



# Cloud Computing (**LAB**)



Giuseppe Bisicchia  
`giuseppe.bisicchia@phd.unipi.it`

Dipartimento di Informatica, Università di Pisa

# Cosa faremo?

- Svilupperemo un (micro)servizio REST e la relativa specifica OpenAPI
- Utilizzeremo un set di strumenti per automatizzare sia la creazione delle specifiche partendo da esempi concreti di risorse che la generazione di un microservizio di base
- Analizzeremo il codice generato e lo testeremo

# Swagger Petstore

- Petstore è un esempio classico di specifica OpenAPI. Potete trovarlo al seguente link:
- <https://petstore.swagger.io/v2/swagger.json>
- Suggerimento: visualizzatelo su <https://editor.swagger.io/>
- Domande:
  - Quanti tipi di risorse sono presenti?
  - In che modo possiamo interagire con ognuno di essi?
  - Provate a modificare la specifica, cosa accade?

# Caso d'uso: Libro di Ricette

- Proviamo ora ad ipotizzare un microservizio che permette la gestione condivisa di un libro di ricette.
- Domande:
  - Di quante e quali risorse abbiamo bisogno?
  - Che informazioni associamo ad ogni tipo di risorsa?
  - Come possiamo interagire con ogni tipo di risorsa?
- Esercizio: sulla base della specifica per Petstore provate a scrivere tramite editor la specifica del nostro libro di ricette.

# OAS Tools

- OAS Tools è un ecosistema di strumenti server-side per la creazione di servizi REST compatibili con le specifiche OpenAPI.
- Sono inoltre presenti tool per la generazione automatica delle specifiche a partire da esempi concreti di risorse.

# Generazione automatica delle specifiche OpenAPI

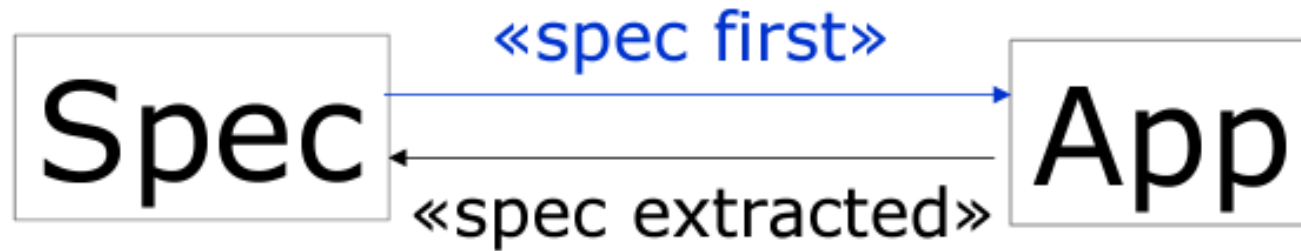
- Generate ora un file YAML contenente una lista di esempi di risorse concrete (uno per ogni tipo di risorsa).
- Una volta ultimato il file utilizzate il comando «oas-tools init» tramite terminale.
- Scegliete dunque «OpenAPI File» e seguite le istruzioni di configurazione.
- Avete appena creato automaticamente un file di OpenAPI! Confrontatelo con quanto fatto manualmente, che differenze notate?

```
- recipeId: pasta_carbonara
name: Pasta alla carbonara
description: Gli spaghetti alla carbonara sono un classico primo piatto della cucina italiana.
complexity: 3
ingredients:
  - spaghetti
  - tuorli
  - guanciale
  - pecorino romano
  - pepe nero
```

```
? Choose OpenAPI version 3.0
? Enter a title for the described API LibroDiRicette
? Enter a description for the described API Un libro di ricette
? Provide a name for the resource containing {recipeId,name,description,complexity,ingredients} Recipe
? Which will be the 'id' property for 'Recipe' recipeId
? Specify which operations will be available for 'Recipe' GET, POST, PUT
? Choose the preferred format for the OpenAPI Document YAML
? Where will be file generated? oas-doc.yaml
? Generate file with this options? Yes
```

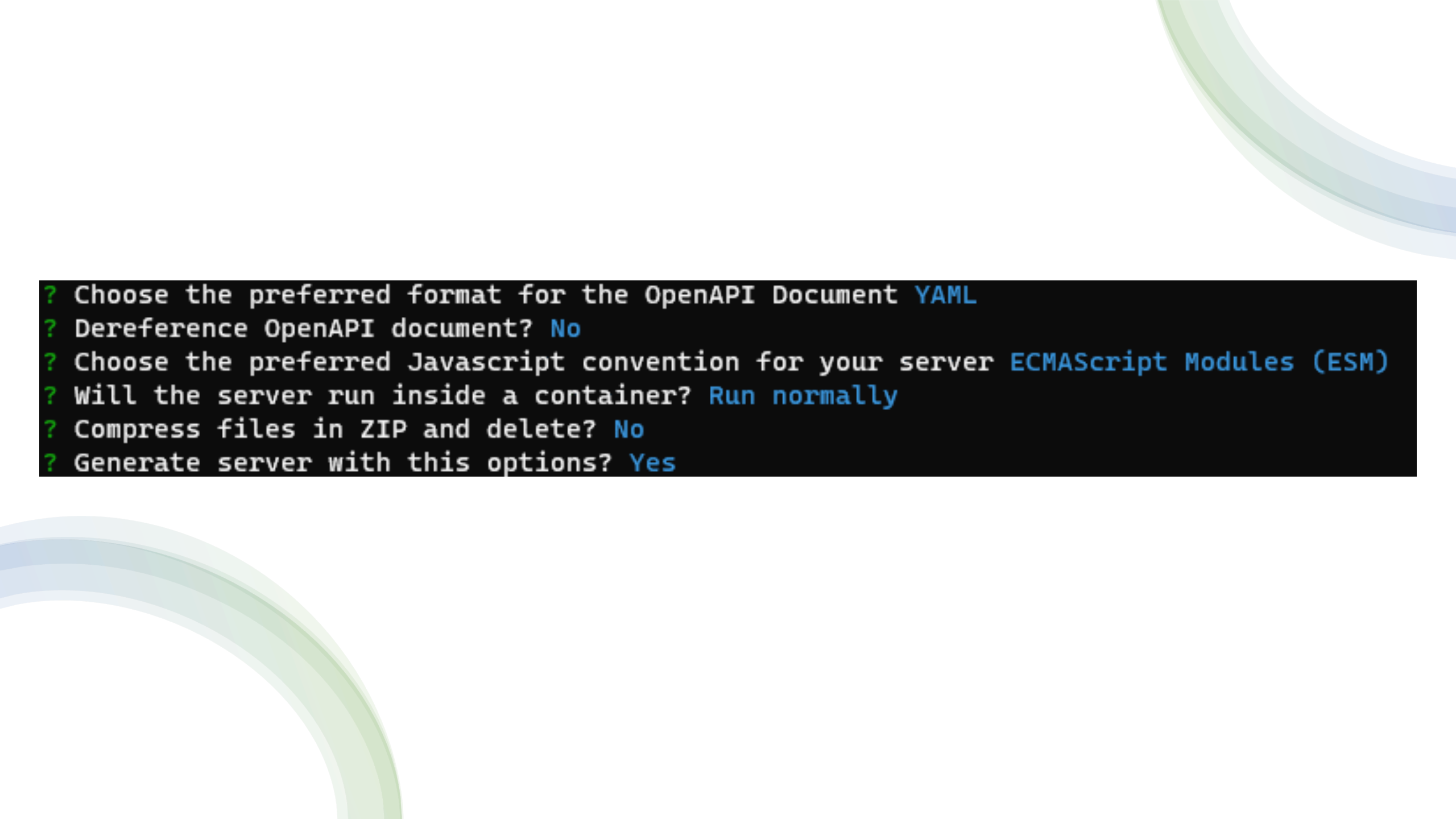


E.g., Connexion framework for Flask  
automagically handles HTTP requests based  
on OpenAPI Specification of your API

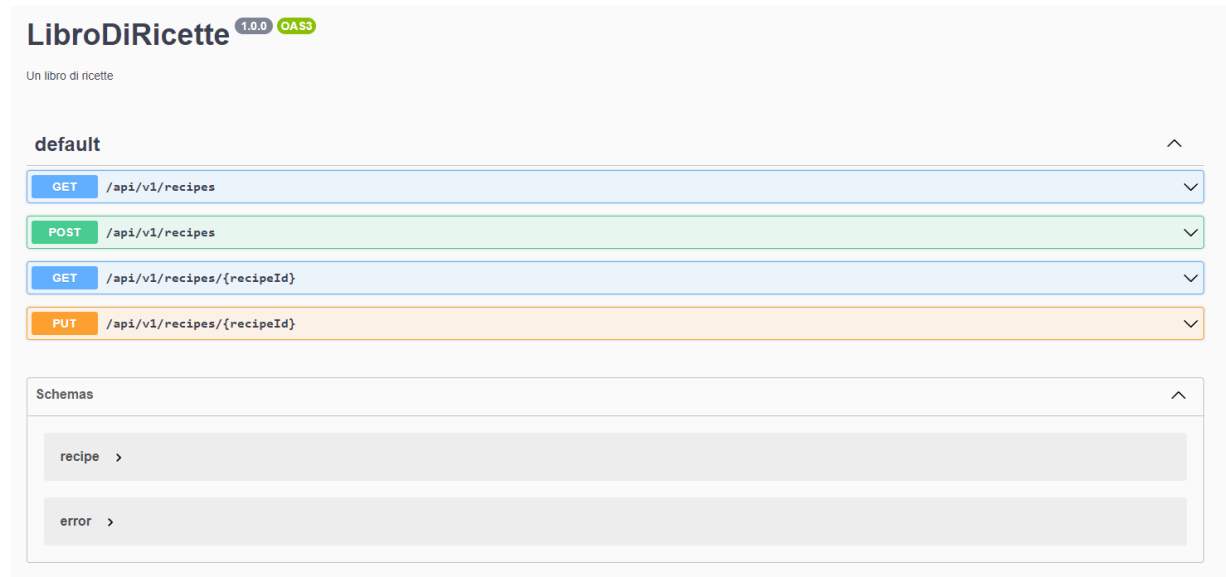


## Generazione automatica delle specifiche OpenAPI

- Un ottimo stile di sviluppo è lo «spec first» che consente di avere forti garanzie sulla compatibilità del codice nei confronti delle specifiche.
- Una volta sviluppate le specifiche OpenAPI è possibile generare lo scheletro di un microservizio in maniera automatica.
- Per farlo possiamo usare sempre «oas-tools init» specificando questa volta «server».



```
? Choose the preferred format for the OpenAPI Document YAML  
? Dereference OpenAPI document? No  
? Choose the preferred Javascript convention for your server ECMAScript Modules (ESM)  
? Will the server run inside a container? Run normally  
? Compress files in ZIP and delete? No  
? Generate server with this options? Yes
```



# Lanciare il server

Per lanciare il server ora basta accedere da terminale alla cartella e digitare «npm start»

# Esercizio

- Provate ad eseguire qualche azione, cosa accade?
- Modificate il codice in modo da farvi restituire una risposta plausibile ad ogni azione.

# Esercizio

- Oltre agli OAS Tools possiamo usare anche l'editor di swagger stesso per generare lo scheletro del nostro microservizio.
- Swagger ci consente anche di generare automaticamente anche un client verso il nostro servizio REST. Provate a generarne uno nel vostro linguaggio preferito, com'è strutturato?

# Esercizio

- Modificate il file con le risorse d'esempio aggiungendo un nuovo tipo di risorsa e rigenerate il file OpenAPI, il server e il client. Come sono cambiati?