Università della Svizzera italiana

Faculty of Informatics

# Lecture Mobile Computing, fall term 2020
# Android Programming Project (APP) – Final project report

| | |
|---|---|
| Group name: | PSA |
| Group members: | Alecci Lidia, Pasi Samuele, Stan Bianca M. |

**Instructions**

- Please answer the questions reported in the following two pages using the foreseen text boxes.
- Do not change the format, font size or any other elements of this template.
- Be concise and precise.
- Do not exceed the given limit of maximal number of characters. The given limits are intended including white spaces (e.g., the sentence "*This is a nice sentence*" contains 23 characters).
- "The app" mentioned in the questions refers to the Android-based application that you develop in the context of the Mobile and Wearable Computing class.
- Once compiled, please upload the document on iCorsi.
- The deadline for sending the document is Thursday, December 10[1], 2020, 19:00 CEST
- If you have questions: post your question(s) in the iCorsi forum or e-mail elena.di.lascio@usi.ch and shkurta.gashi@usi.ch.

[1] Since the final project report template was available in iCorsi only on December 7, we will accept submissions oft he compiled final project report also until December 18.

1. **What is the name of your app?**

Unsteppable

2. **Which problem does the app solve?** (Max. 200 characters)

Increase users' daily step count

3. **How does your app solve this problem?** (Max. 650 characters)

Our app is an adaptive fitness tracker that regulates the daily step goal based on weather and suggests that the user change their goal based on past performance. Our goal is to encourage users to move more by slowly pushing their limits and having them incrementally raising the daily target.

4. **Why is this problem relevant?** (Max. 300 characters)

Fitness is an important part of a person's health, and we believe that having an adaptive goal that changes based on daily circumstances could help encourage users to walk more

5. **Do other apps exist to solve this (or a very similar) problem?**

[✓ Yes] No

6. **If you answered *yes* to question 5, list the existing apps that are most related to yours and explain how these solutions differ from your own. If you answered *no* to question 5, explain why do you think nobody else has solved this problem before.** (Max. 650 characters)

There are a lot of step tracker apps (Samsung Health, Google Fit, ZombieRun ecc). We have found that having a fixed goal regardless of our past performance or how bad the weather is, can get disappointing. While doing a quick search on the PlayStore to see what has been done, we couldn't find apps that took weather or previous performance into consideration (Google Fit adjusts the goal, but without explicitly saying how).

7. **Which of the built-in sensors of your phone does the app make use of?** (Max. 200 char.)

GPS, step detector, screen

8. **Which of the built-in actuators of your phone does the app make use of?** (Max. 200 char.)

Screen, vibration, speakers

9. **Does your app store sensor data locally, remotely, or in both ways?**

[✓ Locally] Remotely Both

10. **Motivate your answer to question 9 (i.e., explain why your app stores data and why it does so only locally/remotely or in both ways).** (Max. 650 characters)

Our app stores data locally, for ease and speed of access. It does access remote data in order to get the current weather for the location. Further developments could move the data to the cloud in order to allow social media features.

11. **Which type of data visualization your app offers and why. If you answered *no* to question 11, explain why data visualization is not necessary for your app.** (Max. 650 characters)

We show the daily amount of steps, as well as weekly and monthly statistics. This is fundamental for users to gauge their performance.

12. **Does your app perform any type of data processing on the collected sensor data?**

[✓ Yes] ◯ No ◯

13. **If you answered *yes* to question 13 explain which type of data processing your app performs on the collected data and why. If you answered *no* to question 13 explain why data processing is not necessary for your app.** (Max. 650 characters)

We use past performance in order to make suggestions to the users about possible adjustments to their daily base goal.

14. **How do you evaluate whether your app performs correctly and achieves its goal (i.e., solves the problem described in question 2)?** (Max. 650 characters)

A way to measure our app's performance would be by doing a comparison between average steps taken before and after starting to use our app, or by issuing surveys on users' mental state (i.e., if the adaptive goal helped motivate them)

15. **Which permissions does your app require to be granted by the user?** (Max. 200 characters)

Position, local storage, physical activity

16. **Does the app raise any ethical issues?**

Yes ◯ [✓ No] ◯

17. **Motivate your answer to question 16** (Max. 300 char.)

Since we store data locally, there are no ethical qualms about our app - privacy isn't in any way put at risk.

18. **Indicate how each member of the group contributed to the App. Provide the name of the member, his/her main task (UI, database, data processing) and the parts of the code he/she wrote.** (Max. 650 characters)

Alecci Lidia - dealt with setting up the database and handling data processing
Stan Bianca M. - dealt with the UI&refactoring
Pasi Samuele - handled the weather API calls and data visualization

19. **What are the main challenges you had to cope with in order to build your app and how did you manage to overcome them? Please also describe "failed attempts", i.e., ideas and solutions you might have explored but did not bring to the expected result.** (Max. 1 page)

For the UI part, we had some trouble deciding how to show all the statistics. We decided on putting a Tabbed Activity inside a NavigationDrawer Activity to make statistics very accessible, since the user only needs to swipe in order to navigate. Things which the user might more rarely access - Settings, Badges - were instead put in harder-to-reach places (top right, top left). We thought of only doing either one or the other, however that would have meant either having many tabs or making it harder for the user to access statistics one-handed. The difficulty with integrating these two different types of navigation came with setting up the PageAdapter in order to allow for swiping between tabs, which was our main goal.

Another issue was making two different themes available - we had to look through many tutorials in order to find information about how to access a theme's colors ("?attr/[…]") rather than actual hardcoded colors ("@color/[…]"). Furthermore, this proved to be quite tricky when it came to acessing them programmatically in order to make the graphs consistent with the theme (having to go through resolveAttribute()).

Lastly, deciding where to put the current weather wasn't immediate - at first, we thought of putting it in the navigation drawer and having it change during the lifetime of the app, but as pointed out by the TAs, this compromised the app's consistency (and also, meteo is such a fundamental part of our app that displaying it front-and-center was important). We chose instead to place it in the Today tab, which is the first thing the user sees when they open the app.

During the implementation of the backend, the most challenging thing was the service. Android doesn't allow always-on services unless they display notifications. In the beginning, we tried to override this constraint, but it became apparent after some attempts and talking with the TAs during a project review that there was no alternate solution.

For the weather, the most difficult part was getting the current location based on GPS data - in particular, fitting the tutorial to our app and creating a utility class in order to avoid putting everything in the MainActivity.

This latter issue was also present throughout our whole development process: keeping code tidy, enforcing separation of concerns, trying to avoid God classes and in general developing according to good design practices. We often had to have quite tight coupling between classes, and MainActivity more than once had a tendency of becoming a God class. We had a tendency to work according to "make it work, then make it pretty", so we had to do multiple refactorings in order to displace functionality into dedicated classes.