Technical Project Report - Flutter Module

Pet Friends App

Subject:	Introdução à Computação Móvel
Date:	Aveiro, 14/06/2023
Students:	103645: Manuel Diaz 89131: Leonardo Freitas g3
Project abstract:	A Pet Sitting app from the perspective of a pet sitter, that allows him to find and book for pet sitting services. Key achievements of the app include a user-friendly interface like a social network, where users can create a profile, post photos, discover other sitters, send messages, see the location routes between him and pet and have a tour interface when a service starts.

Report contents:

1 Application concept

Our pet sitting app is designed with the primary objective of addressing the growing demand for professional pet care services. The motivation behind the app stems from the increasing number of pet owners seeking reliable and trustworthy individuals to care for their beloved pets when they are unable to do so themselves, due to work, travel, or other commitments.

We developed the application from the perspective of pet sitters so, the target users of the app include experienced and qualified professionals who like to take care of pets and, in this way, can earn some money doing something they like.

With our app, we want to ensure that pet owners can benefit from the peace of mind that comes with knowing their pets are in good hands. Overall, Pet Friends app aims to make the process of finding and booking a pet service simple and easy for pet sitters and provide them a user-friendly interface, where they can have a profile, post photos, discover other people who like animals, have a good experience walking animals, etc.

2 Implemented solution

At the beginning of the project we defined some features and requirements that we had to include in our application. We have implemented the following:

- **QR Code:** validation with the qr code is necessary for the pet sitter to be able to start the pet walk. Each request is identified with a code and in order to confirm that it is the correct pet, the pet owner will have to show the qr code corresponding to his request, thus confirming the pet's identity.
- **Map:** we integrate the map twice. One to mark a route between the pet sitter's current location and the pet, so that the pet sitter can move to the pet location and start his service. And another corresponding to the service itself, where he can track his location while walking the pet.
- **GPS:** it is related to the above, because the pet sitter can follow his location and know the way where he has to go, for example, to find the pet.
- **Camera:** the camera can be used for the pet sitter to add photos to his profile.
- **Distance counter:** when the pet sitter is on a service, a kilometer counter is displayed, so he can see how far he and pet has already traveled.
- **Timer:** when the pet sitter is on a service, too, a timer is shown counting down the time of the job.

Architecture overview (technical design)

The architecture devised for our software solution is based on an atomic design by Brad Frost. In this methodology he uses 5 different levels for the design of the app. These consist of: Atoms, Molecules, Organisms, Templates and Pages.

ATOMS
MOLECULES
ORGANISMS
TEMPLATES
PAGES

- Atoms are the basic building blocks of matter. Applied to web interfaces, atoms are our HTML tags, such as a form label, an input or a button.
- Molecules are groups of atoms bonded together and are the smallest fundamental units of a compound. These molecules take on their own properties and serve as the backbone of our design systems.
- Organisms are groups of molecules joined together to form a relatively complex, distinct section of an interface.
- Templates consist mostly of groups of organisms stitched together to form pages. It's here where we start to see the design coming together and start seeing things like layout in action.
- Pages are specific instances of templates. Here, placeholder content is replaced with real representative content to give an accurate depiction of what a user will ultimately see.

For Persistence we used a local Database called Isar DB. Isar is a fast, NoSQL, Cross/Platform DB for flutter made by the same people behind Hive DB.

To interact with the database, we use a file/class called **isar_service** that implements the necessary methods to adapt the already very user-friendly DB to our specific solution. It lets us customize queries to be used in the other project classes and widgets.

In the DB we have created 3 entities: Petsitter, Pet and Image_entity. A Petsitter can have multiple pets as requests for work/services, and multiple images in his profile. A Pet for now can only have one image and one petsitter.

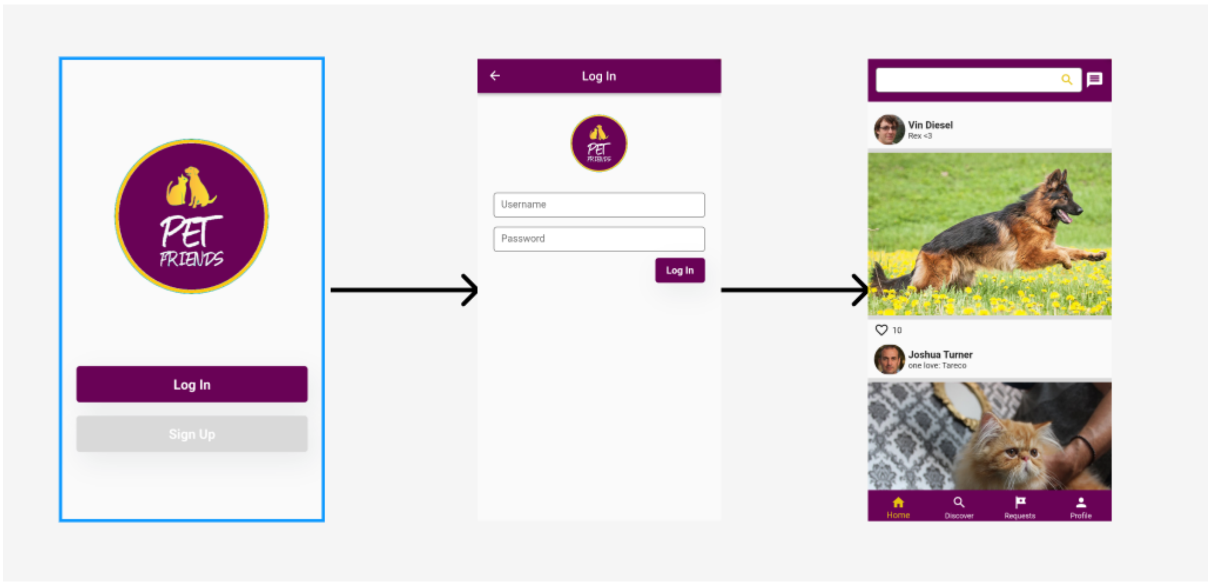
To ensure privacy and allow multiple users to use the app we choose to use a BLoC approach to manage the current logged user of the app. This BLoC or Business Logic Component has two functions: a **ChangeUser** and an **UpdateUser.** The first is invoked every time a user logs in the app and emits a new state informing all interested listeners of the new instance of Petsitter. The latter method is used to change the user's settings in the profile settings page and it basically receives a new Petsitter object, next copies its changed attributes to the current user and finally saves the changes in the DB.

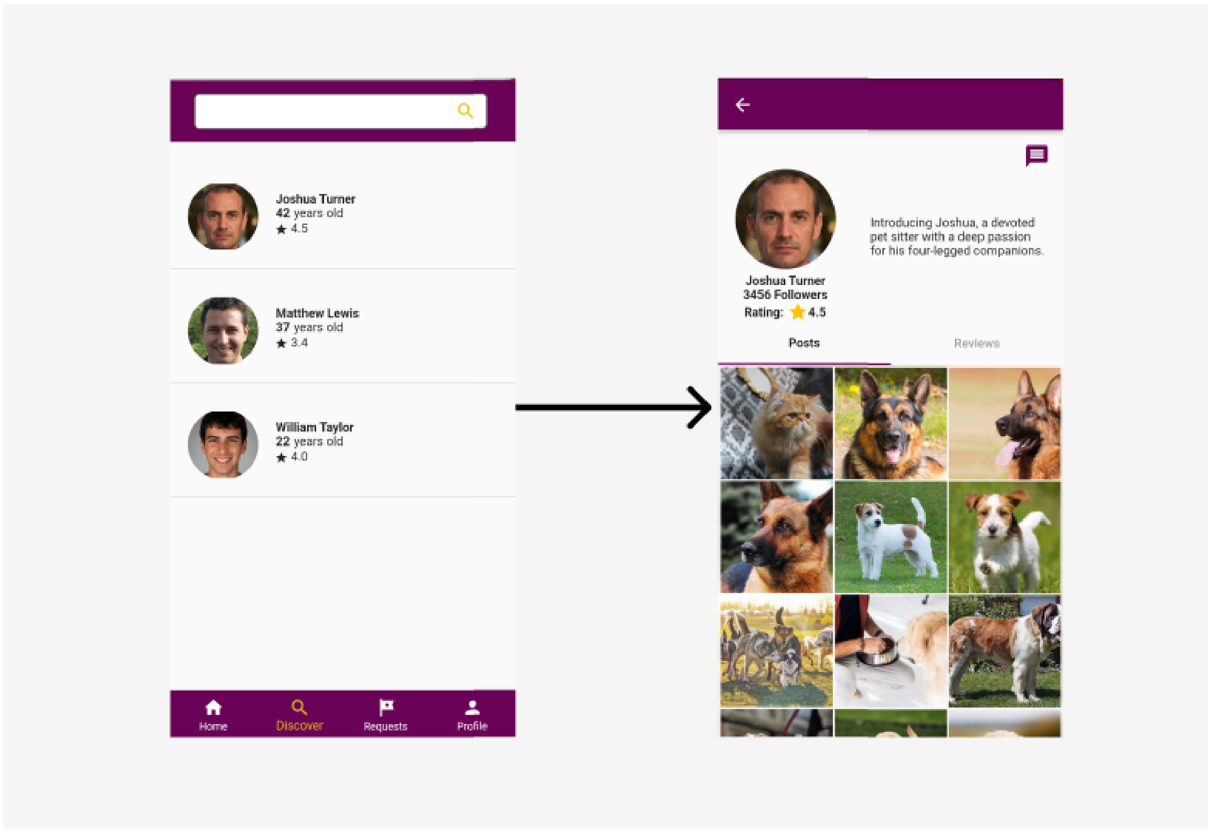
The second and final BLoC used is to ensure we get the correct information carried over from the requests page(In the bottom NavBar) ,which is a list inside a FutureBuilder, to its further pages. In this page we also implement a SlidableWidget approach.

In the profile Page we can see the current user's information, change settings, logout, scroll and zoom individual pictures and finally add pictures to the profile via phone's camera or gallery. To use the camera and gallery we used a library called image_picker.

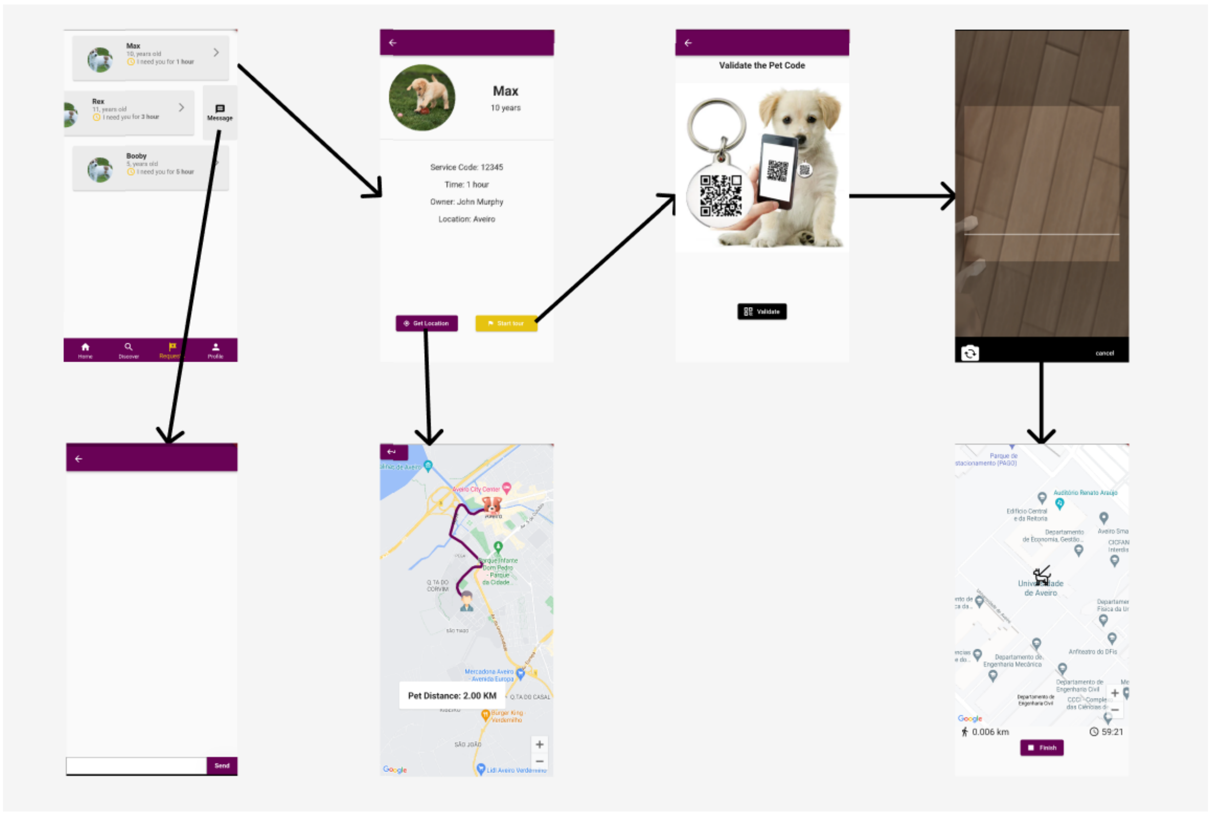
When it comes to integrating with Internet-based external services, one of the strategies that we've utilized is leveraging the Google Maps API. By doing so, we were able to provide our app's users with a seamless experience of searching for, visualizing, and interacting with maps and location-based data. The Google Maps API offers a wealth of features and functionalities that developers can leverage, such as geocoding, routing, distance matrix and more. It also provides SDKs and plugins for different platforms and languages, making it easier to integrate with your app's tech stack.

Implemented interactions





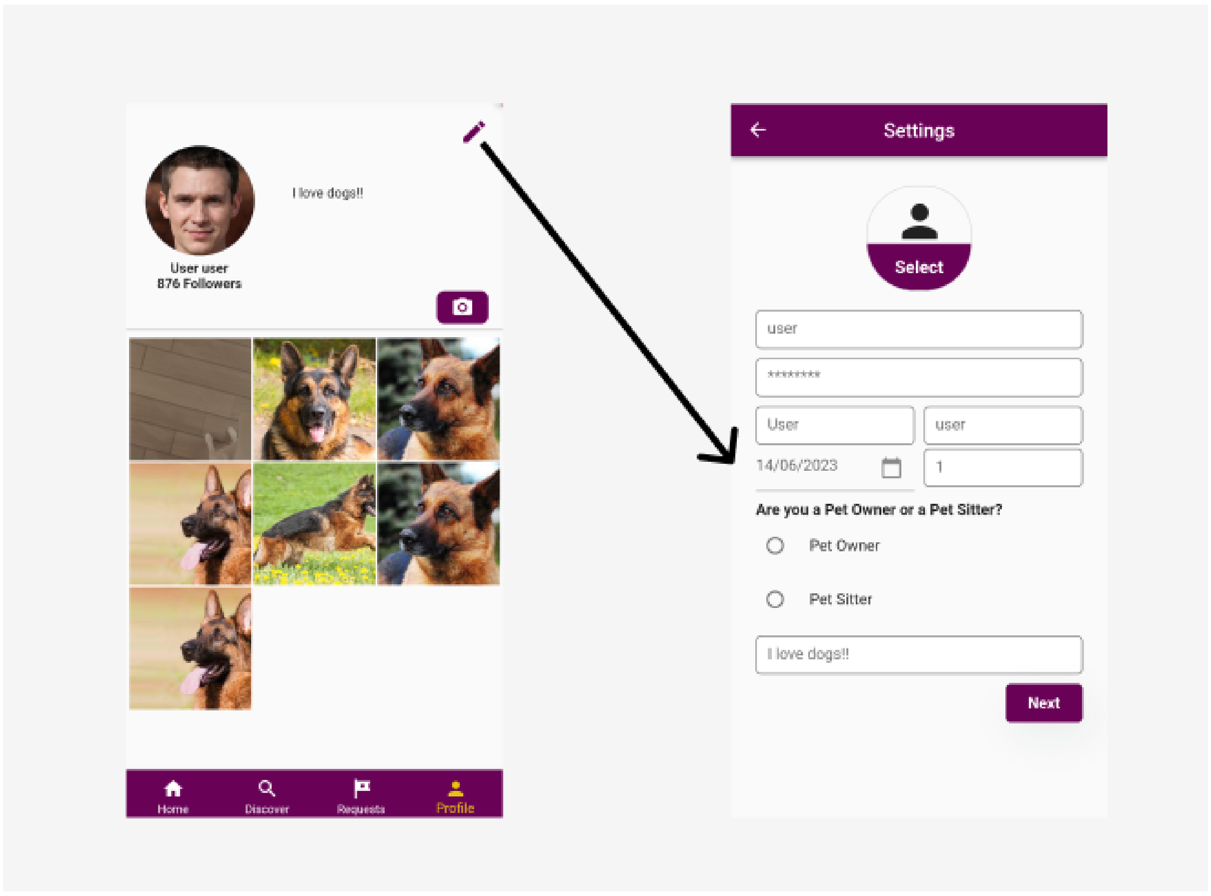
Introducing Joshua, a devoted pet sitter with a deep passion for his four-legged companions.



Service Code: 12345
Time: 1 hour
Owner: John Murphy
Location: Aveiro

Pet Distance: 2.00 KM

0.006 km
59:21



I love dogs!!
User user
876 Followers

user
****
User
user
14/06/2023
1
Are you a Pet Owner or a Pet Sitter?
Pet Owner
Pet Sitter
I love dogs!!

Project Limitations

Overall, we feel that we were able to complete most of the functionalities we wanted to implement in our application, but unfortunately there were a few things that we were not able to finish or implement.

One of these things was the feature of the step counter. But, to compensate we decided to implement a kilometer counter, for the user knows about the distance he has already traveled with the pet.

We also wanted to put the Home and Discover pages with database access instead of static, and give the user the opportunity to make a post, but due to lack of time we decided to focus on what were the main features of our application. We thought it would be easy to do, just follow the same process we used for other pages, but we couldn’t find the time to make sure the whole application had the database connection.

Certainly, with more time and if we have the opportunity in the future to continue improving our platform, we are committed to implementing the features in fault and providing the best possible experience for our users.

New features & changes for the future

We recognize the value that the features that we said before can bring to our platform and in the future, if we have time, we would like very much to implement them.

Another feature that we would like to implement is to integrate our local storage with Firebase. By doing so, we can take advantage of Firebase's real-time database capabilities, which will enable us to sync data across multiple devices and provide a better user experience.

It would also be interesting to implement a notification system. This will help keep pet sitters informed about important updates, such as when a pet owner makes a request or when they receive a message.

3 Conclusions and supporting resources

Lessons learned

Throughout the development of this app, we faced several major challenges and obstacles that required careful consideration and creative problem-solving. One of the biggest hurdles was designing a seamless user experience that balanced functionality with simplicity. We learned that it's important to prioritize features and functionality based on user needs and feedback, and to continually test and iterate on the app's design to ensure it meets those needs.

Integrating Bloc providers into our application was a challenging task and at first it takes a while to realize. It involves understanding the specific integration patterns and adapting the Bloc pattern to fit within the framework’s ecosystem. This includes handling widget lifecycles, managing dependencies, and effectively updating the UI in response to state changes. However, we think that we had the proper understanding and we were able to overcome the difficulties and, at the end, we can say that they helped us a lot.

We also learned a lot about the external features an app can have, namely integration with the Google Maps API, QR Code, Camera, etc, and about the value that these can add to a mobile application.

It is important to mention that from the beginning we were concerned about maintaining a whole structure and organization of the code so that, the more pages and the more code that was developed, we could maintain its easy comprehension, which helped us a lot until the end of the work.

Overall, we think that we have learned a lot from developing this application and have gained skills in flutter, to develop a much more complex mobile application in the future and with a longer development time.

Work distribution within the team

Taking into consideration the overall development of the project, the contribution of each team member is distributed as follow: Manuel Diaz did 50% of the work, and Leonardo Freitas contributed with 50%.

Manuel Diaz:

- QR Code integration
- Map integration with location routes
- Map integration for the pet tour
- Timer and distance counter

Leonardo Freitas:

- Gallery and Camera Integration
- Profile/ Request/Settings Page
- BLoc Logic
- Isar DB integration

Project resources

Resource:	**Available at:**
Code repository:	https://github.com/leofreitas99/PetSitter

Reference materials

Google Maps API's : https://console.cloud.google.com/google/maps-apis/api-list

https://pub.dev/packages/qr_code_scanner

https://pub.dev/packages/google_maps_flutter https://pub.dev/packages/location