Laboratory Session #01

Distributed Systems Programming

Daniele Bringhenti

Topics of the Laboratory Session

Laboratory Session #01 covers two main topics:

Design of **JSON Schemas**

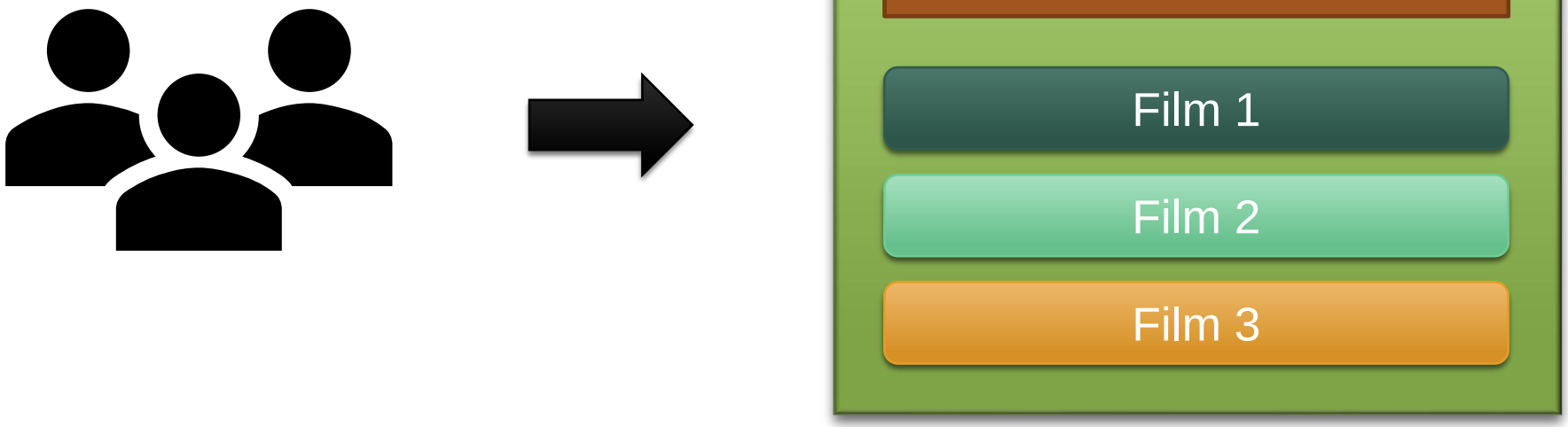
Design of **REST APIs**

The complete experience of Laboratory Session #01 includes also:

Implementation of designed REST APIs

ToDo Manager service

- The context of Laboratory Session #01 is the ***Film Manager* service**:
  - users can keep track of the films they have watched and/or they want to review;
  - basic concepts of this service derive from *Web Applications I* course, A.Y. 2021/2022 (https://github.com/polito-WA1-AW1-2022/)



Design of JSON Schemas (I)

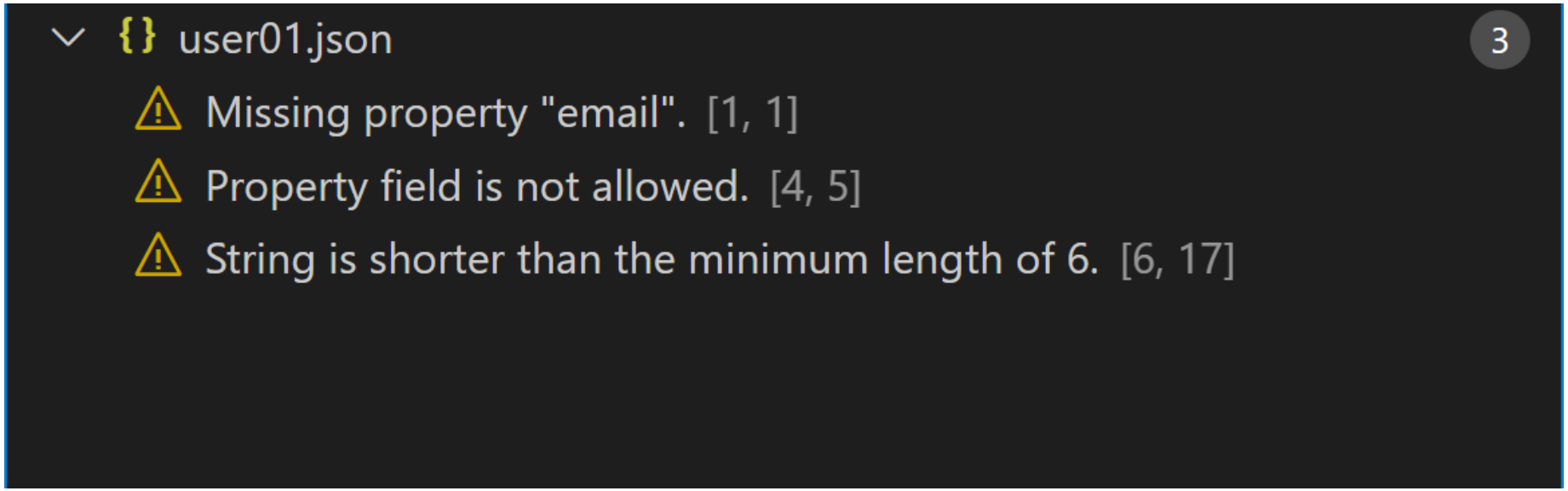
- The first activity is about the design of JSON schemas for three core data structures of the *Film Manager:*
  1) the ***users*** who want to manage their film lists;
  2) the ***films*** *that* the users have watched and/or that must be reviewed;
  3) the film ***reviews*** that users may issue another users.

- The JSON Schema standard that must be used for this activity is the **Draft 07** (http://json-schema.org/draft-07/schema#).

- Recommendation: after designing the schemas, write some JSON files as examples and **validate** them against the schemas!

Design of JSON Schemas (II)

- Tool suggested for the design of JSON schemas and the validation of JSON files:
  - **Visual Studio Code** (*Problems* view).



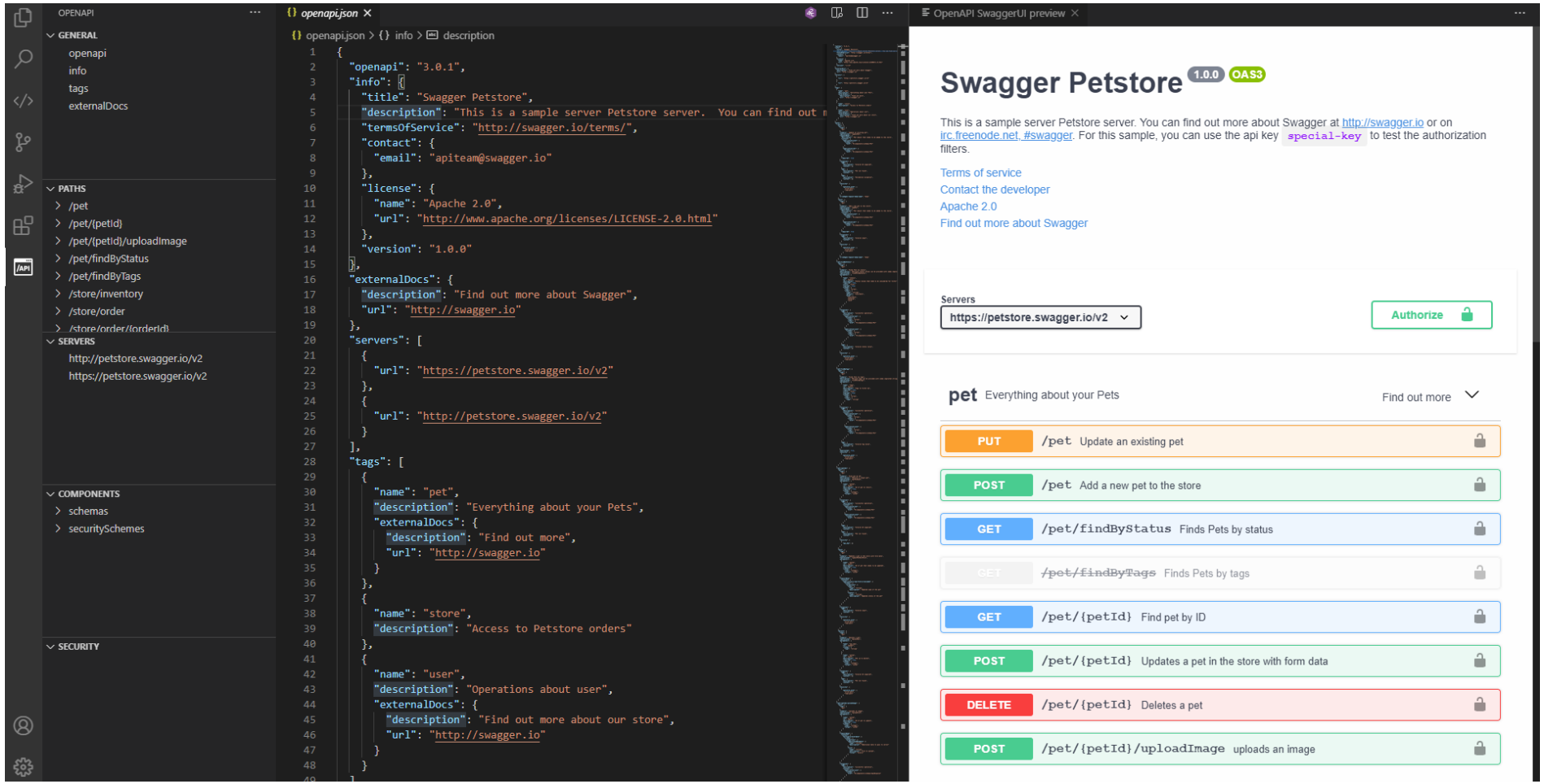
Design of REST APIs (I)

- The second activity is about the design of **REST APIs** for the *Film Manager* service:
  - the design must be documented in an **OpenAPI** file (https://swagger.io/docs/specification/about/).

- In this activity, you can use:
  - the **schemas** developed in the first part of the assignment, customizing them for being used in the REST APIs;
  - the "**OpenAPI (Swagger) Editor**" extension of Visual Studio Code (https://marketplace.visualstudio.com/items?itemName=42Crunch.vscode-openapi).

Design of REST APIs (II)

- **OpenAPI (Swagger) Editor**



Implementation of REST APIs (I)

- The resulting OpenAPI document can be used as the starting point to develop an implementation of the designed REST APIs in a **semi-automatic way**.

- After importing the OpenAPI file to the **stand-alone Swagger Editor** (the online version or the locally installed version), you can automatically generate a **server stub**, corresponding to the design of the REST APIs.

- The server stub must be the filled with the **functionalities** described in the document of Laboratory Session #01.

Implementation of REST APIs (II)

How to generate the server stub, and **make it run**?

Issue in the server stub

How to make the **server stub** run?

- There is a bug, not still fixed, in the most recent version of **oas3-tools** module.
- A possible solution is to define a dependency to a previous version:
  - "**2.0.2**" version is fine;
  - be aware not to write "^2.0.2", otherwise the dependency is solved with the most recent version.

```
"dependencies": {
  "connect": "^3.2.0",
  "js-yaml": "^3.3.0",
  "oas3-tools": "2.0.2"
}
```

Thanks for your attention!

Daniele Bringhenti
daniele.bringhenti@polito.it

netgroup