Laboratory Activity #01

Distributed Systems Programming

Topics of the Laboratory Activity

Laboratory Activity #01 covers two main topics:

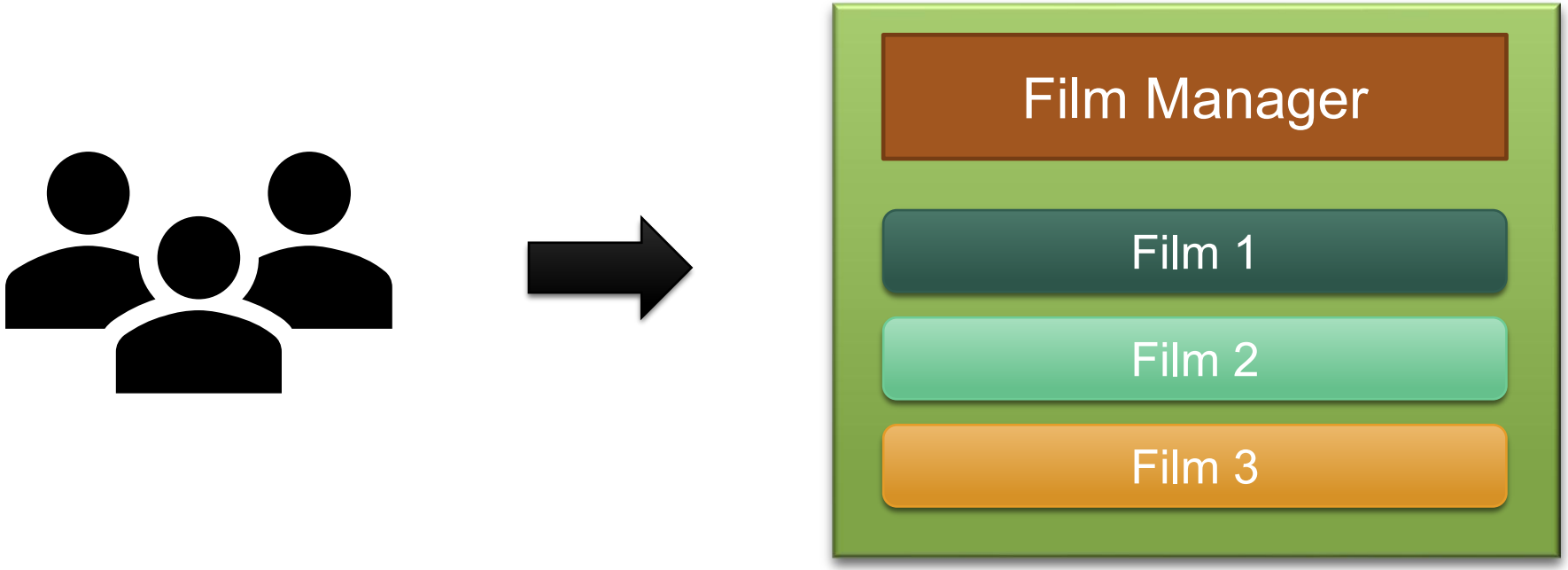
- Design of **JSON Schemas**
- Design of **REST APIs**

The complete experience of Laboratory Activity #01 includes also:

- Implementation of designed REST APIs

Film Manager platform

- The context of Laboratory Activity #01 is the ***Film Manager*** *platform*:
  - Users can keep track of the films they have watched;
  - Films can be private or public and have various attributes;
  - Users can issue a review to other users (or themselves) for their films.



Design of JSON Schemas (I)

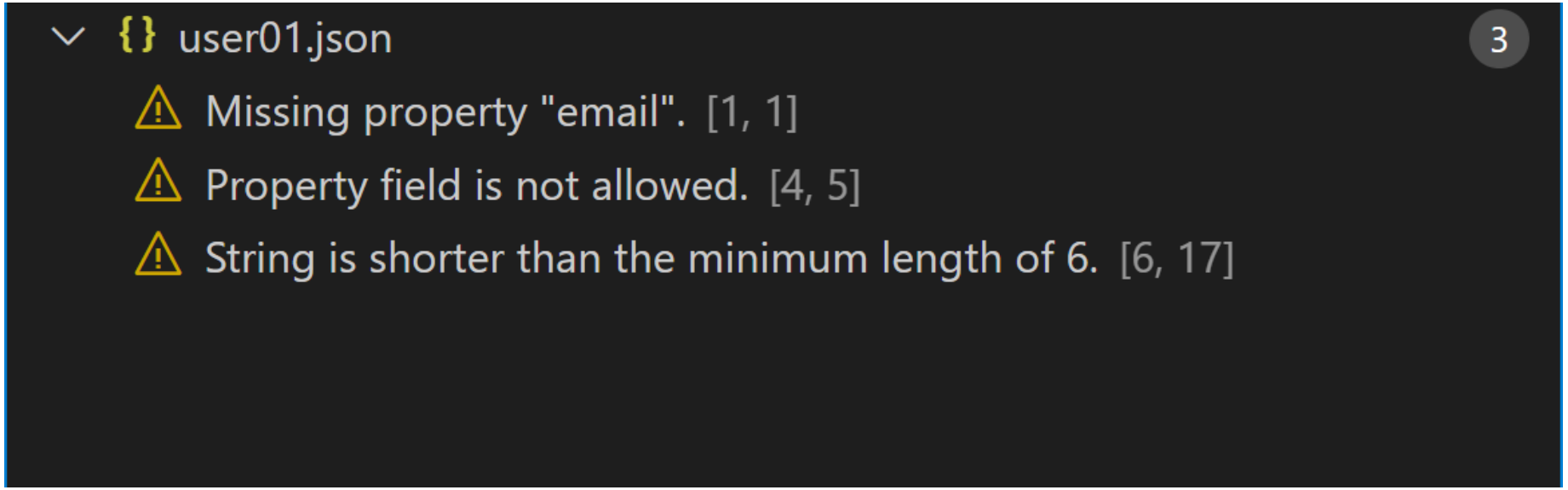
- The first activity is about the design of JSON schemas for three core data structures of the *Film Manager:*
  1) the ***users*** who want to manage their film lists;
  2) *the* ***films*** that the users have watched and/or that must be reviewed*;*
  3) the film ***reviews*** that users may issue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



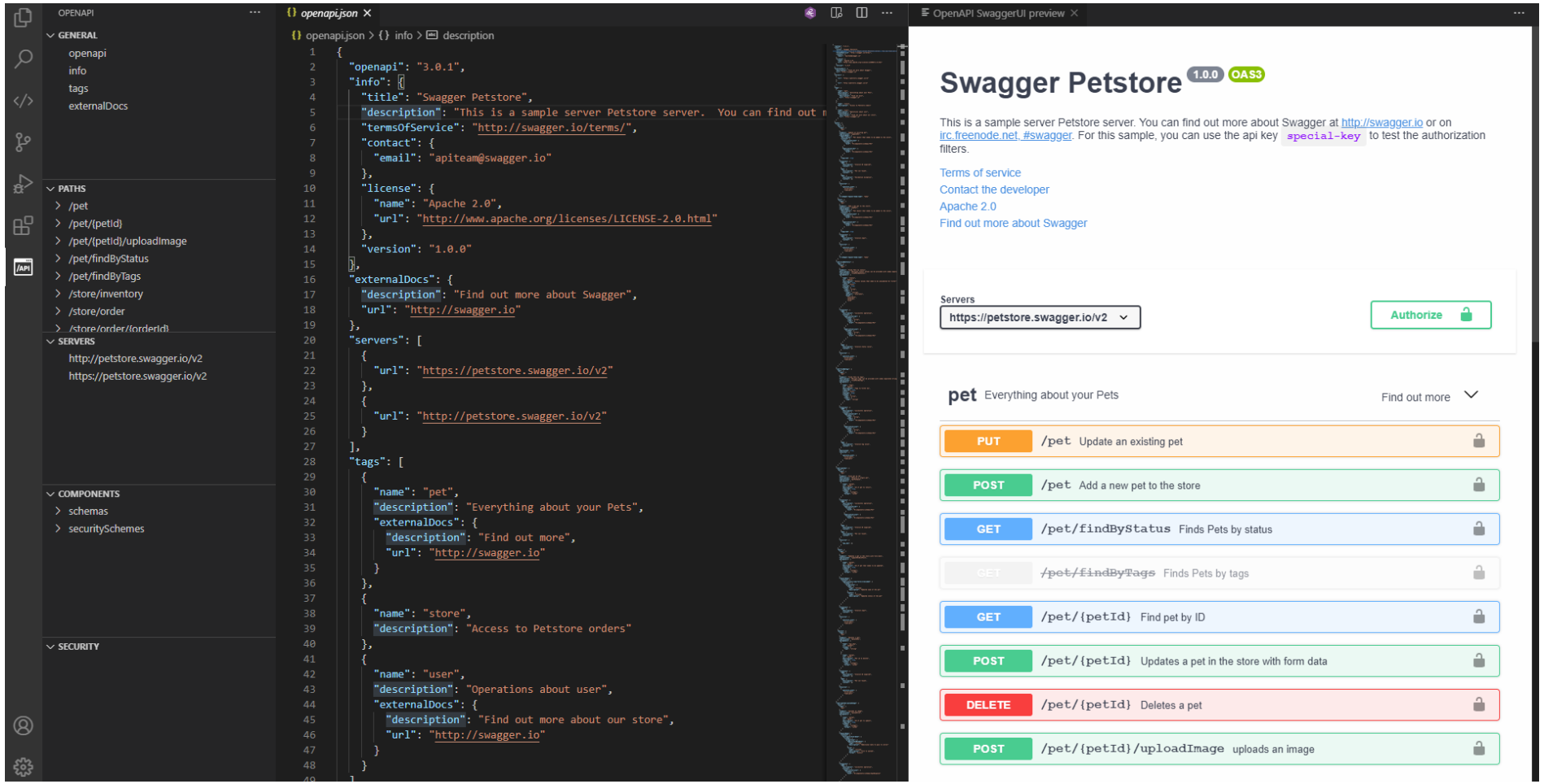
Design of REST APIs (I)

- The second activity is about the design of **REST APIs** for the *Film Manager* platform:
  - 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**



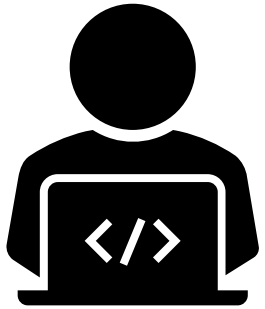
Implementation of REST APIs (I)

- 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** 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**?

Completing the server stub

- We have provided you with a repository (https://github.com/polito-DSP-2025-2026/lab01) that contains a skeleton to be integrated into the server stub you generated.
- This skeleton already includes authentication and service modules that handle database interactions and implement the core functionalities of the platform.
- You should integrate this skeleton into your generated server stub and complete the missing parts required to make the server fully functional — such as controllers and route definitions.
- A detailed description of the missing components can be found in the **Lab 1 specification document** and in the **README** included in the repository.

Thanks for your attention!

If you have further questions use the Slack Workspace or send an email

Rosario Rizza

rosario.rizza@polito.it

Slides made by Daniele Bringhenti