

# Mobile Interaction Summer 2024

Prof. Dr. Michael Rohs, Shashank Ahire, M.Sc.

## Assignment 9

All exercises that are not explicitly declared as group tasks must be done individually and handed in individually. Identical submissions are treated as plagiarism. Plagiarism may lead to loss of exam bonus points.

You can submit the solution to this task in English or German until Wednesday, June 12, at 23:59 via <https://assignments.hci.uni-hannover.de>. Create a pdf file that contains the text and images of your solution, name it "Assignment-09-<Firstname>-<Lastname>.pdf", and save it together with the exported project (Android Studio: File → Export → Export to Zip File) in a single zip file. Your submission must consist of a single zip file containing all necessary files. The name of the .zip file, as well as the names of the contained files, **must not contain any umlauts**. Therefore, please resolve umlauts in file names.

**Android Phones:** Please let us know if you need an Android phone for the assignments. We can lend a limited number of simple Android phones for the duration of the semester. These must only be used for the assignments.

### Exercise 1: Experience Sampling App (24 points)

In the previous assignment you created a paper prototype of an experience sampling app for students. The goal of the current exercise is to implement your paper prototype as an Android app. If you have not completed the previous exercise, implement the app directly (without the paper prototype) using questionnaire items as specified in the previous assignment.

The app should prompt the user to answer questions at reasonable intervals. Depending on your design in the last exercise, the app may measure different aspects, like the mood of a student in a particular lecture. The data should then be sent anonymously to a server, evaluated and visualized in the app.

At [https://couchdb.hci.uni-hannover.de/\\_utils/index.html](https://couchdb.hci.uni-hannover.de/_utils/index.html) we provide a CouchDB [1, 2] database server. You may log into the web interface with user "surfer" and password "CouchingOnTheCouch". A database named "mobint" already exists. The database contains objects in JavaScript Object Notation (JSON) [3]. An object can be queried in a web browser as well as via an HTTP-GET request within an app, e.g.

<https://couchdb.hci.uni-hannover.de/mobint/esm1687348771561>

The list of all documents may be queried as follows:

[https://couchdb.hci.uni-hannover.de/mobint/\\_all\\_docs](https://couchdb.hci.uni-hannover.de/mobint/_all_docs)

The list of all documents with key = ["Mobile Interaktion","MyExperimenterId"]:

[https://couchdb.hci.uni-hannover.de/mobint/\\_design/esm/\\_view/ctxtexp?key=%5B%22Mobile+Interaktion%22%2C%22MyExperimenterId%22%5D](https://couchdb.hci.uni-hannover.de/mobint/_design/esm/_view/ctxtexp?key=%5B%22Mobile+Interaktion%22%2C%22MyExperimenterId%22%5D)

Encoding: [ → %5B, " → %22, , → %2C, ] → %5D

Use the database "mobint" to solve the task. You can create and delete individual documents using a Web browser. Be careful to only delete the documents that you created yourself.

On Stud.IP you will find a template of the Experience Sampling App. The code for most of the UI, for communicating with the database, and for periodically creating notifications is already provided. Modify the template app according to your paper prototype as necessary.

Explain/justify when you need to deviate from your paper prototype. You may use additional attributes for your database documents. If it would be too much effort to implement some of the features of your paper prototype, you may deviate from your paper prototype. Please follow the Android Material Design Guidelines as much as possible.

Please solve the following subtasks:

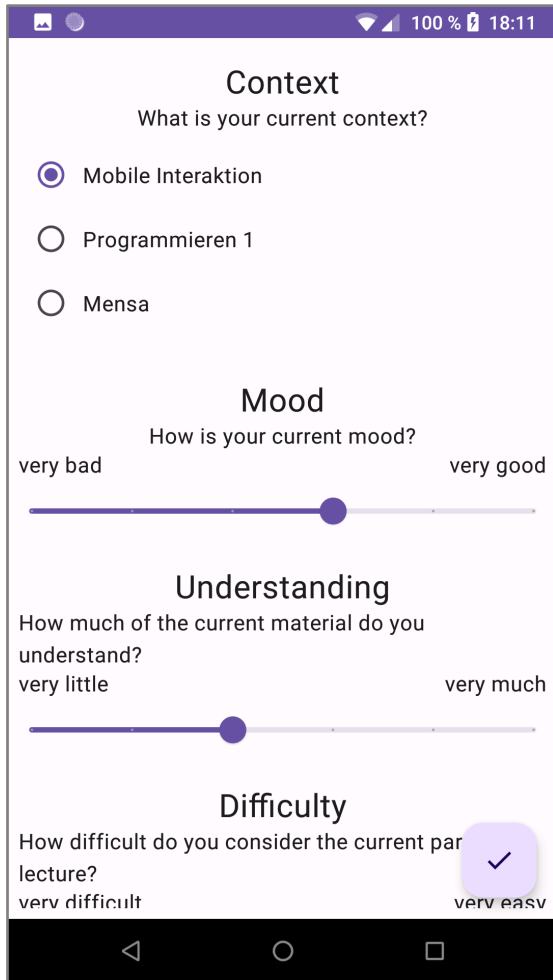
- a) Use your own unique experimenter ID (global constant in ExperienceSampling.kt) to identify your documents in the database. State your experimenter ID. **(1 point)**
- b) Explain the architecture of the app in terms of layers (UI layer: view, view-model; data layer: repository, data source). **(8 points)**
- c) **CouchDB.kt:** Explain why JSON documents are sent to and retrieved from the server asynchronously. What approach has been taken to implement asynchronous send/receive? Why is withContext necessary and what does it do? How are IO exceptions handled? **(4 points)**
- d) **Sampling GUI** (package sampling): Modify the sampling GUI to match your paper prototype. Use four suitable items from the UEQ-Short and two more specific items/questions (see previous assignment). The entered data needs to be sent to the server when the "Submit" button is pressed, to be stored on the server. **(5 points)**
- e) **Statistics GUI** (package statistics): Modify the statistics GUI to show averages for selectable contexts. See right screenshot below. **(6 points)**

Please provide screenshots of the **Sampling GUI** and the **Statistics GUI** in addition to the **source code** in your submission.

[1] <http://couchdb.apache.org>

[2] <http://docs.couchdb.org/en/latest/index.html>

[3] <http://www.json.org>



**Context**  
What is your current context?

☒ Mobile Interaktion  
☐ Programmieren 1  
☐ Mensa

**Mood**  
How is your current mood?

very bad 

very bad

very good

 very good

**Understanding**  
How much of the current material do you understand?

very little 

very little

very much

 very much

**Difficulty**  
How difficult do you consider the current part of the lecture?

very difficult 

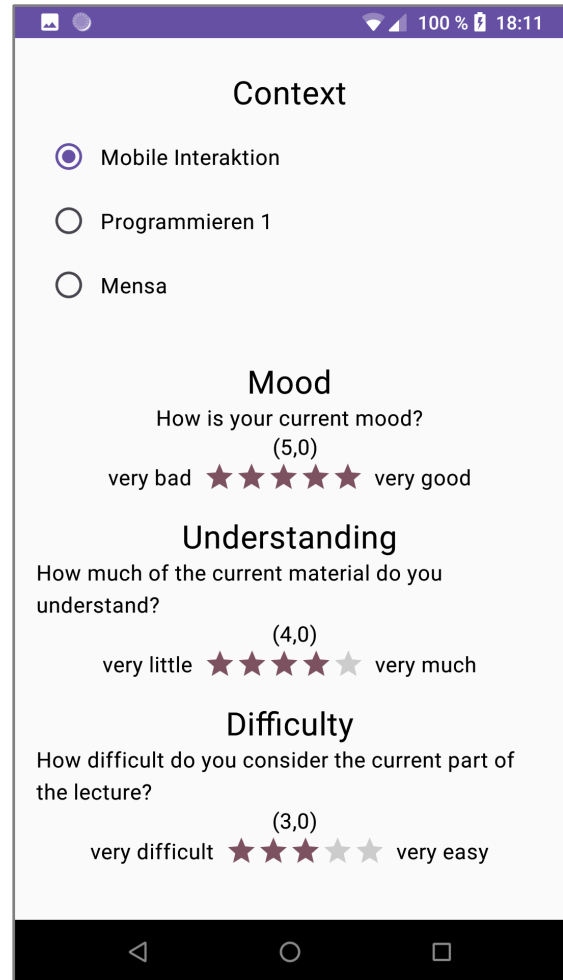
very difficult

very easy

 very easy

☒

Example sampling (input) screen.



**Context**

☒ Mobile Interaktion  
☐ Programmieren 1  
☐ Mensa

**Mood**  
How is your current mood?

(5,0)  
very bad ★★★★★ very good

**Understanding**  
How much of the current material do you understand?

(4,0)  
very little ★★★★★ very much

**Difficulty**  
How difficult do you consider the current part of the lecture?

(3,0)  
very difficult ★★★☆☆ very easy

Example statistics (output) screen.

## Exercise 2: CouchDB Design Documents (11 points)

Given a CouchDB View<sup>1</sup> with the following design document:

```
{
  "_id": "_design/mapreduce1",
  "language": "javascript",
  "views": {
    "ctxt": {
      "map": "function(doc) { if (doc._id.search(\"test\") == 0 &&
                                doc.c.length >= 1) emit(doc.b, doc.a) }"
    }
  }
}
```

... and given the following example documents:

```
{"_id": "test1", "a": 10, "b": "x", "c": "hello"}
```

<sup>1</sup> <http://docs.couchdb.org/en/2.0.0/couchapp/views/intro.html>

```
{ "_id": "test2", "a": 20, "b": "x", "c": "world" }  
{ "_id": "test3", "a": 30, "b": "x" }  
{ "_id": "test4", "a": 40, "b": "x", "c": [1, 2, 3] }  
{ "_id": "test5", "a": 50, "b": "x", "c": "" }
```

- a) What is the general purpose of the **map** and **reduce** functions for databases (independent of the above example)? (4 points)
- b) Explain the map function given above. What does it select and what key-value pairs does it emit (i.e., include in the result list)? Your explanation should be general and not be limited to the example documents given above. (4 points)
- c) Assume that the above view is invoked using:  
[https://couchdb.myserver.de/mydb/\\_design/mapreduce1/\\_view/ctxt?key=%2x%22](https://couchdb.myserver.de/mydb/_design/mapreduce1/_view/ctxt?key=%2x%22). Which of the five example documents given above are selected and what is the output? Important: Give the result as a formatted JSON-Array. (3 points)