UKRAINIAN CATHOLIC UNIVERSITY

BACHELOR THESIS

Development of the mobile application

Author:
Serhii ROSOVSKYI

Supervisor:
Serhii MISKIV

*A thesis submitted in fulfillment of the requirements
for the degree of Bachelor of Science*

in the

Department of Computer Sciences
Faculty of Applied Sciences

Lviv 2021

Declaration of Authorship

I, Serhii ROSOVSKYI, declare that this thesis titled, "Development of the mobile application" and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed:

Date:

"First, solve the problem. Then, write the code"

John Johnson

UKRAINIAN CATHOLIC UNIVERSITY

Faculty of Applied Sciences

Bachelor of Science

Development of the mobile application

by Serhii ROSOVSKYI

Abstract

The Ukrainian Catholic University has many services that are scattered on different web platforms. Consequently, the main problem is that the university can not get all the information in one platform, such as a mobile application. This thesis will be related to developing a cross-platform application for both platforms, Android and IOS, that will contain all the necessary information for students and staff of UCU.

Acknowledgements

Firstly, I want to thank my supervisor Serhii Miskiv for always giving wise advice. I am deeply grateful to the team of business analysts who were helping to research this project. These people are Nazar Mamonov, Yewgen Domoretskyi, Khrystia Sliusarchuk, Oleksandr Dubas and Yarema Fylypchuk.

Contents

List of Figures

List of Abbreviations

BLoC	**B**usiness **L**ogic **C**component
SSO	**S**ingle **s**ign-**o**n
UX	**U**ser e**X**perience
UI	**U**ser **I**nterface
SDK	**S**oftware **D**evelopment **K**it
API	**A**pplication **P**rogramming **I**nterface

Dedicated to my mom and dad

Chapter 1

Introduction

1.1 Relevance of the topic

Mobile development is gaining momentum. It is much more profitable for people to look at the phone, open the downloaded application and see what they need. Searching for something on the Internet, going from one site to another, spending nerves searching for the necessary information is increasingly not liked by people who are much better off viewing it in a mobile application. It's more about convenience.

That is why creating an application for UCU students and staff will facilitate the faster search for information and many other valuables that are collected in one place.

After registering in the mobile application, you will access the offline locale card, filter UCU events and much more.

1.2 The problem

UCU students and teachers need a long time to search for the necessary information on various sites on the Internet. For example, finding a suitable event is caused by a significant challenge or filtering of events because there are quite a number of them. The solution to this problem will be creating a real-time mobile application for UCU students and teachers, which will always be at hand on the phone. You will no longer need to go to different sites to find the information you need.

1.3 Purpose of the study

The motivation to write a thesis on such a topic was not so difficult. During my four years at UCU, I noticed that the university lacked a modern mobile application. Its primary purpose will be:

- Collection of all UCU events;
- Filtering events by categories;
- Phonebook: search person's info by name and surname;
- Attaching a Lokal card that can be restored;
- Register UCU's corporate email address;

Chapter 2

Related works

2.1 Mobile solutions

The starting point is to make sure that solutions exist. There are many universities in the world, which could have mobile applications created by them.

- The University of California, San Francisco (UCSF Mobile)
  Description: "UCSF Mobile: access more info and easy navigation including location-awareness technology, real-time info (what's open, what's not) and convenient access to the UCSF directory to help you get more out of your UCSF experience."[California San Francisco, 2020]
  The rating of the application in the App Store is inferior(avg. rating is 1.5 stars).
- Northern Arizona University (NAUgo)
  Description: "NAUgo is Lumberjacks' one-stop mobile app for staying informed, connected and up-to-date. Choose your experience: Flagstaff Campus or Statewide Students: Students can log in to see the course schedule, grades, advisors and more while enjoying their favourite map, dining, transit and event modules - plus a whole lot more! Guest: The same classic NAUgo that you've grown to love."[Store, 2021]
  The rating of the application in the App Store is superior(avg. rating is 4.5 stars).
- National University of Water Management and Environmental Sciences
  The rating of the application in the Play Marker is superior(avg. rating is 4.0 stars), a number of app reviews - 732.

* * *

- Results of the research:
  - There are universities' applications, which students/teachers use.
  - Almost all applications are only for a specific platform(Android or IOS).
  - The UI of applications follows trends of 2010.
- Main functionalities of the applications:
  - Events calendar
  - Directory
  - Emergency number

- News and advertisements
- Grades
- Information about teachers, faculties

2.2 Native/Cross platform approach

In 2021 there are a bunch of options on how to build a mobile application. We can do it in two different ways: natively or cross-platform.
Native development can make the application only for one particular platform at the moment. For example, for Android, we can use Kotlin for IOS - Swift.
Firstly, we need to figure out which operating system students use. For this, I made a poll and here are the results:

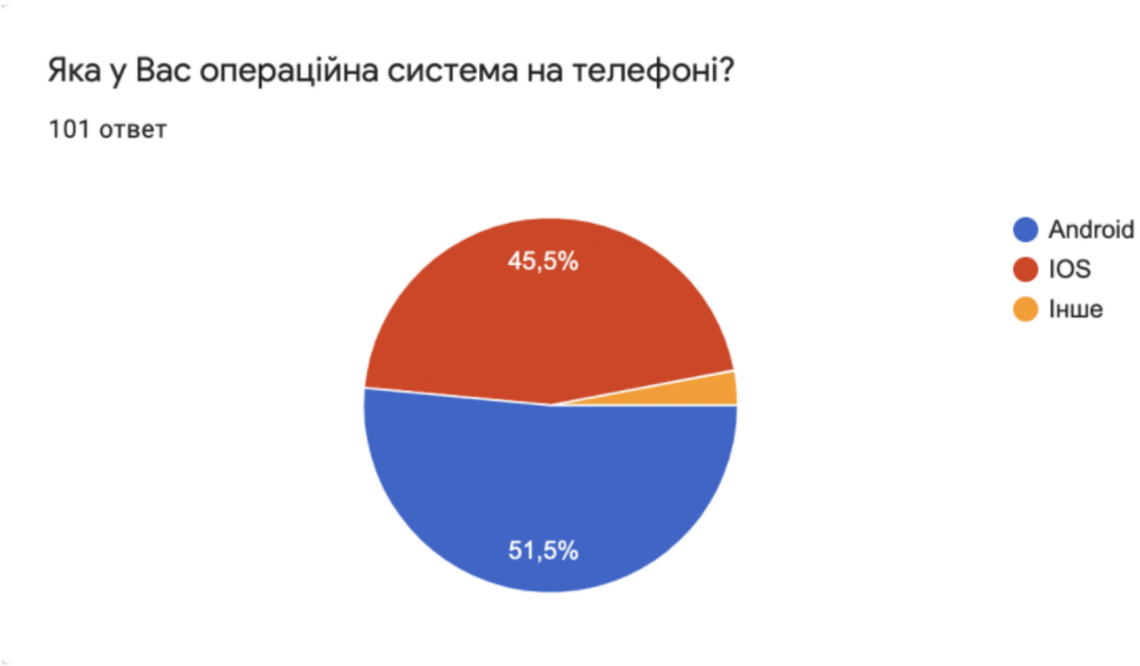


FIGURE 2.1: Student's mobile OS distribution

So, as we see, the difference between Android and IOS users in the university is not essential. So it would be an inferior way to build the application natively, just for one platform, because we would lose a vast amount of users.
The second way is cross-platform. Cross-platform is the ability of the mobile application to work on a few platforms. It allows having two different applications for Android and IOS using one code. So, for further development, the cross-platform has been selected. The technology for cross-platform development will be discussed in section 4.1

Chapter 3

Project information

3.1 Project description

The application for Ukrainian Catholic University will connect most of the significant university services used by students, teachers and university guests. Some of these services are “UCU Phone Book” to find information about some other student/teacher, “UCU Events” service for targeting university events and news.

3.2 Project scope

The project involves the creation of a prototype in the form of an application for a mobile phone (both IOS and Android), which will provide two implemented use cases:

1. UCU Events
   A platform that will allow students to conveniently receive information about events at the university and effectively manage their time, and the university information department to conveniently notify students. The platform will display events in the mobile application and give you the possibility to filter them according to the preferences of users who can customise them in the filter settings.

2. UCU Phonebook
   This service will allow students and teachers to find short information about UCU students based on their name.

3. UCU Calendar
   It will look like a typical calendar, but it will have many valuable functions that users can use. With this calendar, you can filter certain information: weekends, holidays for students, weeks of independent work, exam sessions, graduation celebrations, etc.

Chapter 4

User Experience

4.1 User Research

Before the development of the application, I need to know the needs of the community in the university.
On this task, I was working with my team(Nazar Mamonov, Khrystia Sliusarchuk, Oleksandr Dubas, Yarema Fylypchuk, Yevgen Doom). We found ideas of the functionality:

1. Events' filter - the user can view all upcoming events and filter them by interest;
2. Phonebook - a database with user data (name, mail, phone number), where the user could find the contacts of the person he needs;
3. Class schedule;
4. Academic schedule (vacations, holidays);
5. Dining room - menu, prices, schedule;
6. Audience booking - create a more user-friendly interface and a simpler procedure than the existing one;
7. Search for books in the library - whether they are available, on which campus they are, the opportunity to book a book and then go pick it up;
8. Creating or restoring a "Lokal" card, the ability to add it to the application itself;
9. Receiving corporate email when registering in the application;
10. Opportunity to see the balance of payment for tuition or other services;
11. Opportunity to see your grades, closed rates, debts, ratings, deadlines.
12. Opportunity to view the UCU hierarchy for employees (who is the boss, his contacts, who is the subordinate, etc.);
13. Campus map - a general map of the campus, where you can see the schedule of the Central Office, library, dining room, cafe, search for audiences.

To find out what we should implement in the application, we interviewed 13 students of the university. After that, we decided to apply these features: Events' filter Phonebook Academic schedule

- Events' filter
- Phonebook
- Academic schedule

4.2 User stories

4.2.1 Log In/Sign In

Story №1 As a user I want to sign up in the UCU App, so that I can use it.

Acceptance criteria:
Given that User is unregistered:

1. If he opens the UCU App, the 'LogIn' page opens;
2. If User clicks on the 'Sign Up' button, the system redirects him to the 'Sign Up' page;
3. On the 'Sign Up' page user needs to fill 4 fields:
   - Full Name - string, max - 100 characters, no numbers allowed, required;
   - Email Address - his own email "ucu.edu.ua" domain"
   - Phone - string, of type +38 (country code) + varying number of integers, letters not allowed, required;
   - Password - must be longer than 6 symbols, required;
4. If some of the required fields does not meet the constraints → the incorrect input is highlighted red, the error message: "Invalid input" displays above the corresponding field;
5. If some of the required fields are left empty → the empty field is highlighted red, the error message: "This field is required" displays above the corresponding field;
6. If User clicks on the 'Sign up' button → redirected to the 'Verify Email' page (if all of the fields were correctly filled);

Story №2 As a user I want to log in the UCU App using my email address and password, so that the system can authenticate me.

Acceptance criteria:
Given that User is registered and logged-out:

1. If he opens the UCU App, the 'Log In' page opens;
2. If User fills the 'Email' and 'Password' fields using his authentication credentials and clicks on the 'Log In' button → redirected to the '...' page (if the credentials were valid);
3. Otherwise, the error-message pops up: "Invalid email address or password", User is not logged-in and stays on the same page;
4. In case User does not remember his password, he clicks on the 'Forgot password?' button and the system redirects him to the 'Forgot password' page;

4.2.2 Home

Story №1 As a user I want to know basic information about the university, so that I will be well informed.

Acceptance criteria:
Given that User is signed-in:

1. if he clicks on the 'Home' button → redirected to the 'Home screen' page;
2. On the 'Home screen' page the basic information about departments, student organizations, etc. is displayed in boxes;
3. If User clicks on one of the boxes → a 'Detailed info' pop-up appears with basic information about that department, student organization, etc. (title, contact information and short description);

4.2.3 Academic Calendar

Given that User is signed-in:

1. If he clicks on the 'Calendar' button → redirected to the 'Academic calendar' page;
2. On the 'Academic calendar' page the calendar is displayed starting from the current month and 2 months after;
3. If User clicks on the 'Filter' button → a 'Calendar filter' pop-up appears;
4. On a 'Calendar filter' pop-up User can click and thus choose one or more checkboxes:
   - Weekends;
   - Holidays for students;
   - Week of independent study;
   - Exam session for 4th year students;
   - Exam session for 1st-3rd year students;
   - Graduation celebrations;
5. All the checkboxes are connected with logical AND;
6. If User clicks on the checkbox again → checkbox becomes clear;
7. If User clicks on the 'Cross' button → 'Calendar filter'; pop-up closes
8. If User clicks on the 'Reset' button → all the checkboxes become clear and User can proceed with selecting them from scratch;
9. If User clicks on the 'Apply' button → redirected to the 'Filtered calendar' page;
10. On the 'Filtered calendar' page the calendar is displayed with the corresponding dates circled with different colors;

11. If User clicks on a circled date → the pop-up appears with basic information about that date (date, title and short description);
12. If User clicks on the 'Cross' button → pop-up closes
13. On the 'Filtered calendar' page if User clicks on the 'Calendar' button → redirected to the 'Academic calendar'

4.2.4 Events Filter

Story №1 As a user I want to see all the events so that I will be aware of what is happening in the university.

Acceptance criteria:
Given that User is signed-in:

1. if he clicks on the 'Events' button → redirected to the 'Events' page;
2. On the 'Events' page all the possible events are ordered by date along with the short information about them on the picture: topic, date and lecturer;

Story №2 As a user I want to filter events, so that I can find interesting ones for me to attend.

Acceptance criteria:
Given that User is signed-in:

1. On the 'Events' page, if he clicks on the 'Filter' button → a 'Filter' pop-up appears;
2. On a 'Filter' pop-up, User can filter events by 3 criterias:
   - Location - dropdown: Sheptytsky Center, Kozelnytska 2a, Sventsitskoho 17, Khutorivka;
   - Category - dropdown: Sociology, IT, Religion, General..
   - Date - dropdown: This day, Next 7 days, Next 14 days, next month.
3. All the criterias are connected with logical AND;
4. If User clicks on the 'Cross' button → 'Filter' pop-up closes;
5. If User clicks on the 'Reset' button → filters become clear and User can proceed with selecting them from scratch;
6. If User clicks on the 'Apply' button → redirected to the 'Filtered events' page (in case there exist events which meet the constraints of the filter);
7. Otherwise, a funny meme or message pops-up: "There are no events, which meet your constraints";
8. On the 'Filtered events' page the filtered events are ordered by date along with the short information about them on the picture: topic, date and lecturer;
9. On the 'Filtered events' page if User clicks on the 'Events' button → redirected to the 'Events' page with all of the possible events;

Story №3 As a user I want to go to the "Event details" page, so that I will know more information about the event.

Acceptance criteria:
Given that User is signed-in and is on the 'Events'/'Filtered events'/'Selected events' pages:

1. if he clicks on an event → redirected to the 'Specific event' page;
2. On the 'Specific event' page picture, topic, categories, date, time, location, people who are attending and description are displayed;
3. If User clicks on the 'Register' button → the message pops up: "Successfully registered!";

Story №4 As a user I want to be able to like events, so that I can easily find them later on.

Acceptance criteria:
Given that User is signed-in and is on the 'Events' page:

1. if he clicks on the 'Heart' button→ redirected to the 'Selected events' page;
2. On the 'Selected events' page all the events liked by him are ordered by date along with the short information about them on the picture: topic, date and lecturer;
3. if User is on the 'Specific event' page, if he clicks on the 'Heart' button → event is added to the 'Selected events' page;
4. If User clicks on the 'Heart' button again → event disappears from the 'Selected events' page;

4.2.5 Phonebook

Story №1 As a user I want to find other Users, so that I can easily contact them.

Acceptance criteria:
Given that User is signed-in:

1. if he clicks on the 'Search' button → redirected to the 'Search' page;
2. 'Search' page consists of one 'Search' field with the 'Search' placeholder;
3. if User clicks on the corresponding field → the placeholder disappears, the keyboard appears and User can start typing the search query;
4. if User clicks on the 'Cross' button → search query becomes empty;
5. if User clicks on the 'Reset' button → the placeholder appears, the keyboard disappears;
6. if User clicks on the 'Go' button → the list of Users, who meets the search query, and information about them (their images, full names, faculties, courses) displays;
7. if User clicks on the "Full name' button → a 'User Info' pop-up appears;

8. On the 'User Info' pop-up User can see the information (image, full name, faculty, course, phone number, email) of the corresponding User;

9. if User clicks on the 'Cross' button → 'User Info' pop-up closes

4.2.6 Profile

Story №1 As a user I want to see basic information about me.

Acceptance criteria:
Given that User is signed-in:

1. if he clicks on the 'User' button → redirected to the 'User profile' page;

2. On the 'User profile' page User's image, status (Student/Teacher/Staff), contacts (phone number and email) are displayed. If User is a student, then information about faculty, specialization and rating is also displayed;

Story №2 As a user I want to change the language of the app, so that it is more comfortable for me to use it.

Acceptance criteria:
Given that User is signed-in and is on the 'User profile' page:

1. if he clicks on the 'Settings' button → a 'Settings' pop-up appears → clicks on the 'Change language' button→ a 'Change language' pop-up appears;

2. On a 'Change language' pop-up, User can click 2 buttons:
   - English
   - Ukrainian

3. If User clicks on the 'Cross' button → 'Change language' pop-up closes;

4. If User clicks on the 'Change' button → the language of the whole app changes to the corresponding language and User is redirected to the 'Settings' pop-up;

Story №3 As a user I want to sign out, so that I can login from the new account.

Acceptance criteria:
Given that User is signed-in and is on the 'User profile' page:

1. if he clicks on the 'Sign out' button → redirected to the 'Sign in' page;

4.3 Low Fidelity Prototypes, Features

A **prototype** is a working version of a product that you can interact with. **Low Fidelity** prototypes exclude the presence of design in any form and focus exclusively on logic. Very often, such prototypes are executed stylised like freehand sketches (sketch).

The first image is the prototype of the "Login" page at the beginning of the development. It has two text fields, where the user could type his credentials, they will be sent to the server, and as a response, I would get an access token. But after the discussion with the IT department, we understood that this approach is not secure.

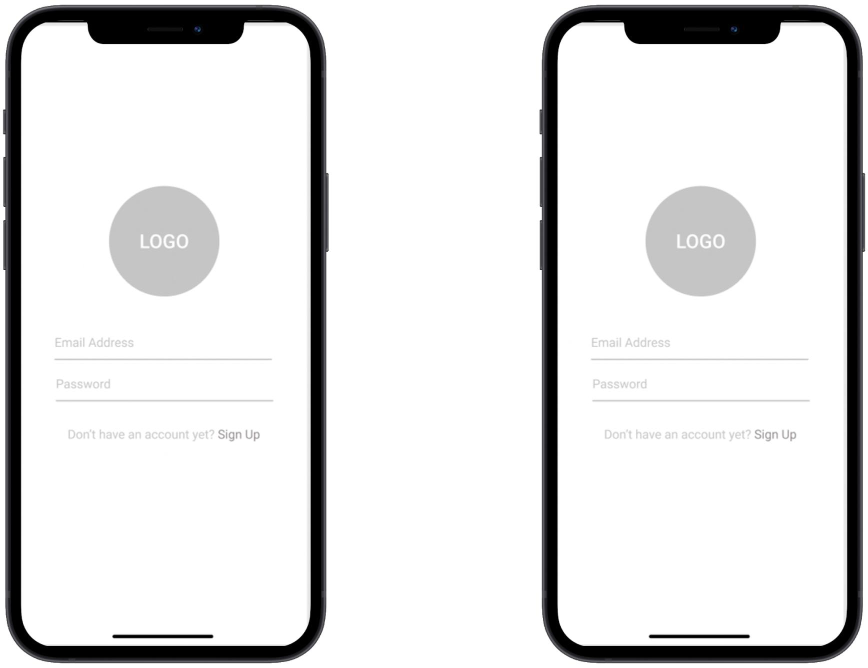
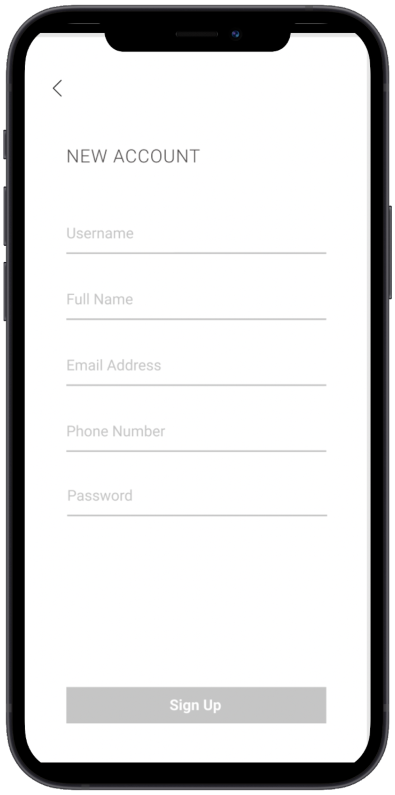

LOGO
Email Address
Password
Don't have an account yet? Sign Up
LOGO
Email Address
Password
Don't have an account yet? Sign Up

FIGURE 4.1: "Login" page prototype

It has a high risk of leaking the credentials.
The best way would be to create a login using Google SSO(Single sign-on). The user would not enter any data in our application; he will be redirected to the Google authentication.

4.3.1 Registration page



NEW ACCOUNT

Username

Full Name

Email Address

Phone Number

Password

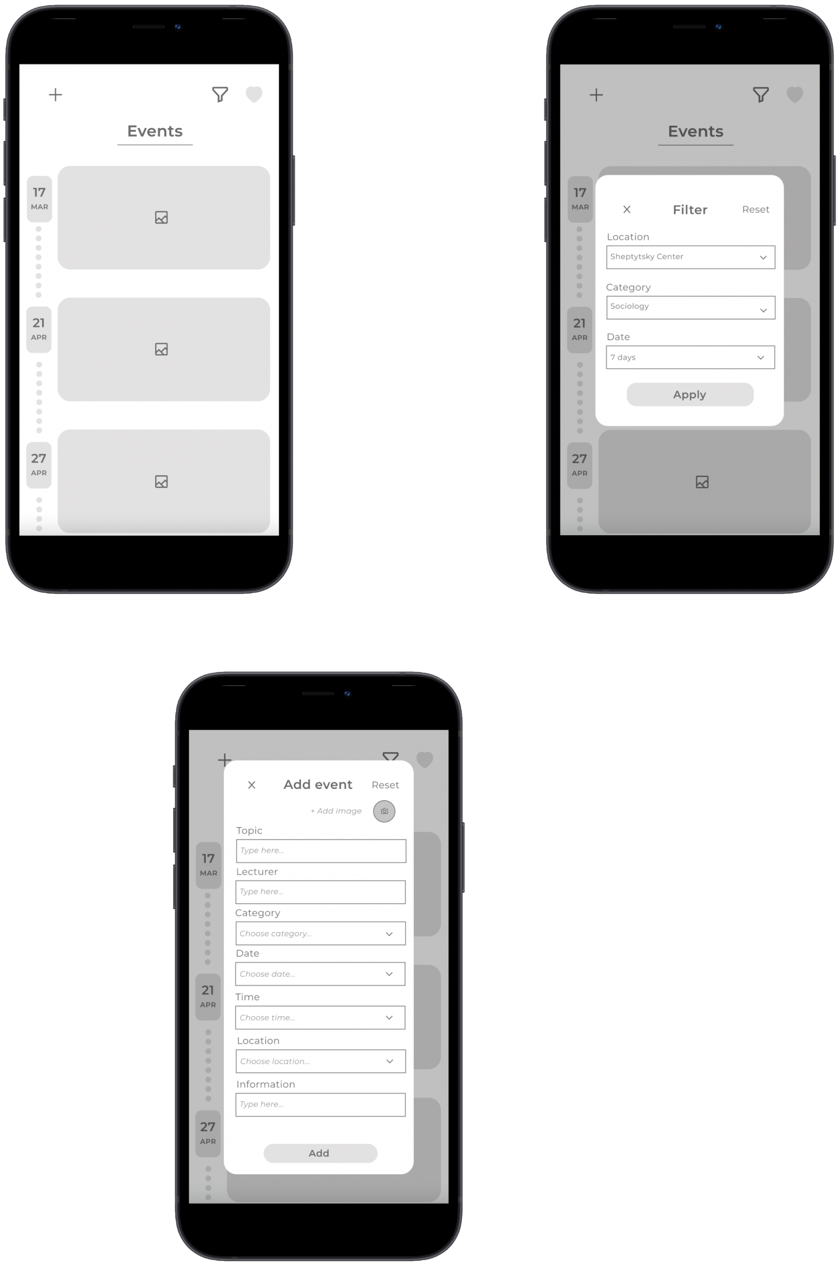
Sign Up

FIGURE 4.2: "Registration" page prototype

In discussion with the administration and IT Department, we decided to add the registration of the new student in the application. How does it work now? When submitting documents to the university, future students come to the particular room, sit down in front of the computer, fill in all the data in some program, and create records in the university's database. This procedure is time-consuming because you need to stay in the queue to the classroom, but you can't do it from anywhere and at any time.

In the application, new users can register, filling in their data. When their university's mail is created, they will be notified by the email entered when registering in the app.

4.3.2 Events page



×
Filter
Reset
Location: Sheptytsky Center
Category: Sociology
Date: 7 days
Apply

×
Add event
Reset
+ Add image
Topic: Type here...
Lecturer: Type here...
Category: Choose category...
Date: Choose date...
Time: Choose time...
Location: Choose location...
Information: Type here...
Add

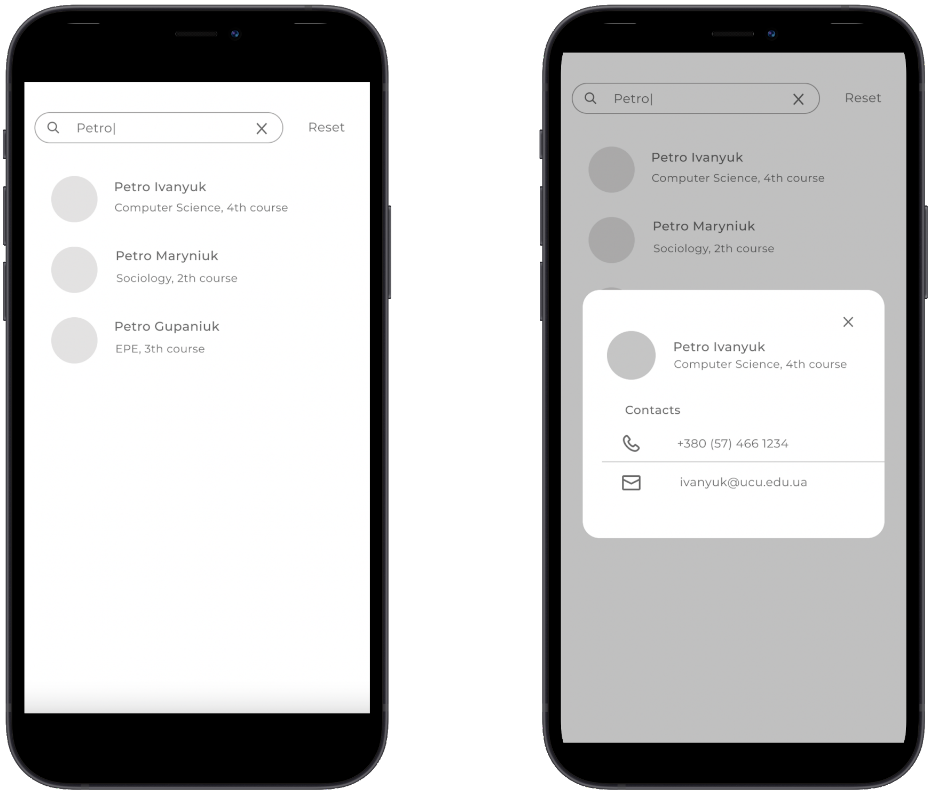
FIGURE 4.3: "Events" page prototype

Events are made as a scrollable list of items. You can see events' basic information and the date, when those will take place. On top of the screen, there are a few buttons:

- "Plus" button - it is only visible for users, who have "Writer" permission, it will show the creation pop-up.
- "Filter" button - will show the pop-up with filter options, and when you apply them, the list will be reloaded with events, which are related to selected options.

– "Favorite" button - your events list will be reloaded and only those events, which you had already selected as favorite, would be shown.

4.3.3 Phonebook page



Petro Ivanyuk
Computer Science, 4th course

Petro Maryniuk
Sociology, 2th course

Petro Gupaniuk
EPE, 3th course

Petro Ivanyuk
Computer Science, 4th course

Petro Maryniuk
Sociology, 2th course

Petro Ivanyuk
Computer Science, 4th course

Contacts

+380 (57) 466 1234

ivanyuk@ucu.edu.ua

FIGURE 4.4: "Phonebook" page prototype

There is a text field on top of the screen. User can type any name and when he stops for 2 seconds, the application makes the request to fetch users with such name and those users will be shown. If you tap on some user, you will show the pop-up with users contacts.

4.3.4 Profile page

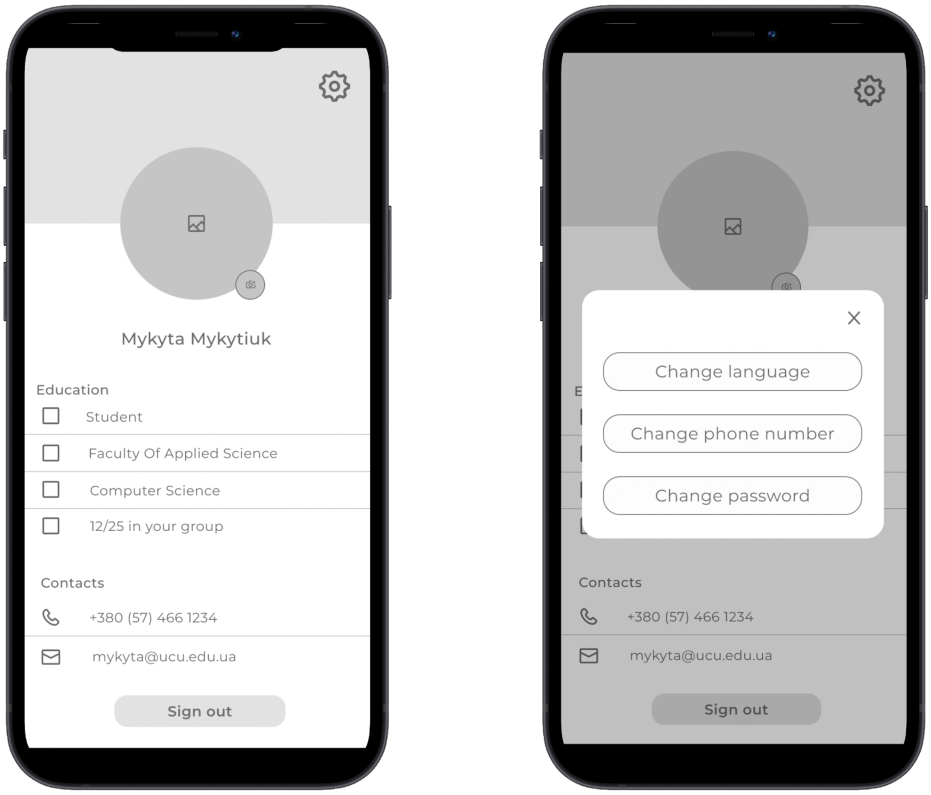


FIGURE 4.5: "Profile" page prototype

On this screen, User can see his general information:

- General user's information:
- Image with the possibility to change it;
- Group(Student, Teacher,..);
- Faculty and department;
- Group success rating;
- Phone number;
- Email address;

On the top of the screen, there is a "Settings" button on top of the screen, which gives you the possibility to:

- Change language
- Change phone number
- Change password

Chapter 5

System development

5.1 Technology

One of the best cross-platform mobile technology for now is Flutter. It is a mobile UI framework created by Google and released in 2017. It lets you create a native applications for both platforms(Android and IOS) with only one codebase.

"Flutter consists of two important parts:

- An SDK (Software Development Kit): A collection of tools that are going to help you develop your applications. This includes tools to compile your code into native machine code (code for iOS and Android).
- A Framework (UI Library based on widgets): A collection of reusable UI elements (buttons, text inputs, sliders, and so on) that you can personalize for your own needs"[Thomas, 2019]

Flutter uses a programming language "Dart", which is object oriented, strongly-typed.

5.2 Mobile application architecture

BLoC (Business Logic Components) is a software architecture pattern. It was introduced in 2018 at the Dart conference. BLoC separates UI and business logic, which makes it much easier to manage and test the code.

First scheme demonstrates the main BLoC's principle, we have widgets(UI components) which send "something" to BLoC with 'Sink' and get "something" back with 'Stream'.

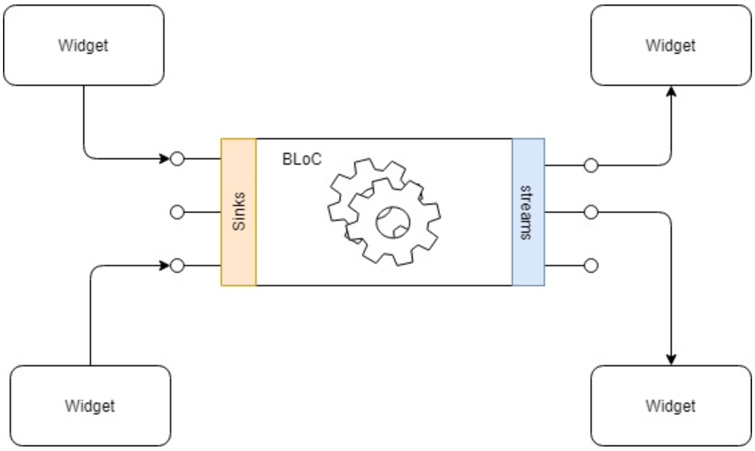


FIGURE 5.1: BLoC architecture

Bloc is based on the Stream. What is the 'Stream'? Steam is an asynchronous events of data. On the scheme we there is 'Sink'. It is an abstraction, which allows us to send data to our stream. We can subscribe on the stream. When new data comes to the stream, its subscribers are notified about that. You can imagine that this is a kind of a conveyer, which accepts data and immediately sends notifications to all subscribers, that his state has been updated.

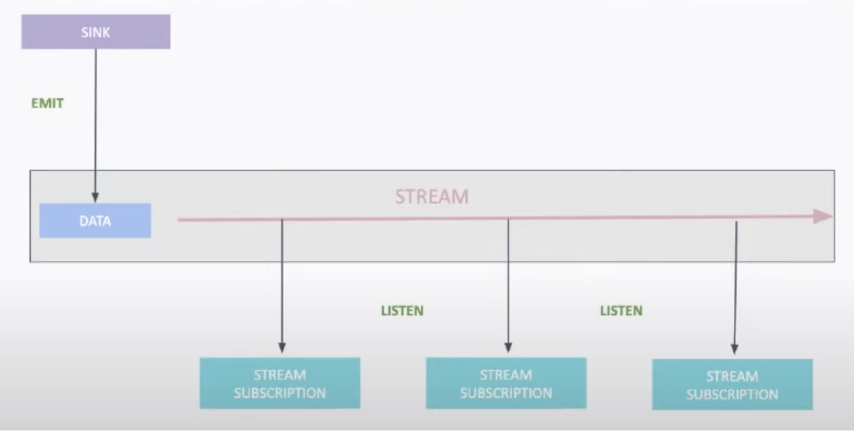


FIGURE 5.2: Stream

Summing up, here is full workflow of how the BLoC works:

1. User does some interactions with the UI
2. The UI sends event to the bloc's stream
3. Bloc knows, that the stream has a new event and what to do, when this event had come: whether he should change data, or make new request to fetch another data, and etc
4. Bloc changes the state of the UI
5. The UI rebuilds itself with new data

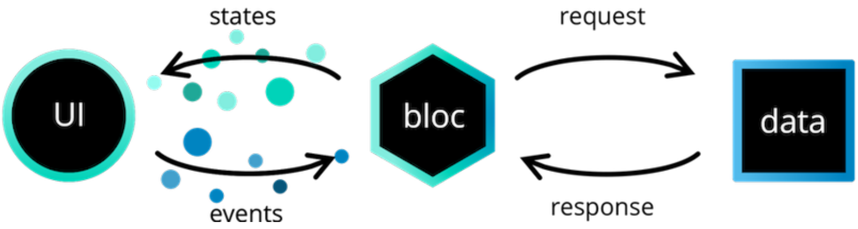


FIGURE 5.3: BLoC communication

5.3 Project structure

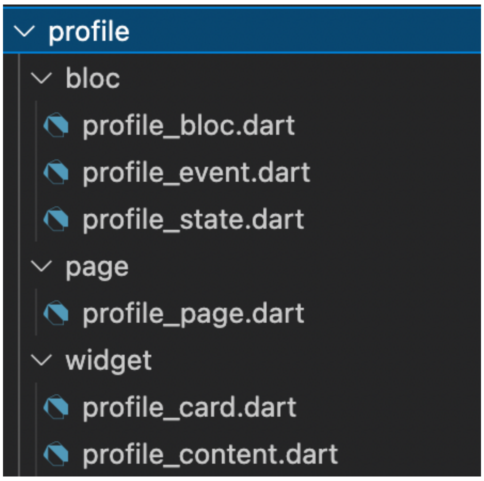
In the project, I separated three main layers:

- Presentation
- Domain
- Date

5.3.1 Presentation layer

In the 'Presentation' layer I have files, which correspond to the user interface and logic(BLoC), which directly communicate with widgets. For example, here I have a directory, which corresponds to the Profile screen.

- Page directory is the main file, which contains the "Scaffold" widget and connects to it's bloc.
- Widget directory contains all the widgets, which this page consists of.
- Bloc directory contains the logic for this page, but it does not have the implementation of any requests, such code is separated.



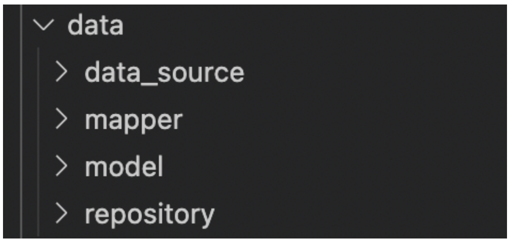
```
profile
  bloc
    profile_bloc.dart
    profile_event.dart
    profile_state.dart
  page
    profile_page.dart
  widget
    profile_card.dart
    profile_content.dart
```

FIGURE 5.4: Presentation layer

5.3.2 Data layer

In the "Data" layer I have:

- Data source works as an API client, where all requests are made.
- Model stores all the parsed objects, which we retrieve with requests from the "Firebase" database.
- "Repository is the brain of the data layer of an app. It handles data from remote and local Data Sources, decides which Data Source to prefer and also, this is where data caching policy is decided upon"[Rešetár, 2019].



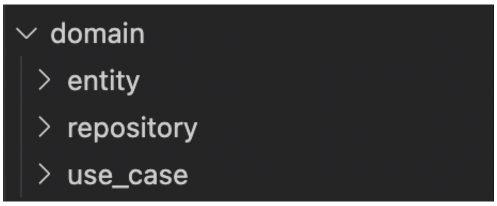
```
data
  data_source
  mapper
  model
  repository
```

FIGURE 5.5: Data layer

5.3.3 Domain layer

The “Domain” layer is the mediator between data and presentation layers.

- Entity is a full representation of our object from the database. The difference with a model is that entity’s fields are non-nullable. It can also have some additional methods.
- Repository in the domain layer is the collection of abstract classes for our repositories.
- Use case is where the business logic is executed.



```
∨ domain
  > entity
  > repository
  > use_case
```

FIGURE 5.6: Domain layer

5.4 Server-side architecture

Before I start coding, I should understand how our application would work and will be integrated with other services, APIs. The whole picture of the architecture(built with ‘miro’):

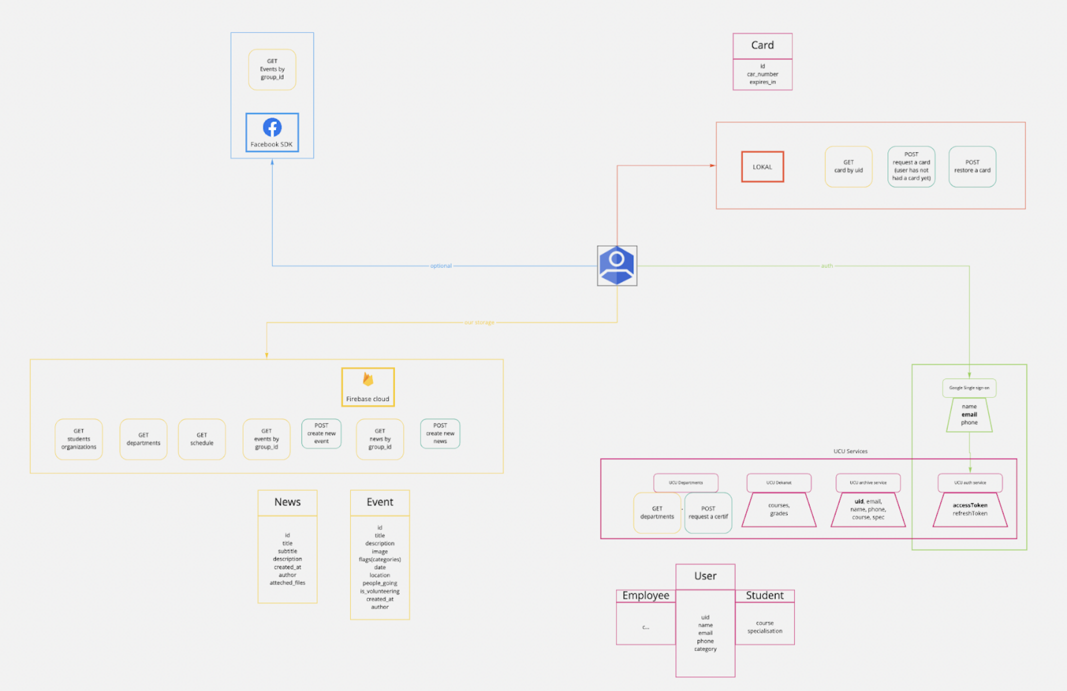


FIGURE 5.7: Server-side architecture

5.4.1 User authentication

User authentication is implemented with Google SSO, making the application more secure because it prevents data leakage.
How it works: the user clicks on the 'Sign-In with Google' button, and he is being redirected to the 'Google authentication' page in the browser. He types his credentials, and as a response to this, Google gives us his information about him and id token, which we can send to the UCU server to identify this user in our system. And in response to this request, we will get an access token, which will determine the user and allows him to browse the application, make requests, etc.

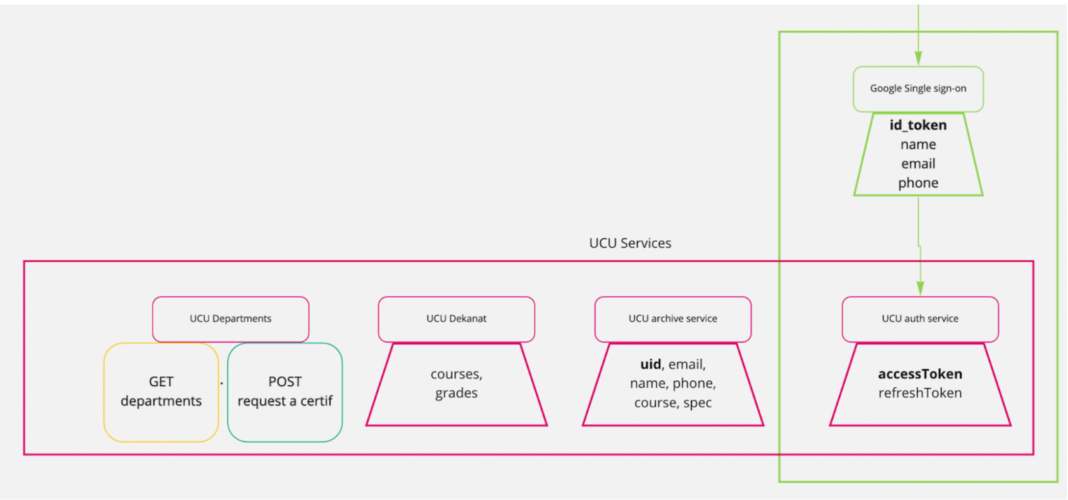


FIGURE 5.8: Firebase architecture

5.4.2 Firebase

The only backend storage, which the application has, is a "Firebase". "Firebase" offers us two different databases. This is discussed in more detail in the section 5.6
In addition to the database, "Firebase" has a Database REST API, which works like a standard REST API. I don't need to write anything by myself, just to have some data in the database.
For example, if I have a "Firebase" project, it has a database with a collection called "events", and we want to retrieve all the data inside of it; we can do this in Flutter like this:

```
@override
Future<List<EventResponse>> getEvents() async {
     QuerySnapshot snapshot = await
dbRef.collection("events").get();
     final events = snapshot.docs ?? [];
     return events.map((json) =>
EventResponse.fromJson(json.data())).toList();
}
```

Where **dbRef** is a reference to our database;

```
final dbRef = FirebaseFirestore.instance;
```

“FirebaseFirestore” is an object from the "Firebase" library in Flutter. Our project is already set up with "Firebase" in the “main” function, so “FirebaseFirstore.instance” already knows which project it is related to.
Set up of the "Firebase" in the project for both platforms. All the data are hidden because of privacy.

```
final FirebaseApp _ = await Firebase.initializeApp(
     name: 'db2',
     options: Platform.isIOS || Platform.isMacOs
          ? FirebaseOptions(
               apiId: "XXXXXX",
               apiKey: "XXXXXX",
               projectId: "XXXXXX",
               messaginSenderId: "XXXXXX",
               databaseURL: "XXXXXX",
          ) : FirebaseOptions(
               apiId: "XXXXXX",
               apiKey: "XXXXXX",
               projectId: "XXXXXX",
               messaginSenderId: "XXXXXX",
               databaseURL: "XXXXXX",
          ),
);
```

I can make several requests with Firebase Database REST API:

- GET students organisation;
- GET departments;
- GET schedule;
- GET events by groupId;
- GET news by groupId;
- POST creates a new event;
- POST create new news;
- GET user;
- PUT user;

5.4.3 Lokal

There is a loyalty program In the university’s dining room that students, teachers and employees have a discount of 25 percent. This is achieved with the company ‘Lokal’ and ‘!FEST”. They give us special discount cards, which we can use when paying for purchases at the checkout. But it always takes a long time to issue or restore the card when losing it. The possibility to have the card in the application, to issue and restore it, would make the life of the community more relaxed.
Therefore, I had a few conversations with managers from ‘Lokal’ to implement such a feature in the application. They are interested in this offering.
For this, I sent them features(requests), that they need to add to their API for this realization.

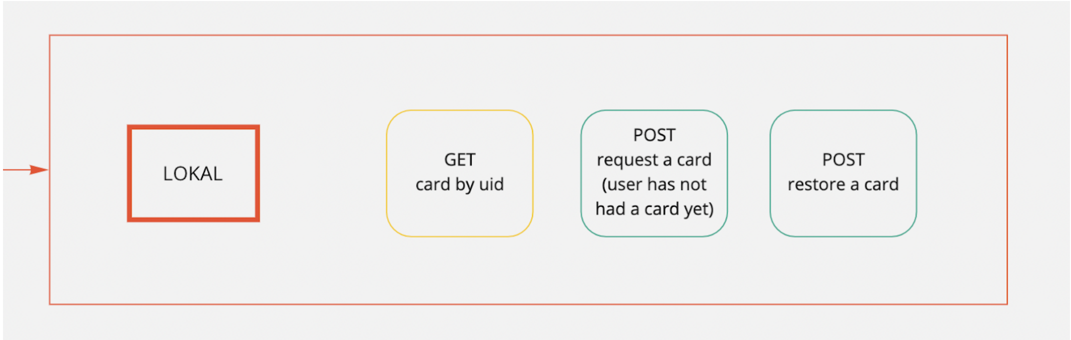


FIGURE 5.9: Lokal architecture

To communicate with 'Lokal', I need several requests:

- GET card by uid(Important note: uid for the user in the 'Lokal' database must be the same as the uid in the UCU's db);
- POST issue a card;
- POST restore a card;

5.5 Security

The main part where we need to think about security is a leak of user's data and credentials.
For the authentication, I use Google Single Sign-On. With this, the user doesn't need to enter any email or password in the application, he does it directly on the Google authentication page.
I do not store any data in the application, each time user opens it, the app fetches new data.

5.6 Database Design

"Firebase offers two cloud-based, client-accessible database solutions that support real-time data syncing:

- Cloud Firestore is Firebase's newest database for mobile app development. It builds on the successes of the Realtime Database with a new, more intuitive data model. Cloud Firestore also features richer, faster queries and scales further than the Realtime Database.
- Realtime Database is Firebase's original database. It's an efficient, low-latency solution for mobile apps that require synced states across clients in real-time."[Google, 2021]

I've already been working with 'Realtime Database' and it was easy and clear. But, for this project, I decided to use a new one - 'Cloud Firestore'. My database should store different collections:

1. Users information
2. Events

3. Calendar events
4. Lessons schedule
5. IT Departments information
6. Student Organizations information
7. Feedbacks

Chapter 6

Conclusion

In this project, we build a front-end application for the Ukrainian Catholic University. In addition to the application, there is a "Firebase" database, where all the data is stored. Unfortunately, there is no real data in the application, I am using mocked one. But it can easily be migrated to the real back-end server, if such ever exists.
The user experience was worked out in[Goodfellow et al., 2014] details, the main problem stills the user interface. Workable features are: UCU Events, UCU Phonebook, UCU Student Organizations, UCU Departments, UCU Profile.

Bibliography

California San Francisco, University of (2020). "UCSF Mobile App". In: URL: https://campuslifeservices.ucsf.edu/bts/services/ucsf_mobile_app.

Goodfellow, Ian J. et al. (June 10, 2014). "Generative Adversarial Networks". In: arXiv: 1406.2661 [stat.ML]. URL: https://arxiv.org/pdf/1406.2661.pdf.

Google (May 11, 2021). "Choose a Database: Cloud Firestore or Realtime Database". In: URL: https://firebase.google.com/docs/database/rtdb-vs-firestore.

Rešetár, Matt (Sept. 12, 2019). "Flutter TDD Clean Architecture Course [5] – Contracts of Data Sources". In: URL: https://resocoder.com/2019/09/12/flutter-tdd-clean-architecture-course-5-contracts-of-data-sources/.

Store, App (2021). "NAUgo". In: URL: https://apps.apple.com/us/app/nau-mobile/id476727986.

Thomas, Gaël (Dec. 12, 2019). "What is Flutter and Why You Should Learn it in 2020". In: URL: https://www.freecodecamp.org/news/what-is-flutter-and-why-you-should-learn-it-in-2020/.