Lappeenrannan teknillinen yliopisto

School of engineering science

Sofware Development Skills

Daniil Komov, 000503581

LEARNING DIARY, Mobile development MODULE

**LEARNING DIARY**

06.05.2022

I read the information I needed for starting the course, such as general information and environment setup the main goal of the course is now clear to me, that is exactly as I wanted: creating a project to practice and show skills and finding inspiration in programming world.

I chose mobile frontend development module because I have a corresponding course on flutter at the moment, and I liked it a lot to work in a mobile environment that I decided to register for this course. I didn’t have to set up code editor because I already had it before starting this. I’m going to use Android Studio, because it is the one I was using for some time now. For the first time learned what is GIT, tried to set it up, created github repository, everything went fine, the video-explanation about GIT is very clear.

13.05.2022

I decided to use Android Studio as my code editor since I’ve used it before already and it feels comfortable. I wrote an e-mail to the teacher asking if I can make an app using flutter and I got positive answer. I decided to create a chatting app. I started by creating the project in Android Studio, then I created the home screen which is the one where you choose if you register or log in. I made the buttons, edited the design, and then I moved forward to creating login and register screens. The idea was to create a general login screen where user would enter his e-mail and password and press log in. I googled how to create TextFields so that they look how I wanted them to look(for example, how to make cursor orange etc.). For most of the time I used official flutter website for this. I had some problems passing e-mail and password variables through the code, but after some brainstorming I figured out how to do it.

14.05.2022

I finished the design of the login page and started to write the register page. I did nothing special on the register page since I wanted it to look kind of the same as the log in page, so I just copied the code from the log in page and pasted it to the register page, then I changed some of the words, log in button to register button etc.

After that a problem for me appeared: I didn’t really know how I’m going to implement all of the cloud messaging into the app. At first I had a thought to make my own server which would store the messages on the SQL database, but then I thought that it would be too difficult for me to make since I’m not very experienced in creating all of this yet. So, I went to the Internet and looked out for what other people make. I started to google and found out about such thing as Google Firebase. I decided to use it since it seemed very simple and straight forward compared to other databases.

15.05.2022

I watched an official video from Google Firebase about how to create a project in their system and how to implement it into the flutter project. I did exactly what is done in the video and added implementation of the Firebase Auth to the app. Firebase Auth is user authentication system which allows to easily identify users in the app. By further googling I found out how to make user log in with variables in flutter. I added the line of code to log in page. At this moment I started to check the system for a long time, pressing log in button and watching if the user actually logs in in the Firebase. For a long time I didn’t know how to make flutter check if the user is logged in or not. Also I wanted to implement an error message if the password is not correct or if there are other errors, but I also had trouble with that. I googled how to make flutter react on the error messages given by the Firebase, and there were answers in the internet, but I tried them all and nothing worked: the program just didn’t react on the error. So, I decided to leave the problem till some other day. I committed changes in git and tried to push it to Github, but Github failed saying that some files weigh more that 100mb. I had to implement Github large storage, which didn’t work for some reason. I tried to troubleshoot the problem in the internet and found some solutions I need to do in order to make this work, but when I tried on of the solutions it returned my git repository to the previous commit and the last commit got deleted. I got mad and postponed the Github problem until the other day.

22.05.2022

Before trying to resolve the problems I had last time, I decided to go for something else so that I have a clear mind on the problems so that I solve them. I decided make my app look better by adding special animation. I have never tried to implement animations, so I thought it would be a nice thing for me to develop because I will learn something new. So, I googled official Flutter site about animations, and found out about a method to add an animation through the route. I tried to implement that on the transition between main page and the log in page, as well as between main page and register page. The attempt was successful after some code editing.

So, after I had something successful, I decided to proceed with the problem connected to the authentication status. I was struggling for a long time, trying to make flutter recognize the error of the server responding negatively, but it just didn’t work and I don’t know why. But, in the end, I realized that for the moment I was doing something I didn’t need to do at the moment, and I read further about Firebase Auth and the line of code “FirebaseAuth.instance.currentUser =null” and I finally made the machine to understand whether the user is logged in or not. It was much more simple than I thought. However, I still have to figure out how to make flutter show error messages specific to different errors the server replies with.

After getting over the problem, I planned to make a loading animation. I googled some loading animations, and I made a separate screen which would work only when the app needs it. So, I made it work after the press of the log in button. I thought of the design(of course), and I think it looks cool for now. I created another file with the “groups” screen on which I placed one “Sign out” button to test if the system actually responds when log in attempt is done. Now, when I press the log in button with the right credentials, it goes to loading screen, then after some seconds it goes to the test screen with sign out button. If writing wrong credentials, after loading it just returns back to the log in screen.

Later this day I also decided to make an alert dialog saying that the credentials are wrong. I had an experience doing that already, so I just added it as a separate widget(with editing the design as well).

23.05.2022

Today I decided to proceed with making the groups page. I decided to implement firestore at first. This time I decided to not watch another youtube video about this, but to dive in it myself. I started firestore in my project on firebase website. I was thinking for a long time how I would implement messaging in the firestore, and at this point I only came up with just creating “Groups” collection, in which there would be Group names. Creating collection was quite simple. But then I wanted to implement something more interesting than just a simple listView widget in the groups page. I searched the internet for the different solutions, and I found the ClickableListWheelScrollView, which is a list that looks like a wheel that you can spin and choose the group. I copied the widget from the website, optimizing it to my own case. I was struggling with the widget, though. The problem was that this widget required ItemCount, and I needed the groups to be counted in the firestore and then the integer had to be passed to the ItemCount field, which was very problematic for me to do. The variable didn’t want to be passed to that field. So, I decided to clear my mind and do this on some other day.

29.05.2022

So, I continued with the widget. I thought about how to make the program count the groups, and I couldn’t figure out anything better than just making another collection in the firestore named “Counter”, in which the number of groups would be showed. That way I created the variable which gets the number of all of the groups and I wrapped this into a method “getCounter”. That way there is a global variable which gets the value from executing the method. After that I successfully created the widget, but there was still a problem. I created some groups in the firestore as for testing, but when I checked the app, it gave error “Bad state”. I thought the reason was that the program just doesn’t have time to load the information from the server. So, I thought that I need to implement some future builders. I looked through some websites and I found good examples of implementing them into the flutter app, and I wanted them to be exactly the same in my program as on website, so I copied them. I liked the future builders a lot because it implements exception handlers for the snapshots. So, I pasted them into my code and changed some of the returnables. For example, I used the loading widget I made so that it is showed when snapshot data is loading. Now the groups load as it is intended, and the group names are now on the screen. I tried to make list items to be clickable and I tried to make it so that you could press the items and so that the program switches the page when you choose the group. Here I ran into the problem again: for some reason, the buttons didn’t work, It didn’t go into the new chat page that I created. However, after some testing, I figured out that the buttons actually do work if the app is being started with the user logged out of the app. I got an idea that this is connected to the first launch of the app. I decided that I’ve done enough for the day and I decided to continue later. Summer was starting and I had my summer holiday trip soon, so I decided to continue my project after my summer holiday.

17.06.2022

After returning from my summer holiday trip, I decided to continue with the project as I don’t have many days left. So, I was trying to solve the problem with the buttons, and I also figured out that not only the buttons in the wheel list view don’t work, but also the sign out button. Moreover, it turns out that the sign out button actually responds by signing the user out, but the page transition doesn’t work. In the console in shows the error connected to the context of the navigator. After some brainstorming and internet searching processes, I figured out that the problem was actually hidden in the way the app starts the groups page after if checks for the user log in status when launching the app. Turns up it was just enough to change the code at the point where the app starts the groups page by adding the MaterialApp(home: Groups()) piece of code. Now the groups open in any condition and the transition happens. So, after myself being satisfied, I moved forward and I started doing the chat page. I thought about what to do first and figured out that the most difficult part for me here would be the messages themselves, so I decided to make the message bar and send button at first. It was actually pretty easy to do: I copied the code from the log in page(TextField widget) and pasted into the chat page, then changed the hint text. After that I copied the log in button, pasted to the chat page and then changed it so that it looks like a send button. I added a nice icon on the button, and made it all look good.

21.06.2022

This day I decided to implement the system of messages being sent to the server. I decided to add functionality to the message bar and send button I created last time. But before doing that, I decided to figure out how the system would work in the firestore. For that I decided to watch a video on youtube, and I found the right one. The link for the video is here: <https://www.youtube.com/watch?v=wHIcJDQbBFs&list=LL&index=1&t=1203s>

I watched through the whole video and I thought that I could do the whole messaging referring to the example of this guy. I looked at how he implements the whole messaging and how makes the app to send messages to the server. So, I decided to do the similar system. However, I thought that what he does in the video is quite difficult and I decided to do it my own way. So, I took his idea of messaging, but I didn’t copy his classes and everything, I decided to make my own thing. At first I made the textField to accept the written message when it is inserted by adding onChanged field. I added a global variable which is being changed when the message is inserted. I also thought that it would be nice to add another container on the top of the screen so that it shows the name of the group. I made a container with the similar design and passed the group variable from the groups page so that it is shown in the chat page. Then I implemented functionality to the send button, so I same some queries that take the variable and make the message on the server with some specific info, such as the message, time when it was written, and the name of the user. At this point I realized that I still didn’t add the ability for the user to write his nickname when registering in the app, So I quickly decided to do that. Thanks to flutter and firebase it wasn’t a hard thing to do: I quickly made another TextField on the registration page, created another global variable and it works now. Also it was easy to make a query that would state the nickname in the message. After that I had to proceed with the messages and how they would be showed on screen, so I decided to watch the video again. But it was too late and I decided to sleep first.

22.06.2022

I watched the video again after breakfast, and I realized that the way he does everything is very complicated and I would spend ages by thinking about every action he makes. But I watched the whole thing, and I didn’t understand some parts, so I decided to copy his doing with my own editing. At some point I just decided to not do the same thing as he does, for example transforming the messages. I successfully implemented messaging by creating an expanded widget on the chat page in which I pasted the container with the message builder as a child. Message builder was something I didn’t really copy from him because I used Stream builder and he didn’t. The message widget itself I kind of copied from him, but of course with my own design. Afterall, I was very satisfied with the result, because it all worked very well when I tested the app. Unfortunately, after that, I faced another problem: after I wrote 10th message in a certain group, the message appeared not on the right place. The reason for that was that I didn’t understand how the guy did the transformation when creating “GetMessages” method, and in transformation he also included ordering by creation time, which I haven’t done. So, my messages had the same order as they appeared in the firestore storage, which was quite random after the 10th message. So, I started to brainstorm on how to make the messages display in order, and it all even came to creating another counter, now for the messages. I decided to continue on the next day.

23.06.2022

Today I had to figure out how to make the messages in order, and I decided to try ordering them like the guy in the video does. I tried doing it and I succeeded. It was much easier than I thought, I just had to change the timestamp a little. Not the messages are shown in order no matter how many messages you write. Before getting to work, I thought a little about what else I could implement into my app, and I thought that my sign out button on the groups page has to be changed to something else, and also I have to add change password feature. I needed something that could open with a press of a button and something that could have some options in it, but not a separate page. So I figured out to make a drawer: a curtain that opens with a tap on the button. It was very easy to make: I just went to official flutter instruction page and watched their video on how to make it. The only problem I faced was that it opened from another side. But then I thought that it looks better when it is opened from the left side. I decided to continue with it tomorrow.

24.06.2022

Today I had to make the drawer look fancy, so I thought that I could add a container on the header of the drawer on which there would be some beautiful icon and the nickname of the user. I already tried using some icons from FontAwesomeIcons when making sign out button, so I decided to use it again because I liked the icons as they look much better than the standard icons. At this point I searched the FontAwesomeIcons website for some icons, and I chose right arrow button for the drawer button, and some austronaut icon for the user nickname container. I changed the sign out button to look like a button for calling drawer, and then I pasted the austronaut icon into the container in the drawer, as well as made a text there which shows the nickname of the user. Then I made a column on the drawer field itself, where at this point I had to put two buttons: change password button and sign out button. I copied the sign out button before changing it to the drawer calling button, so I just pasted the sign out button into the drawer and made it to be on the bottom of the drawer.

Now I had to proceed with the change password function. For that I created a button, on which I placed the FontAwesomeIcon “key” and text. After that I created another file on which I would add the new class for making another page. It was again pretty easy to do, I had to write some code for creating a class, then I wrote a widget with another TextField and submit button, and I created a big text before the TextField stating that the user has to use 6 character password or otherwise the password won’t change. Adding functionality was also pretty easy, as I just created a global variable again, and I made a firebase command on the submit button, so that it changes the password of the current user on the value in the variable. Now I have quite a simple drawer with the user nickname, sign out button and change password function. Also, I edited the design a little, so that the drawer is not just black, but so that it has a line on the edge of it showing clearly its edge. At this point I thought that my app looks finished, as I didn’t have much time left. I decided that I would create some general groups in the firestore and all the users that have the app are gonna be able to use them.

25.06.2022

I decided that I need to get my project done already and I have to make my git repository up to date so that I could use a link in the course. In the past I had a problem with my project: I couldn’t push to GitHub because some of the files were too big for it. I tried Large File Storage today, but it didn’t work again, the terminal just states that the files are too big. I searched the internet a lot today and I still couldn’t figure it out. Then I figured out that the files that are too big are not even on the last commit I’ve made, turns out they are still on that previous commit I’ve made a month ago. For the whole evening I was trying to delete these big files on the previous commit(as I figured out that they are not even needed so much for the project), but I failed, unfortunately. I tried many methods, but nothing works. I decided to create a new repository on the GitHub, on which I would just paste my repository with the last commit on which I won’t have these big files. I hope I won’t get worse grade for that. Below I have a screenshot from android studio, a proof that I did Git commits. On the screenshot there is one commit highlighted with blue: it is the commit which I suspect has big files, because The project folder was named differently after this commit, and the path to the big files still contains the old folder name “Chattt”. I just created another repository folder on the desktop on which I made another Git, then I created another branch on GitHub. Then I made a video on how to open the project, readMe file in which I explain how to open the project, and also I saw that my repository should have my name, so I decided to make another txt file in which I just state my name and my student number.

Graphical user interface, text

Description automatically generated