

<b>SPE-CLO5:</b> My Booking Services
<b>Date:</b> 01/06/2020, <b>Date de fin:</b> 31/07/2020
<b>Groupe:</b> rouman_s, legal_q, bincza_m, massar_t et simoes_t
<b>Ci-joint:</b> Répartition des tâches, architecture des différents services

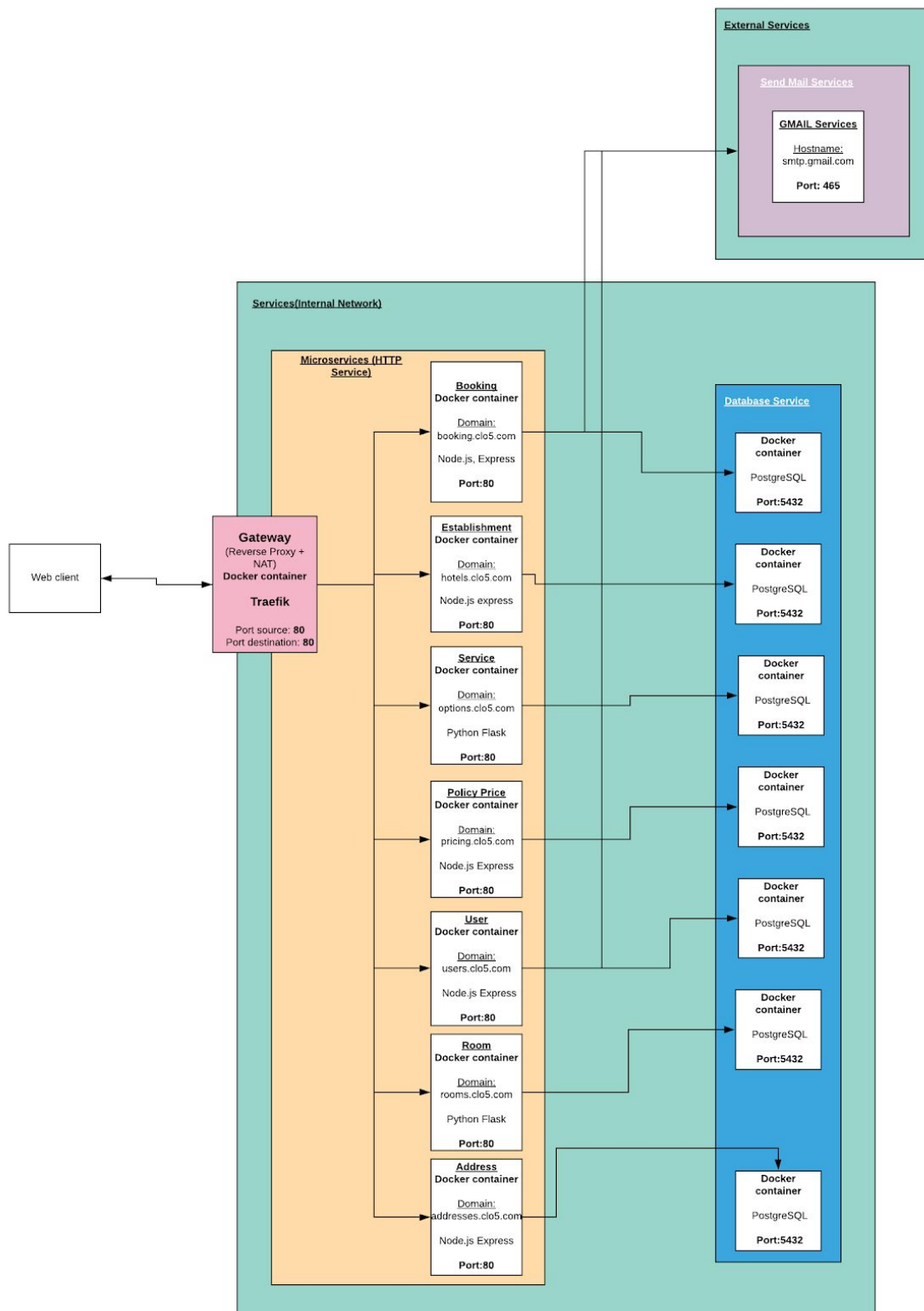
---

## Etape 2 - Schéma architecture applicatif

### Domain Driven Design:

Le Domain Driven Design est une méthodologie qui aide à concevoir des logiciels répondant à des problématiques complexes. Il a été conceptualisé par Eric J. Evans.

## Schéma architecture applicatif:



## Mocking data

Les services seront développés en python ou en node.js.

La mocking data est présent dans sous forme de fichier json, dans un dossier “mock”, appelés lors des tests unitaires et fonctionnels.

On peut générer de la mocking data en cli.

### Générer de la mocking data pour les services node.js:

Un seul objet: ts-node index.ts mock read

Une liste: ts-node index.ts mock list

### Générer de la mocking data pour les services python:

Un seul objet: python run.py mock read

Une liste: python run.py mock list

### Lancer les tests unitaires node.js:

npm test

### Lancer les tests unitaires python:

pytest

## Documentation

Un swagger a été mis en place sur chaque service.

Un swagger est accessible à l’adresse <http://localhost/documentation> pour les services en python.

Un swagger est accessible à l’adresse <http://localhost/swagger> pour les services node.js(II faudra modifier l’url en entré pour qu’elle est la valeur de: <http://localhost/public/swagger.json>).

### Générer un swagger pour les services en node.js:

npm run tsoa spec

## Room Service



The image shows the Swagger UI for the Room Service API. At the top, it displays 'API 3.0' and the base URL 'http://127.0.0.1:5000/swagger.json'. Below this, there are two expandable sections: 'tools' and 'room'. The 'tools' section contains a single endpoint: 'GET /tools/ping'. The 'room' section contains five endpoints: 'POST /room/', 'PUT /room/', 'GET /room/', 'DELETE /room/{id}', and 'GET /room/{id}'. Each endpoint is represented by a colored bar with the HTTP method and path.

**API 3.0**  
[ Base URL: / ]  
<http://127.0.0.1:5000/swagger.json>

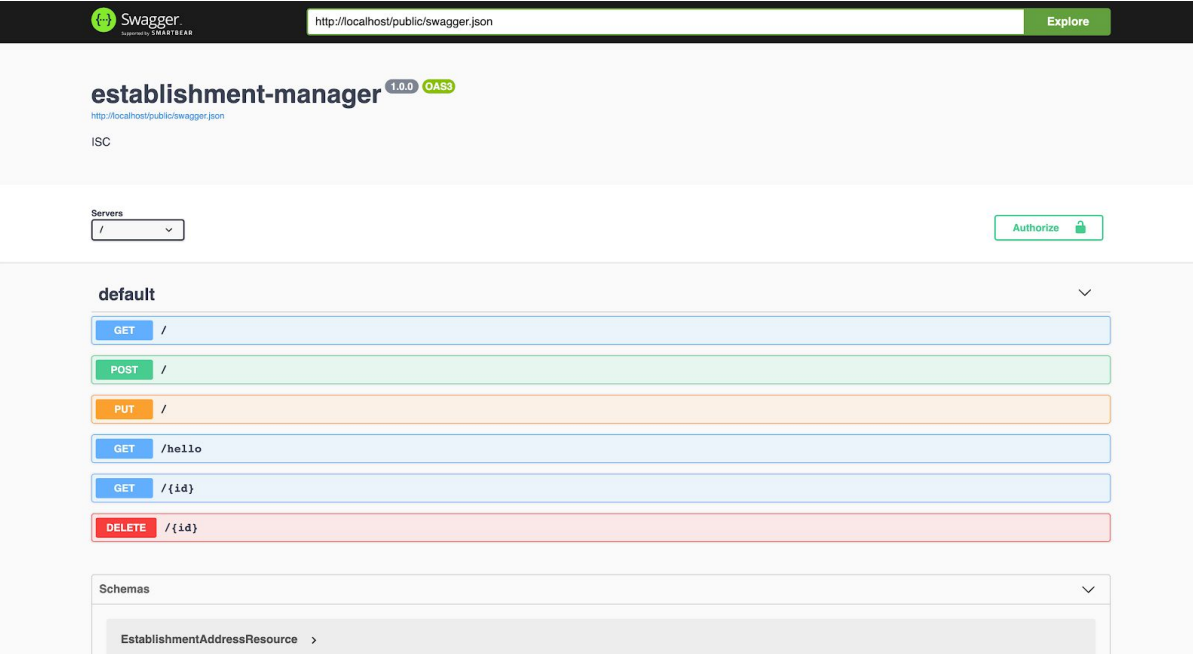
**tools** Related tools

- GET /tools/ping

**room** Related room endpoints

- POST /room/
- PUT /room/
- GET /room/
- DELETE /room/{id}
- GET /room/{id}

## Establishment Service



The image shows the Swagger UI for the Establishment Service. At the top, it displays the Swagger logo, the title 'establishment-manager 1.0.0 OAS3', and the base URL 'http://localhost/public/swagger.json'. Below this, there is a 'Servers' dropdown menu and an 'Authorize' button. The main section is titled 'default' and contains six endpoints: 'GET /', 'POST /', 'PUT /', 'GET /hello', 'GET /{id}', and 'DELETE /{id}'. At the bottom, there is a 'Schemas' section with a single schema entry: 'EstablishmentAddressResource'.

Swagger  
http://localhost/public/swagger.json Explore

**establishment-manager 1.0.0 OAS3**  
<http://localhost/public/swagger.json>

ISC

Servers: /

Authorize

**default**

- GET /
- POST /
- PUT /
- GET /hello
- GET /{id}
- DELETE /{id}

Schemas

- EstablishmentAddressResource >

# Option Service

Option API1.0

[ Base URL: / ]

http://localhost:5000/swagger.json

Service Manager API

option

Option related operations

POST

/option/

PUT

/option/

GET

/option/

DELETE

/option/{id}

GET

/option/{id}

# Policy Price Service

Swagger

http://localhost/public/swagger.json

Explore

policy-price-manager1.0.0OAS3

http://localhost/public/swagger.json

ISC

Servers

/

Authorize

default

GET

/hello

GET

/

POST

/

PUT

/

GET

/establishment/{establishmentId}

GET

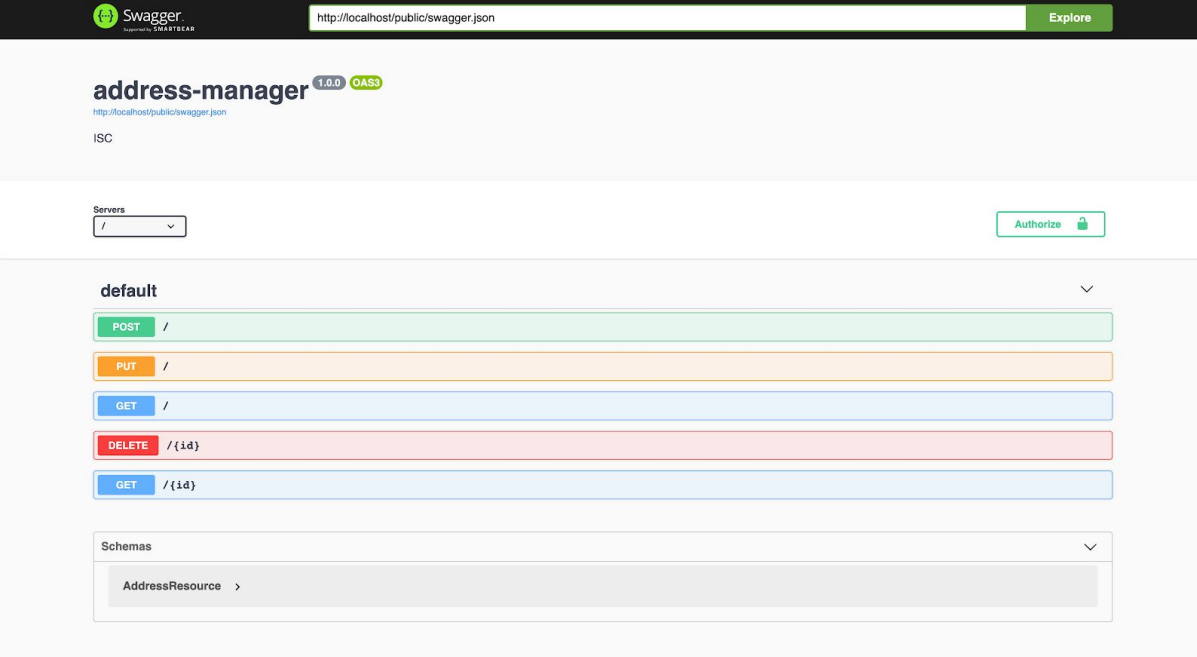
/id}

DELETE

/id}

Schemas

## Address Service



The image shows the Swagger UI for the 'address-manager' API. The top bar includes the Swagger logo, the URL 'http://localhost/public/swagger.json', and an 'Explore' button. The main header displays 'address-manager' with version '1.0.0' and 'OAS3' tags, along with the URL and the text 'ISC'. Below this, there is a 'Servers' dropdown menu showing '/' and an 'Authorize' button. The 'default' section is expanded, showing a list of endpoints: POST /, PUT /, GET /, DELETE /{id}, and GET /{id}. A 'Schemas' section is also expanded, showing 'AddressResource' with a right-pointing arrow.

Swagger  
Powered by SMARTCAR

http://localhost/public/swagger.json Explore

**address-manager** 1.0.0 OAS3  
http://localhost/public/swagger.json  
ISC

Servers  
/

Authorize

**default** ✓

POST /

PUT /

GET /

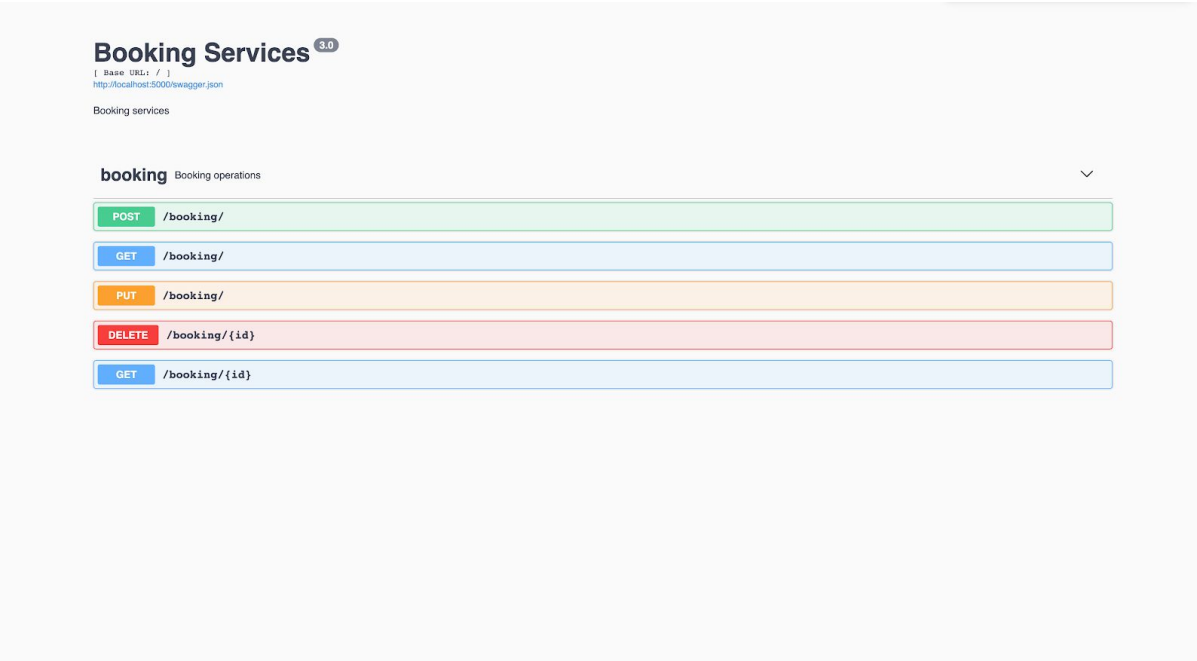
DELETE /{id}

GET /{id}

Schemas ✓

AddressResource >

## Booking Service



The image shows the Swagger UI for the 'Booking Services' API. The main header displays 'Booking Services' with version '3.0' and the URL 'http://localhost:5000/swagger.json'. Below this, the 'booking' section is expanded, showing a list of endpoints: POST /booking/, GET /booking/, PUT /booking/, DELETE /booking/{id}, and GET /booking/{id}.

**Booking Services** 3.0  
[ Base URL: / ]  
http://localhost:5000/swagger.json  
Booking services

**booking** Booking operations ✓

POST /booking/

GET /booking/

PUT /booking/

DELETE /booking/{id}

GET /booking/{id}