UNIVERSITÀ DEGLI STUDI DI MILANO BICOCCA

ADOPTADOG

DOCUMENTATION

PROGRAMMAZIONE
DI DISPOSITIVI
MOBILI

RODRIGO LÓPEZ SÁNCHEZ

922906

r.lopezsanchez@campus.unimib.it

INDEX

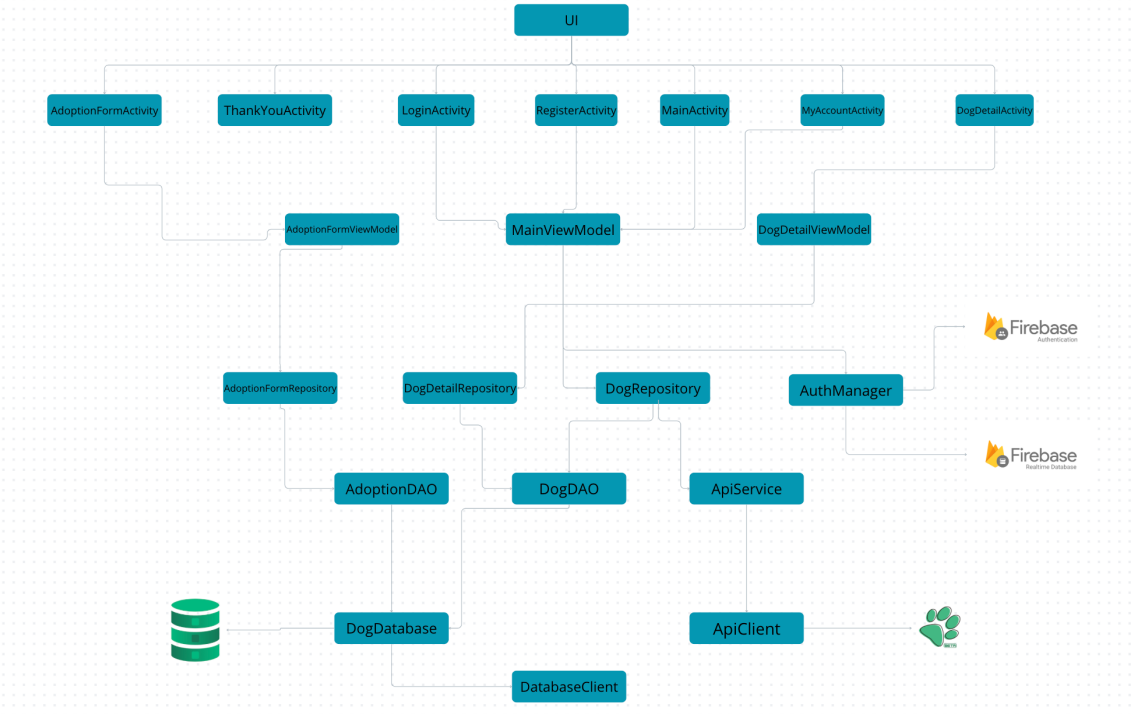
INTRODUCTION

AdoptADog is an Android mobile app designed to simplify the dog adoption process.
When the user enters the app, its greeted with an interface designed to make the user feel comfortable and have a satisfactory experience. He/she will be able to browse and filter by age and gender among the available dogs displayed in the dog list where basic information its displayed such as name, a short photo, a physical description, gender and age. If a dog is selected from the list, a larger photo of the dog, as well as its age, whether it is sterilized or not, a physical description and some personality characteristics will be shown. If the user is interested in one, a login or registration will be requested by the application.

Once logged in, the user can navigate to your account view, log out, or check the details of the account. Once inside, the account data is displayed and, if desired, delete it completely.

AdoptADog offers an **intuitive** and **immersive** interface for animal lovers who want to take the step of adopting a dog that is compatible with their lifestyle and needs.

ARCHITECTURE



The application is essentially composed of three layers:

UI (User Interface): This layer, through the use of activities (Activity), adapters (Adapter), and graphical elements, defines the user interface to interact with the application's various features, such as viewing dogs, adoption forms, and managing user accounts. The data displayed in this layer comes directly from the ViewModel, ensuring a clear separation between presentation logic and data. For instance, MainActivity and DogAdapter handle the display of the dog list, while activities like LoginActivity and RegisterActivity manage user login and registration.

ViewModel: It acts as an intermediary layer between the UI and the repositories. The ViewModel (in this case, MainViewModel) retrieves data from the repositories and exposes it through LiveData. The UI observes these data changes and updates automatically when modifications occur. This ensures a smooth user experience and a decoupled architecture, as the UI does not need to interact directly with data sources such as Firebase or Room.

Repository: DogRepository interacts with the local database (Room) and the external API (Huachitos) to synchronize and provide updated data. Additionally, AuthManager acts as a repository for operations related to Firebase Authentication, while the classes in the Firebase and Database folders handle interactions with Firebase Realtime Database and Room, respectively.

EXTERNAL SERVICES USED

Firebase Authentication: Used for user authentication via email and password. This service ensures secure login and registration processes, primarily handled by the AuthManager, LoginActivity, and RegisterActivity classes.

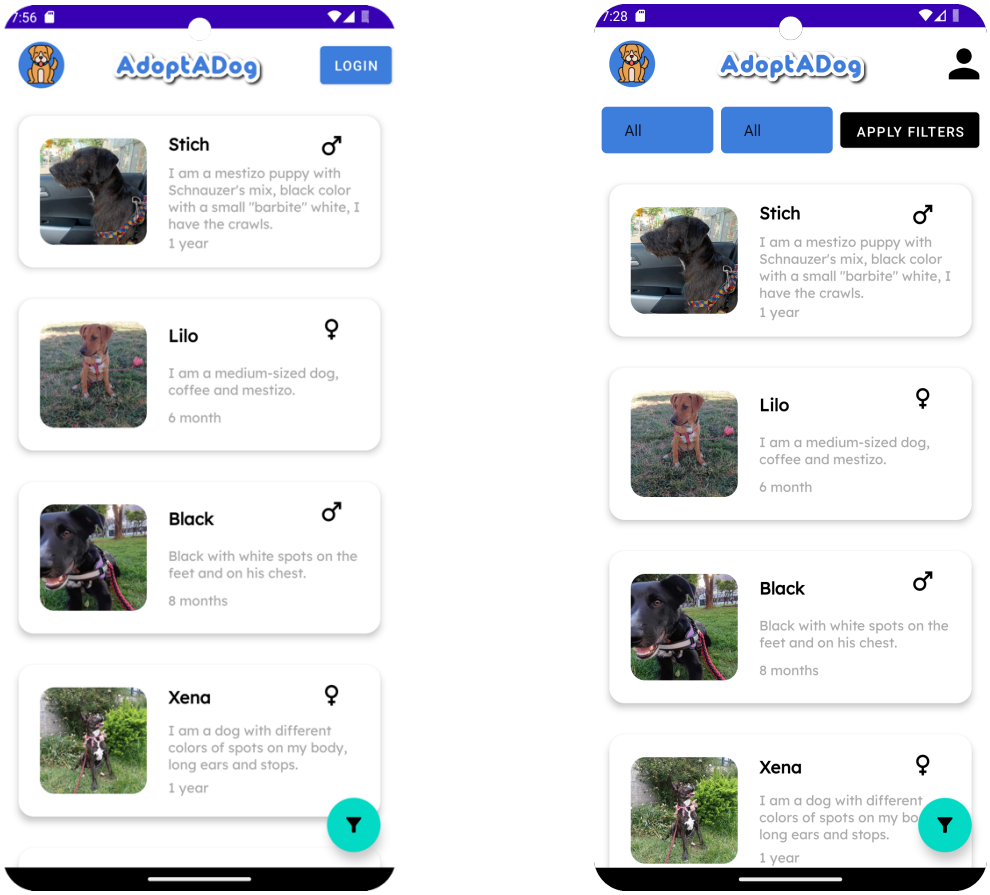
Firebase Realtime Database: Serves as a real-time database to store user-related information and relevant data. For example, properties such as adoption forms and other user preferences are managed in this database.

Room Database: A local database used to cache information about dogs available for adoption, ensuring data is accessible even without an internet connection. This is implemented using DogDAO, DogDatabase, and DatabaseClient.

API Huachitos: An external API used to retrieve information about adoptable dogs, such as their breed, age, gender, and status. The retrieved data is processed through ApiClient and ApiService, then stored or synchronized with Room.

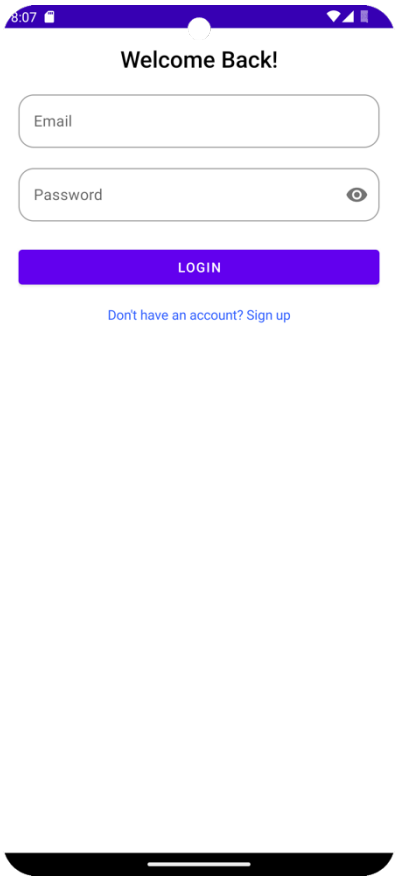
THE APPLICATION

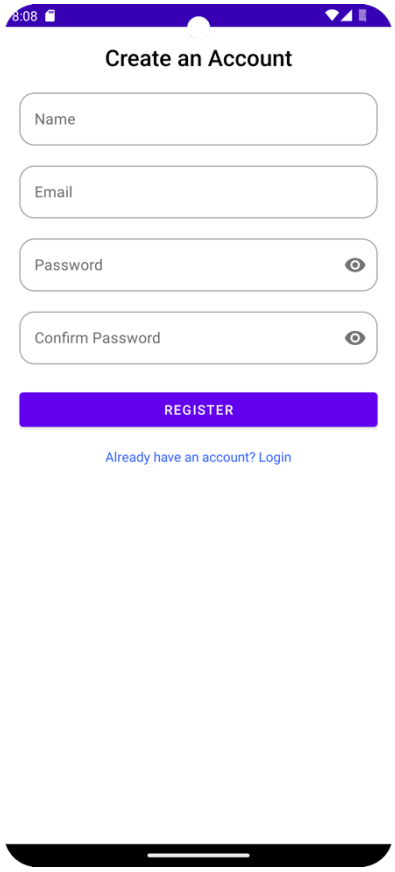
HOME PAGE



When the user enters the application, the first view he/she encounters is the main view. Here, he or she can browse the available dogs, where basic data such as their name, photo, physical description, gender and age are provided. By scrolling down, it's possible to click on the central AdoptADog logo to return to the top of the list in an intuitive way. If it's desired to filter the dogs to be more specific, the user can click on the button located in the lower right corner of the screen to activate the filters and modify them according to his or her preference. By clicking on the upper right corner, the login view will be displayed.

LOGIN/REGISTER





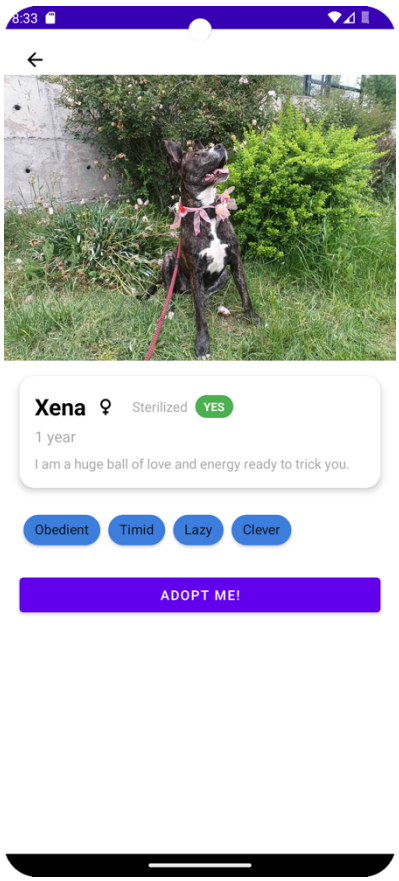
LOGIN

When the user clicks on the login button, he/she will be redirected to the corresponding view. There, he/she can log in by filling in the inputs with his/her account data, or, otherwise, he/she must click on “Don’t have an account? Sign up”
Which will take him/her to the registration view.

REGISTER

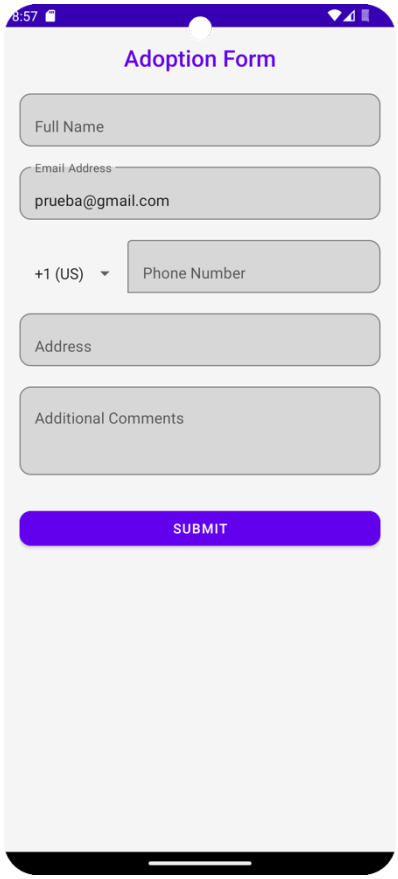
Once in the registration view, the user must fill in the inputs for the registration to be correct. The data will be validated, none of the inputs will be empty and each data have to be apt (for example, the email has an @). After the correct registration, the user will be redirected to the view where the list of dogs is located, and the login button will have changed to an icon of a person, which by clicking on it the user will be able to go to the My Account section or log out.

DOG DETAILS



Clicking on a dog in the list will take the user to the details view. In this view, the user will be able to see the photo of the dog, its name, its gender, and, as a novelty, whether it is spayed or not, as well as its personality description, something very important when adopting a pet. Below the dog, several chips will be seen where some specific characteristics about the dog will be shown (purely visual). Finally, there is an adoption button, which will act depending on whether the user is logged in or not. If the user is not logged in, he will be redirected to the login view, so he can do so or otherwise, register.
If the user is logged in, he will be redirected to the adoption form view.

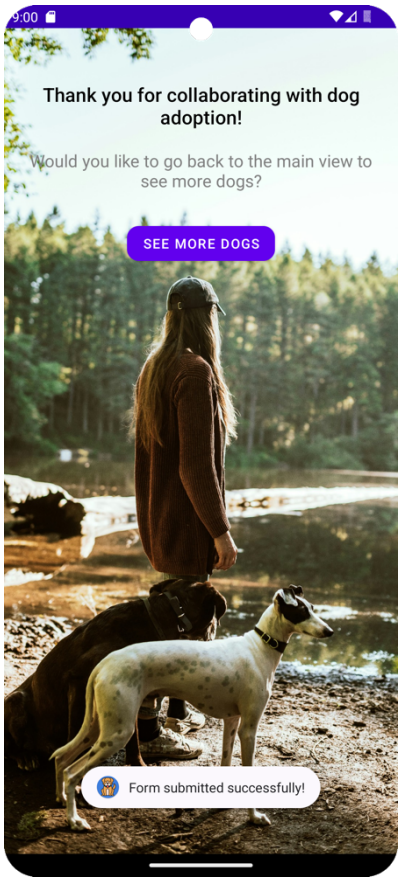
ADOPTION FORM



When the user is logged in and enters a dog, he can click on the adoption button. In this case, he will be redirected to the adoption form view. In it, he must fill in the inputs correctly. The email will be auto filled with his address, so he will not have to touch anything in that field. He will be asked for his name, email (auto-filled), phone number, where there is a selector to select the country code, his address and a space for additional comments.

If the user correctly fills out the form, it will be sent with date and time (DD/MM/YYYY) to the database, and he will be redirected to the thank you view.

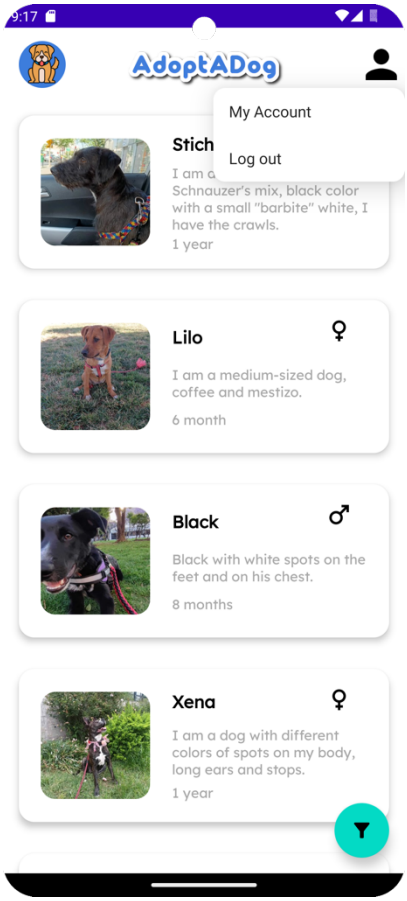
THANKS VIEW

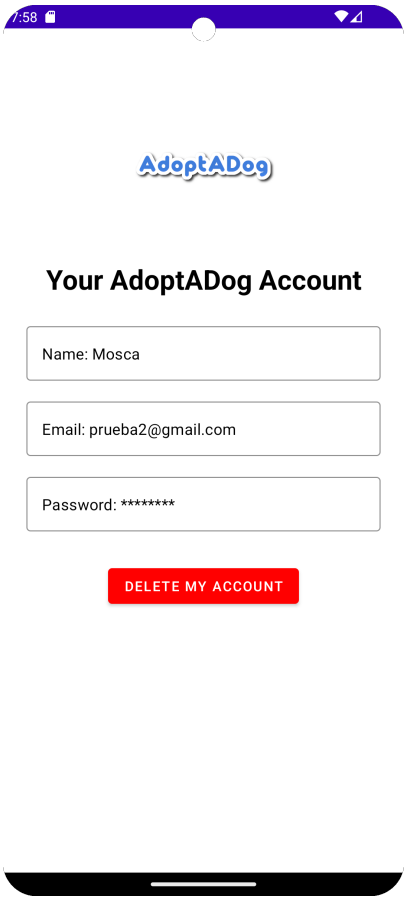


If the form has been submitted successfully, the user will be redirected to the thank you view, where he will be symbolically thanked for completing the form. Then, he will be asked if he would like to return to the dog list to see more via a button that will redirect to the list.

SECONDARY VIEWS

MY ACCOUNT VIEW





When the user is logged in and clicks on the person icon, a popup menu will be displayed where the user can click My Account or Log out.

If My Account it's clicked, the user will be taken to My Account view where the account details are shown. Below that, there is a button which, if it's clicked, will ask if the user wants to delete his account. If accept is clicked, the account will be deleted from the database and the application will redirect the user to the dog list. The login button will be displayed again.

POSSIBLE FUTURE IMPROVEMENTS

-A mailer could be used, that is, a provider through which when the user registers and/or correctly fills out the adoption form for a dog, emails would be sent to them.

-The application could be refactored into AdoptAnAnimal, to include more animals, not just dogs

-A proprietary API could be created to customize the application and thus make it more realistic.

-An administrator panel could be created to add animals to the list and control the forms and registered users.

-The views could be improved, with a more modern design.

NOTES

-The API used is a free-to-use public API. All data is in Spanish, so it was necessary to use Google ML Kit to translate several texts (physical and personality characteristics, age...) so there may be some incorrect texts.

-As it is a public API which is updated every day, certain animals that were not dogs had to be removed from the list (MainViewModel class line 42, “Popeye” and “Karey” were cats)

-The documentation has been written in a language that is not my native language, so there may be some errors.