Biomedical Wearable Technologies
for Healthcare and Wellbeing

Networking

A.Y. 2021-2022

Giacomo Cappon

Outline

- **Recap**
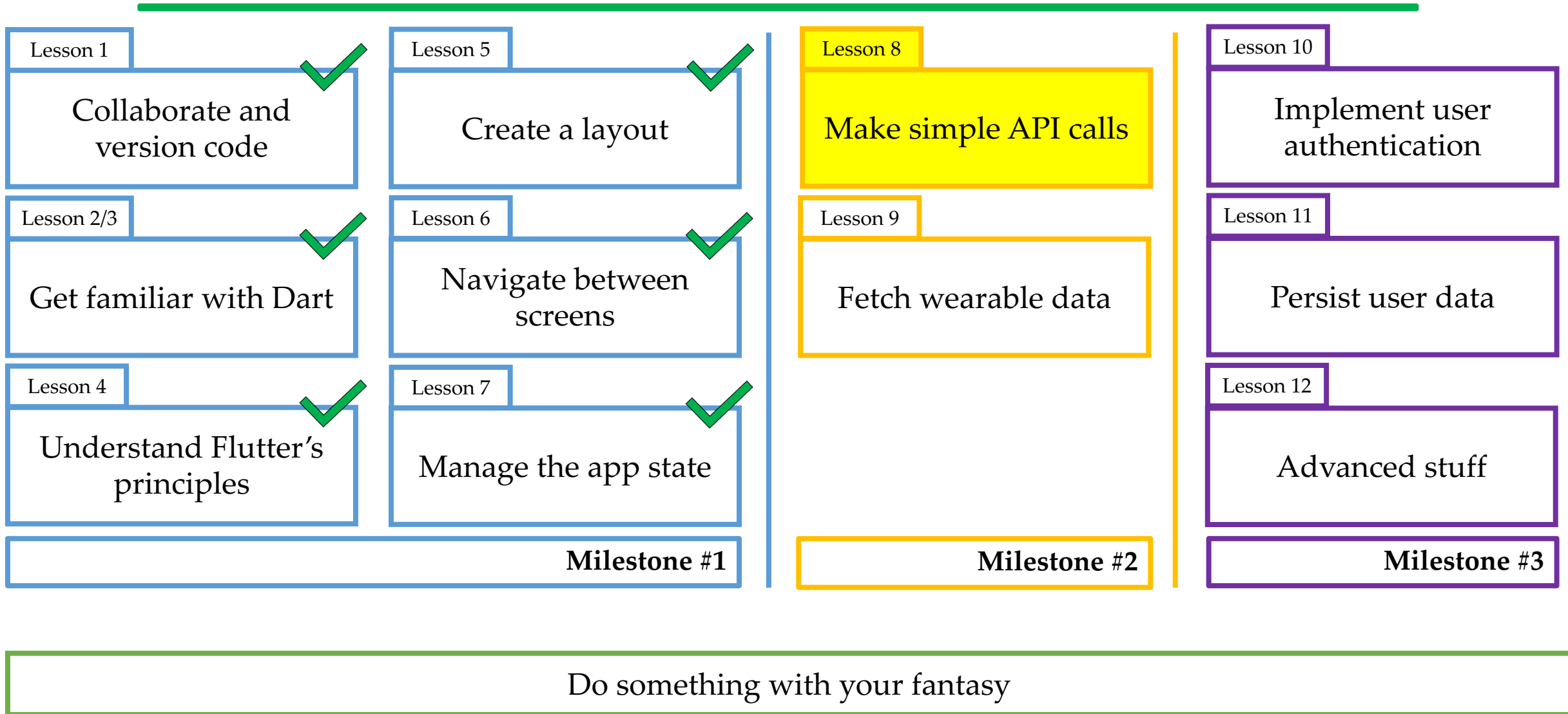- The network flow
- RESTful API in practce
- Case study

- Exercise
- Homework
- Resources

Recap

Lesson 1 Collaborate and version code ✓	**Lesson 5** Create a layout ✓	**Lesson 8** Make simple API calls	**Lesson 10** Implement user authentication
Lesson 2/3 Get familiar with Dart ✓	**Lesson 6** Navigate between screens ✓	**Lesson 9** Fetch wearable data	**Lesson 11** Persist user data
Lesson 4 Understand Flutter's principles ✓	**Lesson 7** Manage the app state ✓		**Lesson 12** Advanced stuff
Milestone #1		**Milestone #2**	**Milestone #3**

Do something with your fantasy

Outline
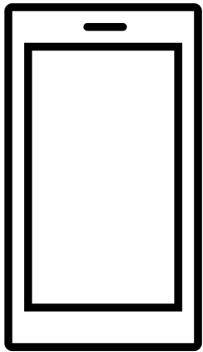
- Recap
- **The network flow**
- RESTful API in practce
- Case study

- Exercise
- Homework
- Resources

The network flow
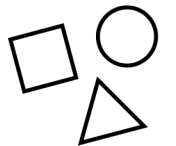
Outline

- Recap
- The network flow
- **RESTful API in practce**
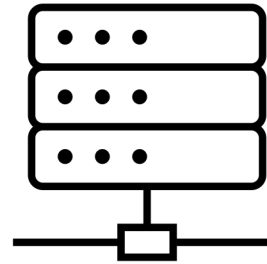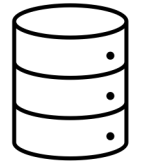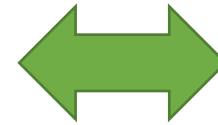- Case study

- Exercise
- Homework
- Resources

RESTful API in practice

- From the practical point of view, RESTful API can be used via http following three steps:
  - Step 1: Send an http request to the RESTful API
  - Step 2: Await for the response
  - Step 3: Process the response (usually in a JSON format)

- The http request has the following structure:

<METHOD> <HTTP or HTTPS>://<DOMAIN>/<ENDPOINT>?<PARAMETERS>

+

<BODY> and <HEADERS>

- <METHOD> and <ENDPOINT> defines the so-called **route, e.g.:**
  GET /heartrate/today/
  DELETE /user/1

RESTful API in practice

➢ Beside its content a response contains an **HTTP status code**, i.e., a special number that tells to the frond-end if the request is successful or, otherwise, why it is not successful. Here's the most common:

- 200: OK
- 401: UNAUTHORIZED
- 403: FORBIDDEN
- 404: NOT FOUND
- 500: INTERNAL SERVER ERROR

➢ Normally, the front-end developer has to manage these codes based on the API specifics
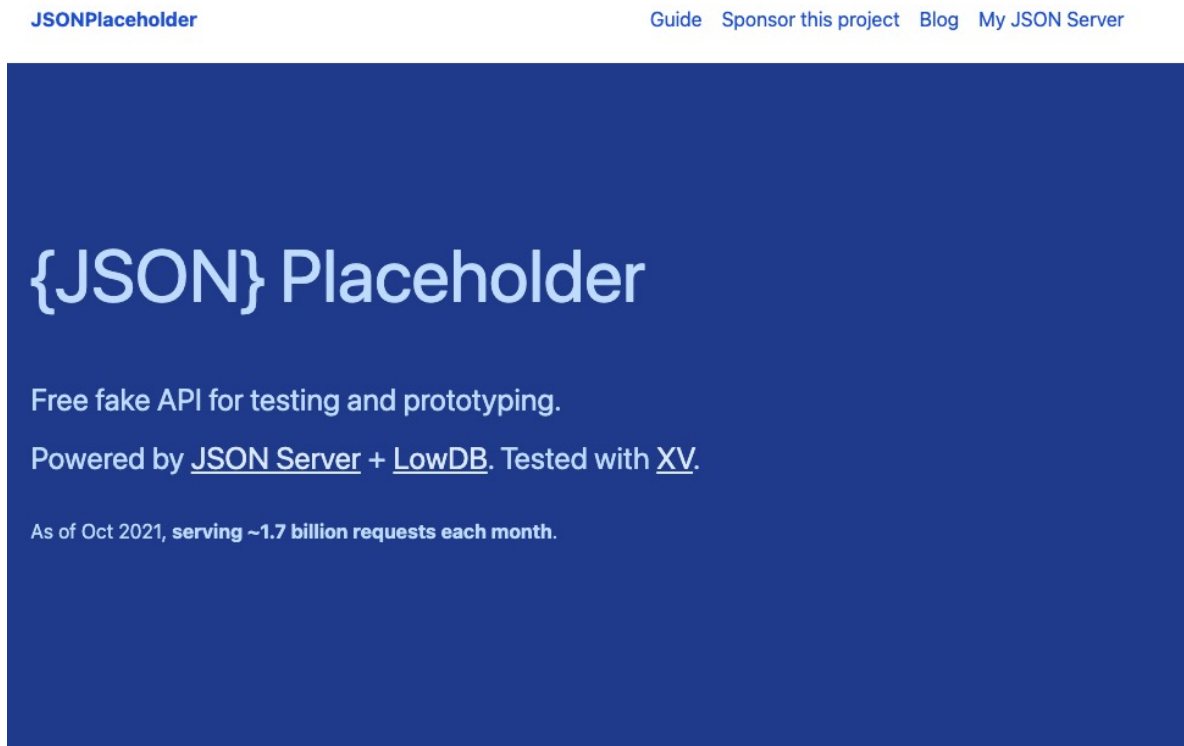
Outline

- Recap
- The network flow
- RESTful API in practce
- **Case study**

- Exercise
- Homework
- Resources

{JSON} Placeholder

➢ To learn how to do request to an API and process its result we will use {JSON}Placeholder

{JSON} Placeholder

➢ {JSON}Placeholder exposes the following resources and routes we can play with

Resources

JSONPlaceholder comes with a set of 6 common resources:

/posts	100 posts
/comments	500 comments
/albums	100 albums
/photos	5000 photos
/todos	200 todos
/users	10 users

Note: resources have relations. For example: posts have many comments, albums have many photos, ... see guide for the full list.

Routes

All HTTP methods are supported. You can use http or https for your requests.

GET	/posts
GET	/posts/1
GET	/posts/1/comments
GET	/comments?postId=1
POST	/posts
PUT	/posts/1
PATCH	/posts/1
DELETE	/posts/1

Note: see guide for usage examples.

{JSON} Placeholder – Post resource

➤ Let's zoom in the post resource:

- ▪ https://jsonplaceholder.typicode.com/posts (this is equivalent to the GET https://jsonplaceholder.typicode.com/posts request) will result to:

```
[
  {
    "userId": 1,
    "id": 1,
    "title": "sunt aut facere repellat provident occaecati excepturi optio reprehenderit",
    "body": "quia et suscipit\nsuscipit recusandae consequuntur expedita et cum\nreprehenderit molestiae ut ut quas totam\nnostrum rerum est autem sunt rem eveniet architecto"
  },
  {
    "userId": 1,
    "id": 2,
    "title": "qui est esse",
    "body": "est rerum tempore vitae\nsequi sint nihil reprehenderit dolor beatae ea dolores neque\nfugiat blanditiis voluptate porro vel nihil molestiae ut reiciendis\nqui aperiam non debitis possimus qui neque nisi nulla"
  },
  {
    "userId": 1,
    "id": 3,
    "title": "ea molestias quasi exercitationem repellat qui ipsa sit aut",
    "body": "et iusto sed quo iure\nvoluptatem occaecati omnis eligendi aut ad\nvoluptatem doloribus vel accusantium quis pariatur\nmolestiae porro eius odio et labore et velit aut"
  },
  {
    "userId": 1,
    "id": 4,
    "title": "eum et est occaecati",
    "body": "ullam et saepe reiciendis voluptatem adipisci\nsit amet autem assumenda provident rerum culpa\nquis hic commodi nesciunt rem tenetur doloremque ipsam iure\nquis sunt voluptatem rerum illo velit"
  },
  {
    "userId": 1,
    "id": 5,
```

➤ So, we have a list of posts, each identified via an id, the id of the user who posted it, a title, and a body.

http package

➢ To be able to make calls we will use the http package. This provides a simple web client to be used to make http calls.

Android-specific action

➢ To be able to access to internet functionalities in Android you are required to provide a specific permission:

➢ To do so, in the android>app>src>main folder open the AndroidManifest.xml file, and add the following after the <manifest …> tag:
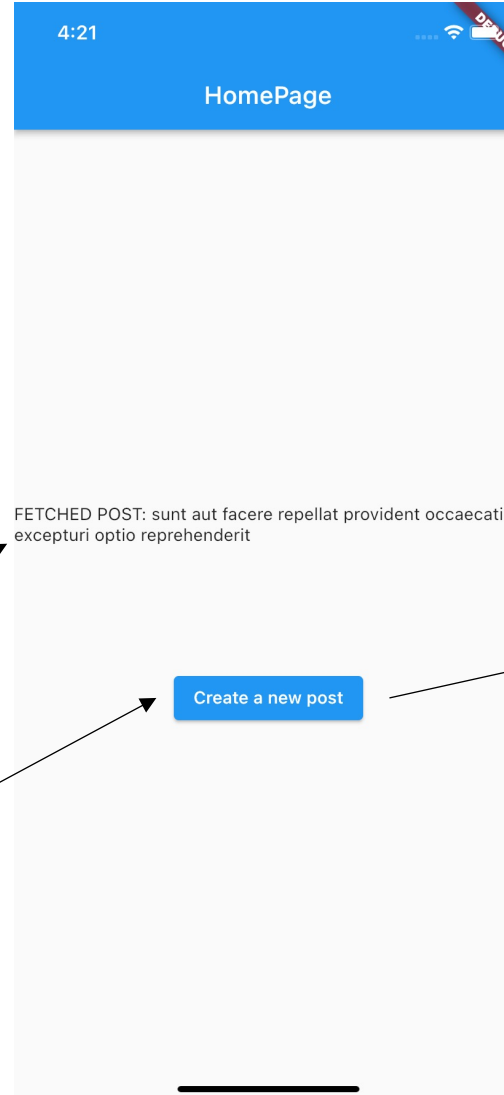
<uses-permission android:name="android.permission.INTERNET" />

➢ No need to do this in iOS.

Case of study - Live

➤ To demonstrate how to make and process API calls in Flutter, we will build a very simple app called "network_master" able to

1. Fetch data of a specific post from {JSON}Placeholder
→ **GET method**

2. Create a new post in {JSON}Placeholder
→ **POST method**

4:21   HomePage

FETCHED POST: sunt aut facere repellat provident occaecati excepturi optio reprehenderit

Create a new post

Routes

All HTTP methods are supported. You can use http or https for your requests.

GET	/posts
GET	/posts/1
GET	/posts/1/comments
GET	/comments?postId=1
POST	/posts
PUT	/posts/1
PATCH	/posts/1
DELETE	/posts/1

Note: see guide for usage examples.

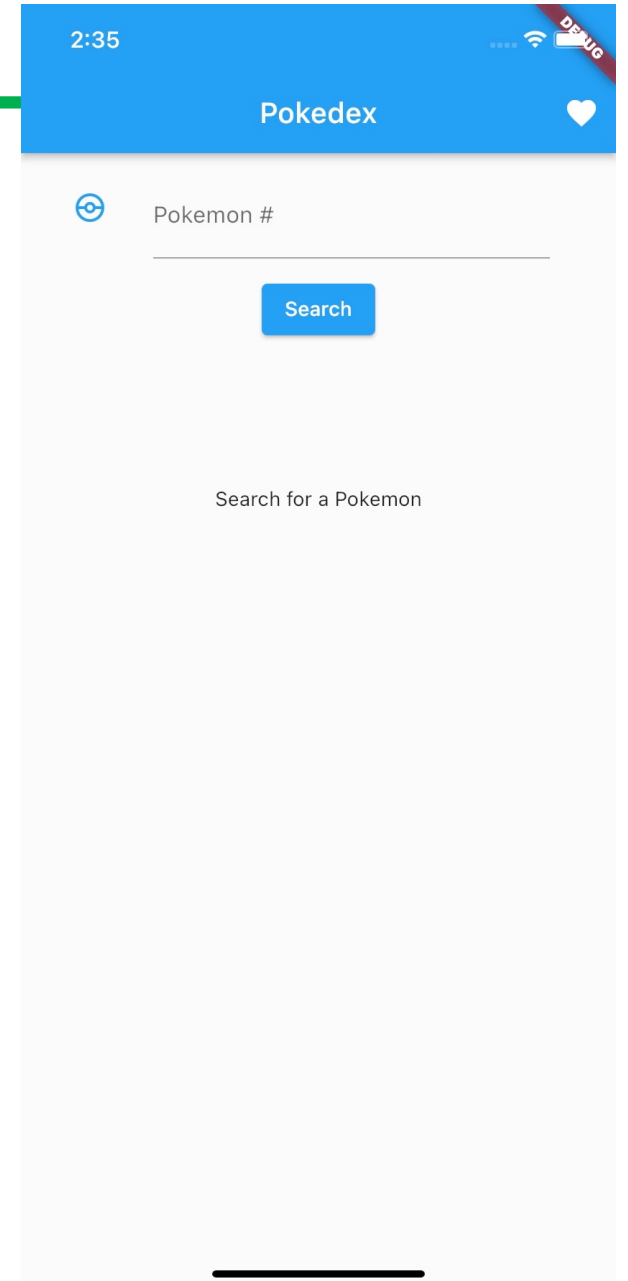Full example in lab_08-networking/network_master/

Outline

➢ Recap

➢ The network flow

➢ RESTful API in practce

➢ Case study

➢ **Exercise**

➢ Homework

➢ Resources

Exercise

➢ Exercise 08.01

- Implement a Pokédex

- The user can search into the search bar for a Pokémon given its number (be aware that valid numbers go from 1 to 898). Once the button is tapped, if the number is valid, some of the Pokémon details are shown to the user.

- The user can add that Pokémon to its favorites using a button.

- Via a button in the AppBar, the user navigates to another screen where all its favorite Pokémon are shown.

- In the "Favorites Pokémon page" the user can look at the list and delete entries.

Outline

- Recap
- The network flow
- RESTful API in practce
- Case study

- Exercise
- **Homework**
- Resources

Homework

- Get familiar with the http package

- Take a look to a powerful alternative to the http package: Dio
  - https://pub.dev/packages/dio

Outline

➢ Recap
➢ The network flow
➢ RESTful API in practce
➢ Case study

➢ Exercise
➢ Homework
➢ **Resources**

Resources

- HTTP Status Codes
  - https://en.wikipedia.org/wiki/List_of_HTTP_status_codes

- {JSON}placeholder APIs
  - https://jsonplaceholder.typicode.com/

- PokéAPI
  - https://pokeapi.co/

- Fetch data from the internet. Cookbook by the Flutter community
  - https://docs.flutter.dev/cookbook/networking/fetch-data

- Send data to the internet. Cookbook by the Flutter community
  - https://docs.flutter.dev/cookbook/networking/send-data