Polytechnic University of Timișoara

Faculty of Automation and Computers

**Department of Computer and Information Technology**

**Split the Bill**

Bachelor Thesis

*Candidate:*

**Emese Máthé**

*Supervisor:*

Lect. dr. ing. **Oana Iulia Casandra HOLOTESCU**

Timișoara

2019

**Contents**

1. Introduction
   1. Problem Statement
   2. Scope of the project
2. Use cases
   1. Application Description
   2. Use cases
   3. Application Features
3. Application design
   1. Architecture
   2. Strucutre of the data model
4. Technologies
   1. Flutter SDK
      1. Dart
      2. Scope Models
   2. Firebase
      1. Authentication
      2. Real Time Databse
      3. RESTful API
   3. Git
   4. Android vs. IOS
5. Application Implementation
   1. Account
      1. Registration
      2. Authetication
      3. Editing user infromation
   2. Packet organization
   3. Billboard page
   4. Supply page
   5. Chores page
   6. More page
      1. Apartment infromation
      2. Change theme
      3. Report issue
   7. Page interactions
6. Examples & Screen shots
7. Conclusion & Perspective
   1. Conclutions
   2. Future work
8. References

1 **Introduction**

Problem Statement

Nowadays a lot of people move into the city, at first people think of getting their own place but it is rather expensive to do so, and for the sake of saving up money people search for rooms in shared flats. Living with more than three people in an apartment can become difficult, especially when it comes to sharing the expensies and splitting the money for the cleaning supplies that everybody uses. It is also difficult to keep track of who contributed to the cleaning of the common areas in the flat.

Conflict can arrise from tha lack of communication between the roommates, because of their busy schedule, and they have no time to check in with each other, thus making it hard to live with one another. Such as having some friends over without asking, or having forgotten that the landlord was coming and people did not know about it. These can lead to conflicts and missunderstandings, thus forcing some people to find a new place to live in.

In order to prevent such things the people who live in shared flats must well organized, and have to have a good money management skills, but a lot of people who live in shared flats are young adult who do not yet posses there skill sets, thus making it difficult to get along with other and split everything correctly.

1.2 Scope of the Project

The scope of the project is to help people who have roommates organize themselves more easily when it comes to trivial matters such as chores, or splitting the money for the cleaning supplies. For this reason I was thinking of develeping a mobile application, the reason for developing a mobile application is because nowadays people are on their phones all the time, and almost everybody has a smart device. Because of this reason it is easier to check the application if something has came up or something has changed or not.

The reason why I did not develop a web page is because users prefer native application instead of web applications. There are a lot of reasons why this is the case, one of the reason is that it is faster and native application are more easily personalized, and users tend to like personalizing their application. Another reason for prefering application is that is more secure, and they can work up until a point without internet connection.

The mobile application is called split the bill, this application helps with spliting the chores and cleaning supplies between the people living in the same flat, and it also has a billboard page for putting announcements for people to see. In short the application will be useful for people who want to keep track of the things above and help them organize themselves more easily.

2. Use cases

Use cases help in organizing system requriments, and services that the software product will provide. They help undestanding how the user will interract with the system, and how the modules of the system will interract with each other.

2.1 Application Description

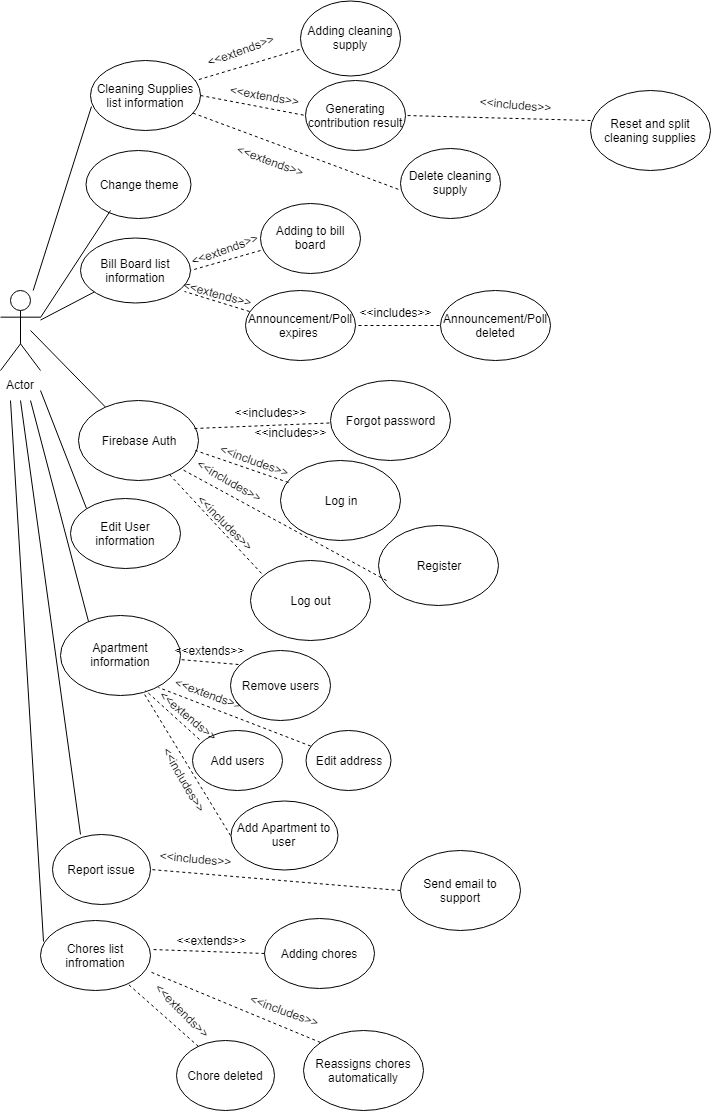
The application is called Split the Bill, it’s main focus is to help mange money resources for people who live in shared accomodation, thus making their life easies when it comes to splitting equally the money for cleaning supplies.

The application offers more than this, it also helps the people living togheter organize themselves when it comes to spliting the chores, and again it help with time mangment. The chores will be reassgined automatically to different people after a certain amount of time that the users in the apartment set.

It is also useful because the people who use this application can also make announcements and polls that the other can see. The announcements are for example when the landlord is coming over to collect the rent, and people need to be home, and someone writes this announcements in the bill board page of the application. People can also ask their roommates question if they can have some friends over, or if they can host a party, and the other people can vote with yes or no.

The application will be a native application, for both Android and IOS devices.

These are the main features of the application that the application is trying to solve and help people get along more easily, and helps them organize themselves more efficently.

2.2. Use case

An important use case is the authetication of the user this is done using firebase. When a user register they will have to provide the following information: first name, last name, nickname, phone number, a valid email address and a password. When sigin in the users will have to use their email address, because this one of the way you can autheticate using firebase. The users will be able to log out from the application. In case the user forgets to their password they can complete a form with their registered email address, to where the user will recieve a message with a link where they can reset their password.

Users will be able to view their user information, they will also be able to edit some of the information if it is outdated or it was misstyped. The followind data can be edited: first name, last name, nick name, phone number.

Another use case is the personalization of the application, the user can enter a dark theme mode, or if they choose the light theme mode they can edit the color of the application.

Reporting some issues is also possible by completing a form, the form contains three fields, a title, a description field where the user will be able to detail the what is wrong with the application, and a type. The type can be as follows: crash, bug or other.

The cleaning supplies information represent a list for all the cleaning supplies for the apartment that the user is apart of. The elements of the list will contain the following information:name, price, the nickname of the userthe type which represent where the supply will be used, the types can be of kitchen, bathroom, living or other. The user can add and delete from this list, and they can also reset the list meaning all the information will be deleted. The deletion will be possible by swiping the element from right to left. Another important use case is the generation for the contribution of each user, the generated data will be as follows: the total amount spent will be calculated and also each user how much paid from the total amount. The calculated result for each user will be the amount that user spent and the precentage from the total amount.

Billboard announcement infromation use case will provide the user the possibility to put an announcement or a poll for the other users in that apartment to see. The billboard will contain a list of announcements, each element regardless of their type will contain a title, a description, the author of the post. If it is a simple announcment it will only have the above information, on the other hand if it is a poll type than the users will be able to vote with yes or no, and the result of the application will be shown and updated each time a user votes. The user will be able to add and delete from this list as in the cleaning supply page. When addin a new element the user must specify the availability of the announcemnts, because after the time specified by the user the element will be deleted automatically by the system.

The chores list informatio list is another important aspect, here the user will be able to view the list of chores that are attached to that specific apartment. The users can add and delete from this list at any given time, deletion will happen the same way as in the bill board use case. Each element of the list will show the following informtation: the name of the chores, a description of the element, the current assigne. After the given amount of time the chores are reassigned to different people, this can be set in the page, and for all the element in the list the changing interval will be the same.

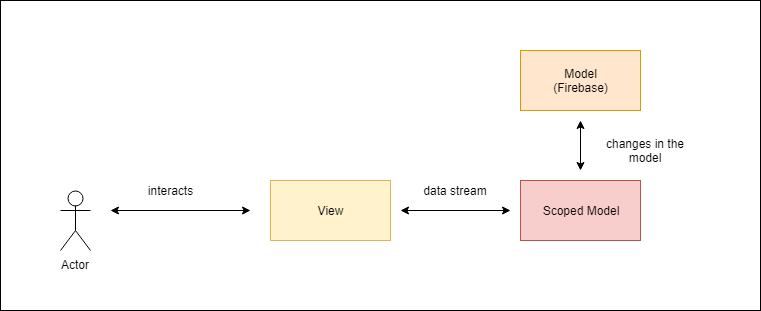
2.3 Application Features

In the table bellow there is the list of features that the application should contain for it to be able to function well.

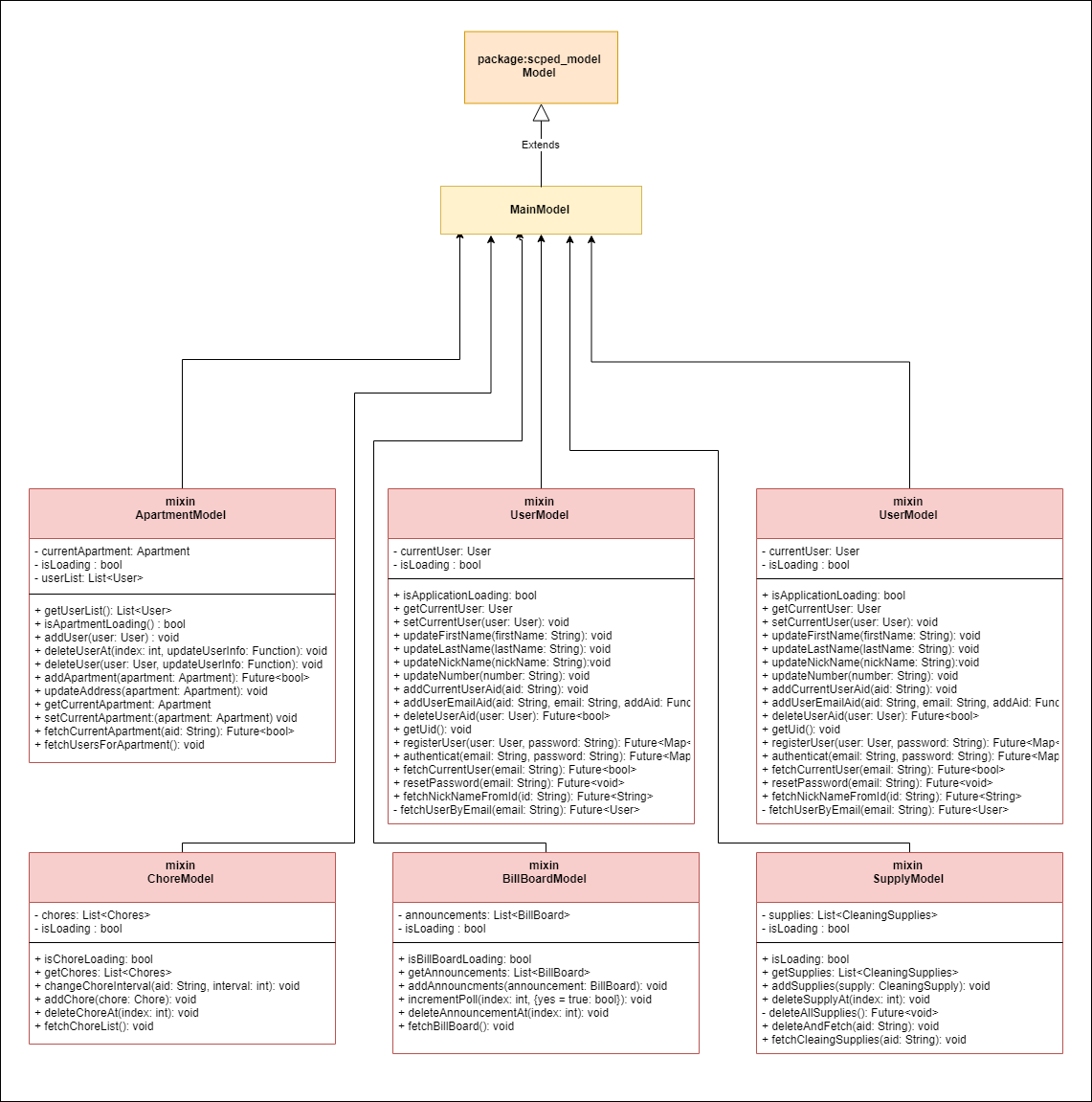
1. Users should be able to register to the application, and when doing so they should give some information about themselves.
2. Users should not be able to use the application without being autheticate
3. In case users should forget their password they are able to reset it.
4. Whe user encounter some problems they can report via gmail.
5. Personalizing the theme of the application.
6. Editing user information is editable in some of the information changed.
7. The application has a bottom navigation drawer, and has multiple pages.
8. The application is functional on android as well as ios platform
9. The application allows the user to add and delete chores, cleaning supplies and billboard announcements
10. The application also generates the contribution of each user in percantage and calculates it is total contribution
11. The billboard announcements can be of two types, simple or a poll, where users can vote with yes or no.
12. The chores are automatically reassigned to the users after a give amount of time.
13. The billboard announcements expires after a given amount of time and they are automatically deleted.
14. Interface should be simple and user friendly.

3. Application design

3.1 Architecture

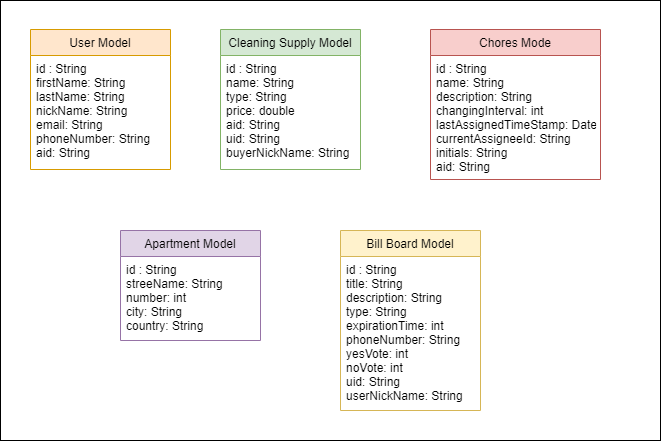
For this application I used layer architecture, in the application I have three layers, a presentation layer, a business logic layer and for the backend I have used firebase, the architecture is shown in the diagram bellow:

The presentation layer repestins the previously mentioned widget tree, that rerenders each time some changes appear in the business logic. This is possible using Scope Model and having references to function using the Scope Model Descendant object that wraps around the widgets. In the business logic section I handle the fetching and writing of the data to firebase. In this layer I also implament some functionality like . The backend layer is completly handeled by Firebase.

Split the bill also takes advante of event-based programming, which is useful when objects change their state. In case of flutter, as mentioned before we have Scope Models and Scope Model Decendants, and using these I can update the state of the model, from the presentation layer and redrawing the layout. This is possible because Scope Model notifies the Scope Model Decendants that can found lower down the widget tree that some changes occoured.

3.2 Structure of the data model

The data that is saved to the Firebase data model is in json format, but in the application there are some object that represent by some custom objects. The classes that define these models have a function that converts the data into a Map, that is accepted by Firebase.

There are five such models throughout the project. In the diagram bellow you can see the models and their fields.

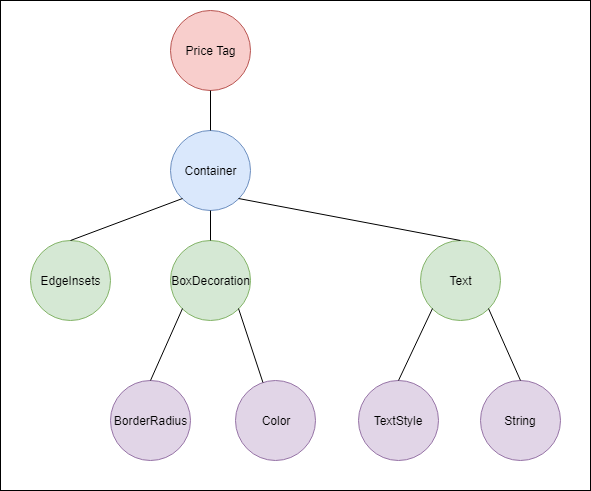
1. Technologies

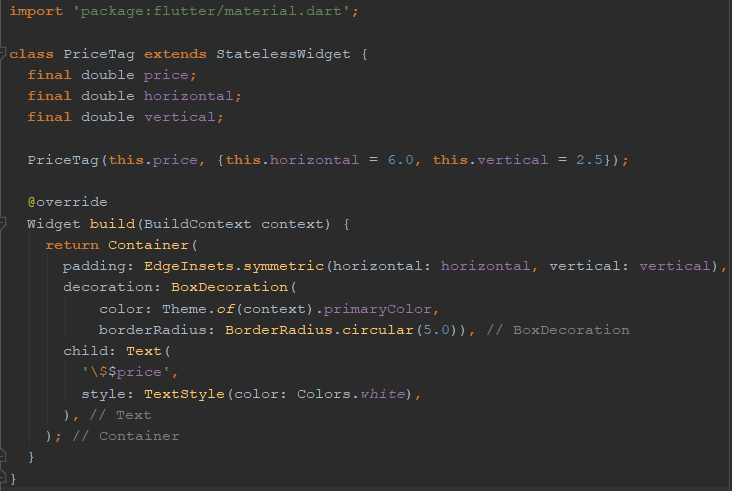
4.1 Flutter SDK

Flutter is an open source  cross-platform mobile app development framework, with which the developers can create native applications  for both the Android and IOS platform, thus the framework has one codebase from which it generates two native application. For Android the generated code is in Java and for IOS the generated code is Swift. This framework is created by Google. The programming language used by Flutter is called Dart.

The framework doesn’t depend on the widgets that are used by the platforms, because it only needs the canvas to draw it’s own rendering.

Flutter implements the OOP principle of the composition over inheritance. Composition and inheritance are both fundamental concepts of object-oriented programming. Composition means that a class has a reference to some other class, thus establishing a HAS-A relationship between these two classes. Inheritance on the other hand is a process where the class that inherits the super class and it’s public and protected methods and fields, this principle establishes an IS-A relationship between the parent and the child class.

A flutter application consists of a widget tree, so the application consists of small widgets which creates a complex user interface. A widget can be of two types. Stateless Widget which doesn’t change thus it is a final widget, and the Stateful Widget, this type of widget changes states by calling the function setState() which notifies the listeners that the state of the widget has changed thus the widget is redrawn.Bellow you can see an example of a widget tree and the code sample for which the tree is built

Because flutter is open-source there are a lot of packages that the developer can use to build the application faster, these packages can be added in the pubspec.yaml file of the projects, thus these packets will be added only to the current project not to flutter. In my diploma project I used packets such as these, for example the scope\_model, percent\_indicator, http, url\_launcher packets. In the above mentioned you can also add other dependencies such as the assets. Assets are considered to be images, or fonts etc.

* + 1. Dart

Dart is an object oriented programming language that uses both JIT(Just In Time) compilation and as well as AOT(Ahead-Of-Time) compilation.

Just In Time compilation is a technique that converts the bytecode that was generated by the compiler into native machine code at run time thus getting the name just in time compilation.

Ahead of Time compilation is a different compilation technique this one compiles the code into native machine code before the code starts to run in the environment.

* + 1. Scope Models

As mentioned before the Scope Model is one of the third-party packages that can be installed by adding the dependency into the pubspec file. This package helps pass data within the widget tree from the parent to the child nodes. And when the data is changed these child nodes will be updated and redrawn.

The Scope Model can be accessed in two ways, by using the widget ScopeModelDescendant, in which case the widget that is the descendant of the model will be rebuilt, or by statically accessing the ScopeModel.of(context) method.

Firebase

Firebase is a platform created by Google, that can be categorized as BaaS (Backend as Service). Firebase offers a lot of features, these features help developers to create application that respond fast, thus creating a great user exprience.

This platform offers a lot of analytics, these are categorised into tree group the development group, which consists of a NoSql real time database, authetication, cloud messaging, storage, hosting, test las and crash reporting. Another of the grpups is the grow froup which focuses on the notifications, remote config, app indexing, dynamic links, invites, adWords. The last category is the earn group which consists of AdMob, so that developers can earn money from in-app adds.

4.2.1 Authentication

For the diploma project I used the authetication, for users to be able to log in to the application, using firebase so that i do not save the user passwords into the database, and letting firebase handle user authetication.

Split the bill tries to autheticate the user via a login page, the credential of the user in our case email and password are sent to firebase, firebase then sends a response wheter the action was succesful or not, the token that is sent back to the application is encoded and if this token is manipulated firebase will know that the user that is trying to autheticate is fake or someone is trying to steal data.

4.2.2 Real Time Database

Another feature that was used in my diploma project is the real-time database, this type of database is a NoSql database, this means that the data is stored in a file with the JSON format.This mean that the datamodel is structured as a tree, and the data is saved by adding children to the parent node. The actual data that the user saves can be found at the leafs of the data model.

One of the main benefit of this type of database is that the data is synchonised in a few ms, thus making the mobile application responsive. When users are using a moblie application they like it to be fast, andto get the data as fast as possible. If an application is slow users tend to search for other options, ones that are faster.

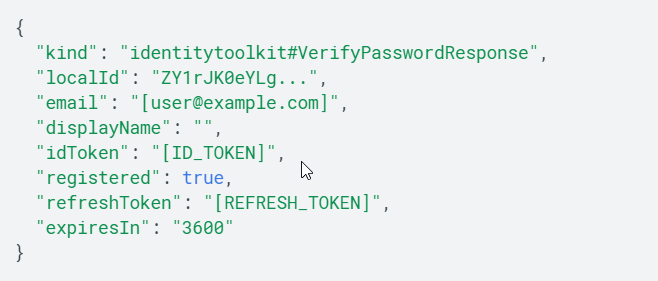
Another great feature of firebase is when that device looses it’s connection to the internet the events are still fired, so when the device goes back online the data will be merged to the database.

4.2.3 RESTful API

REST is short for Representational State Transfer, this API is used with http requests, such as GET, POST, PUT, DELETED. This method is used mostly for communication with web services. The reason is user RESTful API is that this was the server doesn’t exactly know about your application, it doesn’t have concrete infromation about the application itself, unlike with Android development where the user has to connect the application directly to firebase. The application uses some endpoint to firebase for adding, chanding, fetching and deleting data.

GET is used for fetching data from the data model, on the other hand POST is used to created or add new data, while PUT is called when an already existing data needs to changed. DELETE as the name suggests it is used to delete data. The data used by these requests can be block, file or even object, in split the bill the data is encoded in a map, that is formated so that it fits the real time database data model of json format.

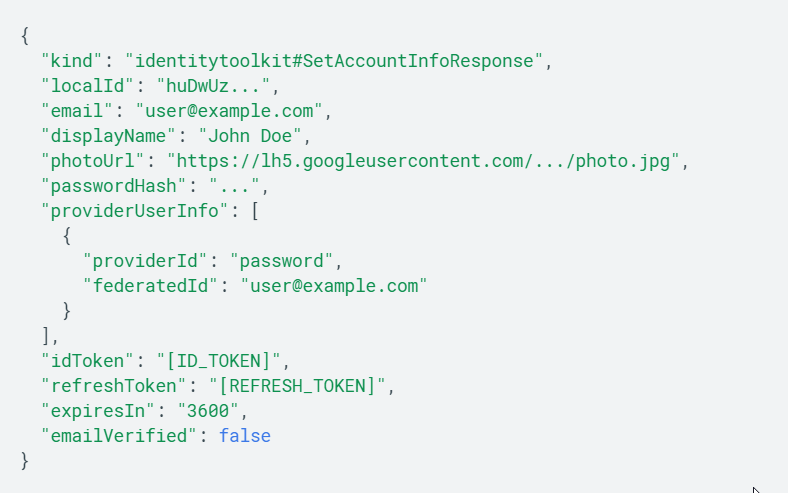
One of endpoint used in split the bill is when user is regitering, signing in, or requesting password reset is provided by google. This http link is an https which means it’s a secure link, the link contains an API key that can be found in the firebase project, that is created for this specific mobile application. After login in the user, we get a response if the authentication was succeful or not, if not we get the following error codes: EMAIL\_NOT\_FOUND, INVALID\_PASSWORD, USER\_DISSABLED. A sample of the response body:



User registration has a similar procedure, but it has different error codes: EMAIL\_EXISTS, OPERATION\_NOT\_ALLOWED, TOO\_MANY\_ATTEMPT\_TRY\_LATER. A sample for registation response body:



Another feature that uses that uses a similar link provided in the documentation is the email reset that is sent to the email of the user. The response body of this later feature is more complex than the one preveiously discussed. This service provided by firebase send the user an email with a link, if the link is accessed the user has to complete a form with the new password, and thus changing his or her password. Below you can see an example of such response:



Another enpoint that is used in split the bill application is the https link for the real time database that is part of the firebase project, from where we got the API KEY mentioned above.

Git

Git is a distributed version control system, it is also open source. A version control system is useful for storing code, and keeping track of which files were modified and by which developer. This is useful when there are multiple developers. A distributed control system is different from a simple version control system, the difference is that the distributed what has a remote repository to where the developers can put their code.

When stating to work on the new repository the command git init has to be exacuted this command will create a new .git subdirectory in the folder that the user was currently in. This subdirectory will contain the metadata for git to be able to register the modifications made by the developers.

There is another important aspect to consider, the git clone command, this command will copy the content of an already existing remote repository. This command will also create a connection to the original repository called origin.

Before publishing the changes to the remote repositoy there are several stept to be taken. The fist step is to call git add, this command will add the changed or newly created files to the staging area. The stagin area maean that git will know that these files will be added to the next commit. The add command can be reversed by running git reset, this will remove the files from the staging area.

After the files are in the staging area the the changes have to applied to the local branch, this can be run by running git commit, this will affect only the local branch/repository. To publish the changes to the remote repository the git push command will be executed.

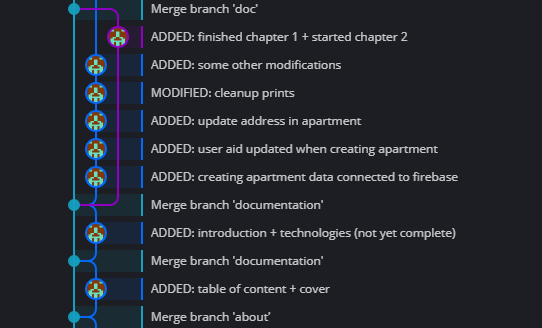
As meantioned above git takes snapshot from the changes made by the developers, this snapshots can be view by running git diff, git will show which lines were removed and which line of code were added in comparison from the last commit that the users has on their local reposiroty.

When developing in parallel programers may need to save some changes, because a task with higher priority came along, in this case they can stash the changes locally, and apply this stash later on.

In a local repository git groups the files into three category tracked, untracked and ignored files. The tracked files are the one that were already staged or commited, the untracked files are the new files that were not staged yet, and the ignored files are the one that git ignores and does not keep track of its history.

Another important aspect of git is branching, the branches can be remote or local branches, these branches represent a pointer to a commit, by default git created a branch called master, this branch will be the root branch of the project. During the development of the project I used branching when I implemented something new. When working with branches we also use the merging feature that git provides, this means that the content of the two branches will be merged into a single branch. During this merge conflicts can arise which the user has to resolve before being able to successfully merge the two branches. Anothe improtant aspect of git is switching between the branches, this operation is called check out.

When it comes to git, the developers need to be able to get the new changes, this can be achieved using the git fetch and git pull. Git fetch will download all the changed and commit from the remote repository, but does not immediately update the local repository. Git pull does exactly that, it fetches the data from the remote repository and immediately merges the changes for the local repository, this can be dangerous because of the merging conflicts mentioned above.

For my project I used github where I created a local repository, I also used local branches when changing or adding code, the reason for this is because I wanted to maintain a working version for my application, and I choose the master to be that branch, that does not contain partial implementations. I also used GitKraken as user-interface for git, in the picture bellow you can see a few examples from the commits and example of how branching was applies in my project.

Android vs IOS

When it comes to mobile devices there are two main operating systems that are popular, the Android operating system and the IOS. There are several differences between the two, one of the main differences is that Android is open-source and it is was published by Google, IOS on the other hand is closed and it was developed by Apple.

Another important difference is that Android is more customizable, but the iPhones may be more user-friendly due to its simplicity in the design.

The application available can be found in different stores, for android devices the google play store, where are significantly more application available for the user, but the danger is that there might be bloatware application, unlike the App Stores that the iPhone provides, the application on this store are tested that they are doing what the description specifies, thus filtering bloatware applications.

IOS is more secure because the source code is locked thus making threat more rare, unlike the Android operating system where the code is open-source thus making it open to cyber attacks.

The above mentioned are the more significant differences between the two operating systems.

1. Application Implementation

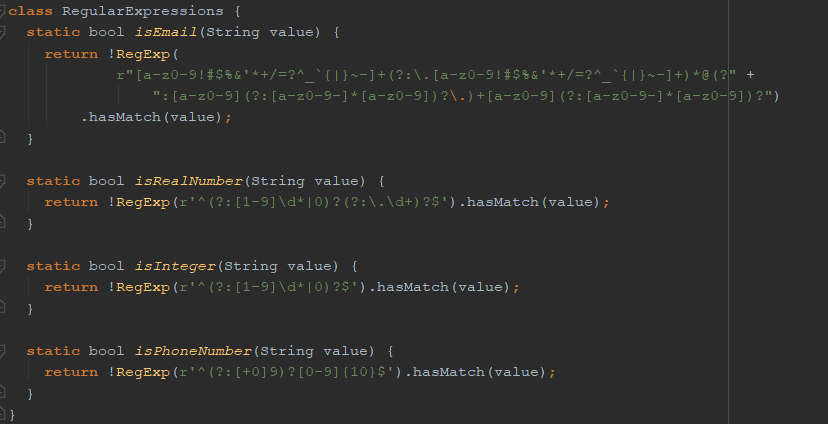
In this chapter I will explain in more detail about how the application was implemented.

Account

Each user must have an account, they cannot use the anonymous authentication. This means that the user must register and then later authenticate before being able to access or view the application.

* + 1. Registration

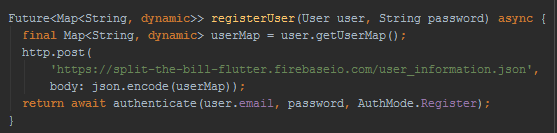
The registration page consists of a form that the user must complete, each element from the page is part of a single form, in my implementation I do use the verification of the field after the user entered all the data. In flutter there is a widget called TextFormField, for each element of a form a new widget of this instance will be added. TextForm field has a validation parameter, and when the user submits the whole form the validator will check if the input is a valid one.

The validation in split the bill is done using regular expression, for this reason I created a class where there are several static functions for different verifications, bellow you can find a code for the verifications:

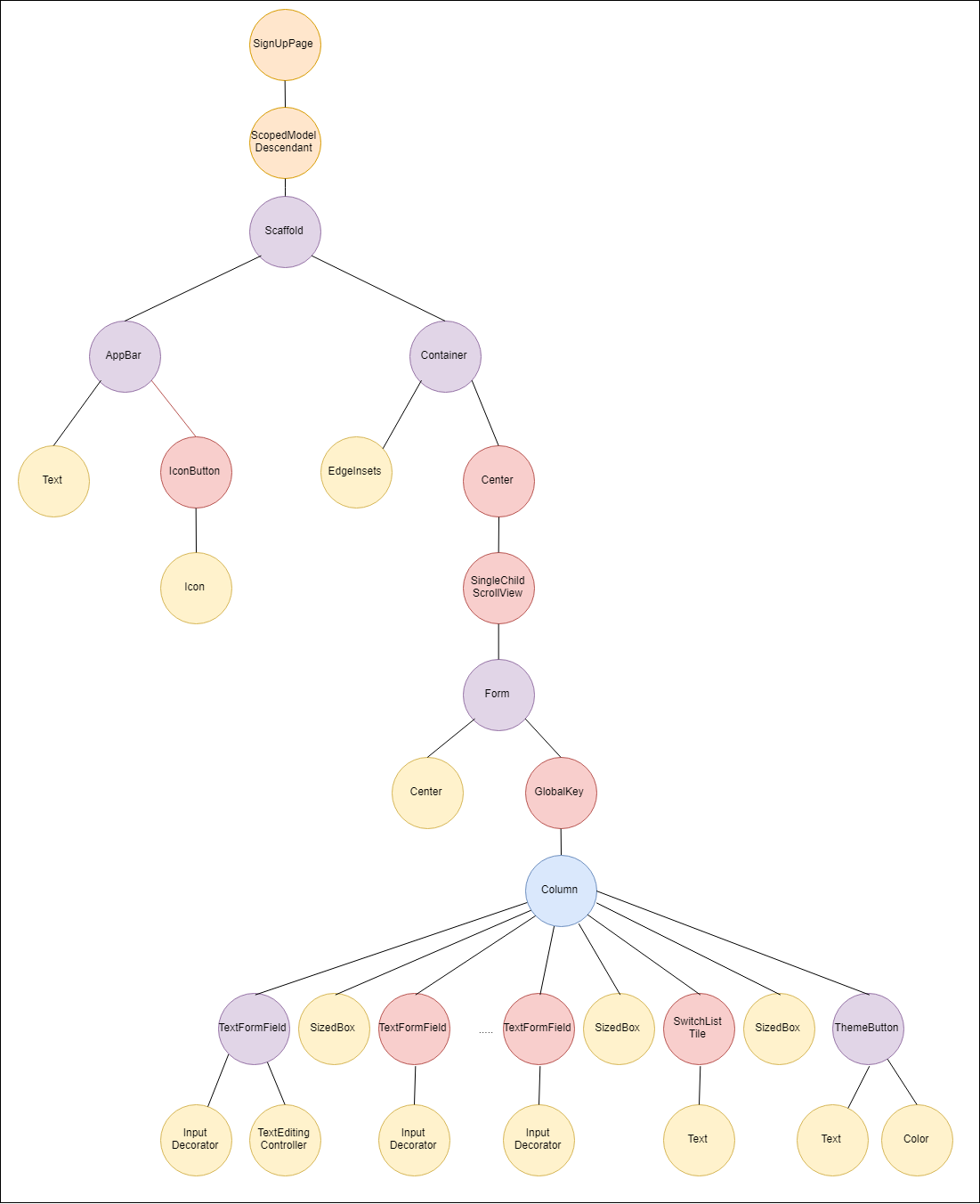
These verifications are also reused in other parts of the project for verification of other forms.

In the previous chapter we talked about RESTful API and how registration works with it, we also mentioned when a user attempt to register there is a response body, and if it was unsuccessful then some error codes are sent as well, in this page some of the most common error messages are handled by creating a pop up window with the error message, so that the user knows whether registration failed or was successful.

In the scoped model part of the registration there are several steps taken until the registration is successful, first the data from the form data is written into the firebase, afterward using the email and the password given by the user the application registers the user using the method with email and password, however if some error occurs the user will be given some details about the error using a pop up window that displays this information.

I would also like to mention that both the function mentioned above are asynchronous, this means that it takes a couple of milliseconds to complete using the await statement the first function waits until the second one completes, and only afterward it returns a response. A function is asynchronous if in the function header there is the keyword async as in the image bellow:

As mentioned before the user interface is built from a widget tree, in the bellow image you can find the widget tree for the registration page:

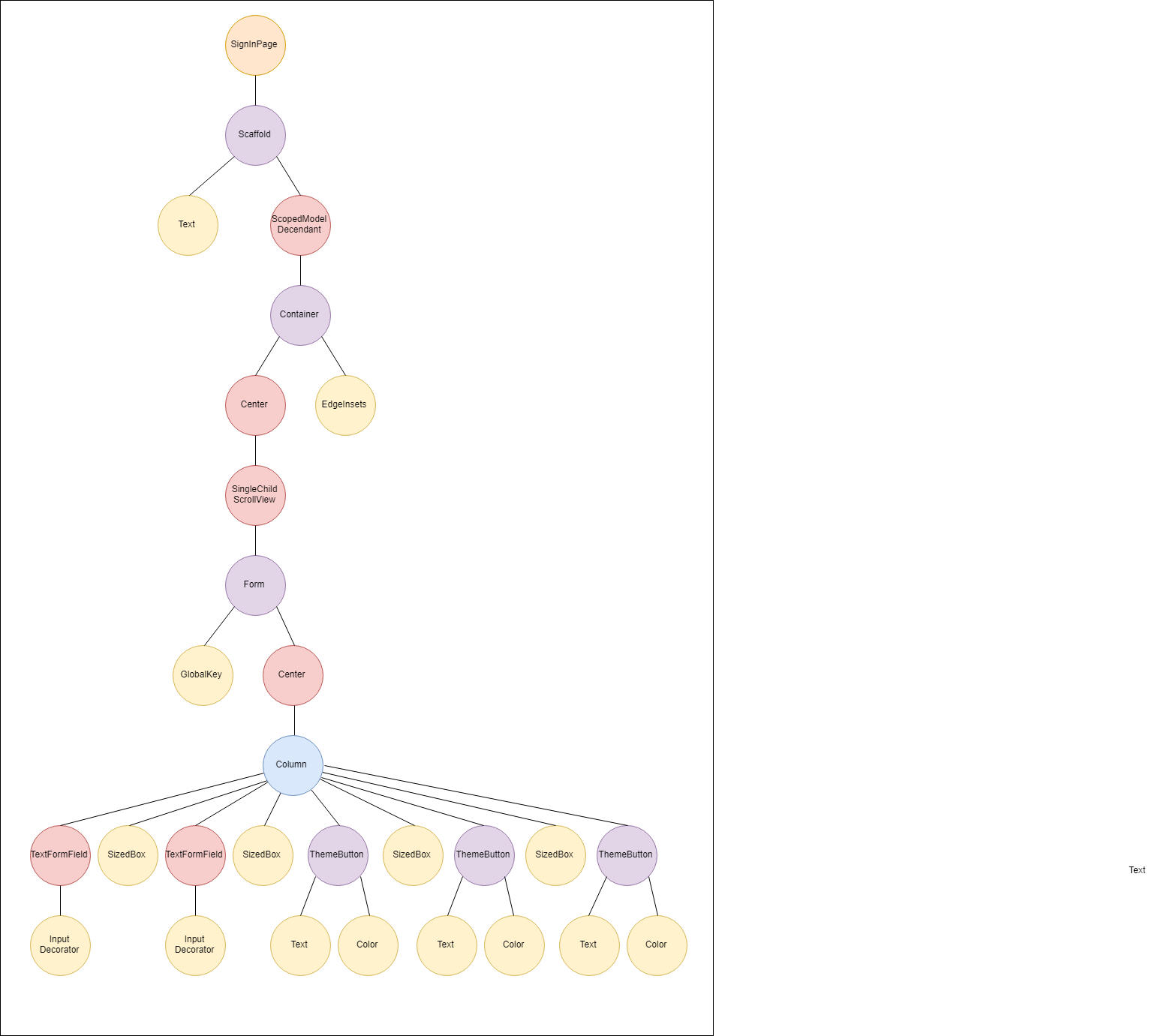


* + 1. Authentication

As mentioned in the previous chapter the authetication is done using RESTful API and using Firebase. Similar to the registration page, there are some error codes that REST can send back if the authentication fails, similarly to reigstration these are handeled using pop up windows, so that the user will know what went wrong.

In the sign in page we also use forms, only this time it is a shorter form, with only two input fields for the email address and for the password, and when the validators are called the functions from the RegularExpression class will be called to check the if the user typed the input correctly.

After the sign in was successful the pages with the values will be fetch from firebase, meanwhile this will show a spinner widget to show the users that the data is loading

In the image bellow you can see how the widget tree for the authetication page is build:

5.2.2 Editing User Information

As mentioned before not all of the user information can be edited, the email address cannot be edited, the first name, last name, nickname, phone number are the editable fields of a user. These fields can edited individually. On the user infromation page there is a list view with the user infromation, at the end of each element that is editable there is an edit icon button, if the user presses that button a pop up window will appear. The pop up window will show which field is currently edited, and there will be a form with the user information currently availble. The user can choose to edit or to cancel the editing progress of that field.

If the user chooses to edit that field, a PUT method will be called, this will update the value of the current user in firebase, using the current user id that is fetch after the sign in happened.

A picture containing text

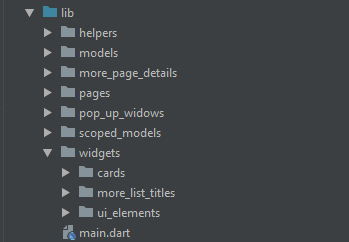
Description automatically generatedThe widget tree for the account information edit pop up and the account infromation page, which also contains the widget tree for editing the user infromation:

A close up of a map

Description automatically generated

Packet organization

In the the image bellow you can see how my the source code is organized into packets:



In the lib folder of the project you can find the dart files, I orgnized them in the following way, the helper packet contains the RegularExpression class that helps in the validation of the forms, and also contains an enum for the type of the authentication if the authetication of simple authentication or registration type.

The model packet contain the class that represent the model for Firebase. The more\_page\_details packet contains classes that represent the pages in the more detail of the application, while the page packet define the main pages of the application such as billboard, supply, sign in, sign up, chores and more pages. The pop\_up\_window packets define all the alert dialogs that are used in the application.

The scoped\_model packets contains the mixins that are used in the Main Model, these mixins as previously mentioned handle the logic and communication with firebase.

The widget packet contains other three subpackets, the first the card packet inside this packet are all the Card widgets that are used throughout the application, in the more\_list\_title packet are defined how are defined the list elements from the more page. And last the ui\_elements are contain some custom widget as a custom button, and a custom price tage widget.

Billboard Page

The billboard page shows the announcements published by the users of the apartment, each of this will have as a field the id of the apartment which so the application knows which announcements to load. The page consists of a list view, a floating action button when that button is pressed a pop up window will show a form that the user needs to complete in order to post a new announcement. The form consist of three input fields: a title, description and choosing from the drop down list the type of the announcements.

As mentioned in chapter 3, the users should be able to create polls, expiration time in hours, the type of the announcement indicate wheter the announcement is a simple one or a poll, where the users of the apartment can vote with yes or no. The result will be updated in the user interface as well as in firebase. In the user interface the progress of the voting will shown by a progress bar. This progress bar is a third-party package.

The deletion functionality is imlpemented using by sliding from right to left, the reason for this direction is that most application use this type of directional sliding when deleting an item, and I wanted to make the application as user friendly and easy to use as possible.

The billboard model has a timestamp as an attribute as well an expiration time, after sign in the application calculates the time that has passed since the announcement was posted, and compares the result with the value of the expiration time, if it greater than the expiration time it will delete the element before loading the data, if is smaller than the expiration time, then the application will set a timer, which will delete the announcement when the time is up.

A close up of a map

Description automatically generatedIn the bellow image you can see the widget tree that make up the billboard card, billboard, bill board form and billboard page widget trees:

A picture containing map, text

Description automatically generated

A close up of a map

Description automatically generated

Supply Page

In the the users can view the supplies that he or she added as well as the supplies added by the other users. To add a new cleaning supply the users will have to add press the floating action button an complete the form of the application. The application will contain three fields, the name of the supply, the price and the type which refers to where that product will be used, the types can be as follows: bathroom, kitchen, living, other.

After adding the new element both the user interface and firsebase will update their value, each element in the list will have a liding icon that will represent the type, a title that will be the name and a price tag at the end of the element representing the price of the product.

The user will also be able to generate some data, the data generated can be done by clicking the statistic button in the app bar of the page, when doing a pop up window will apear that will have the list of the following elements, the name of the user to which a progress bar is attached which will show how much of the total amount the user contributed in percents, and next to the progress bar the total amount of money the user contributed to the apartment since the last reset. After the user list, the total amount will be shown as well. In this part of the application the user will be able to close the pop up or reset the list, by reseting the data in the supply page all of the data in that page will be deleted.

The elements of the list can also be deleted the same was as in the bill board page mentioned in the previous chapter 5.3.

A close up of a logo

Description automatically generatedIn the images bellow you can see the widget tree of the cleaning supply card, this page, and the money split pop up, as well as:

A picture containing text, map

Description automatically generated

A close up of a map

Description automatically generatedChores Page

The chores page is similar to the structure of the previously discussed two page, it has a list view that contains the existing chores for the apartment that the user is part of. These chores are reassigned automatically using a periodic timer, this timer.

The user will be able to add new chores to the page by clicking the floating action button, and completing the button, the form has a the following field, the name of the chore, a description of what the chore implies. When adding the application will add an assignee automatically to the chore.

The interval of the reassignement is in days, this value is the same for all the chores that are part of the same apartment, this can be set by any user living in that apartment. This can be done by clicking the settings icon in the app bar of the application, which opens a pop up window that enables the user to set an integer number for the number of days that the reassignement happens. The application will automatically reassging these elements without any input from the user.

The deletion is impelmented the same way as for the other two pages mentioned in the previous sections. The widget tree is similar to the prevoius two section.

More page

This page of the application represent some other functionality, and has a list of other funtionalities, which by clicking will open a new page in the application. Such pages are the previusly discuessed account information and user infromation editing page, the apartment information and editing page, the changing the theme page, the about page of the application, the sign out, and the reporting page. Some of these pages will be discussed in more details in the next sections to come.

* + 1. Apartment information

This page will have contain the address of the apartment, if these is not apartment created for a user the user will be able to create a new one by completing its address. The user that creates the apartment will automatically be added to the apartment.

If the apartment exists, the data for it will be fetched using the apartment id, that the user has attached to its data model. Adding new users in the apartment is possible by clicking the floating action button of with the user icon on it, this will open a pop up window where the current user will complete with the email of a registered user. The application will fetch the user with that paricular email, and will set the apartment id for the user.

The users will have the option of deleting people from the apartment, this is useful when someone move out from the apartment. Before the user is deleted the application will show a pop up window with the message the following message: Are you sure you want to delete this?. The address of the apartment is also editable, if the user want to edit an existing address the pop up window with the form will appear completed with the existing data.

The user when viewing an apartment page that is not empty will see the following information: the address of the apartment, and the list of users that are currently assigned to this specific apartment. For each user in the apartment the following information will be shown: the full nam of the user, composed of the first and last name, the nickname of the user, the email address as well as the phone number of the user. This is useful because people living in the same flat might want to be able to reach the other one when that person is not at home.

In the following image you can see the widget tree of this particular page:

A close up of a necklace

Description automatically generated5.6.2 Change Theme

Changing the theme of the application is done by using a third party package called Dynamic Theme. Regardless of the theme types, this widget wraps around the whole application and it is added in the main file, at the root of the widget tree. The dynamic theme establishes some default values, in split the bill the default them is white brightness and the color red.

In changing the theme page the user can switch between the light and dark brightness using a toggle. The use can select the color only in light mode by selecting one of the color from the drop down list, and then pressing change color.

* + 1. Report Issue

The report issues page will have a form that the user needs to complete a title, which is short the issue that the user encountared a description of the issue in more details, and the type. The types can be as follows: bug, crash and other. The type can be selected from a drowp down list. After the user submit the application switches to gmail where the user can sent an email with the issue to a given email address.

The switch between the application is done using the url\_launcher third-party package, this package helps to redirect users to different app or http links.

Page interraction

In this section I will be discussing how the pages interract with each other and how they change between them. The pages of the application are organized in a stack of pages. The pages can be changes or replaces. In the case of pop up windows I open the new window that becomes the top of the stack, and when I close, the pop method is called, this will remove the element from the top of the stack. Another way that I used the navigation was to push and replace the top of the stack with the new page.

A picture containing screenshot

Description automatically generatedIn the picture above you can see how the pages interact with each other and how the pop ups interract with the pages:

The application uses a bottom navigation drawer, this can be found at the bottom of the page, there are four main pages. The last page the more page contains a list of other pages, the will be opened by clicking on the item of the list. If doind so the bottom navigation of the application will dissapear, and only the new page will be visible without any navigation menu. The user will be able to get back to the main part of the application by clicking the arrow button found in the application bar of the page that he is currently residing on.

6.Examples & Screen Shots

A screenshot of a cell phone

Description automatically generatedA screenshot of a cell phone

Description automatically generatedIn this section I will exemplify the page layouts and some important use cases mentioned above. In the bellow diagram you will see the sign in and log in pages layouts

A screenshot of a cell phone

Description automatically generatedIn the next screen shot the form for the forgotten password is shown:

In the next pictures the layout of the loading of the supply page and the layout of the supply page with data can be seen:A screenshot of a cell phone

Description automatically generatedA screenshot of a cell phone

Description automatically generated

A screenshot of a cell phone

Description automatically generatedA screenshot of a cell phone

Description automatically generatedIn the next screen shots you will see the layouts for the chore and the billboard pages:

In the next screen shots the layout of forms for adding new chores, billboard and supply:

A screenshot of a cell phone

Description automatically generated

A screenshot of a cell phone

Description automatically generatedA screenshot of a cell phone

Description automatically generatedA screenshot of a cell phone

Description automatically generatedIn the next screen shot the laybout of the money split pop is vissible

In the next screen shot the more and the about page is visible:

A screenshot of a cell phone

Description automatically generatedA screenshot of a cell phone

Description automatically generated

A screenshot of a cell phone

Description automatically generatedIn the next layout examples you can see the form for editing the users phone number:

A screenshot of a cell phone

Description automatically generatedA screenshot of a cell phone

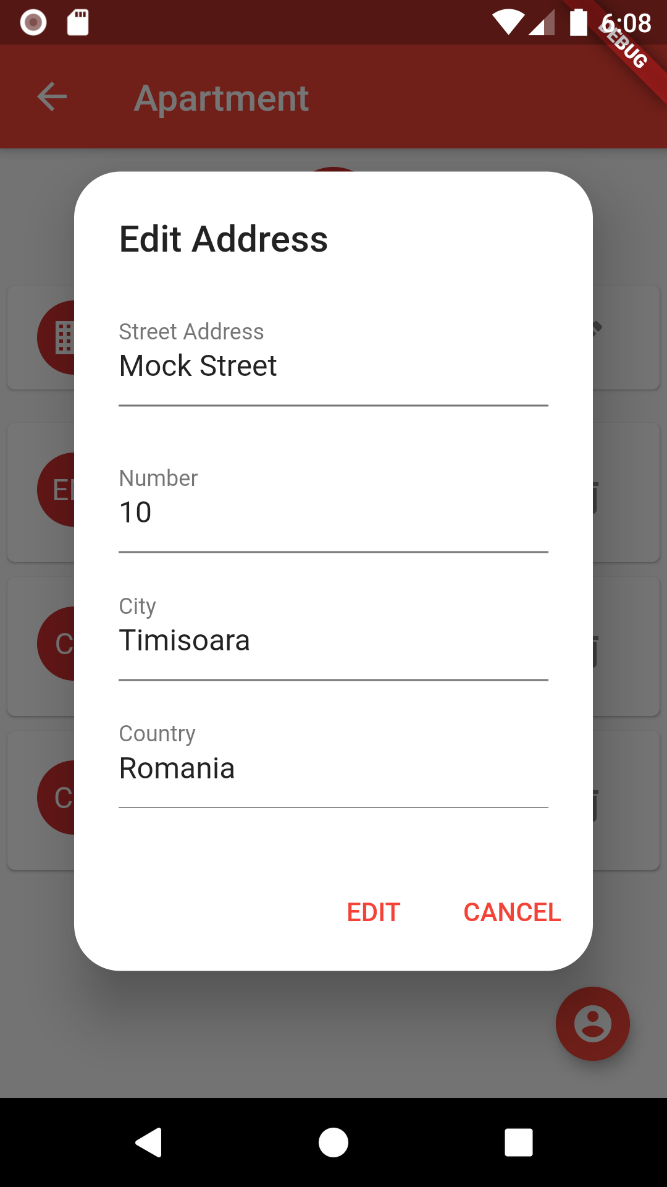
Description automatically generatedIn the next screen shots the user informtation and the apartment information page layout is visible

A screenshot of a cell phone

Description automatically generated The users can add new users by clicking the floating action button and the layout from that form is:

Ithe following picture shows the layout of the editing the address of the apartment and the reporting bug page:

A screenshot of a cell phone

Description automatically generated

In the next screen shot that follow the theme change will be presented, in the application the user can choose between ten different colors. When dark theme is enabled the user can not choose the color of the theme anymore this is dues to the fact that by default it not possible to change the color. In the next example I will present present a dark theme and a two different colors in light mode. Depending on the color flutter will change the backround of the text color, if the color is very light the backround of the text and icons will be black, on the other hand if is a more solid color flutter will leave the button text color and icons as a white.

The layout of the change theme page:

A screenshot of a cell phone

Description automatically generated

The color options for the user:

A screenshot of a cell phone

Description automatically generated

The dark theme, the examples for the themes will be provided for the more page:

A screenshot of a cell phone

Description automatically generated

The pink theme with white text color:

A screenshot of a cell phone

Description automatically generated

The gold theme with black text color:

A screenshot of a cell phone

Description automatically generated

7. Conclusion & Perspective

7.1 Conclusion

Split the bill is an native mobile application that takes care of splitting the money, and mantaining the apartment for people that do not have the time to do it themselfes. It is an application that makes living in shared accomodation more pleasant and helps avoiding conflicts for reasons like money, and cleaning chores.

During the developement of the application I encountered a few problems, one of the most challenging problem that I encountered is to build the widget tree correctly. Each widget has its limitations, and thus a developer can abuse these limitations, in my case I had an exception of an infinite scrolable view, this was dues to the fact that I put two elements that were both scrollable, thus making it infinitly scrollabe. Another important aspect was when I used setState with notify listeners, these two functions esentially do the same thing they notify the widget tree that some changes occoured in on of its nodes, but the differences the that notifylisteners is used by the scoped model, while the set state is used by the Stateful widget, one function updates the informtation from the logic layer of the function while the other one handles the modifications direcatly from the widget tree, this means there is no layered architecture. If the two functions are used toghter at the same time some errors occour, to update the state of a widget only one should be used.

With flutter you can develop applications fast, but at the same time you have to pay attention how you structure the widget tree as well as the scoped models. On the short term it is easier to to make an application that communicates with the database directly from the presentation layer, but on the longt run the code becomes unreadable and unpredicatble, thus separating the presentation and the logic layer is important.

When it comes to the two big mobile platforms it is diffucult to develop an application that looks the same for both IOS and Android. Developing an application that runs on both platforms can become expensive, there are solutions out there like hybrid mobile application, but they are not as powerful as the native applications, and users tend to prefer native applications, over other types. The reason why flutter becomes more and more popular is because it has a single codebase and it generates a native application for both IOS and Android, the generated code for Android is in Java while for IOS it is Swift. The market for the mobile application in the last few years has became very competitive, because of this there are a lot of high quality mobile application that the developers have to compete with, igf users get bored of the application they tend to uninstall the application and search for better ones.

Another technologu used in my project I used RESTful API, the reason for this decision was because using this technology the data is encoded, this means that it is secure. Nowadays cyber attacks are very frequently occouring, making secure connection and encoding the data has became expected of an application. Another great thing about RESTful API is that the server that the data is saved on does not know details about what the data is used for or details about the application.

For my backend I used Firebase which is a BaaS. Firebase provides a lot a of services, in my application I used the real time database and the authetication that firebase provides. The users can autheticate using an email and a password. The registration and the email reset is also handled by Firebase. The email password reset is done outside the application, using Firebase and REST I send a request to firebase to generate a link where the user can reset their password.

In the application there are a lot of pages and pop ups that interract with each other, there are four main pages in the application that are part of the bottom navigation drawer, the fourth page has a list of pages that opens new pages. I considered this method of navigation one of the more efficent ones, because the user has quick access to the frequently used pages, while the pages that are less popular are more complicated to reach. Having quick access to the pages that are more popular help users have a better user expercie. When only having a side drawer navition menu, that type of menu tends to get annyoing and frustrating after a while. If users are not satisfied with the application they will seach for other application that have a better design and are more user firendly. Most of the social media application have this type of navigation system in their application. In flutter pages work as a stack when adding a new page, that page will be added on the top of the stack, when closing a page you can choose from to options, you remove the top element of the stack or you replace it with another one.

Replacing and poping may depend on the situation, usually you replace the existing page when you do not want to do back to that page,by replacing the top element you destroy the top of the stack. In case of the pop you do not want to change the state of the other pages that are part of the stack.

Split the bill has three main focuses, splitting the household supplies, having an announcement billboard where people can inform the other people living in the flat, and distributing the chores that need to be done every few days. There are other setting and functionalities such as personalizing the application, view and editing information about the users, and the apartment, reporting bugs and issues encountered while using the application.

7.2 Future work

I would also like to further improve the application by adding new features, as the cloud messaging. Nowadays a lot of people have social media, but not everybody has an account, for this reason I would like to develop a group chat as well as individual chats between the roommates that live in the same apartment.

Another feature that I would like to add to the application in the chores page, where users can let know the other people if they correctly done their chore for the given amount of interval, the feedback would be anonymous meaning that the assignee does not know who gave her the bad rating.

I would also like to add a new page for the utilities, where people will introduce the type of utility and when the reading for the consumed utility needs to be given to the provider. The time interval in which the bill arrives would also be useful information for the users, and also an assignee that who will pay for that, so that people know who they need to give the money back for that months utility.

Another personalization of the application would be for the user to decide which of the applications pages the user would like to see in the bottom navigation drawer, this would give a lot of flexibility to the application, the use would be able to selecte the amount of pages he or she would like to have in the application, with an upper limit of 6 pages.

The application is only available in english, I would like to develop the applicatio n further so that is can be available in multiple languages, even though nowadays most people understand english still not every one does know it. If there is more langues avaialble that atrackts more users.

I think that the above features might be interesting to implement, becasue they will give the user more flexibiliy, and more management over the apartment, which is the whole purpouse of the application.

1. References

[1] <https://api.flutter.dev/flutter/material/material-library.html#classes>

[2] <https://pub.dev/packages/percent_indicator>

[3] <https://pub.dev/packages/scoped_model>

[4] <https://pub.dev/packages/dynamic_theme>

[5] <https://pub.dev/packages/http>

[6] <https://pub.dev/packages?q=url+launcher>

[7]<https://medium.com/jexia/developer-beware-consider-composition-over-inheritance-a7d4e14556c5>

[8] <https://stackify.com/oop-concepts-composition/>

[9] <https://git-scm.com/about>

[10] <https://www.diffen.com/difference/Android_vs_iOS>

[11] <https://medium.com/flutter-community/flutter-layout-cheat-sheet-5363348d037e>

[12] <https://www.udemy.com/course/learn-flutter-dart-to-build-ios-android-apps/>

[13] <https://www.atlassian.com/git/tutorials/learn-git-with-bitbucket-cloud>

[14]<https://www.oreilly.com/library/view/software-architecture-patterns/9781491971437/ch01.html?fbclid=IwAR0UN7mujNgygczPBdgNUH1SYPNZoqvc21REycv-C5d1v3DByKr8zXl0Tz8>

[15]<https://codeburst.io/software-architecture-the-difference-between-architecture-and-design-7936abdd5830?fbclid=IwAR09aOJzyP2ImOa1GQQCys_7l7NIT25GEgGW9dQfPntZ039q579dBz8xk04>

[16]<https://firebase.google.com/docs/reference/rest/auth?fbclid=IwAR14Qo0GU39yP5ZM4CBzHXQj57vzFBG7ZSfGxtl-m8z1u87UaCAqyxB56jw#top_of_page>

[17]<https://hackernoon.com/googles-flutter-in-10-questions-answers-928e142ec725?fbclid=IwAR3Te1aMiIsMWURxyBKr7PsJPxTaDVmOPpLaE31cEIrT_Sh79NuW3P1LABI>

[18]<https://ontouml.readthedocs.io/en/latest/classes/nonsortals/mixin/?fbclid=IwAR2dj86mUFWDBz-j018n3b9B9etXUV8ILgOMqSdlz7QWnbvE73ZJbAT9UAU#common-questions>

[19]<https://reactjs.org/docs/lifting-state-up.html?fbclid=IwAR0KZq4S8zq4rXdExy2-giyHBKep7qe5HiwWxXfChwEdbP_gBZJSoTeLfAg>

[20]