Description

Intended User

Features

User Interface Mocks
- Screen 1
- Screen 2
- Screen 3
- Screen 4
- Screen 5
- Screen 6

Key Considerations
- Which programming language will be used to implement your app?
- What are the versions of Android Studio, Gradle and all the other libraries?
- How will your app handle data persistence?
- Describe any edge or corner cases in the UX.
- Describe any libraries you'll be using and share your reasoning for including them.
- Describe how you will implement Google Play Services or other external services.

Next Steps: Required Tasks
- Task 1: Project Setup
- Task 2: Implement UI for Each Activity and Fragment
- Task 3: Implement Google Play Services
- Task 4: Notifications
- Task 5: Implement App Widget
- Task 6: Implement IntentService
- Task 7: Accessibility Check
- Task 8: Enable RTL Support

**GitHub Username**: Ashley20

# Pet's Friend

## Description

Animals need a home to stay  and people need others to help them achieve this. If you want to adopt a pet or find a  home for them then this app is for you.

Both options might be quite challenging as you might not know where to find other pet owners or don't have enough finance to visit them. Pet's Friend solves exactly this problem by giving you the chance to communicate  with the other user's of the app and easily accessing the pet information only with a single tap. You can easily search the map to see pets who waits for adoption, display their information or open a chat dialog with the owner. Searching on the map also gives you the pet's current location information which may help you make better choices.

## Intended User

The intended user is the ones who either wants to adopt a pet or post their pets for adoption.

## Features

- Saves user and pet information
- Create and update profile
- Display users current location on the map
- Filter users
- View profiles
- Messaging
- Sign up
- Sign in
- Notifications

# User Interface Mocks

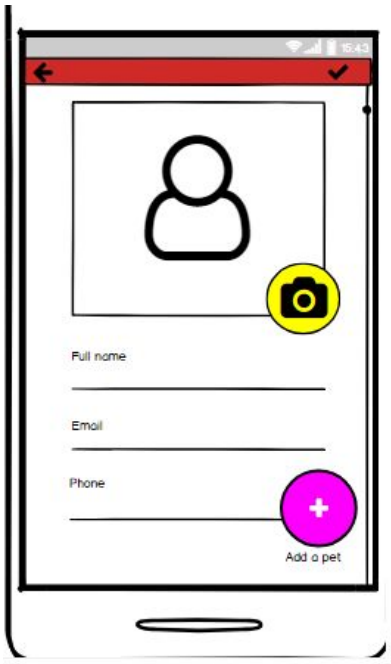
## Screen 1


One can create an account by putting in the full name, email and password. Sign up page.

## Screen 2


Login page. Users have to put in the correct email & password match to sing in to the app.

## Screen 3

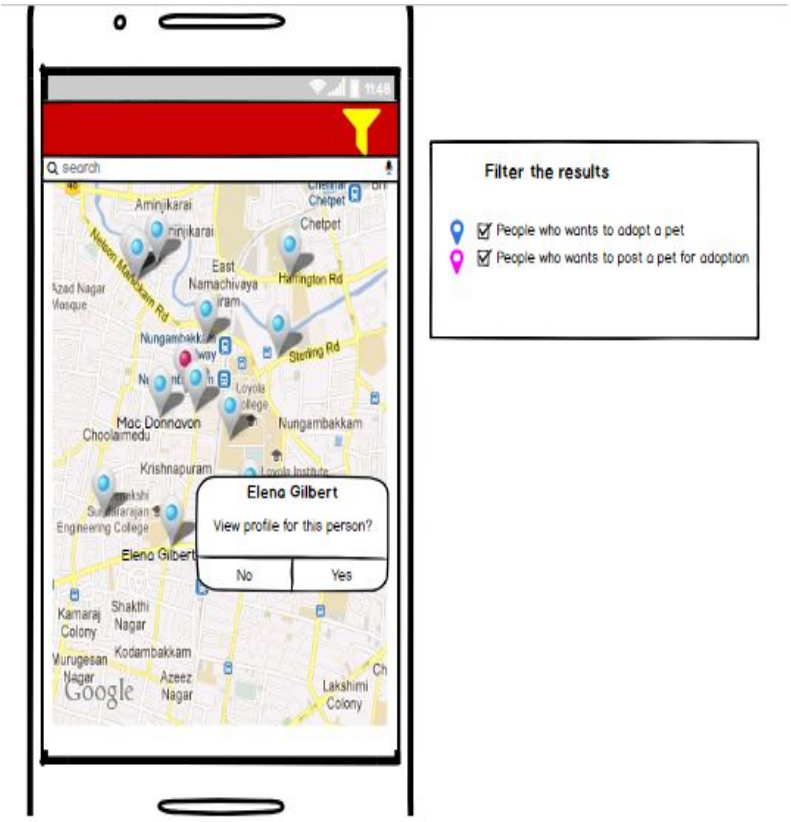



Profile page mockup. This screen shows how an user sees his/her profile page. User profile information can be edited and updated. Adding a new pet that belongs to the user or changing profile information is possible.

## Screen 4


Chat dialog page. Users can communicate by sending each other chat messages.

## Screen 5




Main Page. This is the page which appears first after login process. Users are marked as to their current location on a google map. Clicking on one of those markers will open a dialog asking the user whether or not he/she wants to display the selected user profile. There is a filter feature on the up-right end of the page which filters the users shown on the map according to checkbox selections made by the user. Blue markers on the google map shows users looking for a pet to adopt and the pink markers shows users who wants to post a pet for adoption.

### Screen 6


App widget. Clicking on one of those message dialogs will redirect the user to the actual conversation page with that person.

## Key Considerations

#### Which programming language will be used to implement your app?

The app will be written in Java programming language.

#### What are the versions of Android Studio, Gradle and all the other libraries?

Android Studio - **VERSION 3.1.3**
Gradle - **VERSION 4.4**

Picasso - **VERSION 2.71828**
Butterknife - **VERSION 8.8.1**
Google Maps - **VERSION 15.0.1**
Cloud Firestore - **VERSION 17.0.2**
Location services - **VERSION 15.0.1**
Android Design Support **VERSION 27.1.1**

### How will your app handle data persistence?

Firebase Cloud Firestore will be used to handle data. I picked Firebase Cloud Firestore over  Firebase Realtime Database because it features richer and faster queries and scales better. Firestore's documents and collections will handle the data in a simple way.

### Describe any edge or corner cases in the UX.

If the network goes offline while a user sending a chat message or receiving one the message data will not be lost and will be send again when the network comes back to normal. Those messages will have a weak color until the network comes online and they reach their destination.  The user will still be able see the chat history when there is no network.

The app needs user location permission to do its functions otherwise it cannot be used. If a user at any time  removes the permission that she/he approved before then the app will not crush and ask the permission again clearly describing why it needs that permission to function.

### Describe any libraries you'll be using and share your reasoning for including them.

Picasso to handle the loading and caching of images.
Butterknife for binding Android views.
Google Maps for the map features
Cloud Firestore for storing and retrieving data
Android Design Support Library for beautiful design

### Describe how you will implement Google Play Services or other external services.

Maps services for displaying google maps, markers and all.
Location services for getting the users' current location.

# Next Steps: Required Tasks

This is the section where you can take the main features of your app (declared above) and break them down into tangible technical tasks that you can complete one at a time until you have a finished app.

## Task 1: Project Setup

- Create a new android project
- Create a google and firebase account and get neccessary api keys
- Configure libraries
- Create app logo , icon

## Task 2: Implement UI for Each Activity and Fragment

- Put strings and dimensions in the strings.xml and dimens.xml
- Create a color theme for the app and include it in styles.xml

- Build UI for sign up page
- Build UI for sign in page
- Build UI for profile page
- Build UI for chat page
- Build UI for main page

## Task 3: Implement Google Play Services

- Implement maps services
- Implement location services

## Task 4: Notifications

- Create notification service for incoming new messages

## Task 5: Implement App Widget

- Create UI for the app widget
- Implement AppWidgetProvider class
- Implement IntentService for the widget

## Task 6: Implement IntentService

Implement an intent service which will regularly check if a new message arrives

## Task 7: Accessibility Check

The UI design will take into account Google's Talkback.

- Provide Content Descriptions
- Check if the UI elements clearly visible
- Check if the sufficient contrast and size is provided for the UI items
- Make groupings for the similar items
- Use density independent pixels so that users with low accessibility can change it.

## Task 8: Enable RTL Support

For the RTL languages such as Arabic and many others enable the RTL support and make some changes in the UI.