Group 2
Ronas Baran, Nelly Friman, Hector Gonzalez Campos, Emil Ståhl

Android Prototype

A research project in collaboration with KTH and Stockholms University

1. Implementation

The Android prototype was developed with the Android Studio IDE in the Kotlin programming language. We used last week's Web App prototype built with JQuery as a template to create an Android App with a similar UI. We divided the work so each developer focused on its own part of the application. These parts were Home screen, profile, settings and information where the home screen includes the tests and questionnaires. Following is a description of each of these parts.

1.1 Home Screen

The app's home screen consists of a navigation bar along with cards that represents the different cognitive tests and questionnaires. The logic behind the home screen is written in the HomeFragment.kt file:

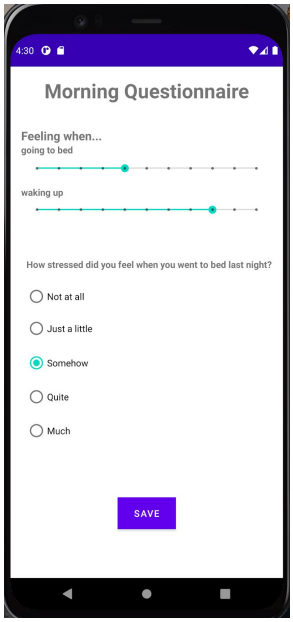
```
class HomeFragment : Fragment() {

    private lateinit var homeViewModel: HomeViewModel

    override fun onCreateView(
        inflater: LayoutInflater,
        container: ViewGroup?,
        savedInstanceState: Bundle?

    ): View? {
        homeViewModel =
                ViewModelProvider(this).get(HomeViewModel::class.java)
        val root = inflater.inflate(R.layout.fragment_home, container, false)
        val cardView: CardView = root.findViewById(R.id.card2)
        val cardLayout: RelativeLayout = root.findViewById(R.id.cardLayout2)
        cardLayout.setOnClickListener {
            val intent = Intent(context, Questionnaire::class.java);
            startActivity(intent);
        }
        return root
    }
}
```

As can be seen in the screenshot above the home screen consists of the navigation bar and the different cards for tests and questionnaires. Upon clicking the questionnaire card the user gets to the screen shown in the next subsection.

1.2 Questionnaire

The questionnaire screen consists of two sliders to determine the feeling when the user went to bed as well as waking up. Below this is a question with five alternatives where the user shall pick one. The code for this is written in the file Questionnaire.kt:

```
class Questionnaire : AppCompatActivity() {

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_questionnaire)

        val goToBedBar = findViewById<SeekBar>(R.id.goingToBedBar)
        val wakeUpBar = findViewById<SeekBar>(R.id.wakingUpBar)

        val radioGroupStress = findViewById<RadioGroup>(R.id.radioStress)
        val saveButton = findViewById<Button>(R.id.saveButton)

        saveButton.setOnClickListener() {
            //save to database
            val goToBed = goToBedBar.progress
            val wakeUp = wakeUpBar.progress

            val selectedOption: Int = radioGroupStress!!.checkedRadioButtonId

            finish()
        }
    }
}
```

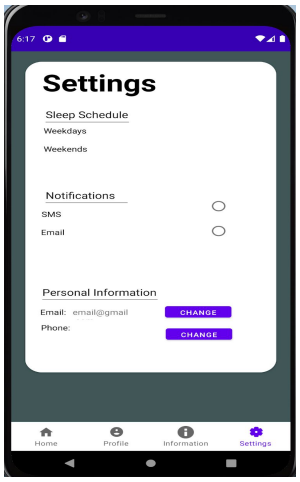


The questions and its alternatives are at this moment hard coded into the application's source code. This is not optimal as the people performing the study cannot easily change the questionnaires. It would therefore be advantageous to add the functionality of fetching the questionnaires from a database which can be updated by the administrators of the study. The same applies to the user inputs which at the moment are stored locally in the application, these answers should be mirrored in a database.

1.3 Settings

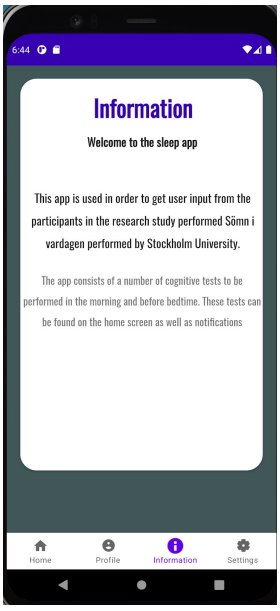
The settings page lets the user adjust the wake up time and bedtime, i.e. when the app should notify the user about doing the questionnaire. Here, the user can also enter its contact information. The settings backend is written in the file SettingsFragment.kt.

```
class SettingsFragment : Fragment() {

    private lateinit var settingsViewModel: SettingsViewModel

    override fun onCreateView(
        inflater: LayoutInflater,
        container: ViewGroup?,
        savedInstanceState: Bundle?
    ): View? {
        settingsViewModel =
            ViewModelProvider(this).get(SettingsViewModel::class.java)
        val root = inflater.inflate(R.layout.fragment_settings, container, false)
        /*val textView: TextView = root.findViewById(R.id.textSettings)
        settingsViewModel.text.observe(viewLifecycleOwner, Observer {
            textView.text = it
        })*/
        return root
    }
}
```



1.4 Profile and information

The last two parts are profile and information where the user can see its user id and contact information. The information page gives info about the app and the study. Future work could focus on displaying the user's statistics on the profile page.



2. Feedback

When the Android prototype was done and functioning properly it was tested on real users and their feedback was documented. Below is a summary of the feedback:

User 1: “It’s user friendly, but it looks like an app from 2015. Good UX but the UI could look a bit better”.

User 2: “UI needs work, maybe an icon next to the cognitive tests to signify its type”

3. GitHub repository

The source code of the Android prototype can be found here:

https://github.com/NellyFriman/SleepApp-AndriodPrototype