

# Tarik FERTAHI

## II-BDCC

Année universitaire : 2022/2023

## Pour le Micro-Service « CUSTOMER-SERVICE »

La création de deux Customers

```
@Bean
CommandLineRunner start(CustomerService customerService){
    return args -> {
        customerService.save(new CustomerRequestDTO( id: "C01", name: "Tarik", email: "Tarik@Tarik.com"));
        customerService.save(new CustomerRequestDTO( id: "C02", name: "OpenLab", email: "OpenLab@OpenLab.com"));
    };
}
```

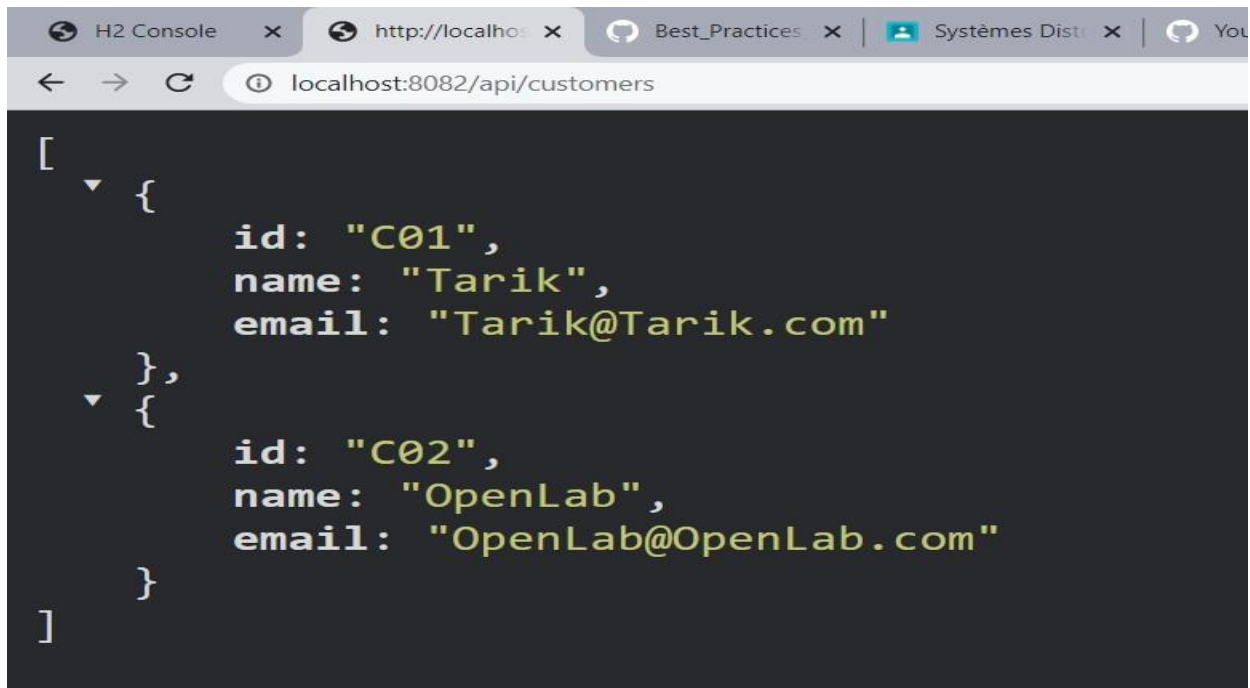
Vérification des clients au niveau de la base de données

The screenshot shows the H2 Console web interface in a browser. The address bar indicates the URL is `localhost:8082/h2-console/login.do?sessionId=95868d2450d277dadcb546a05fe5fbc9`. The interface includes a sidebar with a tree view of the database structure: `jdbc:h2:mem:customer-db`, `CUSTOMER`, `INFORMATION_SCHEMA`, and `Users`. The main area contains a toolbar with buttons like `Run`, `Run Selected`, `Auto complete`, and `Clear`, along with a text input for the `SQL statement:`. The statement `SELECT * FROM CUSTOMER` has been entered and executed. Below the toolbar, the query results are displayed in a table with columns `ID`, `EMAIL`, and `NAME`. The results show two rows: one for `C01` (Tarik) and one for `C02` (OpenLab). A status message at the bottom of the results area indicates `(2 rows, 2 ms)`. An `Edit` button is located at the bottom left of the results area.

ID	EMAIL	NAME
C01	Tarik@Tarik.com	Tarik
C02	OpenLab@OpenLab.com	OpenLab

(2 rows, 2 ms)

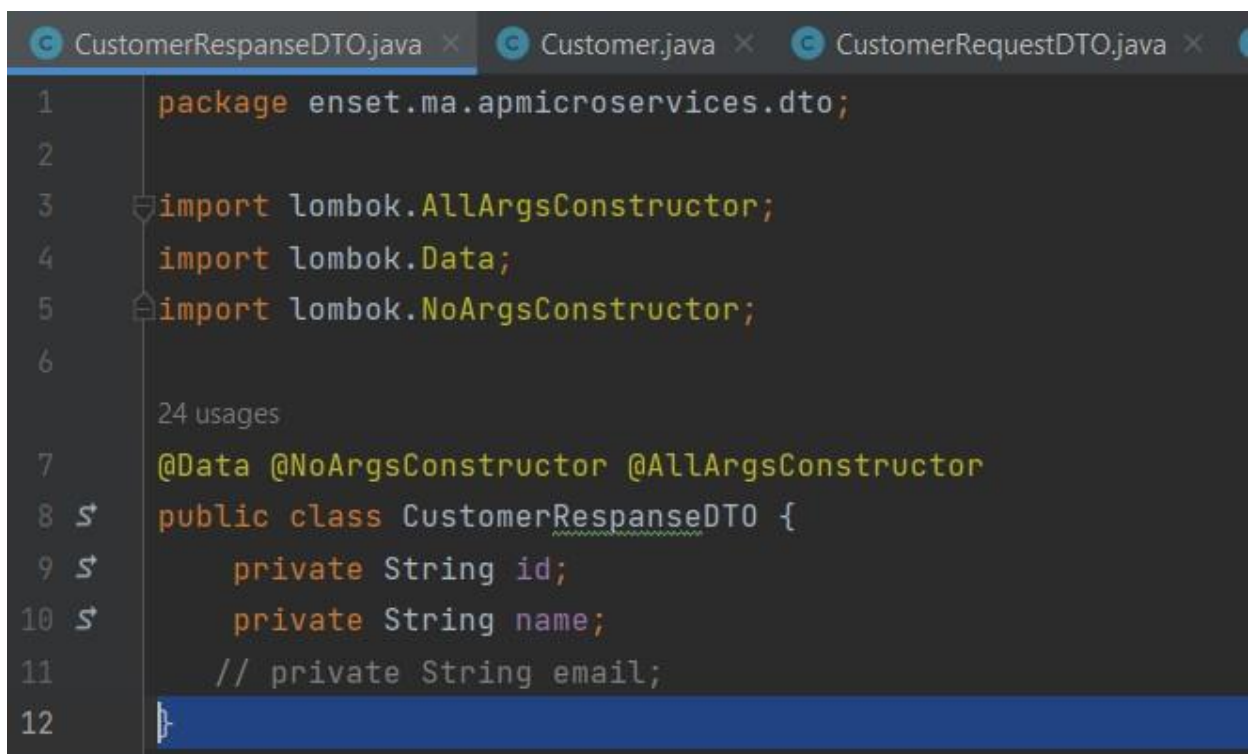
## Affichage des clients



The screenshot shows a web browser window with the address bar displaying `localhost:8082/api/customers`. The main content area shows a JSON array of two customer objects. The first object has `id: "C01"`, `name: "Tarik"`, and `email: "Tarik@Tarik.com"`. The second object has `id: "C02"`, `name: "OpenLab"`, and `email: "OpenLab@OpenLab.com"`.

```
[
  {
    id: "C01",
    name: "Tarik",
    email: "Tarik@Tarik.com"
  },
  {
    id: "C02",
    name: "OpenLab",
    email: "OpenLab@OpenLab.com"
  }
]
```

Grace au DTOs on peut ignorer l'attribut email dans la partie UI



The screenshot shows an IDE with three tabs: `CustomerResponseDTO.java`, `Customer.java`, and `CustomerRequestDTO.java`. The `CustomerResponseDTO.java` file is active, showing the following code:

```
1 package enset.ma.apmicroservices.dto;
2
3 import lombok.AllArgsConstructor;
4 import lombok.Data;
5 import lombok.NoArgsConstructor;
6
7 24 usages
8 @Data @NoArgsConstructor @AllArgsConstructor
9 public class CustomerResponseDTO {
10     private String id;
11     private String name;
12     // private String email;
```



```
[  
  {  
    id: "C01",  
    name: "Tarik"  
  },  
  {  
    id: "C02",  
    name: "OpenLab"  
  }  
]
```

Consulter un client par ID



```
{  
  id: "C01",  
  name: "Tarik"  
}
```

## Test de la méthode Save

Overview **POST http://localhost:8082/** GET http://localhost:8082/a PUT http://localhost:8081/b + ...

http://localhost:8082/api/customers

POST http://localhost:8082/api/customers

Params Authorization Headers (9) **Body** Pre-request Script Tests Settings

none form-data x-www-form-urlencoded **raw** binary GraphQL JSON

```
1 {
2   "name": "Tarik 2",
3   "email": "Tarik2@Tarik2.com"
4 }
```

Body Cookies Headers (5) Test Results Status: 200 OK

Pretty Raw Preview Visualize JSON

```
1 {
2   "id": "8516fc68-4289-4a62-8b15-ae464ab43200",
3   "name": "Tarik 2",
4   "email": "Tarik2@Tarik2.com"
5 }
```

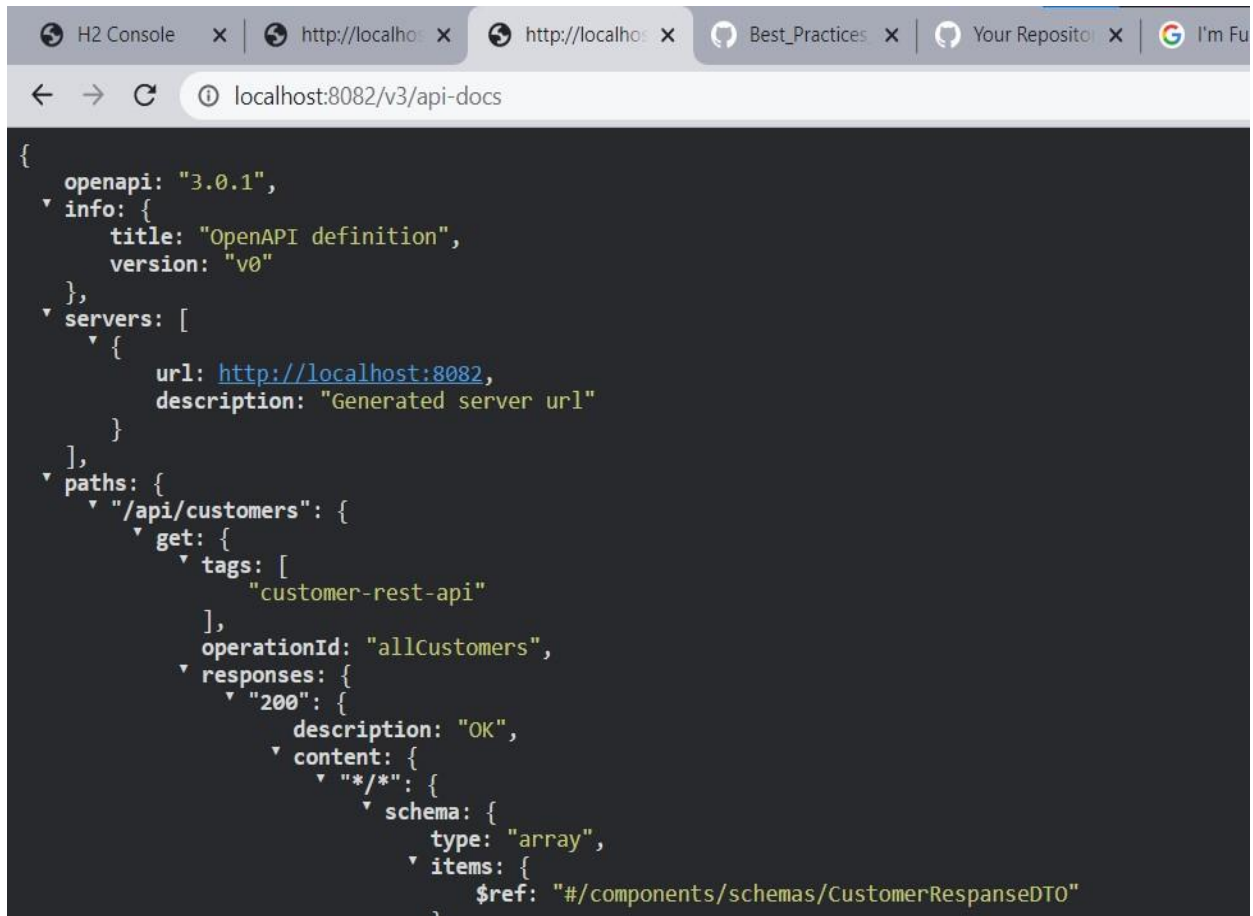
H2 Console http://localhost:8082/api/customers Best\_Practices\_Mi Your Repositories

localhost:8082/api/customers

```
[
  {
    id: "C01",
    name: "Tarik",
    email: "Tarik@Tarik.com"
  },
  {
    id: "C02",
    name: "OpenLab",
    email: "OpenLab@OpenLab.com"
  },
  {
    id: "8516fc68-4289-4a62-8b15-ae464ab43200",
    name: "Tarik 2",
    email: "Tarik2@Tarik2.com"
  }
]
```

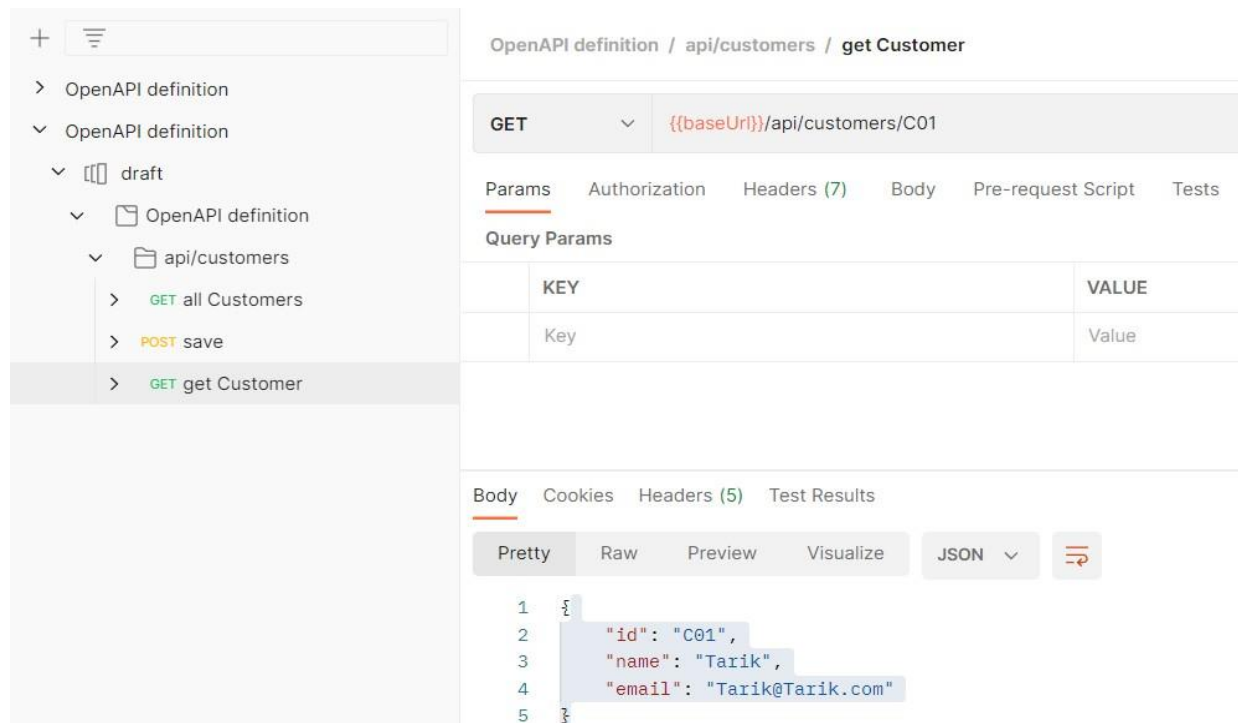


## Documentation Api



The screenshot shows a web browser with the address bar displaying `localhost:8082/v3/api-docs`. The page content is a JSON representation of an OpenAPI definition. The JSON structure is as follows:

```
{
  openapi: "3.0.1",
  info: {
    title: "OpenAPI definition",
    version: "v0"
  },
  servers: [
    {
      url: "http://localhost:8082",
      description: "Generated server url"
    }
  ],
  paths: {
    "/api/customers": {
      get: {
        tags: [
          "customer-rest-api"
        ],
        operationId: "allCustomers",
        responses: {
          "200": {
            description: "OK",
            content: {
              "*/*": {
                schema: {
                  type: "array",
                  items: {
                    $ref: "#/components/schemas/CustomerResponseDTO"
                  }
                }
              }
            }
          }
        }
      }
    }
  }
}
```



The screenshot shows an API client interface with a sidebar on the left and a main panel on the right. The sidebar contains a tree view of the API definition, with the following structure:

- OpenAPI definition
- OpenAPI definition
  - draft
    - OpenAPI definition
      - api/customers
        - GET all Customers
        - POST save
        - GET get Customer

The main panel displays the details for the `GET` endpoint `api/customers / get Customer`. The endpoint is `GET {{baseUrl}}/api/customers/C01`. The `Params` tab is selected, showing a table for Query Params:


KEY	VALUE
Key	Value

The `Body` tab is also visible, showing the response body in JSON format:

```
{
  "id": "C01",
  "name": "Tarik",
  "email": "Tarik@Tarik.com"
}
```

H2 Console x | http://locall x | http://locall x | Swagger UI x | Best\_Practic x

localhost:8082/swagger-ui/index.html

 **Swagger**  
Supported by SMARTBEAR

/v3/api-docs

# OpenAPI definition v0 OAS3

/v3/api-docs

## Servers

http://localhost:8082 - Generated server url

## customer-rest-api

**GET** /api/customers

**POST** /api/customers

**GET** /api/customers/{id}

## Pour le Micro-Service « BILLING-SERVICE »

La création de deux Invoices

```
@Bean
CommandLineRunner commandLineRunner(InvoiceService invoiceService) {
    return args -> {
        invoiceService.save(new InvoiceRequestDTO(BigDecimal.valueOf(7000), customerId: "C01"));
        invoiceService.save(new InvoiceRequestDTO(BigDecimal.valueOf(8000), customerId: "C02"));
    };
}
```

Systèmes Distribués et Big Data F x | Maven Repository: org.springfram x | TarikFERTAHL/Best\_Practices\_Micr x | jav

localhost:8083/h2-console/login.do?jsessionid=54280b74f801aad4931d9d088896df40

Auto commit ☒ Max rows: 1000     SQL statement:

jdbc:h2:mem:billing-db  
+ INVOICE  
+ INFORMATION\_SCHEMA  
+ Users  
i H2 2.1.214 (2022-06-13)

SELECT \* FROM INVOICE;

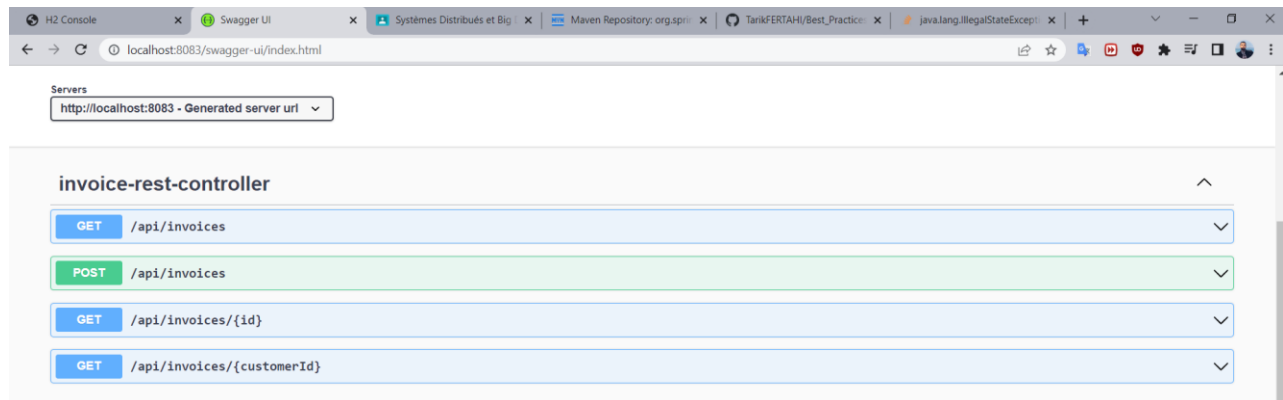
ID	AMOUNT	CUSTOMER_ID	DATE
a2e73802-4808-4e05-9a5f-87db9b19384c	7000.00	C01	2022-10-21 17:31:45.616
c7d5ef91-c91f-445d-a768-39214eb1c5d9	8000.00	C02	2022-10-21 17:31:45.726

(2 rows, 2 ms)



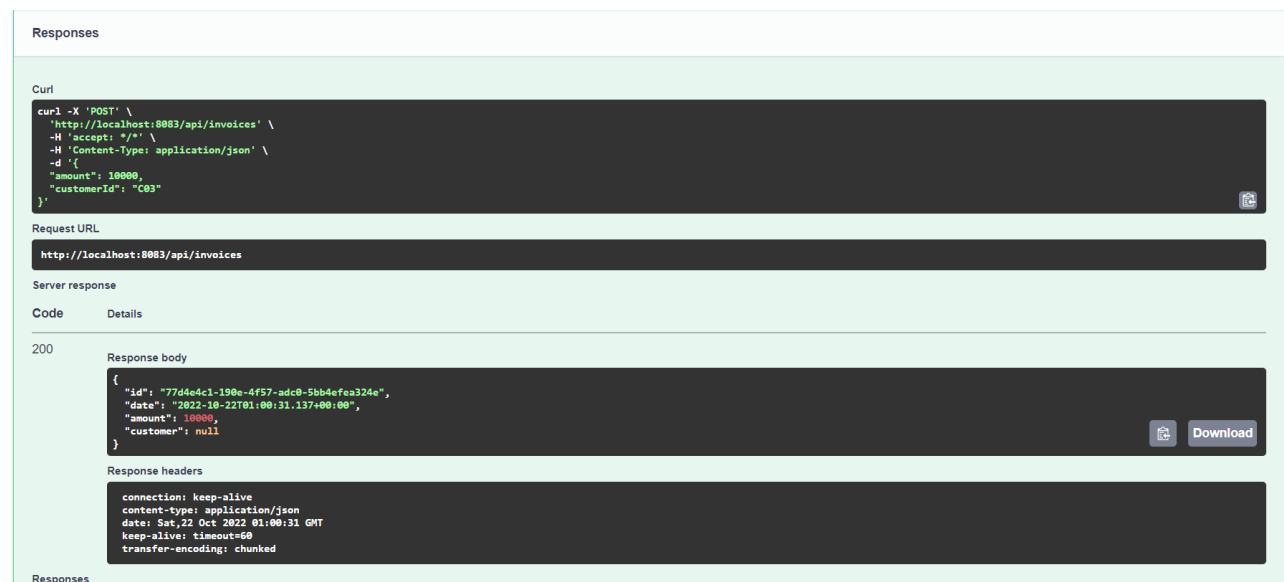
[←](#) [→](#) [↻](#) localhost:8083/v3/api-docs

```
{
  openapi: "3.0.1",
  info: {
    title: "OpenAPI definition",
    version: "v0"
  },
  servers: [
    {
      url: http://localhost:8083,
      description: "Generated server url"
    }
  ],
  paths: {
    "/api/invoices": {
      get: {
        tags: [
          "invoice-rest-controller"
        ],
        operationId: "getAllInvoices",
        responses: {
          "200": {
            description: "OK",
            content: {
              "*/*": {
                schema: {
                  type: "array",
                  items: {
                    $ref: "#/components/schemas/InvoiceResponseDTO"
                  }
                }
              }
            }
          }
        }
      },
      post: {
        tags: [
          "invoice-rest-controller"
        ],
        operationId: "save",
        requestBody: {
          content: {
            "application/json": {
              schema: {
                $ref: "#/components/schemas/InvoiceRequestDTO"
              }
            }
          }
        }
      }
    }
  }
}
```

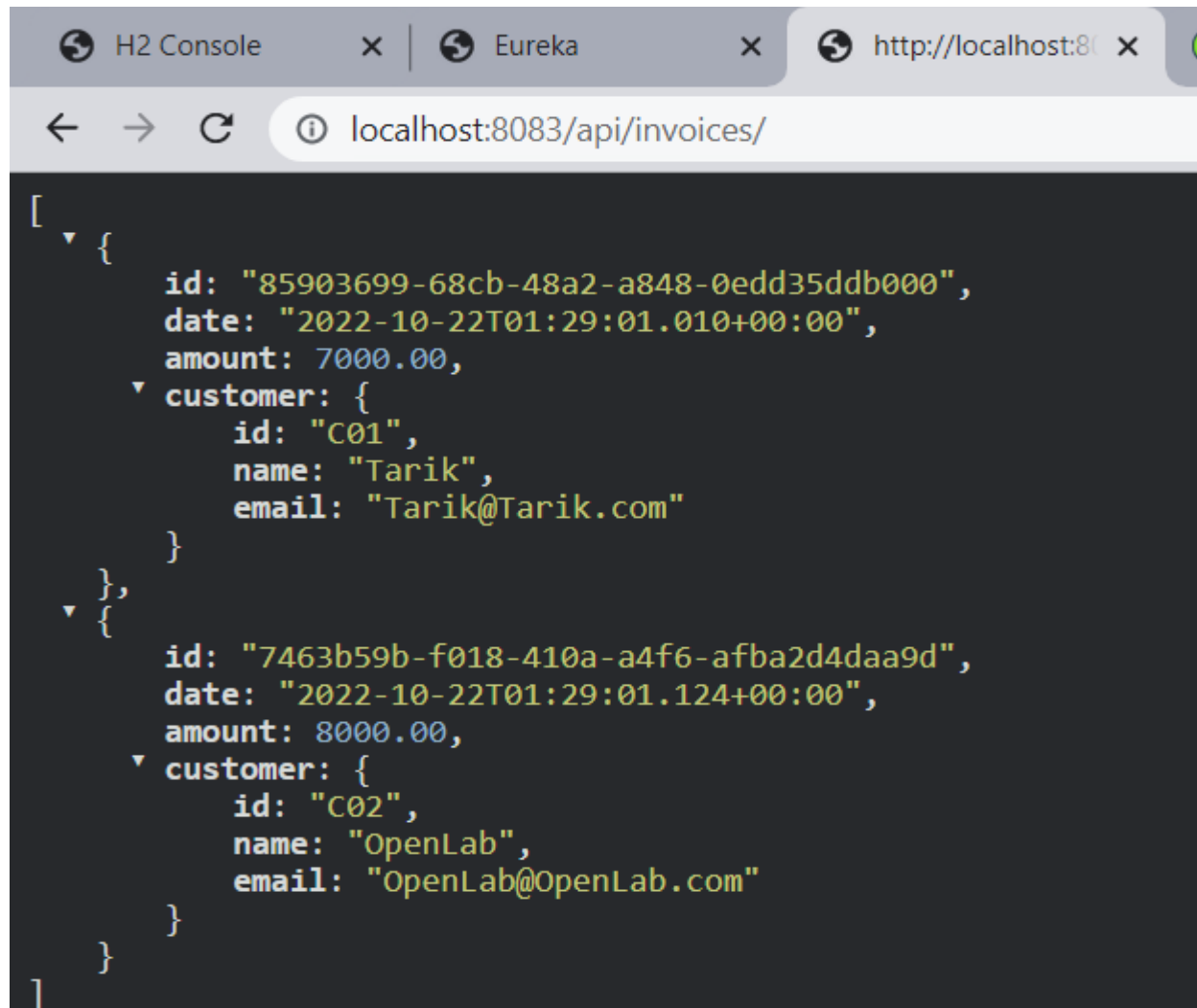


## Méthode Get

## Méthode Save

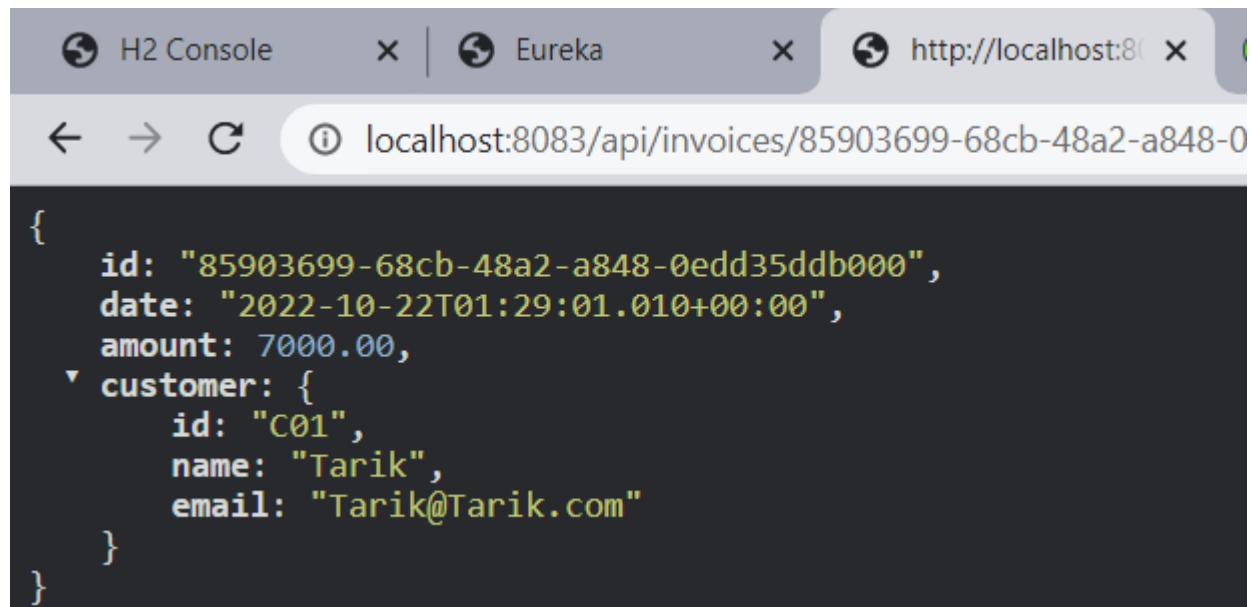


Récupérer toutes les factures



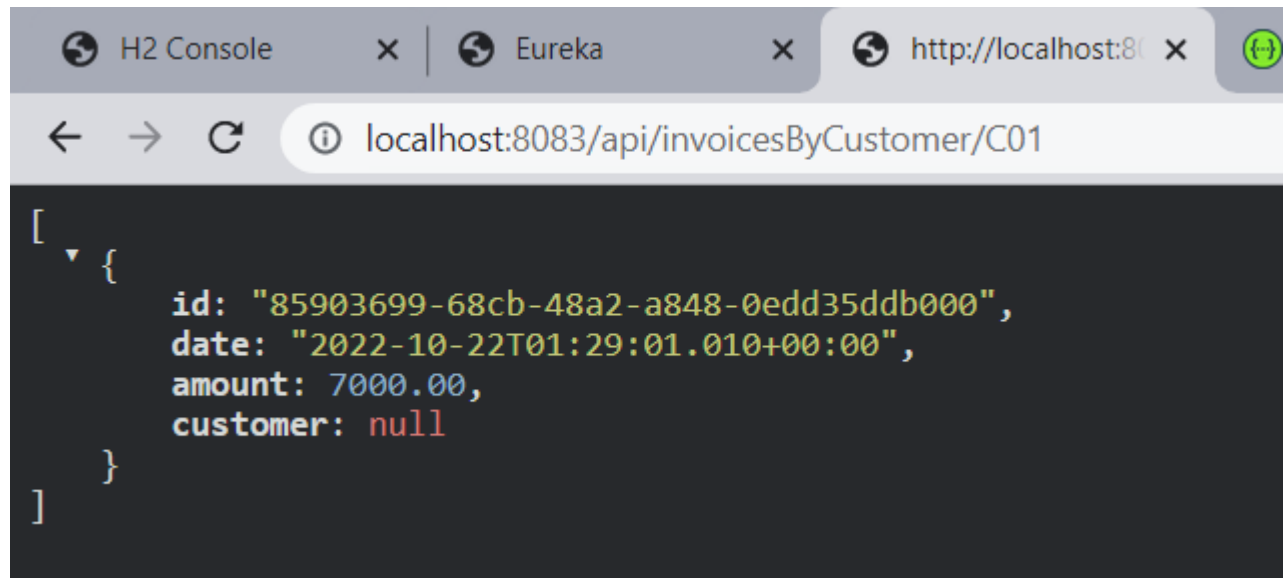
```
[
  {
    id: "85903699-68cb-48a2-a848-0edd35ddb000",
    date: "2022-10-22T01:29:01.010+00:00",
    amount: 7000.00,
    customer: {
      id: "C01",
      name: "Tarik",
      email: "Tarik@Tarik.com"
    }
  },
  {
    id: "7463b59b-f018-410a-a4f6-afba2d4daa9d",
    date: "2022-10-22T01:29:01.124+00:00",
    amount: 8000.00,
    customer: {
      id: "C02",
      name: "OpenLab",
      email: "OpenLab@OpenLab.com"
    }
  }
]
```

Récupérer une facture par Id



```
{
  id: "85903699-68cb-48a2-a848-0edd35ddb000",
  date: "2022-10-22T01:29:01.010+00:00",
  amount: 7000.00,
  customer: {
    id: "C01",
    name: "Tarik",
    email: "Tarik@Tarik.com"
  }
}
```

Récupérer toutes les factures associe a un client



The screenshot shows a web browser with three tabs: 'H2 Console', 'Eureka', and 'http://localhost:8083'. The address bar displays 'localhost:8083/api/invoicesByCustomer/C01'. The main content area shows a JSON response:

```
[
  {
    id: "85903699-68cb-48a2-a848-0edd35ddb000",
    date: "2022-10-22T01:29:01.010+00:00",
    amount: 7000.00,
    customer: null
  }
]
```

## Pour le Micro-Service « EUREKA-SERVICE »


```
@SpringBootApplication
@EnableEurekaServer
public class EurekaServiceApplication {

    public static void main(String[] args) {
        SpringApplication.run(EurekaServiceApplication.class, args);
    }
}
```

```
m pom.xml (eureka-service) x EurekaServiceApplication.java x application.properties x
1 server.port=8761
2 eureka.client.fetch-registry=false
3 eureka.client.register-with-eureka=false
```

H2 Console x Eureka x Swagger UI x Systèmes Distribués e x

localhost:8761



### System Status

Environment	test
Data center	default

localhost:8761

Renews threshold	5
Renews (last min)	2

EMERGENCY! EUREKA MAY BE INCORRECTLY CLAIMING INSTANCES ARE UP WHEN THEY'RE NOT. RENEWALS ARE LESSER THAN THRESHOLD AND HENCE THE INSTANCES ARE NOT BEING EXPIRED JUST TO BE SAFE.

DS Replicas

Instances currently registered with Eureka

Application	AMIs	Availability Zones	Status
BILLING-SERVICE	n/a (1)	(1)	UP (1) - <a href="#">localhost:BILLING-SERVICE-8083</a>
CUSTOMER-SERVICE	n/a (1)	(1)	UP (1) - <a href="#">localhost:CUSTOMER-SERVICE-8082</a>

## Pour le Micro-Service « GATEWAY »

```
@SpringBootApplication
public class GatewayApplication {

    public static void main(String[] args) {
        SpringApplication.run(GatewayApplication.class, args);
    }

    @Bean
    DiscoveryClientRouteDefinitionLocator dynamicRoutes(ReactiveDiscoveryClient discoveryClient,
        DiscoveryLocatorProperties locatorProperties){
        return new DiscoveryClientRouteDefinitionLocator(discoveryClient, locatorProperties);
    }
}
```

```
<dependency>
    <groupId>org.springframework.cloud</groupId>
    <artifactId>spring-cloud-starter-gateway</artifactId>
</dependency>
```

```
<dependency>
    <groupId>org.springframework.cloud</groupId>
    <artifactId>spring-cloud-starter-netflix-eureka-client</artifactId>
</dependency>
```

H2 Console x Eureka x http://localhost: x Swagger UI x Systèmes Distrib. x Maven Repositor. x TarikFERTAH/Be: x java.lang.IllegalSt: x +

localhost:8761

EMERGENCY: Eureka may be inconsistent: scanning instances are off when they're not. Renewals are lesser than threshold and hence the instances are not being expired just to be safe.

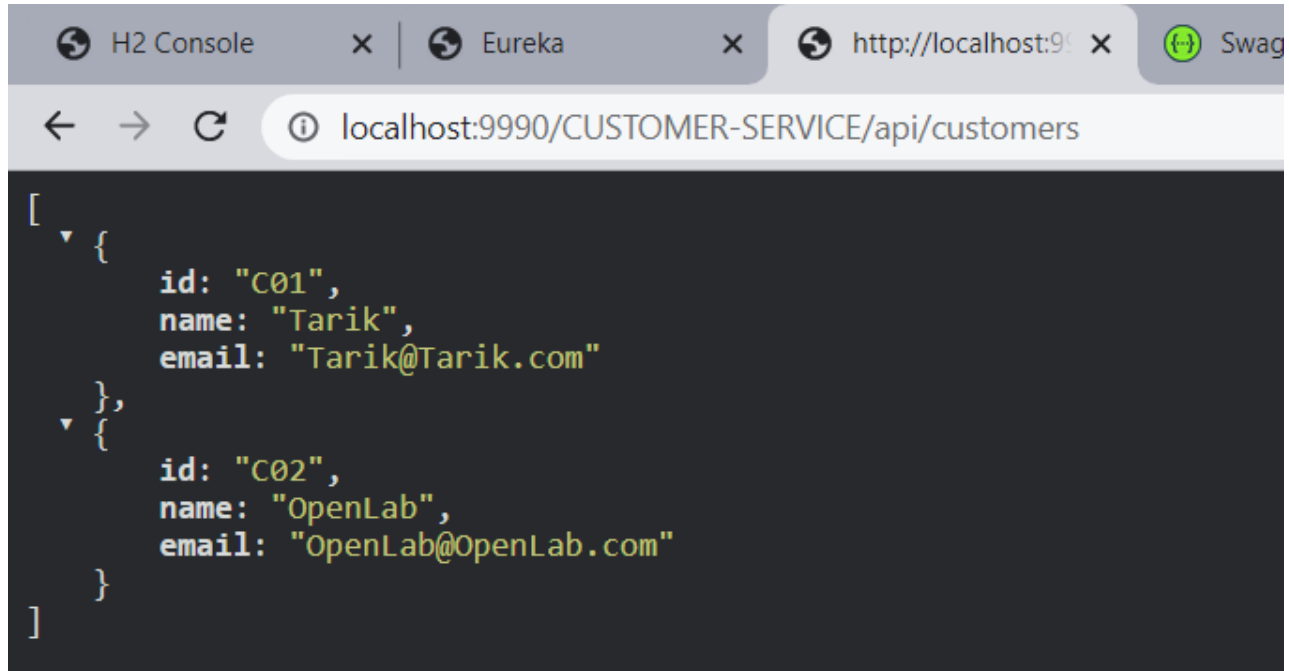
### DS Replicas

Instances currently registered with Eureka

Application	AMIs	Availability Zones	Status
BILLING-SERVICE	n/a (1)	(1)	UP (1) - <a href="#">localhost:BILLING-SERVICE:8083</a>
CUSTOMER-SERVICE	n/a (1)	(1)	UP (1) - <a href="#">localhost:CUSTOMER-SERVICE:8082</a>
GATEWAY	n/a (1)	(1)	UP (1) - <a href="#">localhost:GATEWAY:9990</a>

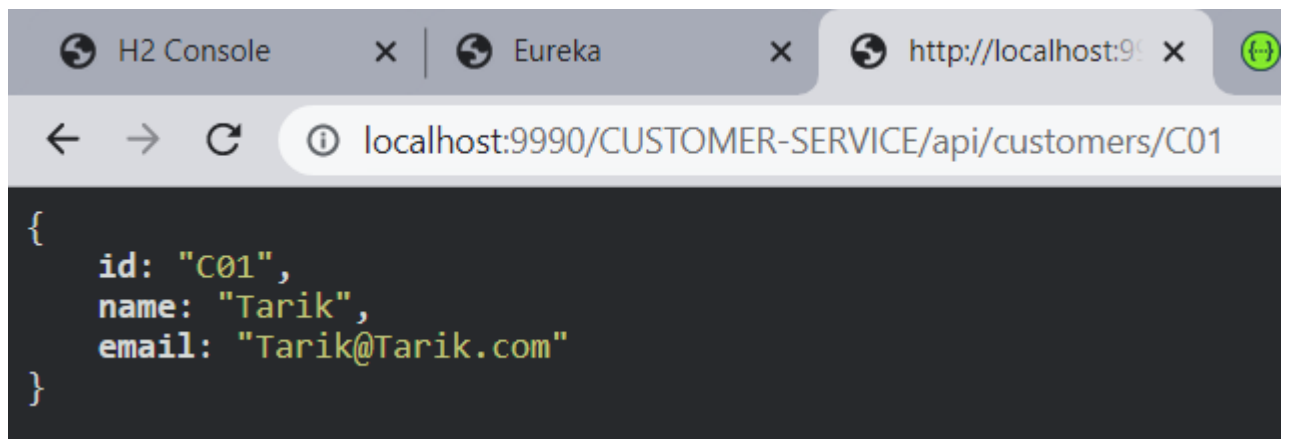


Récupérer la liste des clients en passant par **Gateway**



```
[
  {
    id: "C01",
    name: "Tarik",
    email: "Tarik@Tarik.com"
  },
  {
    id: "C02",
    name: "OpenLab",
    email: "OpenLab@OpenLab.com"
  }
]
```

Récupérer un client par id en passant par **Gateway**



```
{
  id: "C01",
  name: "Tarik",
  email: "Tarik@Tarik.com"
}
```

## Méthode Post

POST

http://localhost:9990/CUSTOMER-SERVICE/api/customers/

Params

Authorization

Headers (10)

Body

Pre-request Script

Tests

Settings

☐ none

☐ form-data

☐ x-www-form-urlencoded

☒ raw

☐ binary

☐ GraphQL

JSON

```
1 {
2   "name": "gateway",
3   "email": "gateway@gateway.com"
4 }
```

Body

Cookies

Headers (3)

Test Results

Pretty

Raw

Preview

Visualize

JSON

```
1 {
2   "id": "d900d1d2-0ba5-428b-a854-c8812f2feb91",
3   "name": "gateway",
4   "email": "gateway@gateway.com"
5 }
```

H2 Console

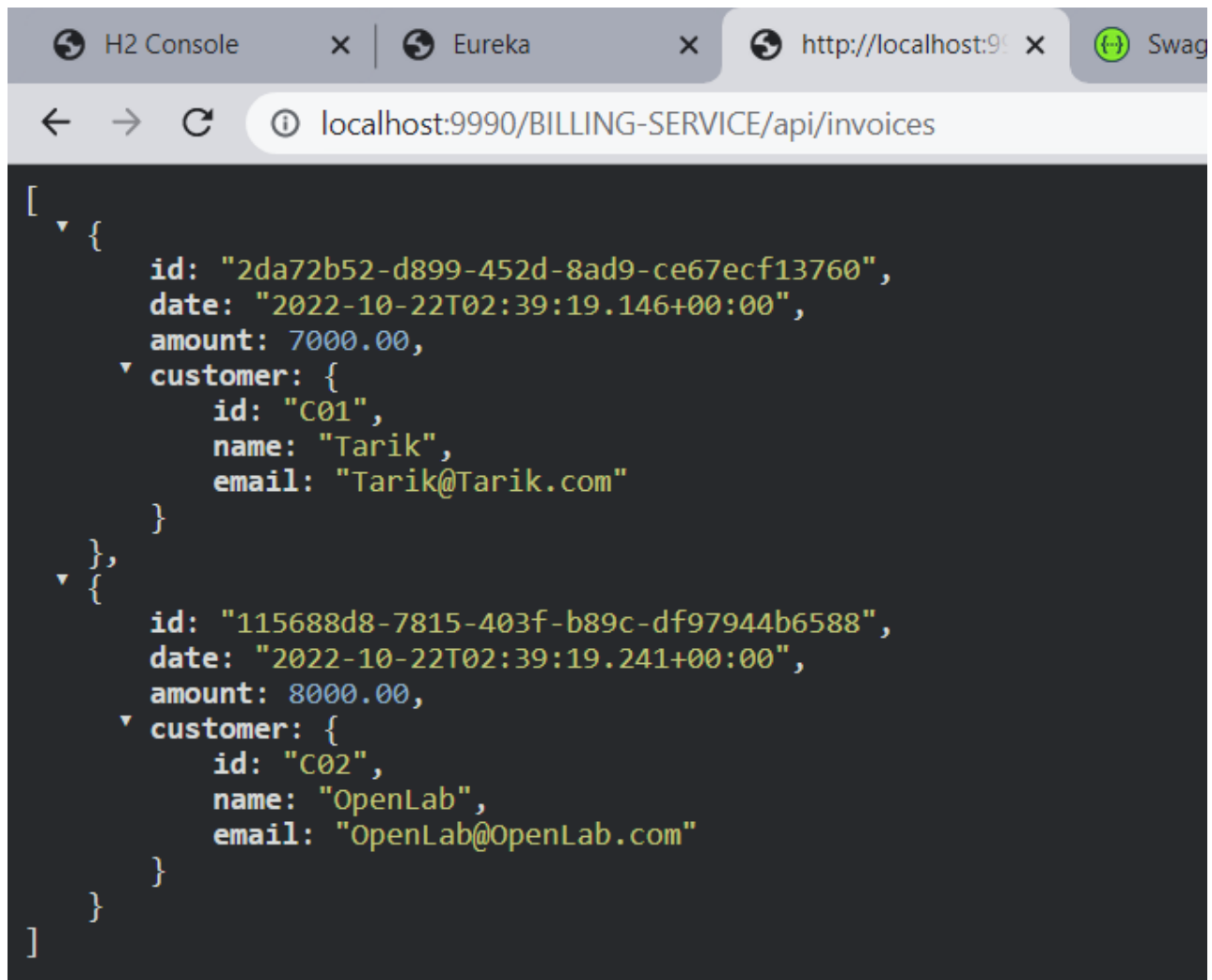
Eureka

http://localhost:9990/CUSTOMER-SERVICE/api/customers/

Swagger UI

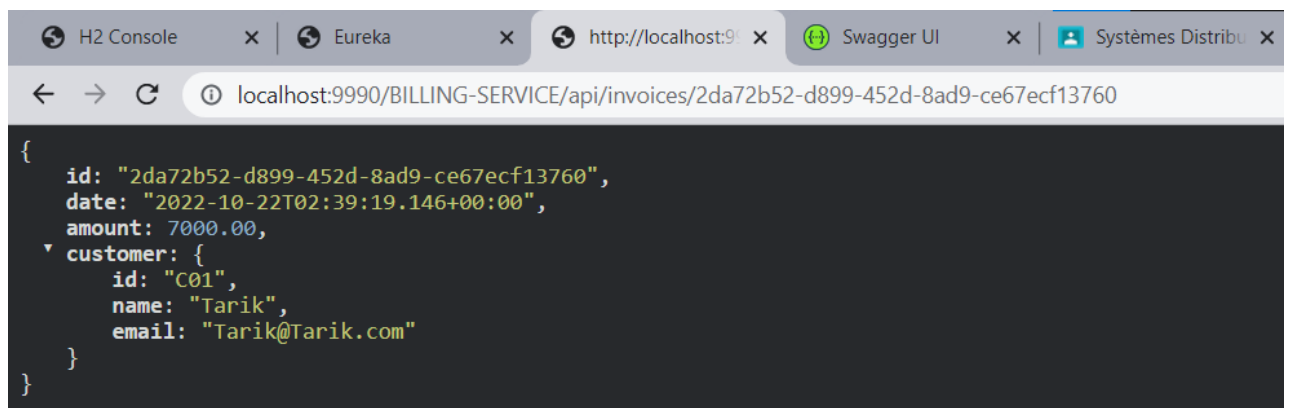
```
[
  {
    id: "C01",
    name: "Tarik",
    email: "Tarik@Tarik.com"
  },
  {
    id: "C02",
    name: "OpenLab",
    email: "OpenLab@OpenLab.com"
  },
  {
    id: "d900d1d2-0ba5-428b-a854-c8812f2feb91",
    name: "gateway",
    email: "gateway@gateway.com"
  }
]
```

Récupérer la liste des factures en passant par **Gateway**



```
[
  {
    id: "2da72b52-d899-452d-8ad9-ce67ecf13760",
    date: "2022-10-22T02:39:19.146+00:00",
    amount: 7000.00,
    customer: {
      id: "C01",
      name: "Tarik",
      email: "Tarik@Tarik.com"
    }
  },
  {
    id: "115688d8-7815-403f-b89c-df97944b6588",
    date: "2022-10-22T02:39:19.241+00:00",
    amount: 8000.00,
    customer: {
      id: "C02",
      name: "OpenLab",
      email: "OpenLab@OpenLab.com"
    }
  }
]
```

Récupérer une facture par id en passant par **Gateway**



```
{
  id: "2da72b52-d899-452d-8ad9-ce67ecf13760",
  date: "2022-10-22T02:39:19.146+00:00",
  amount: 7000.00,
  customer: {
    id: "C01",
    name: "Tarik",
    email: "Tarik@Tarik.com"
  }
}
```