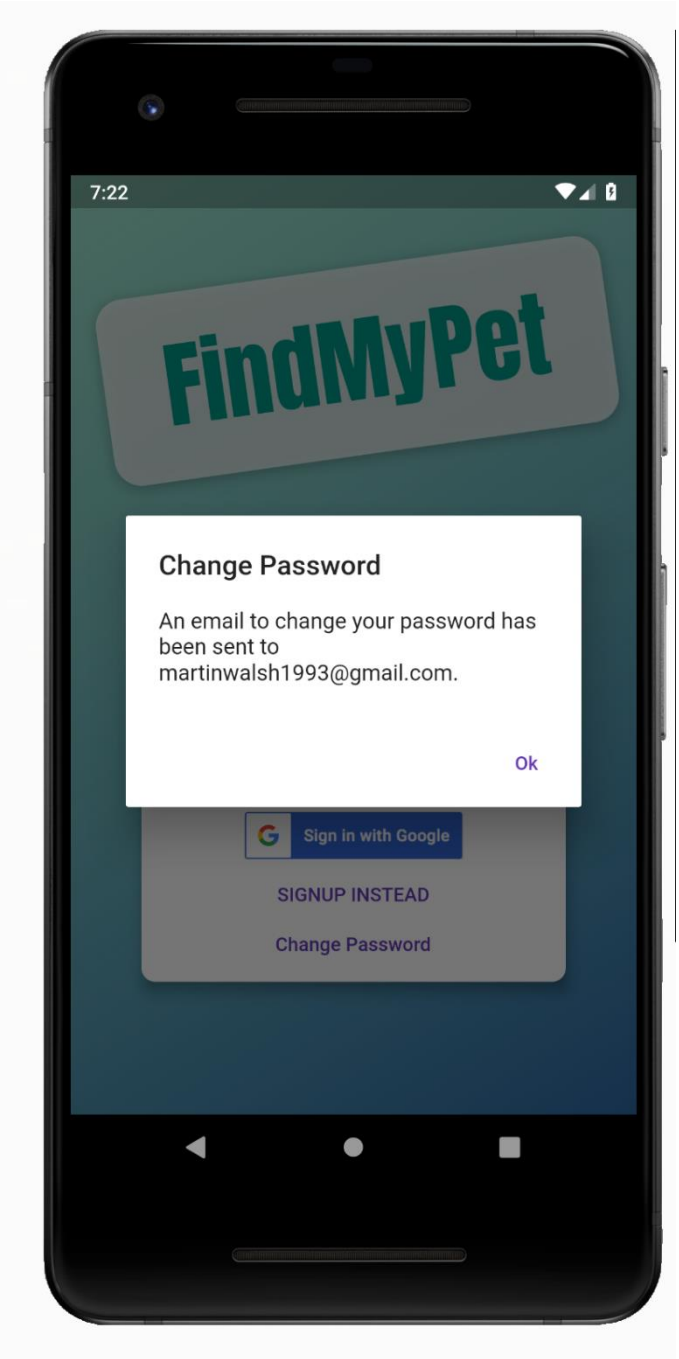


Figure 5: Change Password Message

Screen shown to let the user know an email to reset their password has been sent. They can then follow the link to enter and save their new password, see Figures 6 and 7.

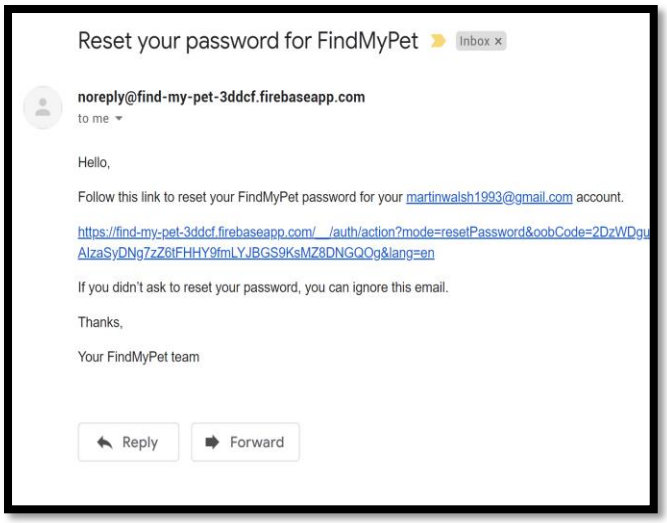


Figure 6: Change Password Email

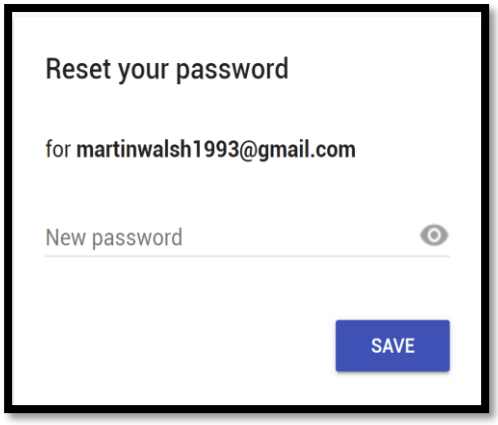
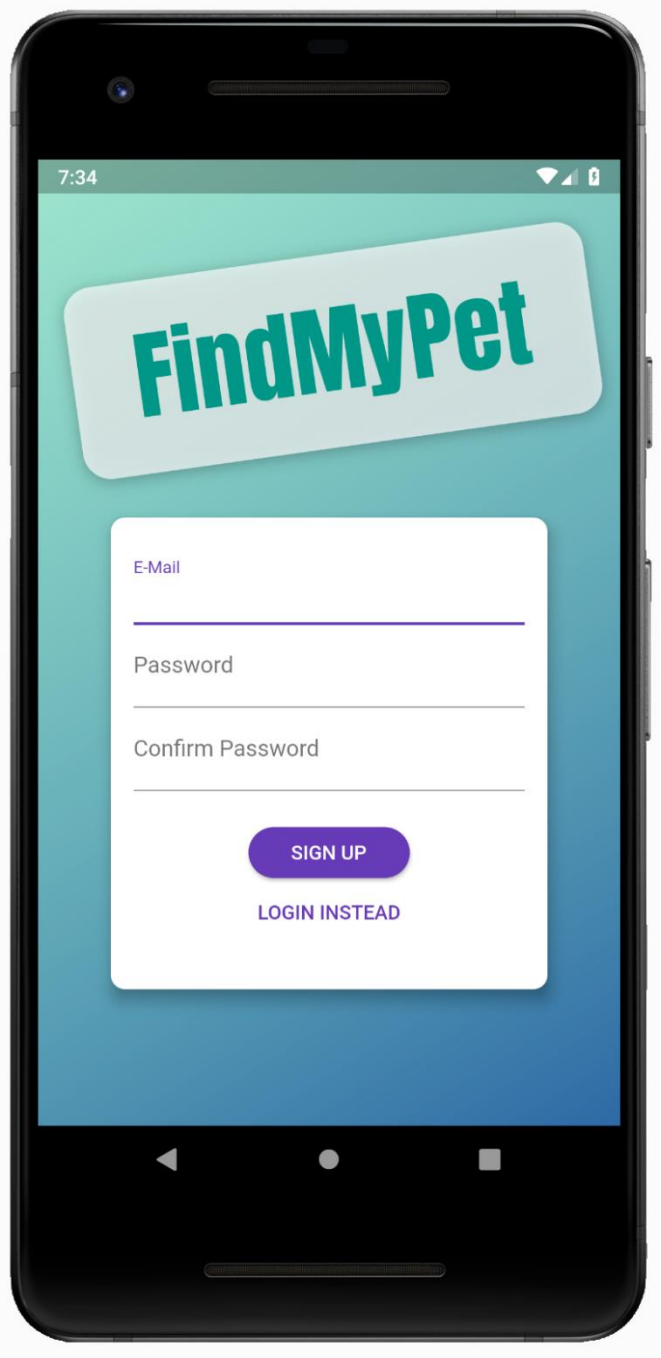


Figure 7: Change Password Screen



Screen show if user opts to sign up. Consists of entering an email, password and confirmation password. There is error checking in place for things such as insufficient passwords and against trying to use an already authenticated email address.

Figure 8: Sign Up Screen

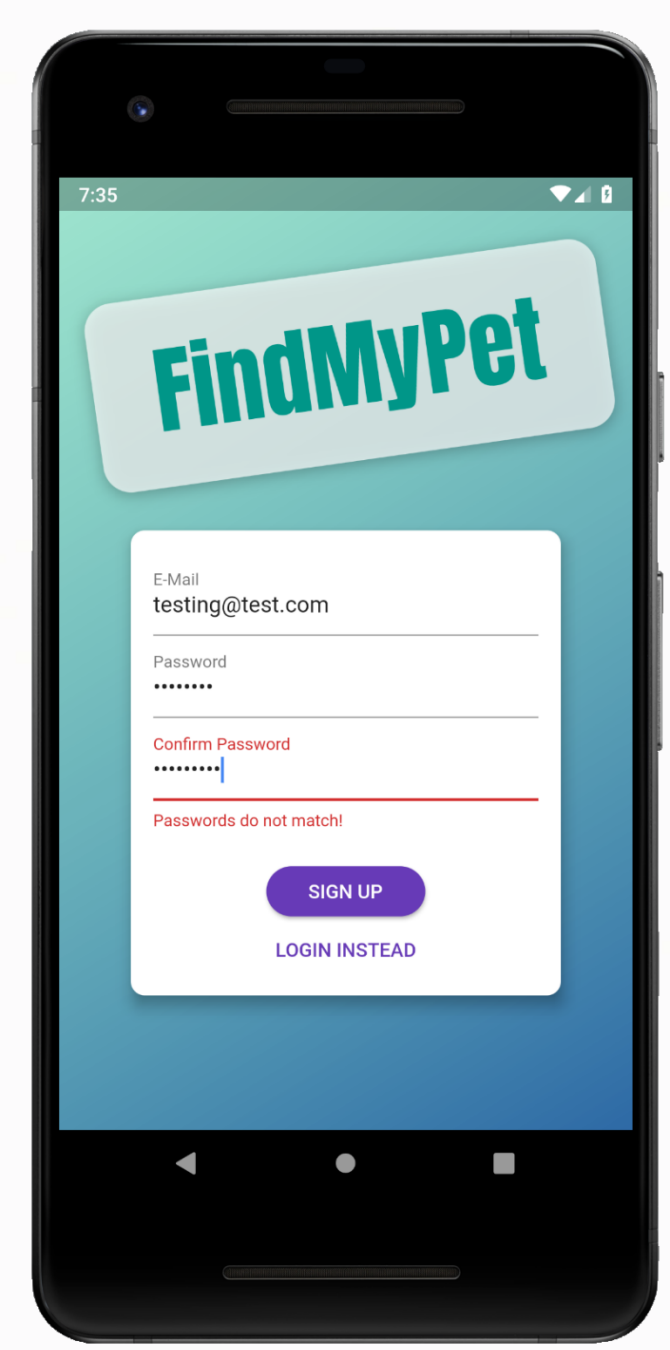
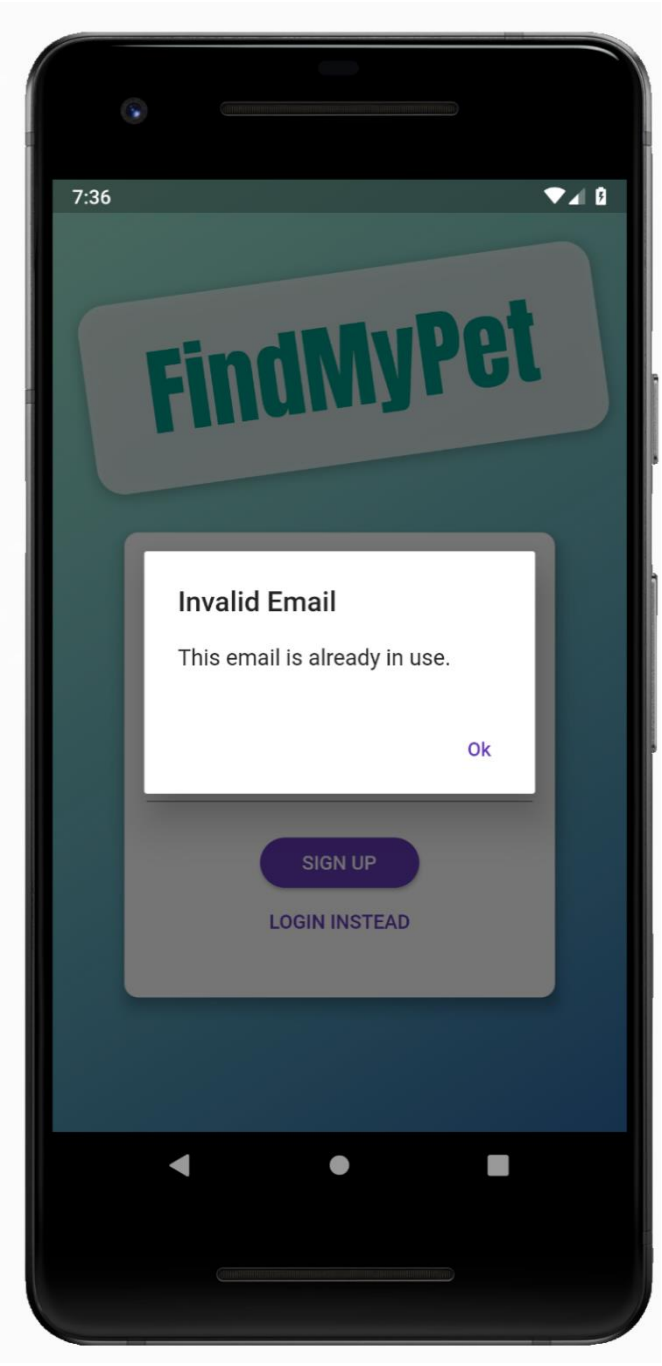


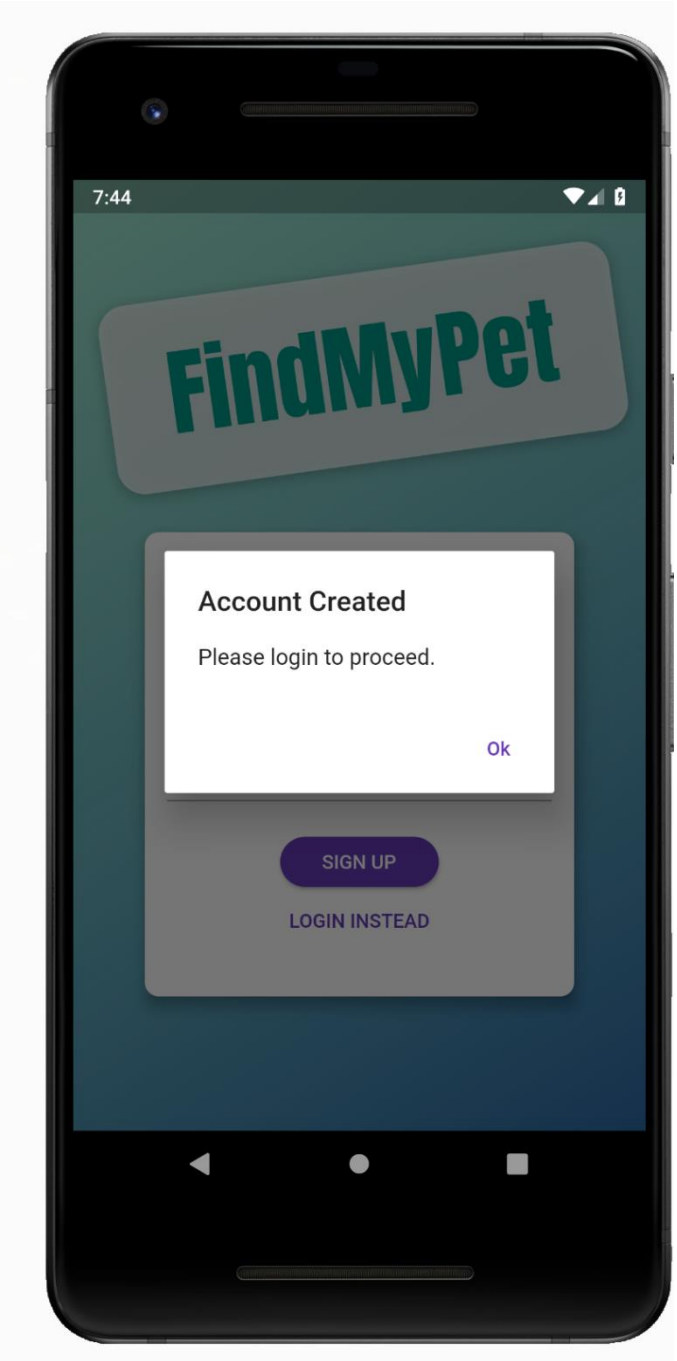
Figure 9: Password Error

This screen demonstrates what occurs when the user's password confirmation is incorrect.



Likewise, this screen demonstrates what occurs when the email being entered is already in use.

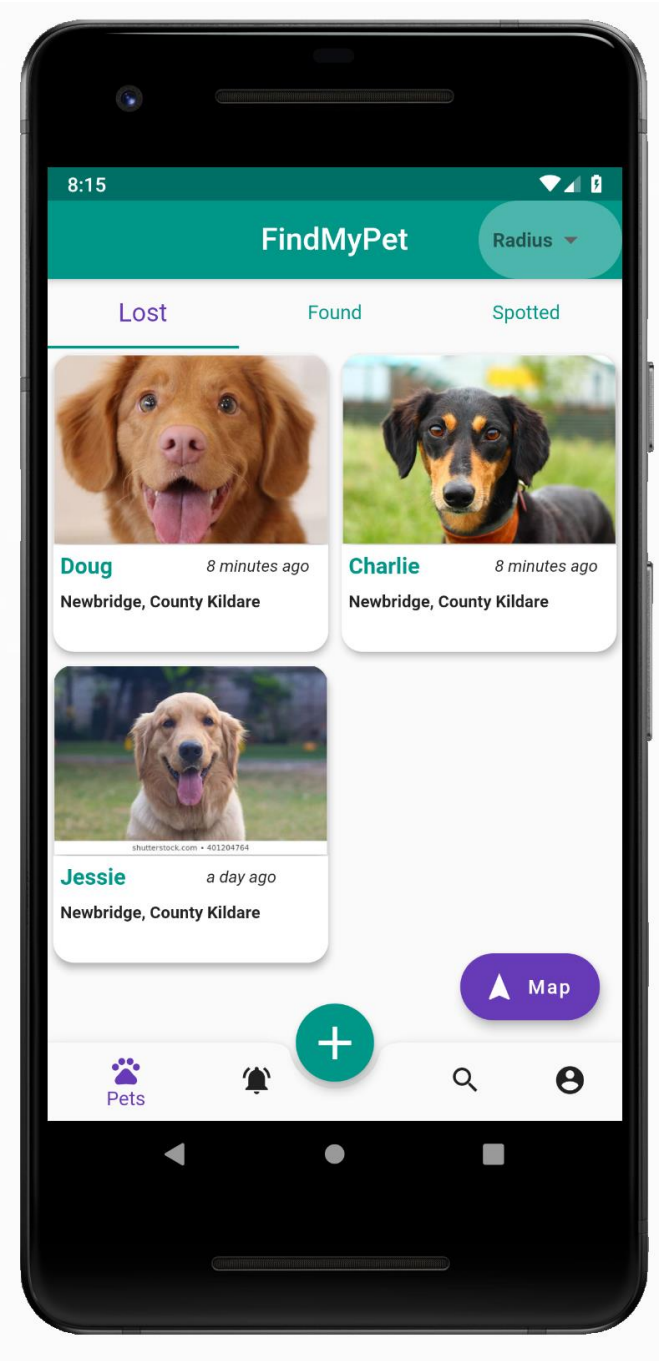
Figure 10: Invalid Sign Up Email



The user will encounter this screen when a they successfully create an account. The application connects with Firebase and creates credentials for that user. There is then an entry added into the Firestore database in order to store information about that user.

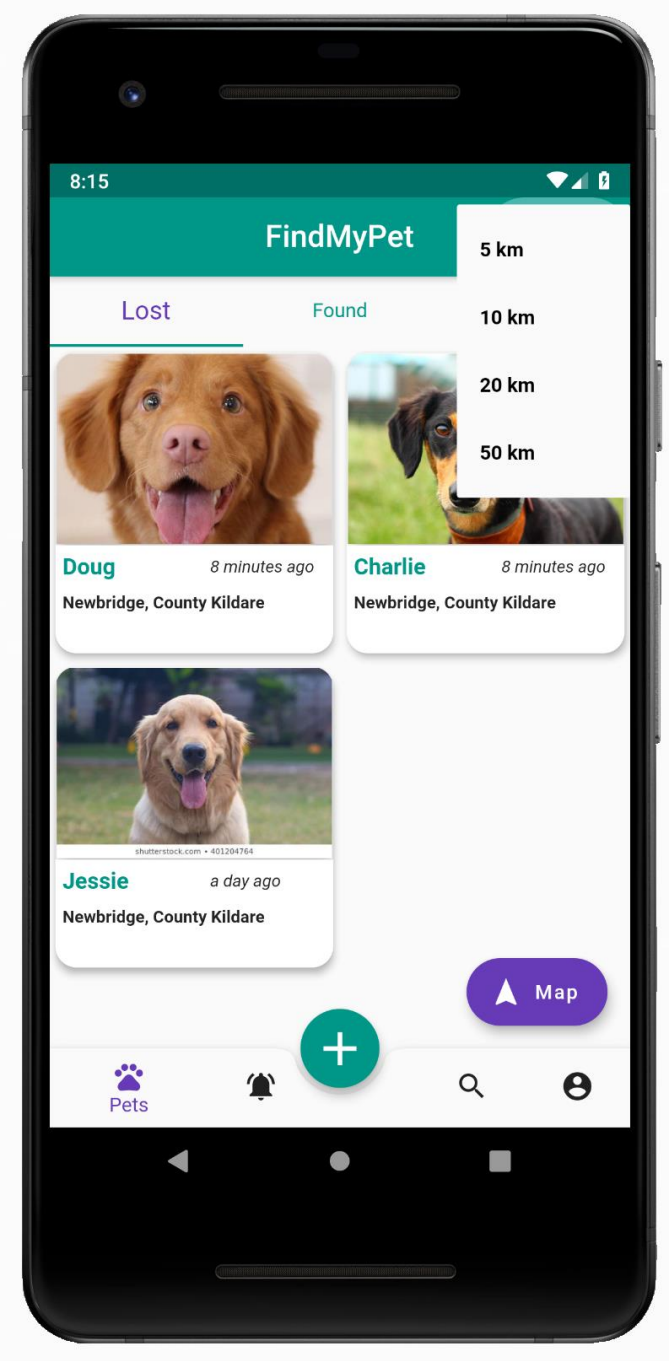
Figure 11: Account Created Pop Up

2.1.2 Posts Overview



The user is led to this screen once they are authenticated. It displays an overview of the posts. The screen itself has three tabs along the top, each one for a different type of posts. “Lost” is for owners to create a post if their pet has gone missing. “Found” is for individuals who have found a lost pet and are keeping it until they can find the owner. While “Spotted” is for individuals who have seen a pet without an owner but are unable to take it in. The users availing of the “Spotted” feature simply take a picture and include a description of where the animal was last seen.

Figure 12: Posts Overview Screen



This screen shows the radius drop down options. Once the user selects an option from the drop-down menu, the pages state will be reinitialised and posts from within that radius will be shown.

Figure 13: Radius Drop Down Menu

2.1.3 Individual Post

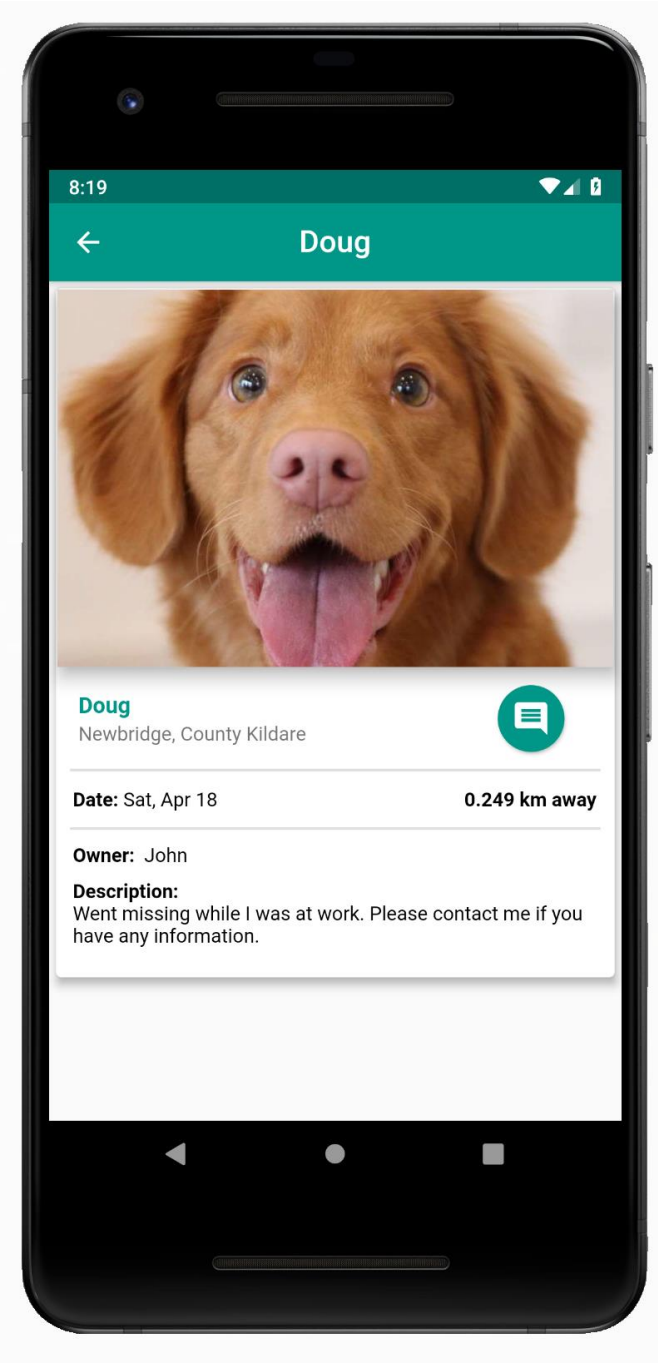
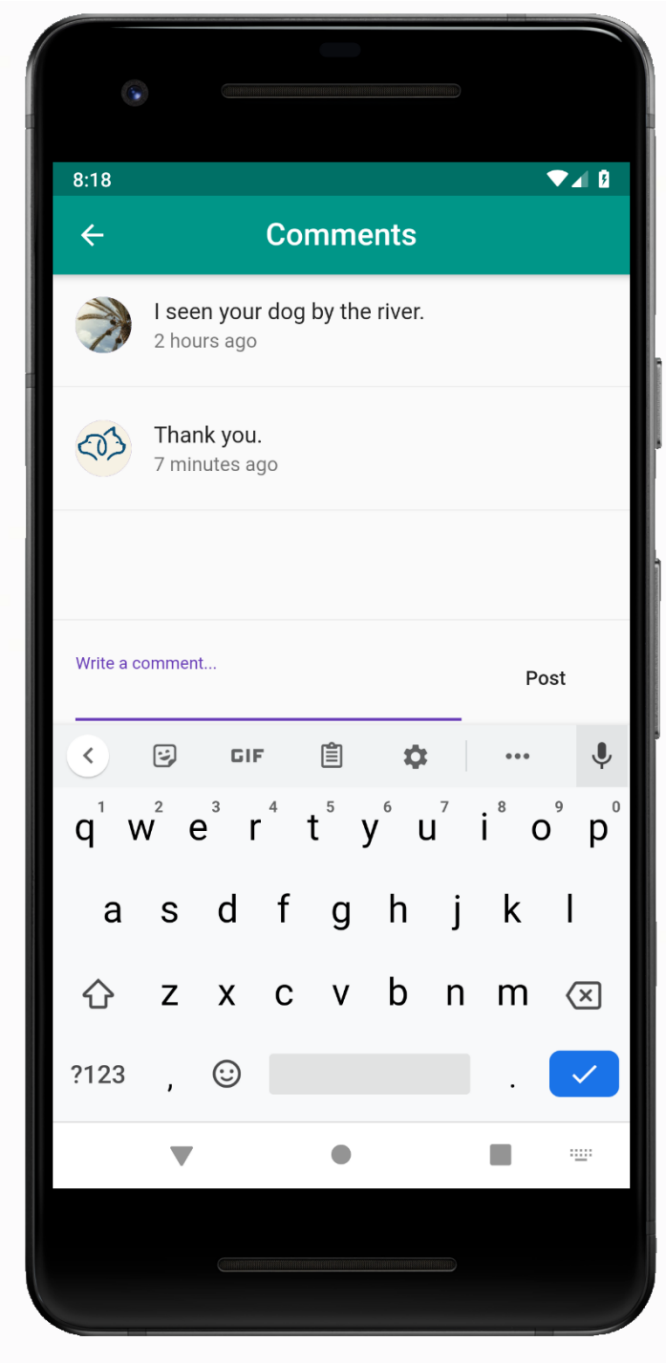


Figure 14: Individual Post Screen

Screen shown when a user taps on a specific post. This page is built in sections, with the image being sourced from Firebase storage. To prevent longer loading times the image is built using a cached image package. This allows for the image to be shown in low quality while waiting for the full image to be ready. The next section is the pet's name, location and a floating action button that if pressed will navigate to the comments page for this post. Followed by the date, distance, owner name and description.

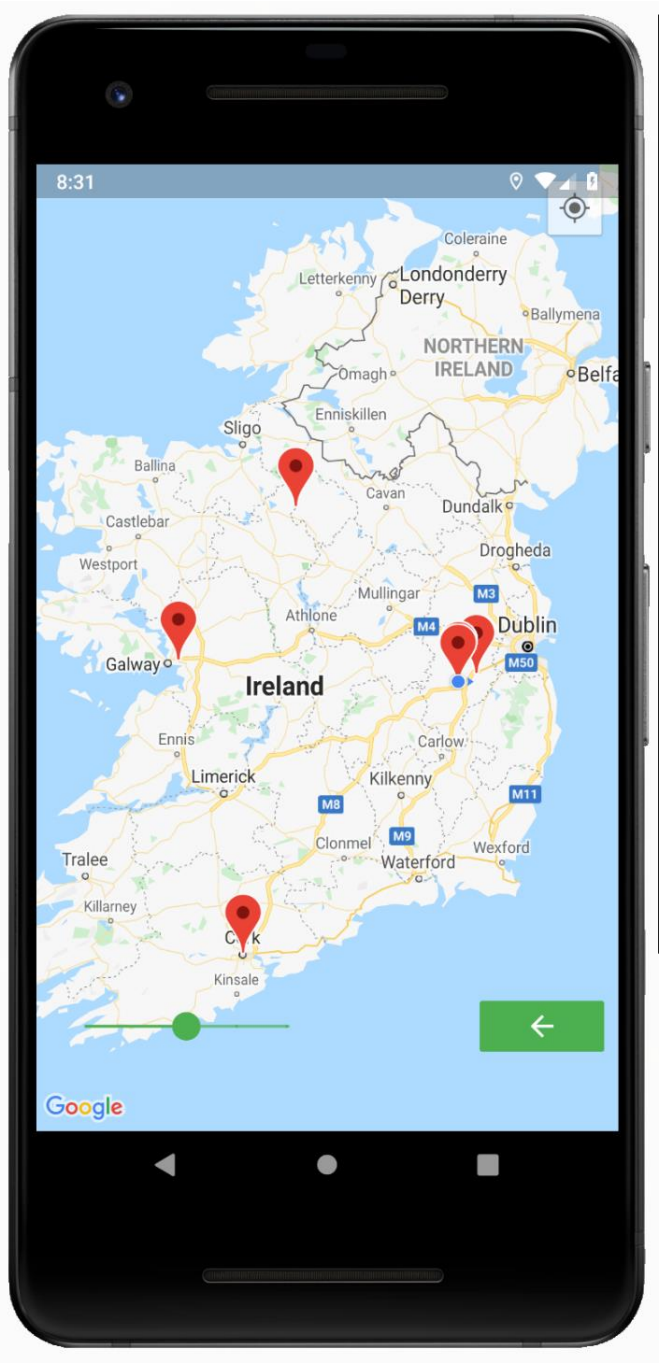
2.1.4 Comments



The screen shown when a user views the comments on a post. The comments are shown in real time.

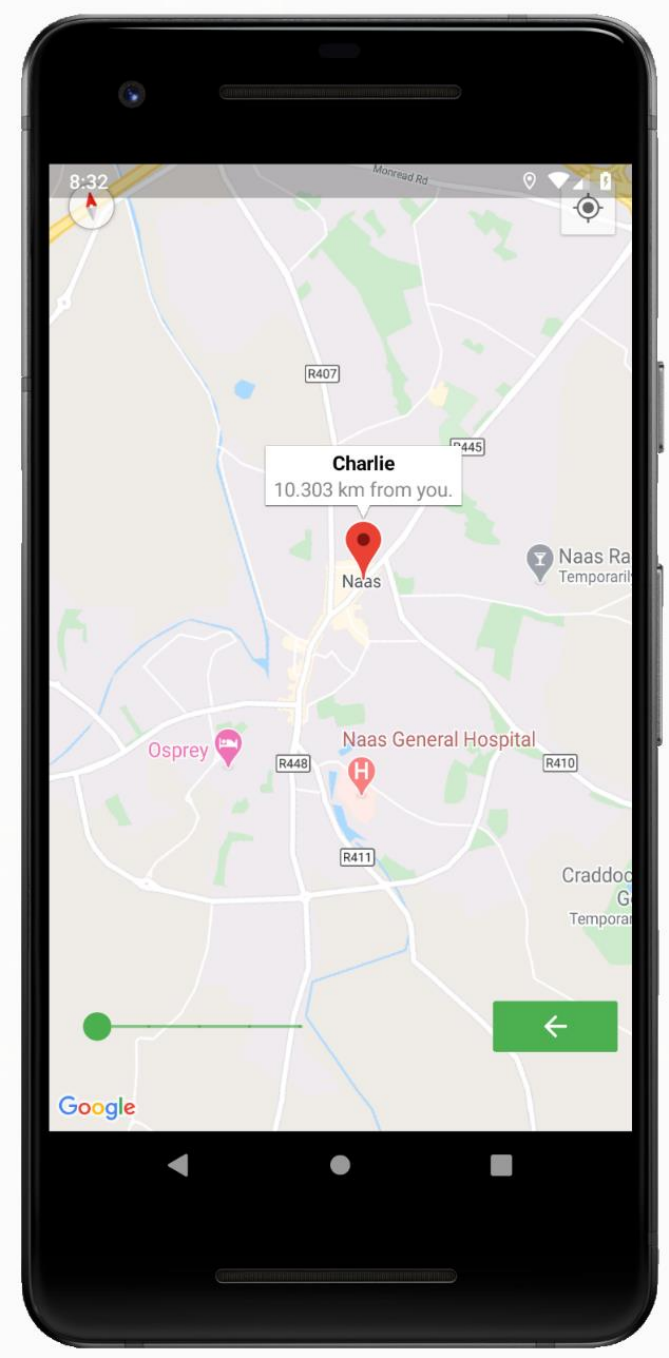
Figure 15: Comments Screen

2.1.5 Map



Screen shown when the user views the map, used to display all posts in Ireland via markers. The map has three possible UI interactions. The slider in the bottom left are different levels of zoom. The second button on the bottom right can be used to navigate back to the previous screen. Lastly, the button on the top right will direct the camera to the user's current location. The map also responds to touch to navigate and zoom.

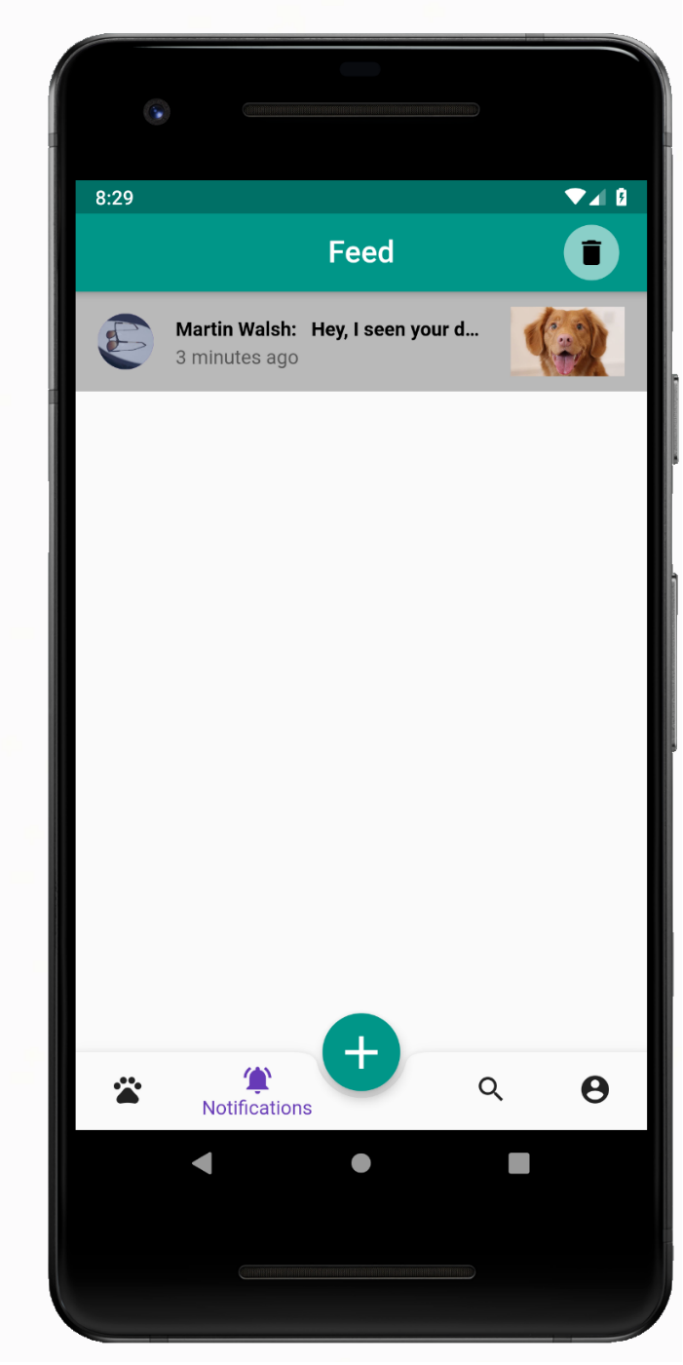
Figure 16: Map Zoomed Out View



Screen shown when a user taps on a marker. It displays the name of the post and the distance from the user in a pop-up information window. The user can then tap on this information window to open the posts page.

Figure 17: Marker Information View

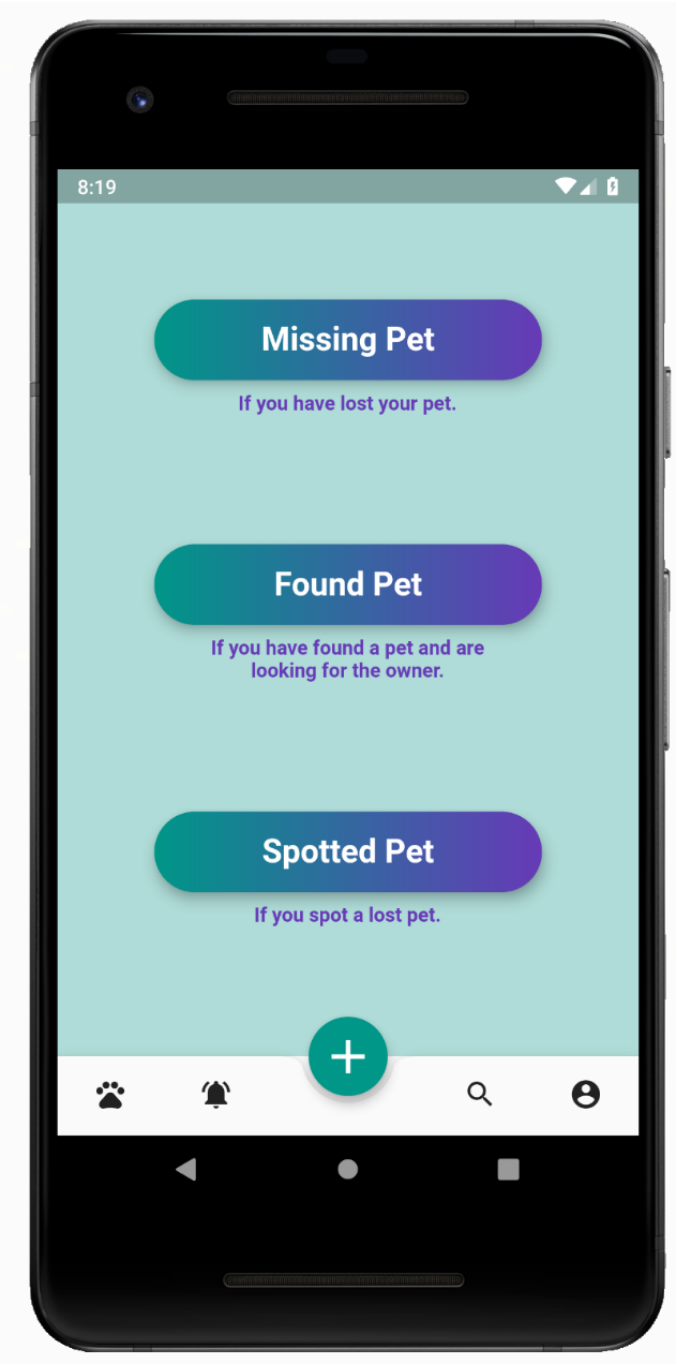
2.1.6 Activity Feed



The screen shown when a user views their activity feed. It displays to the user any comments that have been made on one of their posts. Each activity feed item can also be tapped to navigate directly to the post. There is a delete button located on the top right of the screen that will clear all notifications.

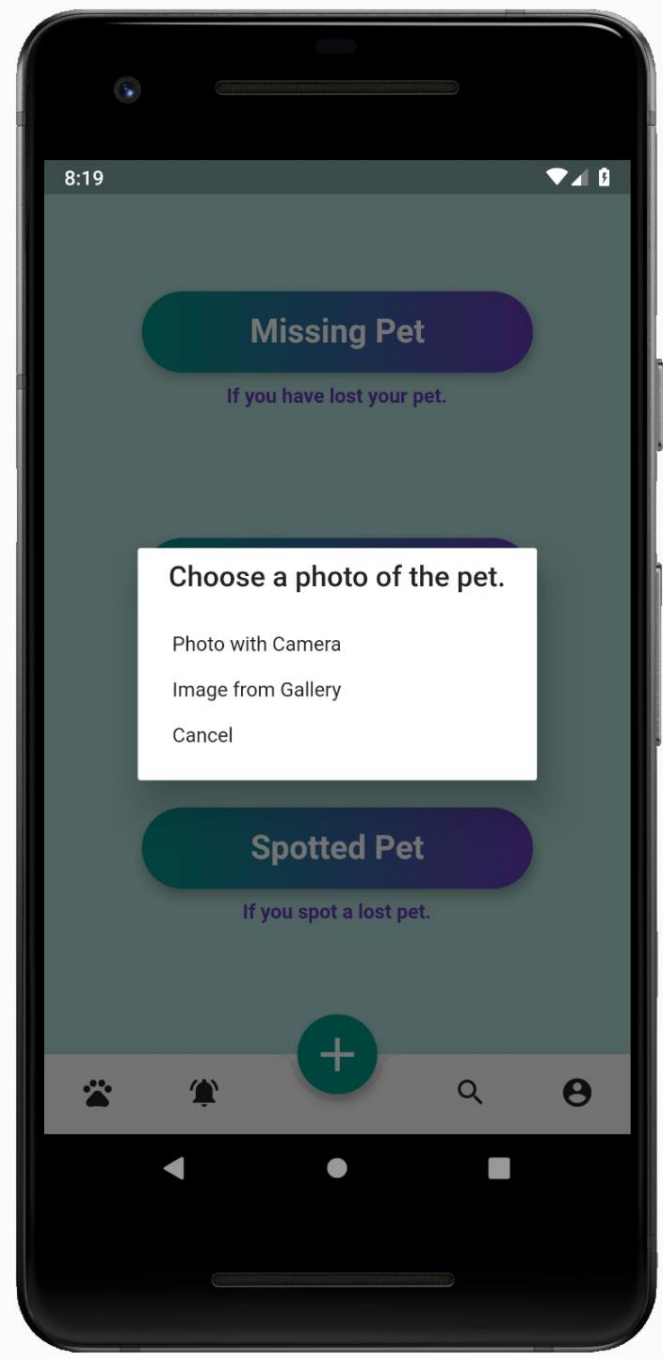
Figure 18: Activity Feed View

2.1.7 Add Post



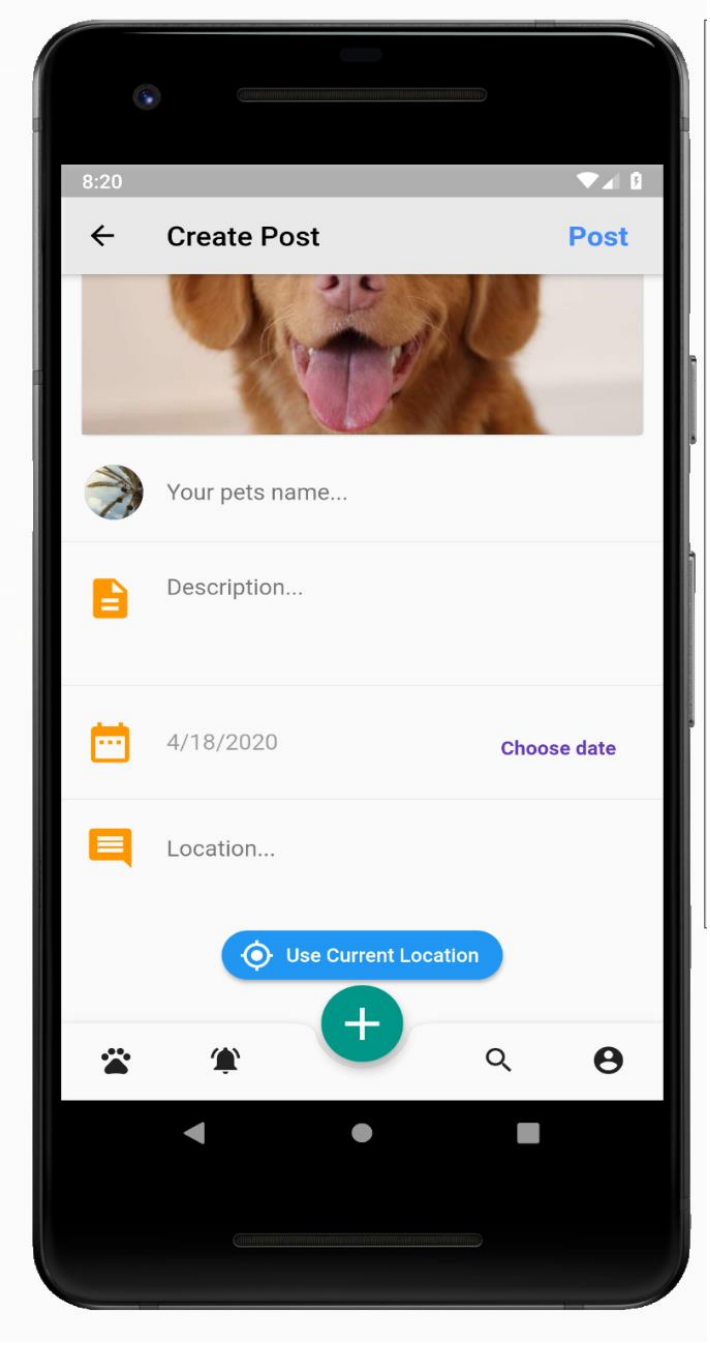
The screen shown when a user attempts to add a post. It displays three different post options with a description of each.

Figure 19: Add Post Screen



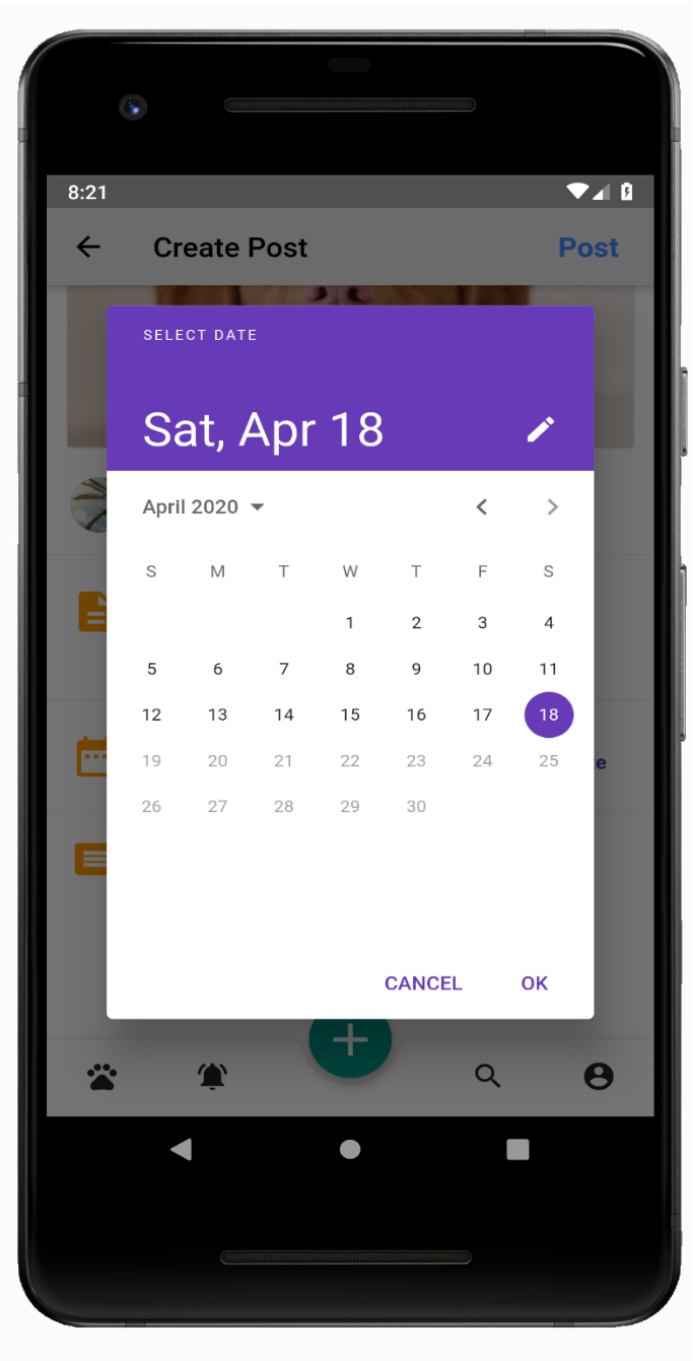
Screen shown after choosing a post option. The user can choose to take a photo and access their device camera or choose from their gallery which will give them access to all their stored images.

Figure 20: Image Option Pop Up



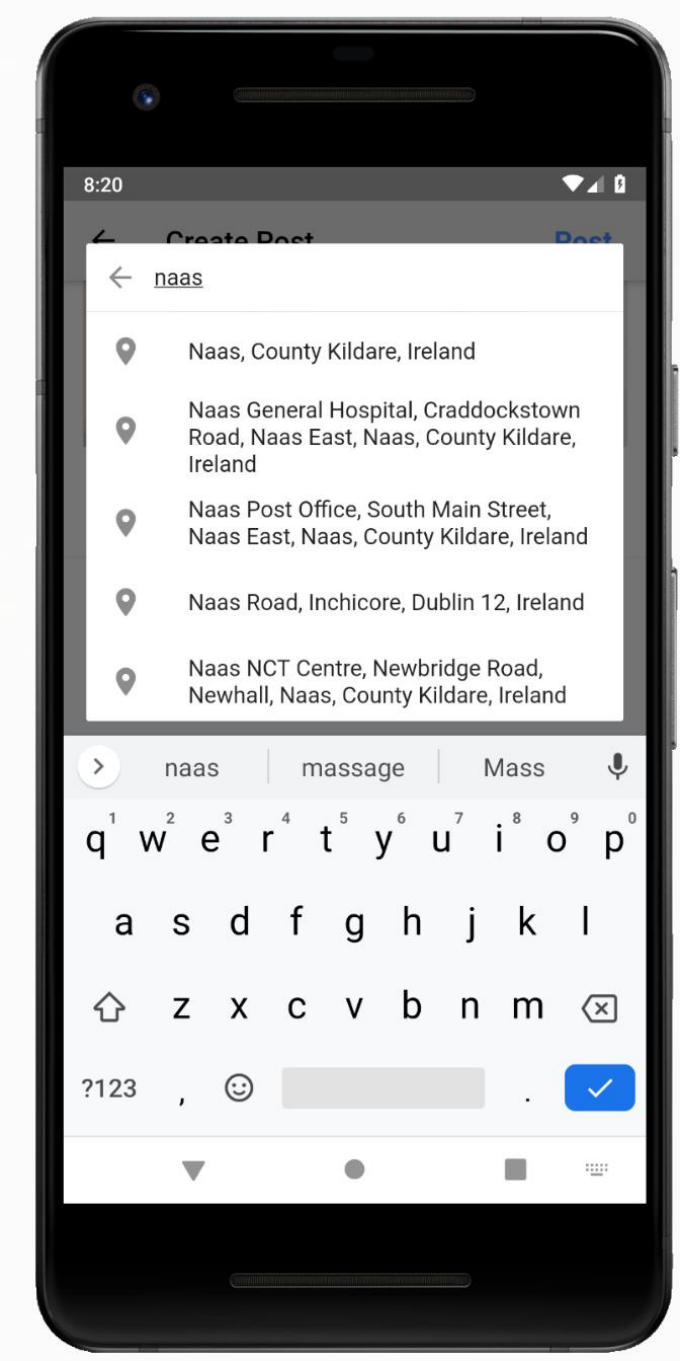
The screen shown after selecting an image for the post. The post form consists of four fields, the first two being normal text fields for the user to fill in. The next is the data field, which defaults to the current date. The user also has the option to enter the date manually using the functional calendar. For location the user has two options. They can hit the blue button to get the current location using the devices GPS or type to bring up Googles predictive search engine.

Figure 21: Create Post Screen



Screen shown if the user decides to choose a different date than the default.

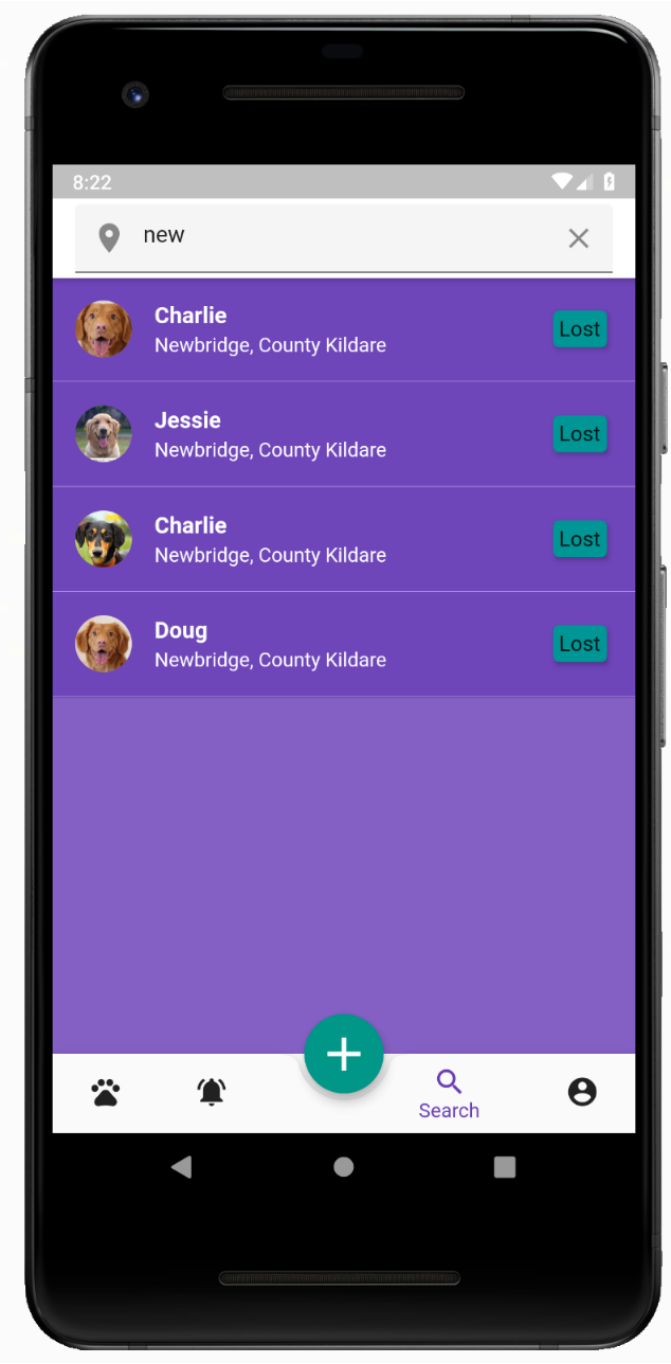
Figure 22: Calendar View



Screen shown if the user searches for an address to enter. It brings up Googles auto-generated location search. This allows the user to quickly and easily select their location.

Figure 23: Google Auto Generated Search

2.1.8 Search

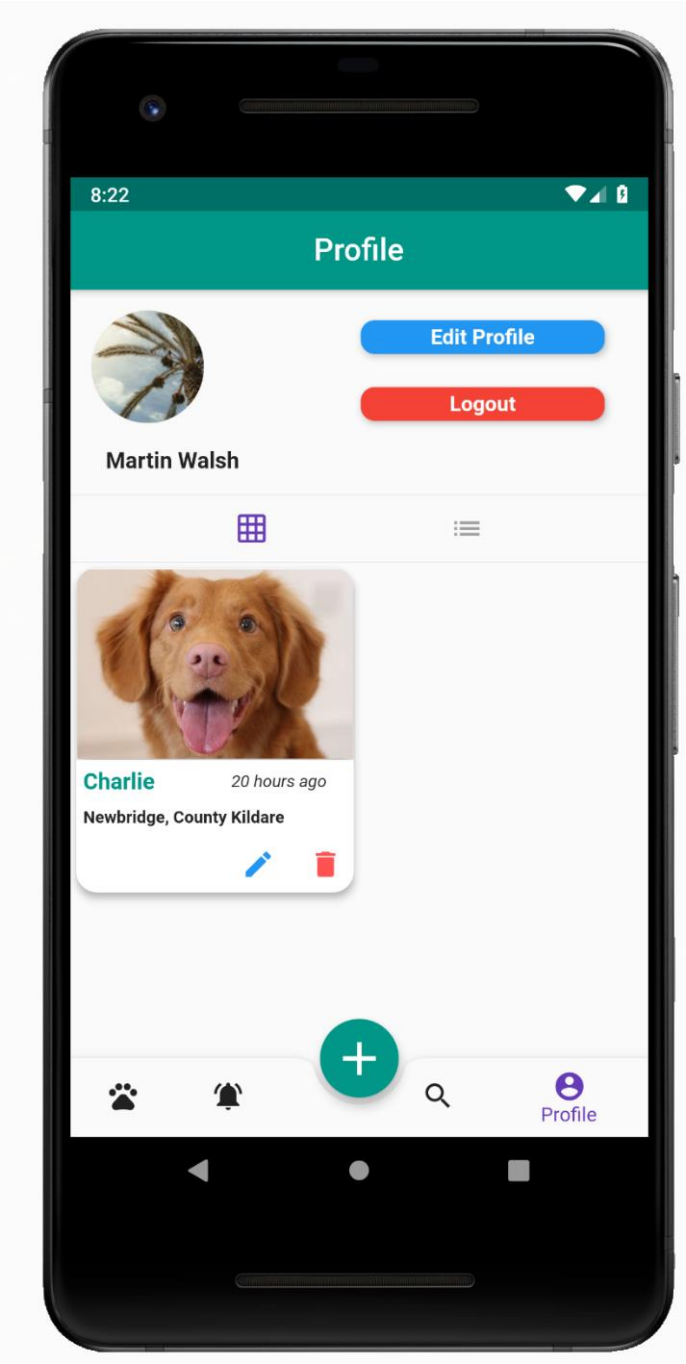


Screen shown if user decides to search for posts based on location. The search returns results that are equal or greater to the search query, so the user only needs to partially type the location.

Within the search bar there is also a location icon, which if tapped will fill in the search bar with the user's current location. The location is retrieved in real time using the users GPS. There is also an icon that displays an X, which if pressed will clear the users search bar.

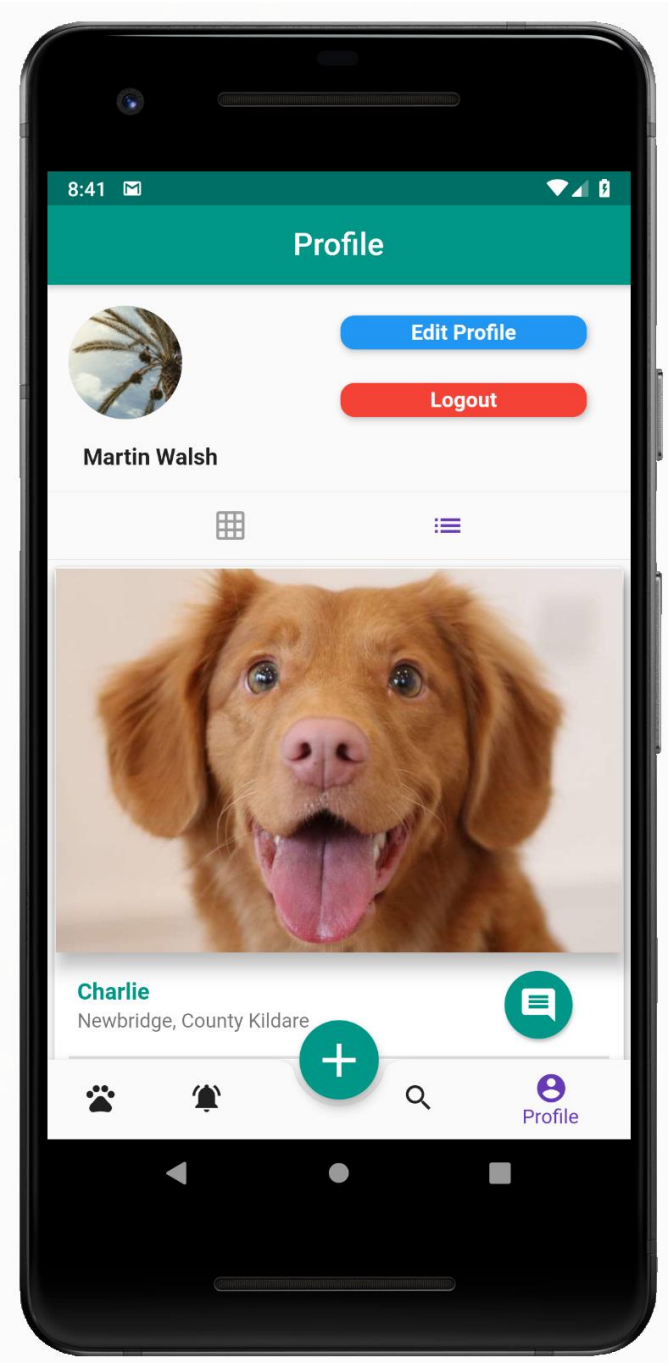
Figure 24: Search Screen

2.1.9 Profile



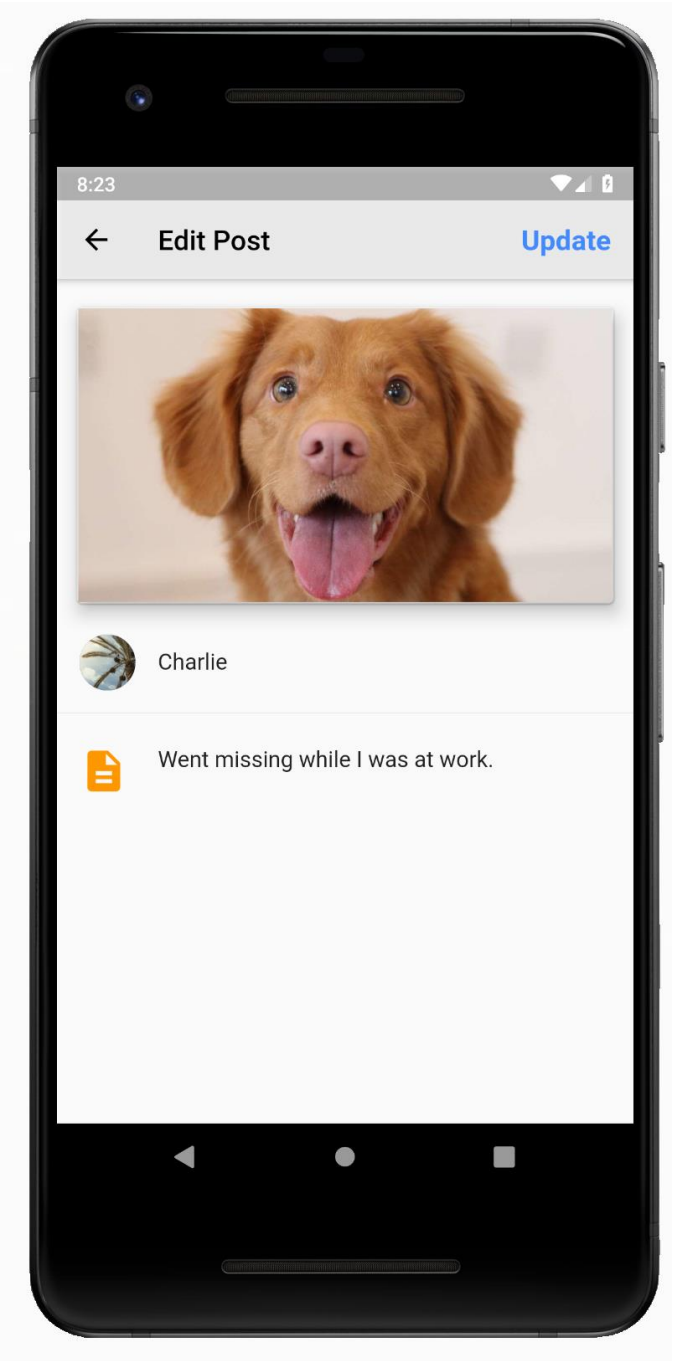
The screen shown if the user views their profile. It is only viewable by the user and will only display their own posts. It displays the user's avatar and name. This page allows them to edit their profile and logout. They can also view, edit and delete their posts with an added option to change how the posts are viewed, i.e. grid or list.

Figure 25: Profile Screen



Screen shown if the user changes how the posts are viewed.

Figure 26: Profile Screen List View



Screen shown if the user edits their post. The form fields are already filled with what the user previously entered.

Figure 27: Edit Post Screen

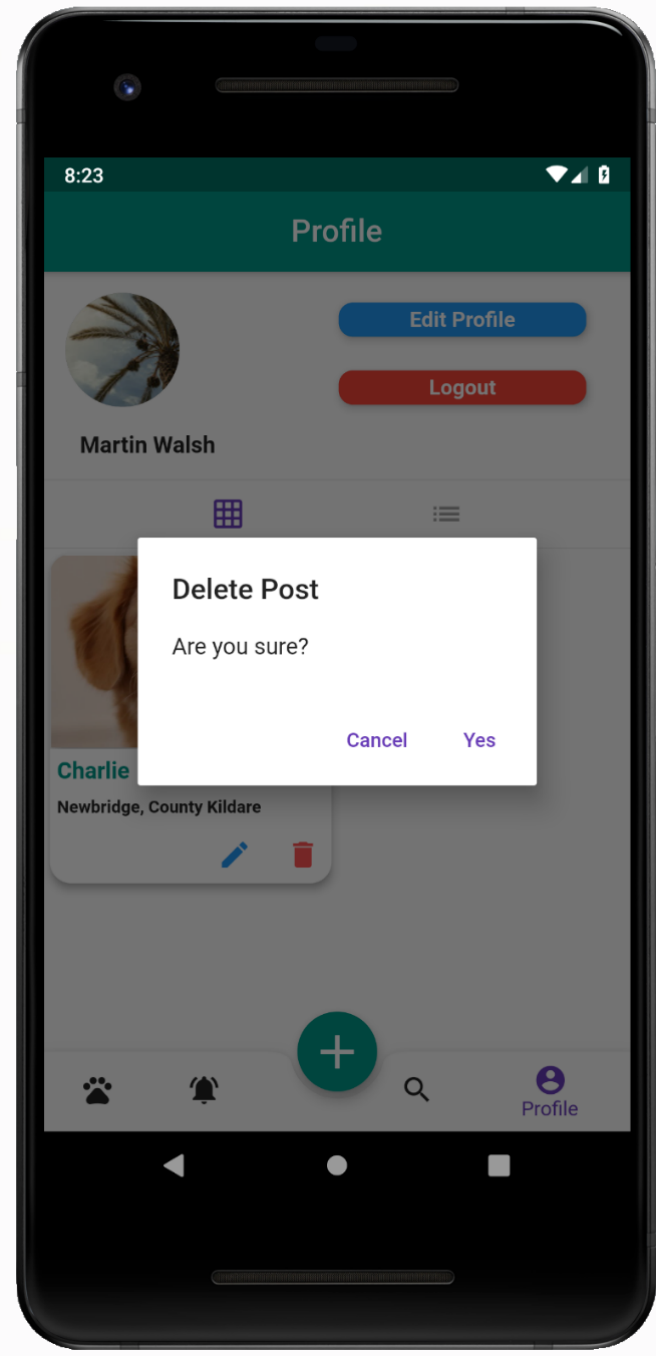
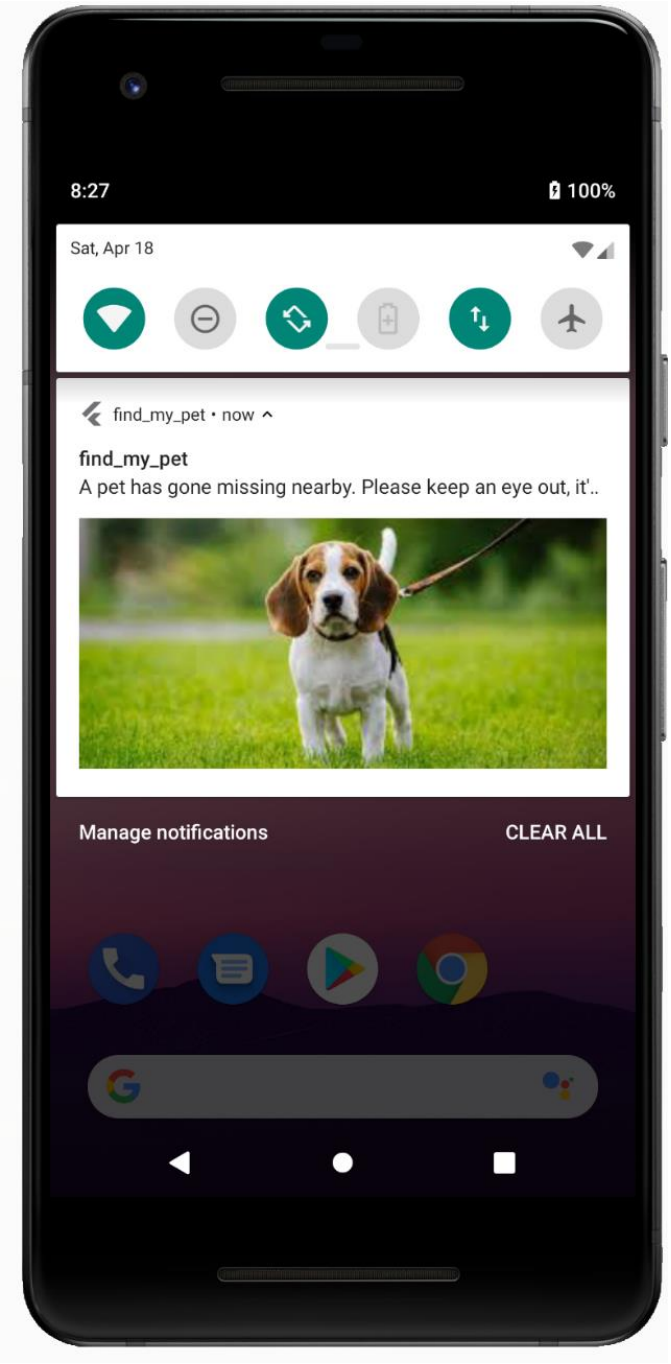


Figure 28: Delete Post Warning

The screen the users are shown when they tap the delete button on a post. If the user confirms, the post will be deleted from the database and disappear from the user's profile. Details of the posts will be removed from multiple other areas of the database, such as the image in storage, any activity feed items relating to the post, comments that were on the post and location information that was used to display the post on the map.

2.1.10 Push Notifications



This is how the post notification will appear when a pet is posted as missing and a user is within a 10-kilometre radius of its location. This notification will let users know within the radius that a pet has gone missing near them and display an image of that pet. This instantly makes users aware of what the pet looks like and that it is missing.

Figure 29: Lost Pet Push Notification

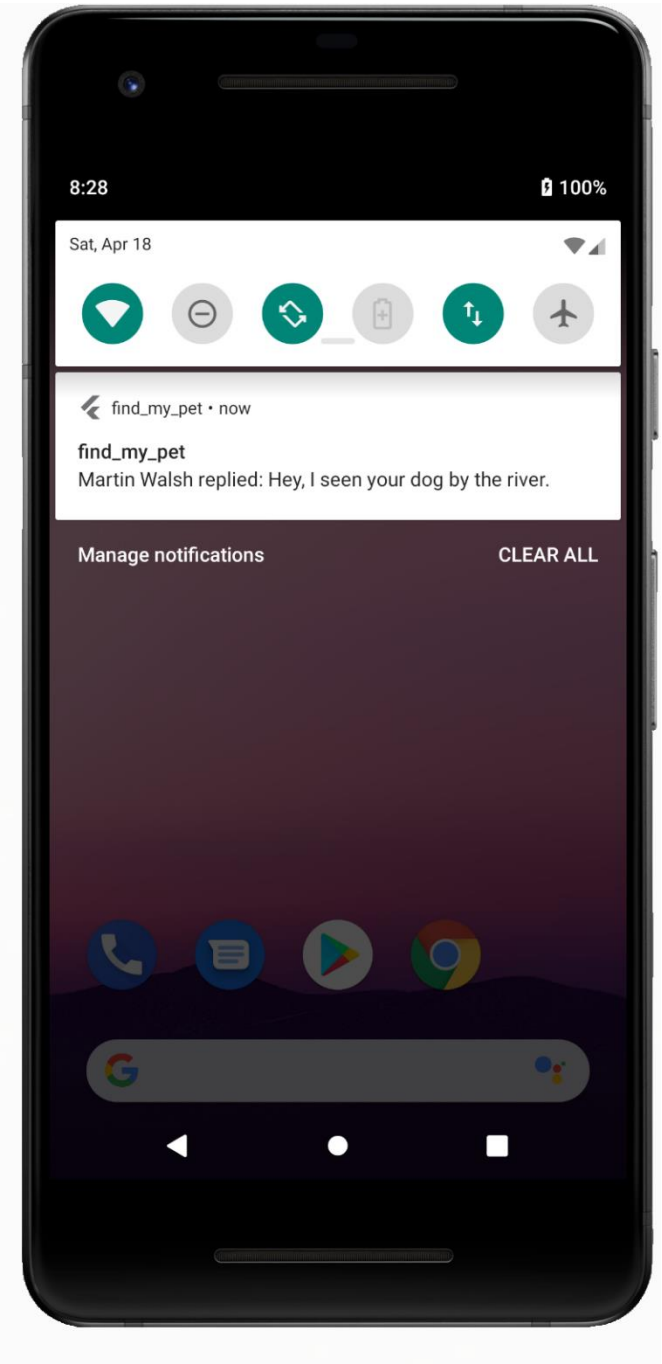


Figure 30: New Comment Push Notification

This is how the post notification will look like when someone comments on a user post. It ensures the owner of the post will be made aware of the comment even if not using the application at that time. It displays the commenter's name and the comment itself.

2.2 Database Structure

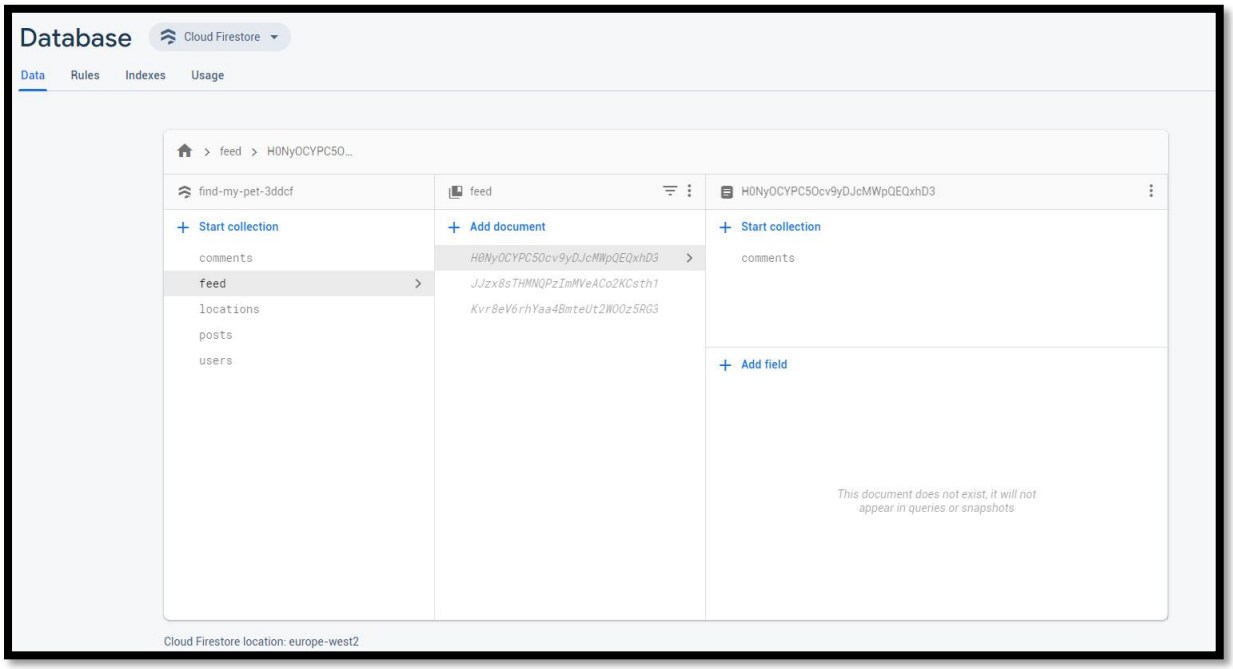


Figure 31: Database Overview

The database is divided into five sections (Figure 31):

1. Comments – Contains a list of all comments for each post in the application.
2. Feed – Contains a list of all activity items relating to each user.
3. Locations – Contains a list of all post locations and their related posts.
4. Posts – Contains a list of posts alongside post and user data.
5. Users – Contains a list of all users and their data.

For more detail please see the technical manual.

3. Testing

Due to the COVID-19 lockdown, testing was done solely between my own mobile device and the emulated device on my laptop as access to other devices was not possible.

Images of the tests can be seen in the previous section. The tests carried out were as follows.

3.1 Authentication

1. Invalid login – Entering an incorrect login to ensure an alert box stating so was presented to the user. This was carried out for email and password.
2. Successful login – Entering correct details and progressing to the home screen.
3. Successful Google login – Choosing the Google sign-in option and choosing the account progresses to the home screen.
4. Switching between authentication modes – tapping sign-up instead navigates to the sign-up page and vice versa.

5. Invalid email layout – Entering an invalid email to ensure an alert box stating so was presented to the user.
6. Invalid email – Entering an email already in use to ensure an alert box stating so was presented to the user.
7. Passwords do not match – Entering a different password in the password confirmation field to see if warning is produced.
8. Sign-up successful – Creating an account to see if user is told it was successful and account is created in backend.
9. Change password invalid – Tapping change password button without an email to ensure an alert box stating so was presented to the user.
10. Change password valid – Ensuring user gets reset password email when change password is pressed.

3.2 Posts Overview

1. Post types – Switching between tabs to ensure the correct posts are being shown.
2. Radius – Choosing each radius option from drop down menu to see if correct posts are displayed.
3. Navigating to posts – Tapping on posts to see if they are loaded correctly.

3.3 Individual Post

1. Data – checking that correct data has loaded based on backend information.
2. Comments – Tapping on comments button to see if app navigates to comments page.

3.4 Comments

1. Adding comment – Adding comments from different devices to ensure if they appear in real-time.
2. Comment data- Checking that comment data, time and image are correct.

3.5 Map

1. Markers – Checking if there is a marker for each post within the database.
2. Information window – Checking that tapping on a marker produces an information window containing post title and distance.
3. Navigation – Checking that the map can zoom in and out and move any direction.
4. Centre – Checking that the centre button return map view to user's location.
5. Back button – Checking that back button navigates user to previous screen.
6. Marker tap – Checking that tapping on a marker navigates the user to that posts screen.

3.6 Activity Feed

1. Display – Commenting on a post to see if it is displayed in the user's feed.
2. Data – Checking that the data in the feed item is correct, such as time, name, image and comment data.
3. Navigate - Tapping on a feed item to navigate to the post its from.
4. Delete – Checking that pressing the delete button clears all feed items.

3.7 Add Post

1. Post types – Checking that each post option is stored under the correct heading in the database.
2. Image gallery – Checking if the user can upload a photo from their gallery.

3. Camera - Checking if the user can upload a photo taken in real time.
4. Current location – Checking that current location button enters the correct location for the user.
5. Generated location – Checking that Googles auto-generated location search works.
6. Calendar – Checking a user can choose a date from the calendar and it is reflected in the date field.
7. Warnings – Trying to post with empty field to see if warnings are shown.

3.8 Search

1. Location button – Pressing location button to check that it automatically fills search bar with user's location.
2. Clear button – Pressing clear button to check that it clears search bar.
3. Search – Partially entering an address to check that it shows all posts with location greater than or equal to the search query.

3.9 Profile

1. Edit button – Pressing edit profile button to check that user is navigated to edit profile screen with their previous data already in the field option.
2. Update profile – pressing update profile to check that the change is made in the database and application.
3. Post orientation – switching between the post orientations to check that posts change their display types.
4. Edit post – Pressing edit post button to check that user is navigated to edit post screen with their previous data already in the field options.
5. Update post - pressing update post to check that the change is made in the database and application.
6. Delete warning – Pressing delete button to check that user is present with alert box asking them to confirm deletion of post.
7. Delete post – Deleting post to ensure it is deleted from all necessary areas of the backend, such as posts, comments, location and feed collections. As well as the associated image being deleted from storage.
8. Logout – checking that logout button navigate to the authentication screen and prevents automatic login from happening.

4. Learning Outcomes

This project allowed for learning in multiple areas of software development. Each one of which can be applied to future projects and sets a great foundation for any further mobile application development. The practise with working on a large project, with multiple technologies over a long period of time will no doubt be invaluable in future endeavours.

4.1 Flutter

Flutter is an open source mobile UI toolkit created by Google. With one codebase, Dart, it allows the development of cross-platform native mobile applications. Meaning one programming language and codebase allows the creation of an application for both iOS and Android.

A major learning outcome from this project is the experience and knowledge built with this framework. Now having the ability to create and design highly functioning mobile applications will no doubt be beneficial. Being highly competent in this area now opens new doors to build potentially impactful applications that were previously not possible.

Through working with Flutter knowledge of Dart was also gained. Dart focuses on front-end development but is an object-oriented language, so its ideas and concepts are transferable to other more common languages.

4.2 Firebase

Firebase, the Backend-as-a-Service (Baas) mentioned in the project description is another major learning outcome of this project. Having used most of Firebases tools on offer during the development of this project, the knowledge gained essentially means that projects that may have otherwise been too large to build due to the backend requirements are now feasible. Authentication, NoSQL database, storage, serverless automated backend code and a messaging system can now all be comfortably implemented into an application using what was learned.

It can be argued that the combination of both Firebase and Flutter makes it possible to create almost any mobile application.

4.3 Personal

My main learning outcome from this project was learning to plan and prioritise my time. During the projects development I noticed at times that although I was putting a lot of time into it, I was not seeing much progression. This led me to take a step back and plan out what I wanted to accomplish each time I started development. Writing a list of the minimum viable outcomes I wanted to achieve during a session allowed me to see a higher return on the time I was investing. Knowing exactly what I wanted to develop each time I started coding allowed me to develop quicker and research more effectively when a problem arose.

Time management in general was an important learning outcome of my time working on this project. Having a project this size while also carrying out the work for other fourth year modules proved difficult. Ensuring time was divided adequately between subjects and project proved of the utmost importance to the success of the project.

5. Flutter Evaluation

As Flutter is a relatively new technology, a part of this project was to evaluate it once the project was complete. Personally, having no previous app development experience, it cannot be compared against other popular mobile application development frameworks. However, the following were stand out points noticed during development.

5.1 The Language

As mentioned throughout the project the language for the development of Flutter applications is Dart. The process of learning and implementing Dart code was an enjoyable one. It was relatively easy to learn, and its syntax was similar to that of Java. Although a relatively new language all the necessary extensions that would be required were available during

development. Also, for the most part, any errors that occurred throughout were well defined. Making it easier to implement a fix. In summary, Dart is powerful and easy to use language that makes developing Flutter application enjoyable.

5.2 Convenience

Flutter appears to be designed to help make development of mobile applications a faster process. With a feature called “Hot Reload” that enables you to completely refresh the UI during development to display changes made within a second. The speed and reliability of this feature makes testing different UI options quick and easy. Also, providing shortcuts to all the essential UI building widgets, such as a containers, columns and rows.

5.3 Documentation

The documentation for Flutter is excellent. Multiple examples can be found for most widgets and classes available. Plus, the development team behind Flutter regularly produce short high-quality videos explaining small aspects of the technology and widget ideas to integrate into applications.

5.4 Overall

In conclusion, Flutter is an efficient and accessible framework. It was ideal for a first-time user as there is a wealth of information available on the technology, including easy to follow videos. In addition, Flutter is convenient to utilise. It provides users with a variety of extensions, along with a widget library and shortcuts. Moreover, the Dart language is easy to grasp which makes using Flutter a more fluid and enjoyable experience. Personally, having no experience with other mobile application development frameworks, it is hard to see anything that Flutter is missing.

6. Project Review & Conclusions

6.1 What was achieved?

The core functionality the project set out to do has all been achieved. The location-based push notification was a major milestone as it works exactly as initially envisioned. Only users within a 10-kilometre radius of a lost animal’s post will get a notification about the missing animal. At times it seemed that the notification would need to be based on towns as trying to get the radius of location appeared too complex. However, this would lead to users getting notifications of animals possibly far from home which they could not possibly find. Which in time could lead users to remove the app from their device. Therefore, achieving the initial format of post notification is a great achievement.

Utilising many facets Firebase has to offer is another achievement. Including the needed database but also its authentication, storage and functions services makes for a very advanced first mobile application. That due to this diverse backend, will run smoothly, keep data synchronised across devices and allow for backed code to be run remotely.

An unexpected achievement which was not really considered before starting the development is the work put into the User Interface. As more time was spent in development and more was learned about the possible ways of changing and improving the UI, a lot of time was put into the application looking and acting nicely. Resulting in an attractive easy to use mobile application that adjust dynamically to phone sizes.

6.2 What was not achieved?

Due to the COVID-19 lockdown there was no access to an Apple computer to test and deploy the mobile application for iOS. Although all steps were taken throughout development for the application to run on iOS. Including permissions, configuration and viewing the app on a virtual iPhone to ensure look and feel was the same as on Android. It meant that the final checks and release plan for iOS could not be carried out.

6.3 Problems encountered

The main problems encountered, although eventually solved, were based around the Firebase Cloud Functions. These are the functions written that will trigger the required push notification when certain events took place. Unlike the rest of the application this code had to be written in JavaScript and then deployed to Firebase via the command line.

This meant previously used packages would not work as the others were designed for Flutter. As a result, more research was required, particularly into packages that would help with procuring and configuring locations based on latitude and longitude.

There was no way of running these functions quickly to test if they work as expected. Instead they had to be deployed to Firebase, which can take up to 5 minutes. Then the event must manually be triggered in the app, i.e. add a post. Console log statements had to be placed throughout the code in order to determine how far the code was running before an error was thrown. Console log statements would be shown in the Firebase functions UI.

Eventually through trial and error the functions worked but it was the most time-consuming debugging process throughout the project.

6.4 What would be done differently?

As work progressed it became apparent that additional research into potential packages would have been beneficial. For example, it was difficult to source a package that included all the functionalities required to implement the location radius. Some appeared to meet the criteria initially, but oftentimes when implemented they proved to be missing something essential. More research into various packages earlier in the project could have potentially saved time as there would be no need for trial and error.

It might also be more beneficial to invest time into gaining in-depth knowledge of the Flutter framework before commencing the project. Though the project was undertaken with a comfortable grasp on the framework, a more nuanced understanding could have potentially saved time. A project of this size with a strict timeframe leaves little room for error. Lack of in-depth knowledge during this project necessitated the foundation be rebuilt. This was

because it became apparent that there were more efficient ways to keep the state management of the application.

6.5 Deviations from Initial Specification and Design

In terms of core functionality, the project stayed true to the initial design and specification planned. All features that were set out were achieved. The design of the applications UI varies from the initial in some ways. However, overall, the planned aspects have been reflected in the application

The areas in which the project does deviate however is with the secondary functionality. With that being said, there isn't less functionality in the final product, but it is different. Initially it was planned to have sections for pet adoption and forums for users to discuss certain topics. The final application does not include these elements but instead comprises of several other useful features. This includes real time commenting for users to communicate, a navigable map that allows users to see all posts throughout the country, an area to notify users of new comments and a way of searching for posts based on location.

6.6 Future Work

All the required features for the application to be entirely functional have been successfully implemented. It is in the process of being released on the Google Play Store. The application was also created with the intention of releasing it on the Apple Store too. Unfortunately, due to the current circumstances it was impossible to test iOS functionality. This is something to work on in the future with the intention of releasing the application across both platforms.

An additional feature that was explored in the research stage but not implemented is the introduction of a machine learning element to the application. The idea was to run a machine learning algorithm on each photo uploaded in order to ensure the photo is of an animal. This is a feature with a lot of potential and can be added to the application in the future.

Acknowledgements

Firstly, I would like to thank my project supervisor, Paul Barry. Who supported my decision to pursue this project idea and continually provided feedback and support to enhance it.

I would like to thank my Mum for her continued support over my academic career. Also, my girlfriend Jennifer, for always pushing me to pursue my goals.

Lastly, a special thanks to Bernard, Osama and Rhyder whose company and support has helped make this college year a great one.

Plagiarism Declaration

Declaration

- I declare that all material in this submission e.g. thesis/essay/project/assignment is entirely my/our own work except where duly acknowledged.
- I have cited the sources of all quotations, paraphrases, summaries of information, tables, diagrams or other material; including software and other electronic media in which intellectual property rights may reside.
- I have provided a complete bibliography of all works and sources used in the preparation of this submission.
- I understand that failure to comply with the Institute's regulations governing plagiarism constitutes a serious offense.

Student Name: Martin Walsh

Student Number: C00170339

Signature: 

Date: 20/04/2020