



POLITECNICO
MILANO 1863

Design and Implementation of Mobile Applications
“Design Document”



“Homie”

Professor: Luciano Baresi

Group members:

Parniya Saeedzadeh (Person code:10740816)

Saman Fekri (Person code: 10694432)

July, 2021

Table of Contents

Introduction	3
1.1 Purpose	3
1.2 Homie scope	3
1.3 General milestones.....	4
1.4 Platform.....	4
1.5 Definition, acronyms, abbreviations	4
1.5.1 Definition	4
1.5.2 Acronyms	4
1.5.3 Abbreviations	4
1.6 Revision history	5
1.7 References	5
1.8 Document structure	5
1.9 Document version.....	5
Requirements	6
Use cases analysis	7
3.1 Use cases	7
Architectural Design	10
4.1 Component View	10
4.2 Runtime View.....	13
4.3 Architectural styles and patterns.....	16
4.4 External services and Libraries	17
4.5 Design Decision	19
User Interface Design	20

1.Introduction

1.1 purpose

The purpose of this document is to provide an overall guidance to the architecture of the mobile application software product concerning Homie, an application created and developed for people who are searching for both apartments and flat mates. In this document, the following topics are discussed:

- Main components such as classes;
- Runtime behavior;
- Employed Design Patterns;
- libraries;
- User interfaces;

1.2 Homie Scope

The purpose of this application is to connect people who want to rent an apartment with searching for other people to give their bed/room rented. You can make new advertisement whenever you want but just one advertisement at a time and not more. By tapping the Home button, you will be directed to a page in which you can see all advertisement plus some ad suggestions that app will give you. By clicking on each advertisement you will be lead to detailed page about all the features that the apartment has and you can send a request to the owner. and by accepting through owners account you will be in contact with him/her personally. In detail, the functionalities offered by Homie are the following:

1. create an account;
2. possibility to make new advertisement, one at a time;
3. search for the flat mate and the apartment;
4. manage your requests;
5. contact with the owner or the one who is searching personally without any third parties;
6. update your profile;

1.3 General Milestones

Our application has lots of characteristics of usability. The user interface is easy to use, so that the user quickly learns what to do. The design has the best experience possible.

In detail, our mobile reaches these milestones:

- **Usability:** the main actor of the system is the end user. For this reason we decided to make the user interface as easy as possible, but still keeping all the functionalities needed to provide the best user experience.
- **Nice User Interface:** we tried to make our application as nice as possible, following Google's Material Design guidelines to have a clean and simple design.

1.4 Platform

The choice of the platform was left to us. We decided to develop for Android, choosing Android framework because with **Android** Studio native **Android** applications can be created which preferable **better** features over the applications have created with cross platforms.

1.5 Definitions, Acronyms, Abbreviations

1.5.1 Definitions

- User: a generic user of Homie application.

1.5.2 Acronyms

- DD: Design Document
- VM: View Model
- AD: Advertisement
- DB: Database
- DBMS: Database Management System

1.5.3 Abbreviations

- APP = mobile application

- INFO = information

1.6 Revision history

- Version 1.0: First release

1.7 References

- www.draw.io: for all the diagrams
- Adobe XD for UI designs

1.8 Document Structure

Chapter 1: Here an introduction of the design document is given. This chapter contains the purpose and the scope of the document, as well as some details on used terms in order to provide a better understanding.

Chapter 2: this chapter specifies the Homie application requirements and use cases. All the functional requirements of the application are represented here.

Chapter 3: this chapter describes two examples of use cases of the Homie application.

Chapter 4: this chapter is about the architectural design of the application. It gives an overview of the architecture and it also contains the most relevant architecture views:

- Component view;
- Runtime view;
- Architectural Styles and Patterns;
- Libraries used;
- Design Decisions.

Chapter 5: this chapter specifies the user interface design. The real screenshots of Homie application are presented here.

1.9 Document Version

Version 1.0

2.Requirements

In this section we present the requirements necessary to the correct behavior of the system.

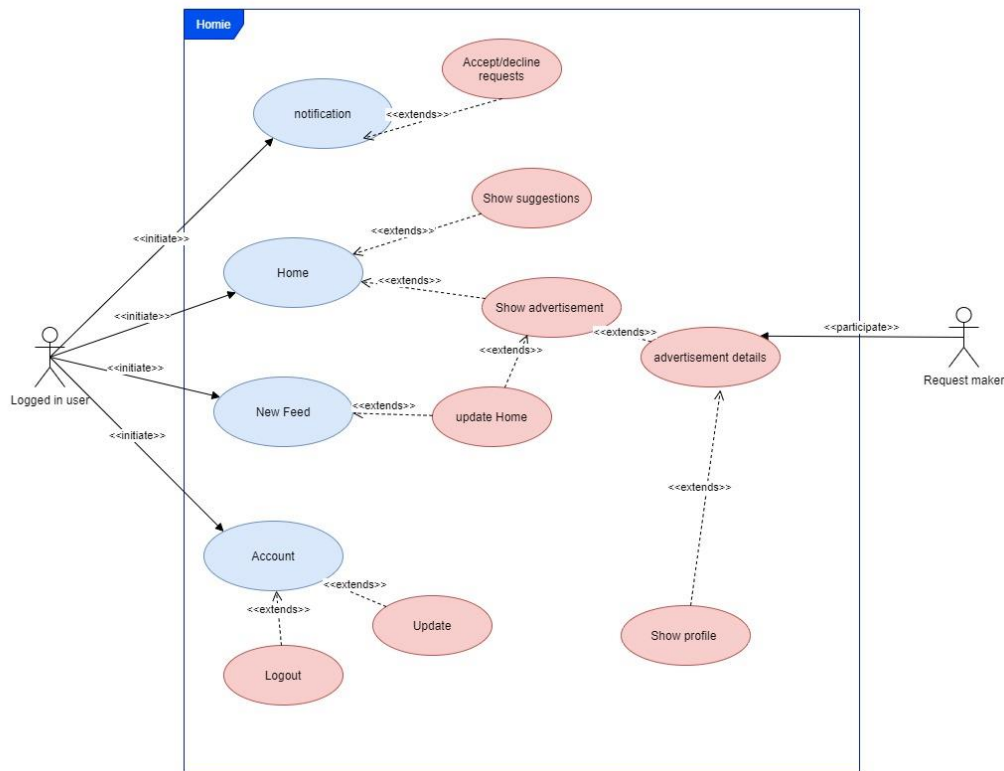
- Since the application is for both national and especially international students, we implemented it in English.
- The application has to start with a splash activity, meanwhile the initial settings are done.
- The application needs to allow the user to create an account.
- The application has to provide immediate access to all features of the application through a navigation bar at the bottom of the screen.
- On the first run, the app should introduce the application features to the users through splash screens and the possibility to login/sign-up.
- The app has to provide users the possibility to see their personal information to choose the perfect flat mate match for themselves.
- The app has to provide users to see some advertisement suggestions according to their preferences in the Home page
- The app has to provide users the possibility to upload their advertisement as an owner.
- The app has to provide users the possibility to see all information about both people including personal habits, picture and etc. and also the apartment properties such as price, and any amenities that it has.
- The app has to provide users the possibility to send a request to the advertisement owner.
- The app has to provide users the possibility to accept the request of people who search for the apartments.
- The app has to provide users the possibility to accept or decline other users' requests.
- Account page should permit users to logout or delete account if logged in.
- Authentication activity should allow users to register or login using email and password.

3.Use Cases Analysis

3.1 Use Cases

In the use case diagrams, we have distinguished two specific cases of usage of the app: one is when the user is logged in using an account, one when is not logged in and wants to register.

Use case: User Logged in



Name: Home

Actor: User

Entry conditions: The user has tapped on the “Home” on his/her navbar.

Events Flow:

- the user sees all the advertisements available
- the user sees some suggestions
- the user taps on an advertisement
- he/she can see more details about the tapped advertisement on advertisement page
- he can tap on the profile picture of the owner and see his profile

- he will press the button to come back to the advertisement page
- since the advertisement is what the user wants, he taps on “request” button to have contact with the owner

Exceptions: The device has no internet connection

Name: New Feed

Actor: User

Entry conditions: The user has tapped on the “New Feed” icon

Events Flow:

- The user fills out the information about his advertisement and add his home
- User can see his advertisement on Home page

Exceptions: The device has no internet connection

Name: Account

Actor: User

Entry conditions: The user has tapped on the “Account” on his/her navbar.

Events Flow:

- The user updates his personal information
- The user logout from his account

Exceptions: The device has no internet connection

Name: Notification

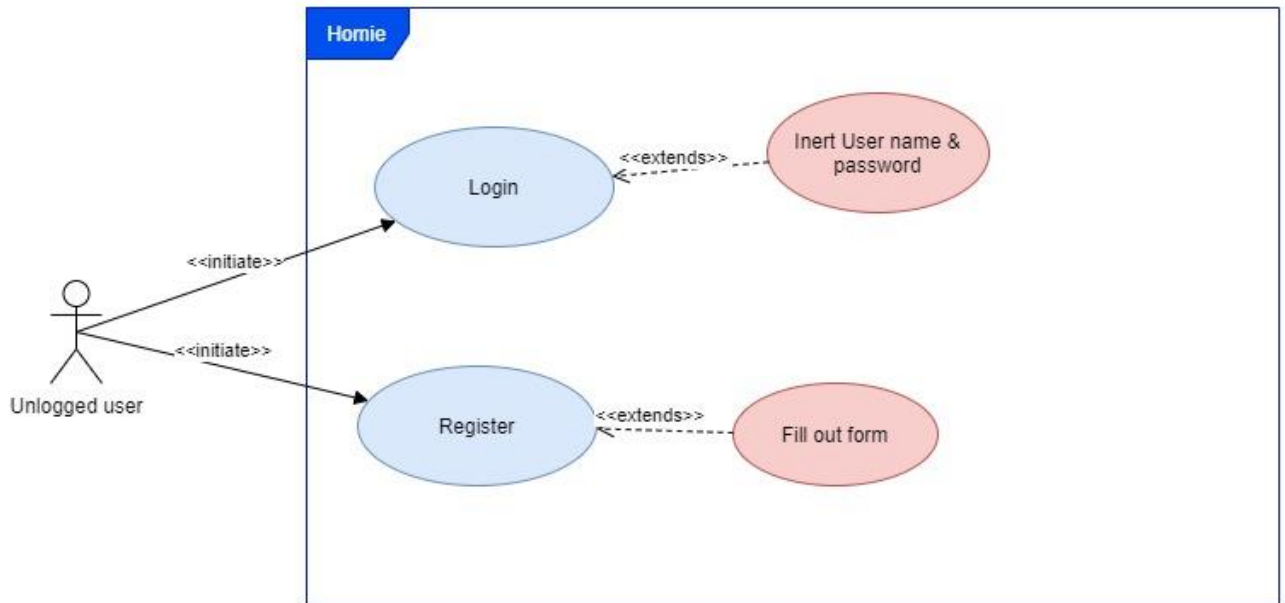
Actor: User

Entry conditions: The user has tapped on the “Notification” on his/her navbar.

Events Flow: The user accepts or decline the request

Exceptions: The device has no internet connection

User registers:



Name: Login

Actor: user

Entry conditions: The user has tapped on the “login” button on welcome page

Events Flow: The user enters his username and password

Exceptions: The device has no internet connection

Name: Register

Actor: user

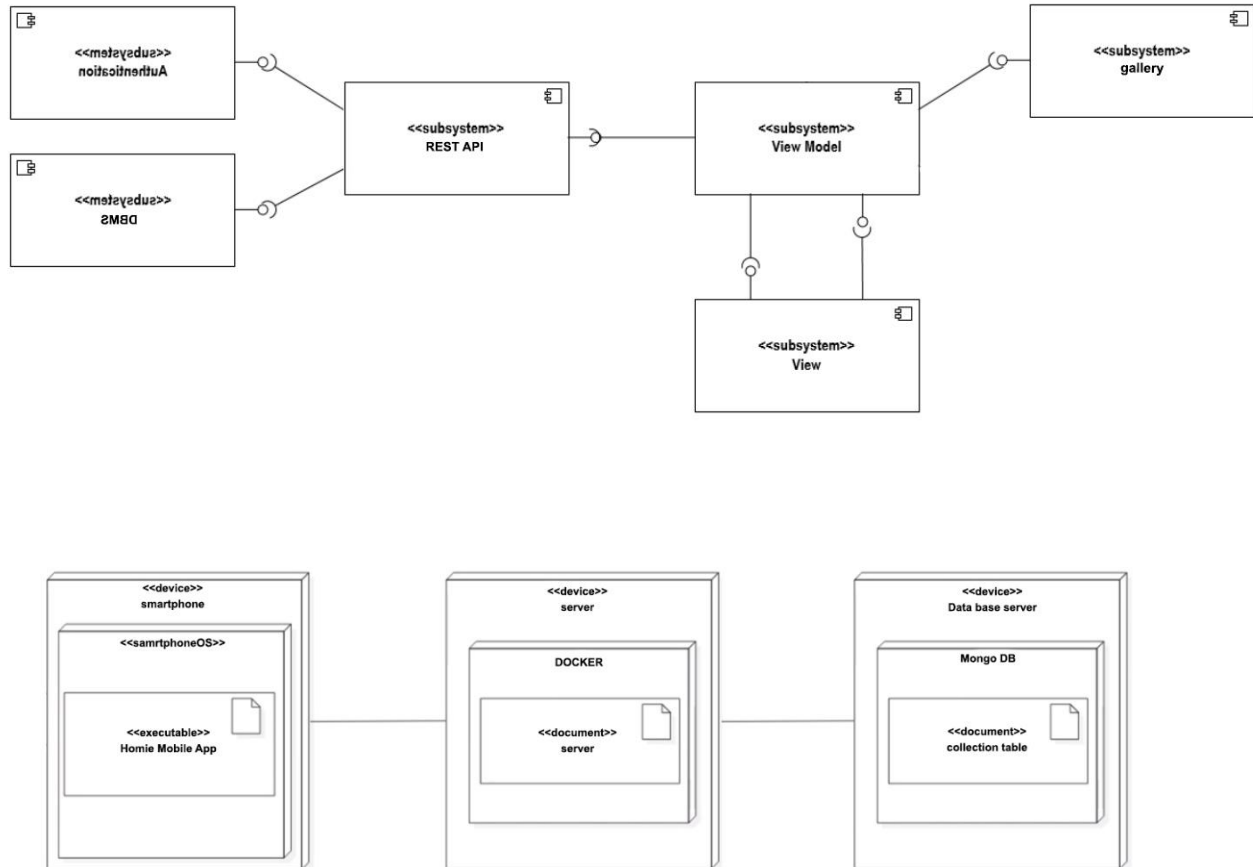
Entry conditions: The user has tapped on the “register” button on welcome page

Events Flow: The user fills out a form about his info.

Exceptions: The device has no internet connection

4.Architectural Design

4.1 Component view



The component diagram, described in the figure above, represents the structure of our mobile application.

Let's see in detail each component of the component diagram:

Mobile App

Looking at the architecture design, at the first we can see the MVVM structure. As stated ViewModels are usually assigned to a View component. You can see we usually update the ViewModel and then observe on the change of them. Also, we bind some ViewModels to the View so some part of view will change automatically after we change the view model.

ProfileViewModel

This component is connected to the ProfileFragment at the first it sends a Get request by RetrofitClientInstance to the server and will fetch data to the proper fields. After someone clicked on profile picture It will use gallery to take the picture from the user. Also, if someone click on update this component sends the updated value to the server to update the user profile.

HomeViewModel

This component is connected to the HomeFragment at the first it sends a Get request by RetrofitClientInstance to the server and will fetch data all homes data to the vertical recyclerview and suggested homes to the horizontal recyclerview.

HomeAdViewModel

This component is connected to the HomeAdFragment at the first it sends a Get request by RetrofitClientInstance to the server and will fetch detailed data of home to the proper fields. besides, if you click on the profile picture or name of user it will switch you to the profile of that user. At the bottom of the page you will see a “Request” button. If you click on that button it will sends a request to the other user.

LoginViewModel

This component is connected to the LoginFragment, it will bind the proper fields to the proper variable. if you click on login button it will send your data to the server by RetrofitClientInstance if you logged in successfully you will be redirect to the MainFragment. In addition, there is signup button when you click on it you will be transfer to the RegisterFragment.

NotificationViewModel

This component is connected to the NotificationFragment at the first it sends a Get request by RetrofitClientInstance to the server and will get all request other people sends to you. you can accept or reject their request by tick and trash button, In addition if you click on the profile picture of request maker you will be transfer to their profile.

ProfileOtherViewModel

This component is connected to the ProfileFragment at the first it sends a Get request by RetrofitClientInstance to the server and fill the fields by proper data. if you click on the “to the house” button it will redirect you their HomeAdFragment of them.

RegisterViewModel

This component is connected to the RegisterFragment, it will bind the proper fields to the proper variable. After someone clicked on profile picture It will use gallery to take the picture from the user. if you click on register button it will send your data to the server by RetrofitClientInstance if you register successfully you will be redirect to the LoginFragment. In addition, there is login button when you click on it you will be transfer to the LoginFragment.

WizardViewModel

This component is connected to the all the fragment and stepper adapter, it will bind the proper fields to the proper variable. the stepper adapter will control your step through the wizard. At the last fragment you could select multiple photos. if you select multiple photos from gallery it will be loaded on slider. At the end, when you click on the complete. if that user doesn't have a home in the server we create a home that user have a home we update the home values.

RetrofitClientInstance

As you can see above, this component connects us to the REST API which is an external service for us. This component mainly shows the routes and fields of HTTP calls. Retrofit is one of the famous libraries for http call in android.

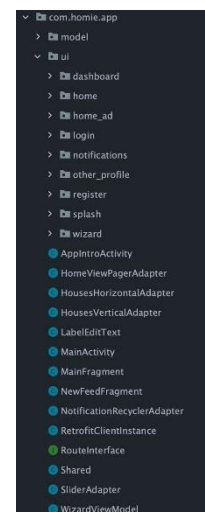
Gallery

This component is a built-in component in android and you can use Intents and ask this component to get you the pictures in the gallery of the phone.

Server

This component is an external component that we build for the purpose of this application. It is a HTTP server based on node.js and on the docker container. the authenticaor and DBMS are connected to this component.

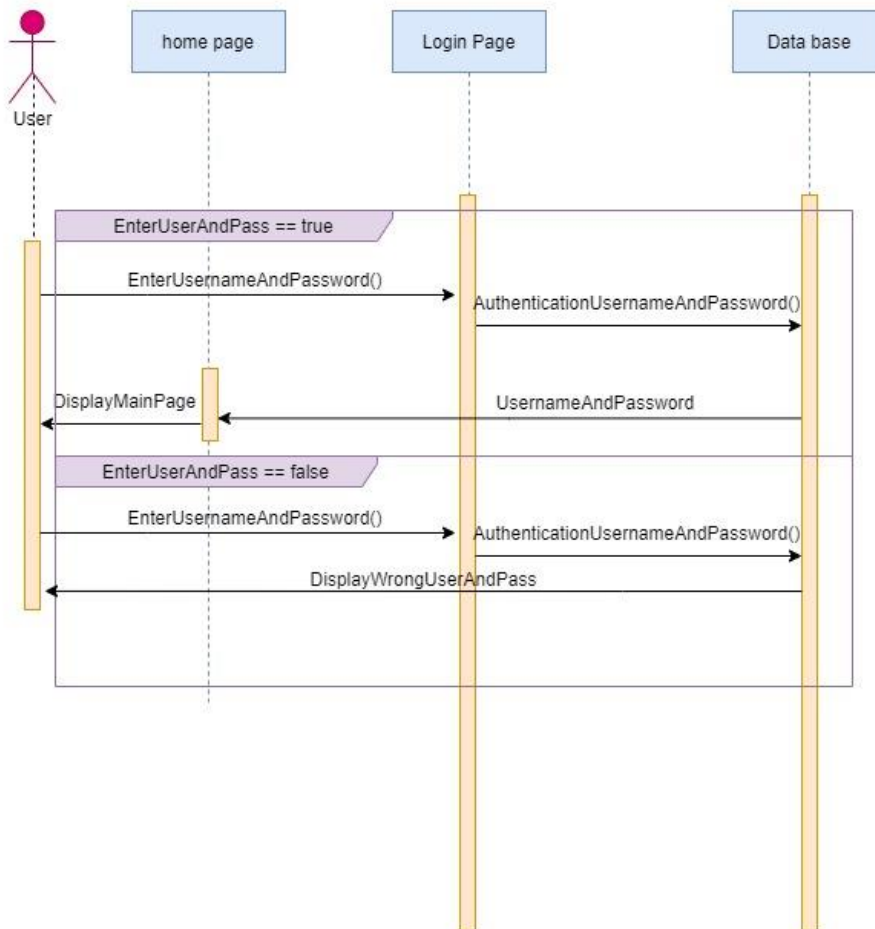
Here's how our project files are organized:



4.2 Runtime View

Below are the sequence diagrams analyzing in detail the interaction between the different components and showing the messages that are exchanged between them

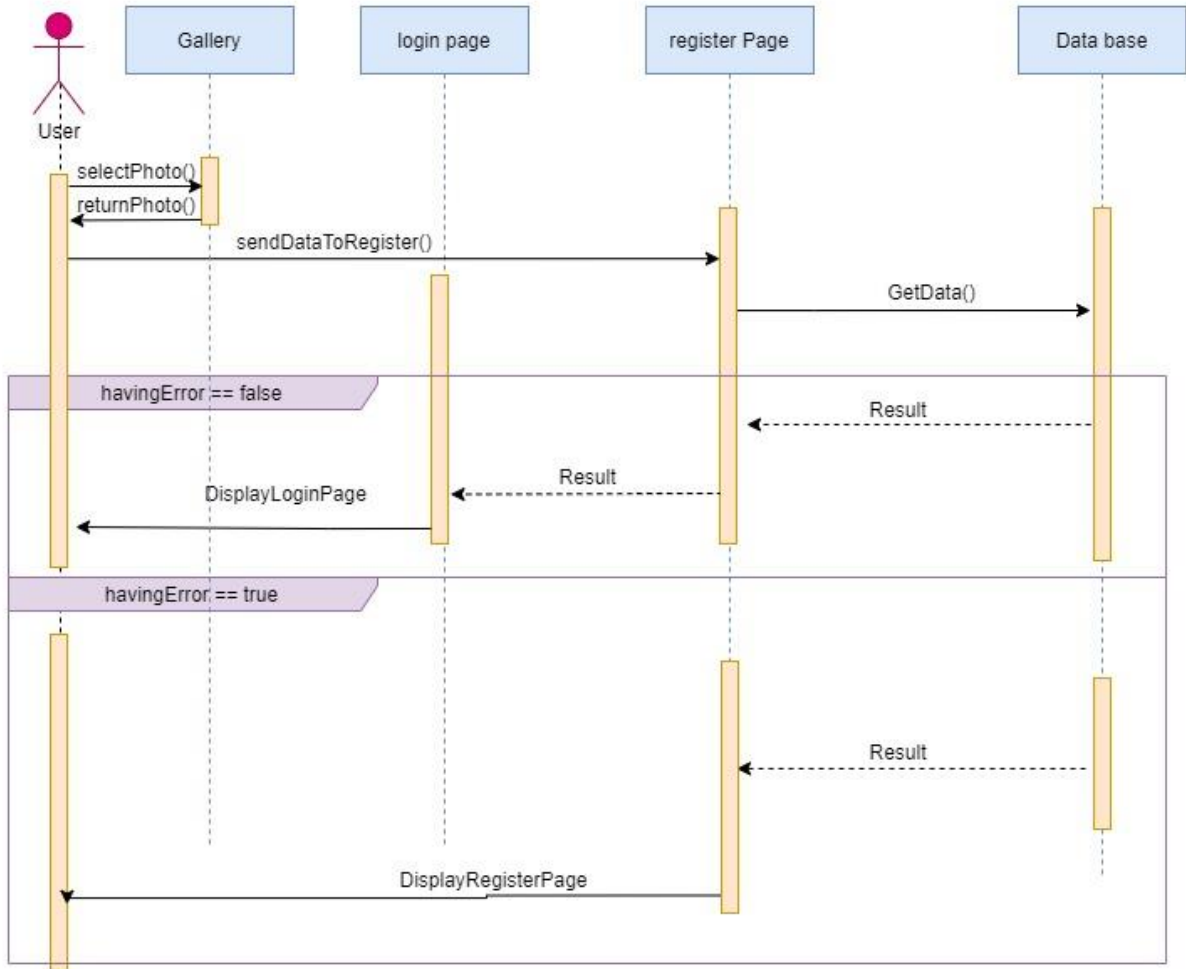
Login:



Once the login page is displayed to the user, we implement a true or false loop in which, the user firstly enters his username and password. And by calling it through our database, they will be authorized and if they are true, the data base sends them to the Home page and finally display that to the user.

But if the information sent by the user was not correct, the authentication fails and finally show the login page again to the user again.

Register:

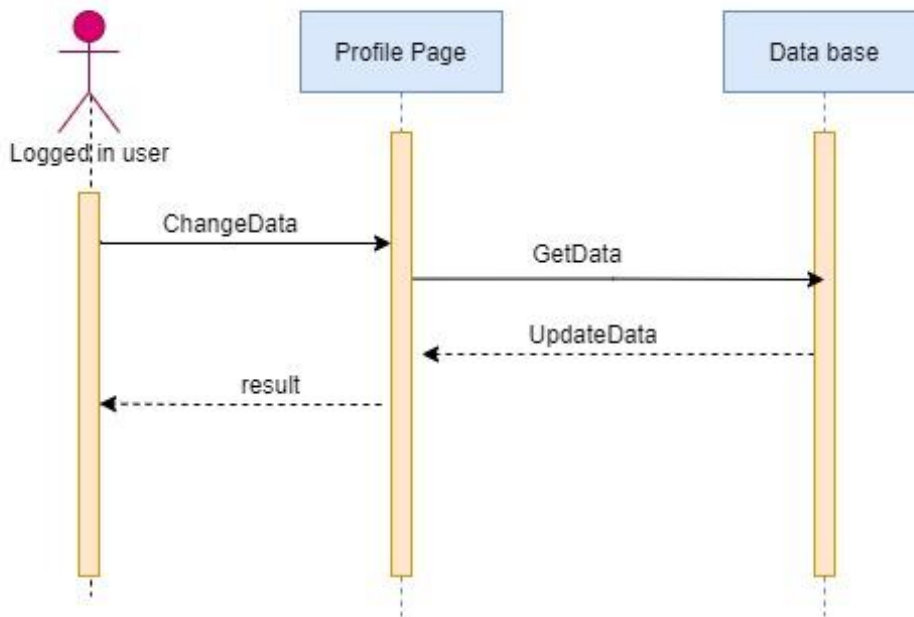


In the register part we imagined that user for instance select a photo from gallery and then this data which is a picture will send to the register page and then the data base will be called and will get the data.

Now we will be in a loop in which if we are not going to have any error happens, the result will be return to register and then login page and finally the login page appear to the user.

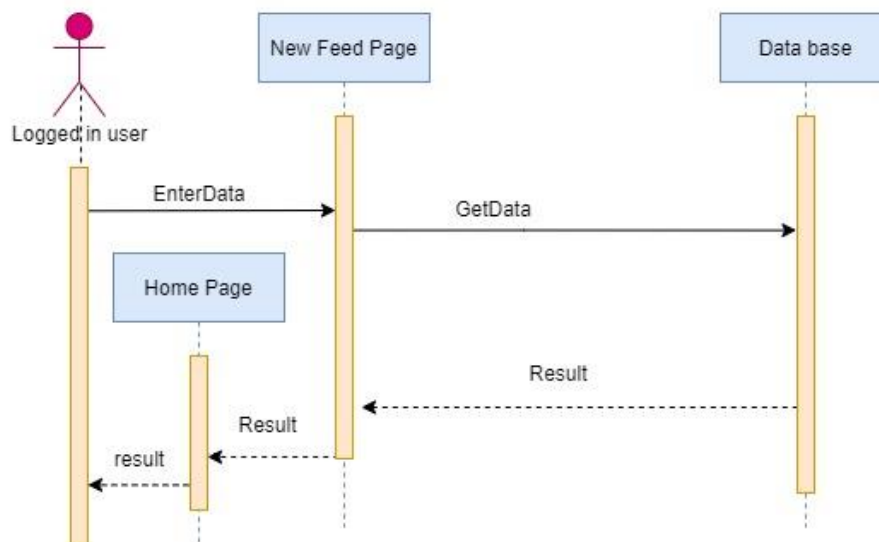
But if we have any error happen during registration, once the database get the data and understand that we have any problem, the result will be return to the register page and the register page will be shown to the user again.

Update Profile:



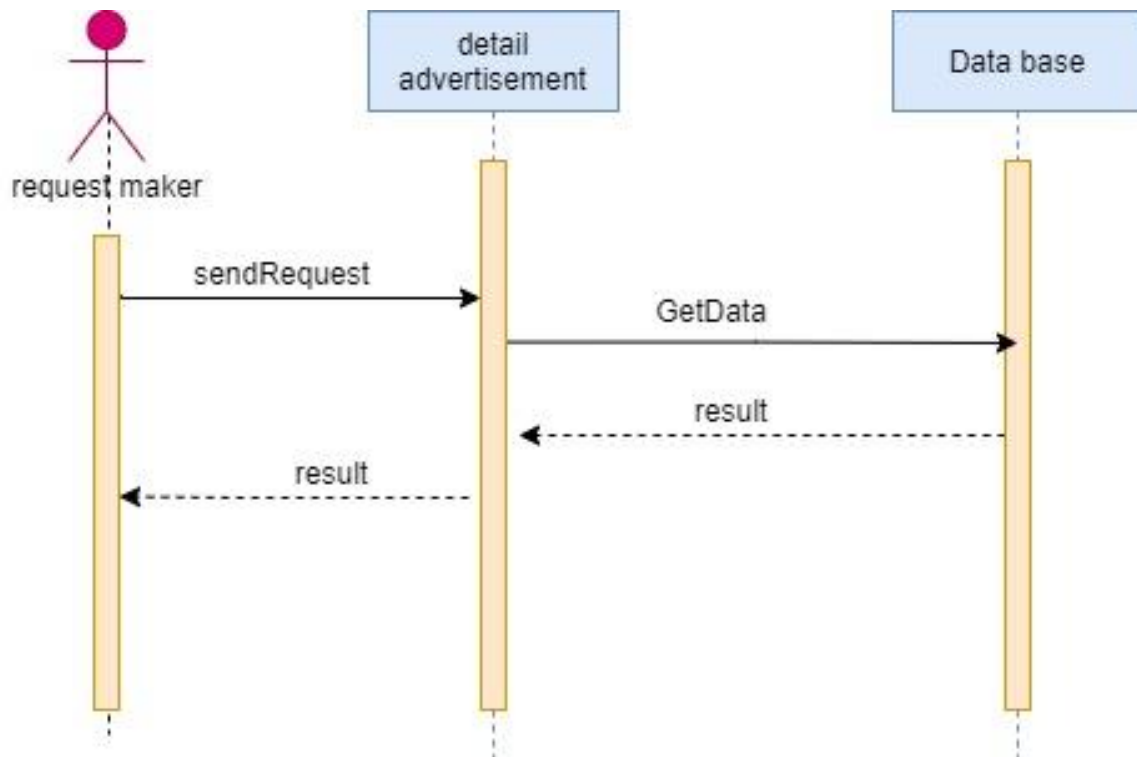
The user wants to update any data in his profile account. Once he changed data in the profile page, the database will get the data and will send the updated data to the profile page and show the result to the user.

Get Home/New Feed:



As soon as the user enter any data into the new feed page, database will get the data and put the result in the new feed and then into the home page and finally send it to the user to display.

Send Request:



The request maker sends a request in the page of detail advertisement and immediately after, the database has been called the data from the advertisement page. And then the result came back to the page of advertisement first and then to the request maker. So that the request in which the user sends has done in this way.

4.3 Architectural styles and patterns

The style and pattern of this app is MVVM. As in the model–view–controller (MVC) and model–view–presenter (MVP) patterns, the view is the structure, layout, and appearance of what a user sees on the screen. It displays a representation of the model and receives the user's interaction with the view (mouse clicks, keyboard input, screen tap gestures, etc.), and

it forwards the handling of these to the view model via the data binding (properties, event callbacks, etc.) that is defined to link the view and view model.

The binder frees the developer from being obliged to write boiler-plate logic to synchronize the view model and view. When implemented outside of the Microsoft stack, the presence of a declarative data binding technology is what makes this pattern possible, and without a binder, one would typically use MVP or MVC instead and have to write more boilerplate (or generate it with some other tool).

The MVVM pattern attempts to gain both advantages of separation of functional development provided by MVC, while leveraging the advantages of data bindings and the framework by binding data as close to the pure application model as possible.

4.4 External services and libraries

for the external services, what we used is a server in which it does have this responsibility to maintain our data. so this an external service of our since it is not in our app and we ae using the external service that we provided.

For the libraries our application is using these libraries which are mentioned below. Some of these libraries are for our user interfaces, some of them are for out data bindings and some of our http requests that we send. However, we commented the codes so that each and every part is obvious and clear that is for what section.

...

// using androidx and material elements

implementation 'androidx.appcompat:appcompat:1.0.2'

implementation 'androidx.constraintlayout:constraintlayout:1.1.3'

implementation 'androidx.cardview:cardview:1.0.0'

implementation 'androidx.recyclerview:recyclerview:1.1.0'

implementation 'com.google.android.material:material:1.3.0'

implementation 'androidx.navigation:navigation-fragment:2.3.5'

implementation 'androidx.navigation:navigation-ui:2.3.5'

implementation 'androidx.lifecycle:lifecycle-livedata-ktx:2.3.1'

implementation 'androidx.lifecycle:lifecycle-viewmodel-ktx:2.3.1'

implementation 'androidx.legacy:legacy-support-v4:1.0.0'

```
testImplementation 'junit:junit:4.12'
androidTestImplementation 'androidx.test.runner:1.1.1'
androidTestImplementation 'androidx.test.espresso:espresso-core:3.1.1'

// adding glide
implementation 'com.github.bumptech.glide:glide:4.12.0'
annotationProcessor 'com.github.bumptech.glide:compiler:4.12.0'

// app intro
// AndroidX Capable version
implementation 'com.github.AppIntro:AppIntro:6.1.0'
// *** OR ***
// Latest version compatible with the old Support Library
implementation 'com.github.AppIntro:AppIntro:6.1.0'

// add material design
implementation 'com.google.android.material:material:1.4.0-beta01'

// slider library
implementation 'com.github.smarteist:autoimageslider:1.4.0'

//stepper library
implementation 'com.stepstone.stepper:material-stepper:4.3.1'

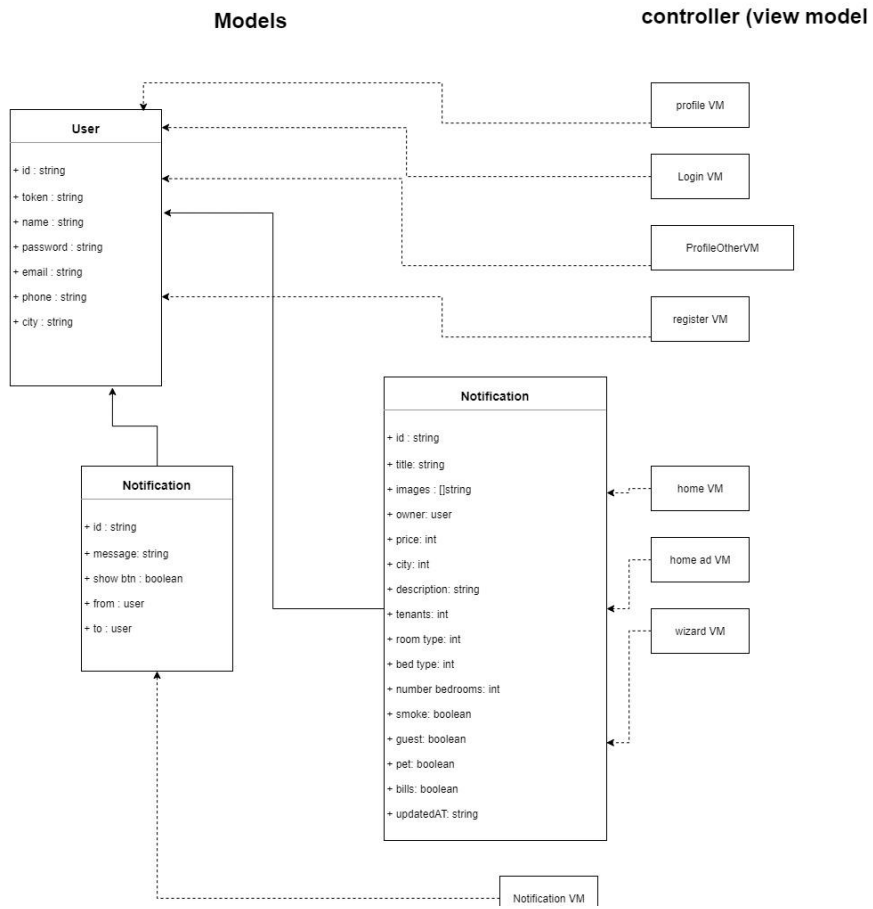
// data binding
implementation 'androidx.databinding:databinding-compiler:3.2.1'
implementation 'androidx.lifecycle:lifecycle-extensions:2.1.0'
```

```
// permission For developers using AndroidX in their applications  
implementation 'pub.devrel:easypermissions:3.0.0'  
  
// retrofit  
implementation 'com.squareup.retrofit2:retrofit:2.9.0'  
implementation 'com.squareup.retrofit2:converter-gson:2.9.0'  
implementation 'com.squareup.retrofit2:converter-scalars:2.0.1'  
implementation 'com.squareup.okhttp3:logging-interceptor:4.8.1'  
...
```

4.5 Design Decisions

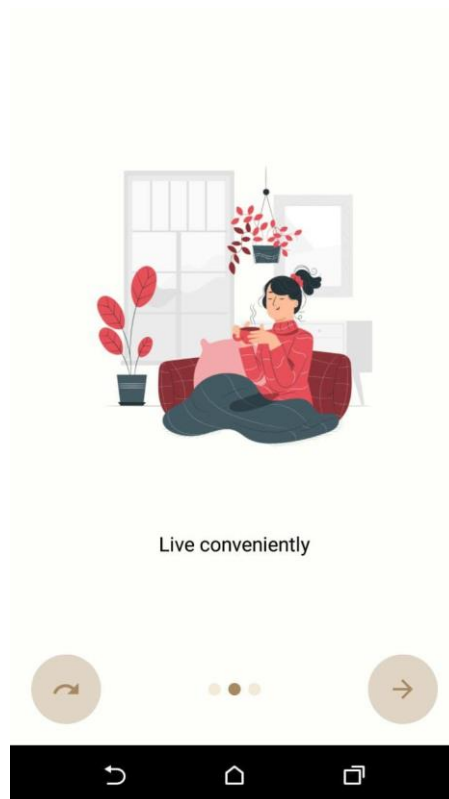
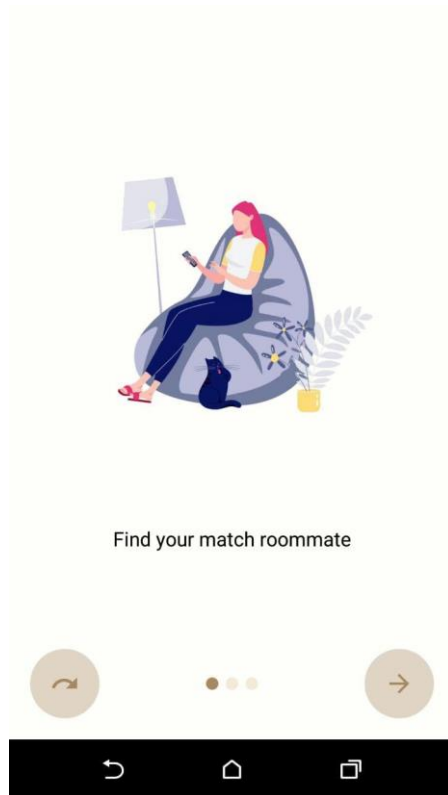
In this part we are going to mention the class diagrams where we put the models and the view models in which control the models.

The diagram is the following:

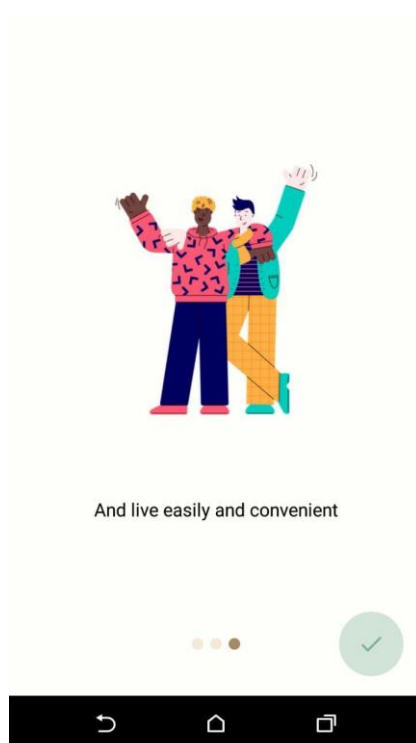


5. User Interface Design

Splash Screens



Here we can find the splash

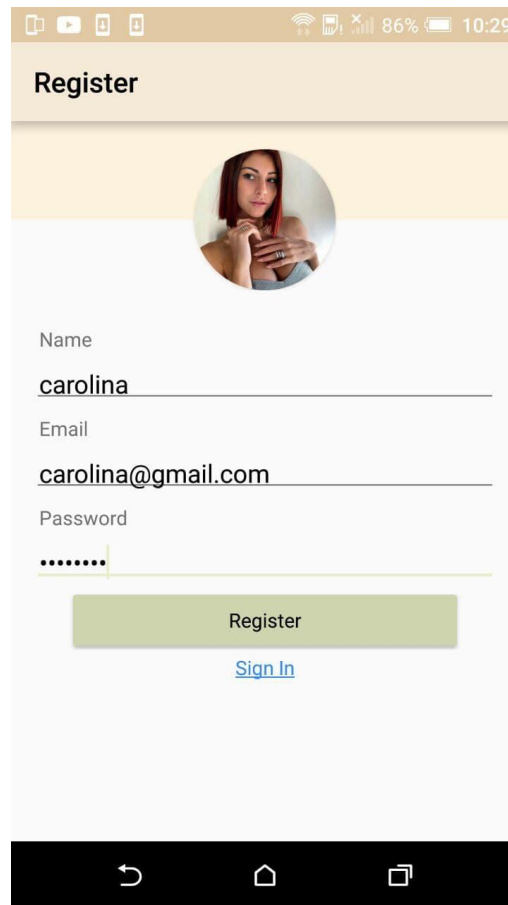
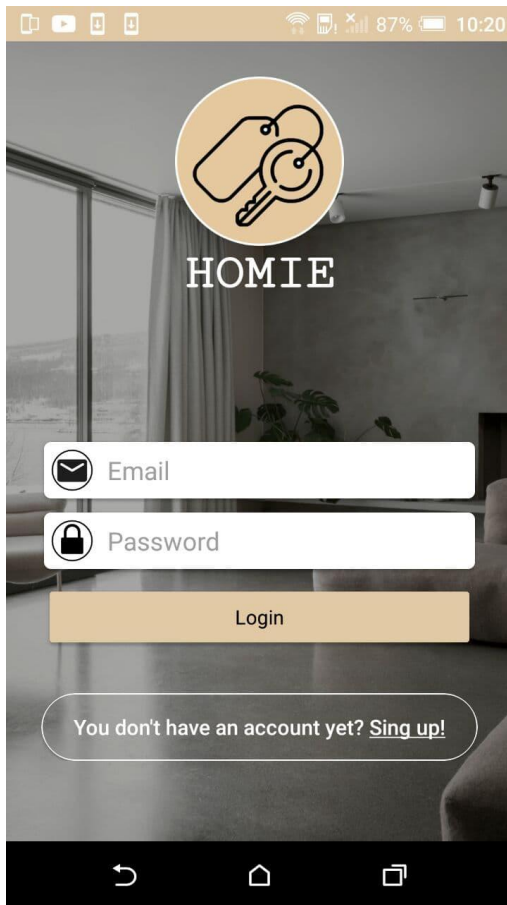


screens which

appears before someone registers for this app.


It helps users to see what are the main features of our app and motivate them to go further to register and be part of HOMIE communication.

Login/Register



Here you can find the welcome page which has the button for login for people who have already signed up in the application. And also there is a button at the bottom which mentions for signing up. If you have not signed up yet you have to tap on that to register.

Profile







YOUR FULL NAME
carolina




YOUR EMAIL
carolina@gmail.com

PHONE NUMBER
3665472148

CITY
Milan

Update 

Profile


By clicking on the register button in the previous page, you will lead to the register page which u can see here. In this page the user should put his/her personal information such as name, email and set a password for his account.




The information such as name and email will appear on his profile page. In addition, there are some other info that they can set to show on their profile account and we will see in the next designs.

Make new advertisement
New Feed

Choose a TITLE for your advertisement

New fun roommate

 NEXT >

New Feed

by clicking on the fab icon on the previous page, you will lead to this page whih is a new feed. In this page you can make new advertisement in some steps. You will be asked about the apartment features and you have to fill them out all.

As you can see in the first step it is asked about the title that you want to show for your advertisement and this title will be appear for other users who are searching for the apartments.

Location and description
New Feed

Choose City
Milan

Add any more description you want
this home have good furnitures

< BACK NEXT >

About Apartment
New Feed

What is the occupancy number of People?
2

Type of the Room
Single room

Type of the Bed
Double Bed

< BACK NEXT >

About Apartment
New Feed

How many bedrooms does this apartment have?
2

How many bathrooms does it have?
1

< BACK NEXT >

Apartment rules
New Feed

Smoking Allowed ☐

Pet Allowed ☒

Occasional overnight guests ☒

< BACK NEXT >

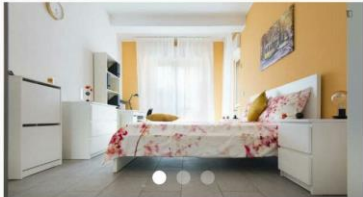
Rental Price
New Feed

Bills included ☒

Price/Month 350

< BACK NEXT >

Select Pictures
New Feed



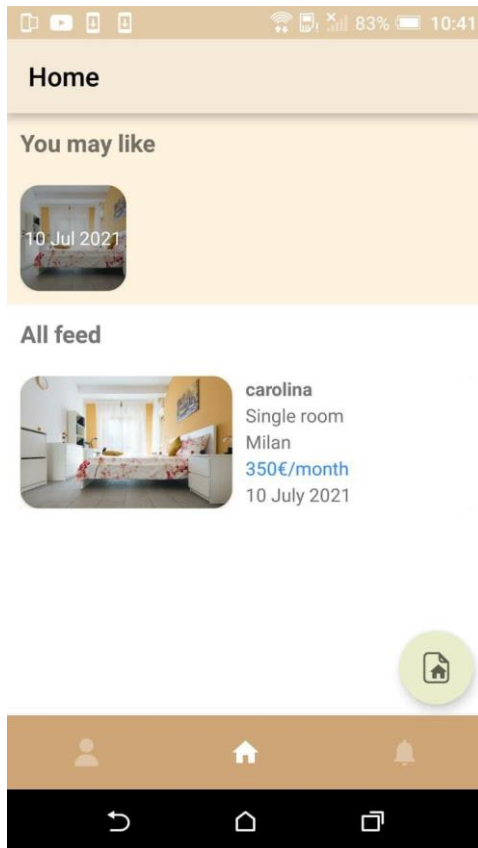
SELECT PICTURES

< BACK COMPLETE

As you can see there are some other steps which are about apartment itself like the occupancy, type of room and bed, price, uploading the pictures of the apartment, number of bedrooms and bathrooms and other things like the location and a description if you want to add. Moreover we put some options about personal information like smoking, pets allowed and etc to make users filter in more detail and find their best match flat mate.

All these steps are shown with a progress bar which is a very motivating tool to make users finish the steps.

Another noticeable feature that we implement is that we separate the form into some pages. This makes the application more user friendly and the experience better.

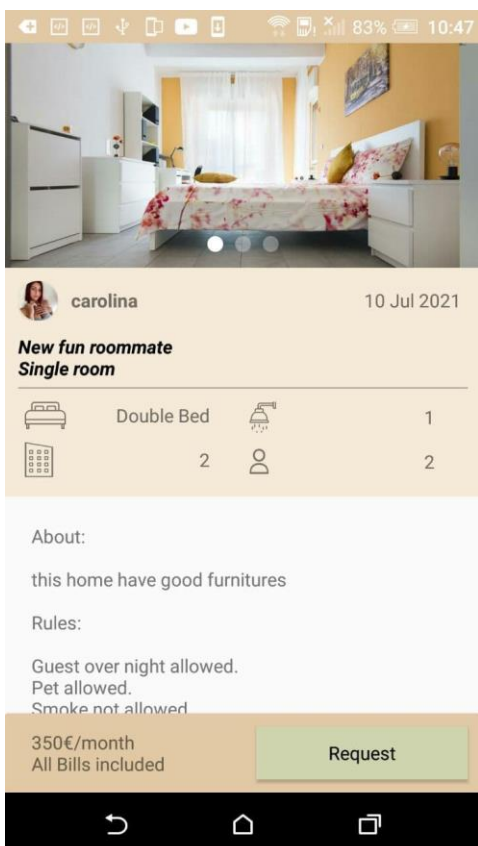


Home Page

By tapping on the Home button on the navbar you will see this page.

The application will suggest you some advertisement you may like and also you can see all the advertisement on the next sections.

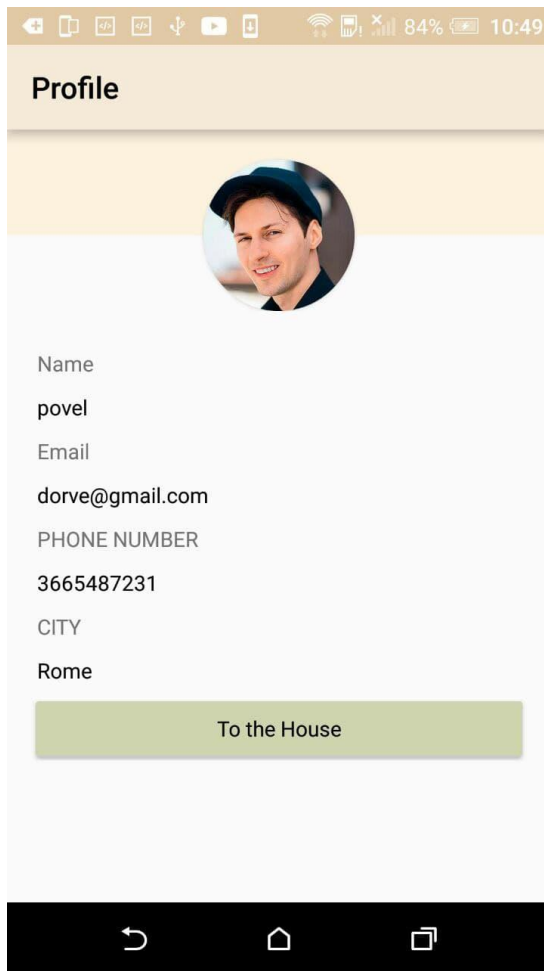
Notice that the fab icon for New Feed is always there in all pages and you can make new advertisement whenever you want. You just need to pay attention that you can make a new advertisement one at a time.



Advertisement page

by clicking on each advertisement you will go into this page. Yu can see the details about the advertisement and also the owner of the advertisement. So if you want to know more about the personal details and contact addresses of the owner, you can easily click on the picture of the owner which appears under the apartments pictures. (you will see the profile pages of the people in the next designs)

so if you will be intrested in the advertisement, you can tap on the request button at the bottom of the page and the owner will receive a notification in his request section of the app and he can accept or decline your request. After accepting or declining you will be toasted and you will understand that the owner is intrestes in you to be his/her flat mate or not, so in this case if you will be accepted you can contact him/her.



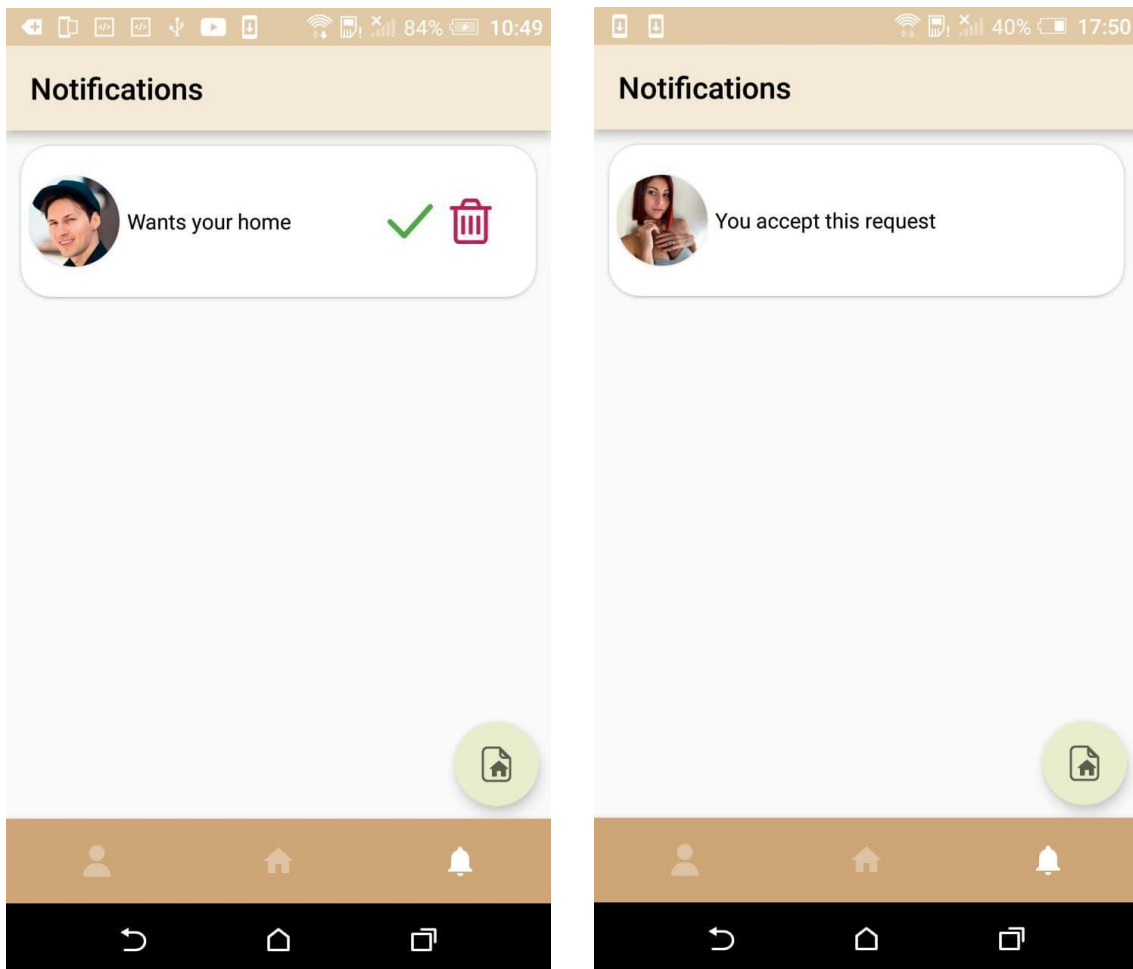
profile

and here is the profile page of the people that will appear like this.

You can see their information such as city phone number and etc. notice that your request have to be accepted by the owner firsrt and then you can contact him. If you wont, its against the applications rules.

For more navigation between pages in our application we put a button at the end which leads you to his advertisement again.

Notification



And the last part is the notification button on the navbar. Whenever you click on that you can access to the requests you have.

You can easily checkout their profiles by tapping on their picture to see if you want to accept or not.

If you accept or decline, the screen turns out like the right page (in this case we show the accepted one).