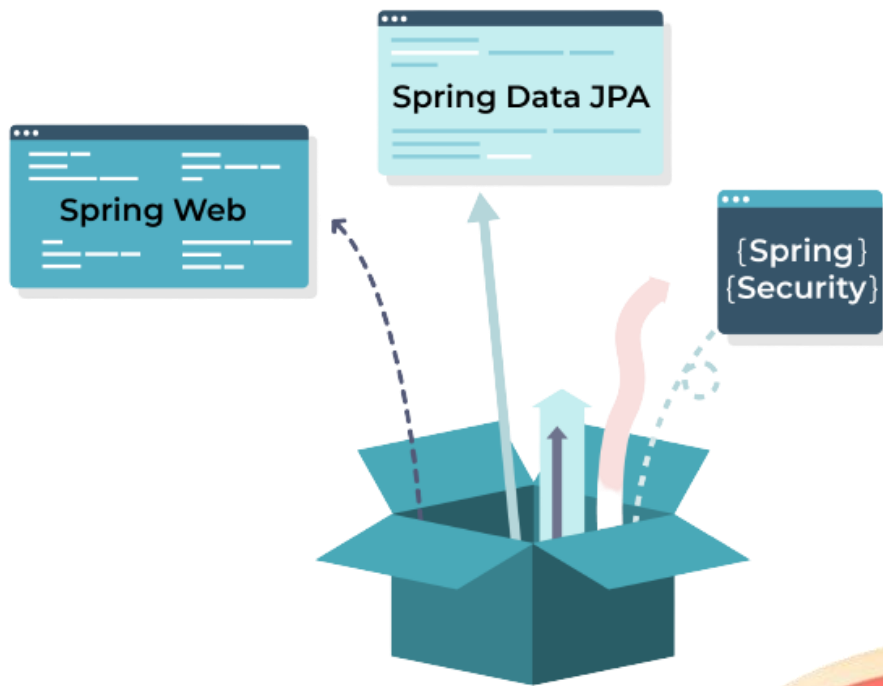


Compte rendu de :

SEANCE 09



Réalisé Par

: AYOUIJIL Soukayna

Département

: Mathématiques et Informatique

Filière

: II-BDCC

Professeur

: M. Mohamed YOUSSEFI

Etape 1 : Création de l'entité Product

```
Product.java  ProductRepository.java  EcomAppApplication.java  application.properties
1 package org.sid.ecomapp.entities;
2
3 import lombok.AllArgsConstructor;
4 import lombok.Data;
5 import lombok.NoArgsConstructor;
6
7 import javax.persistence.Entity;
8 import javax.persistence.Id;
9
10 @Entity
11 @Data @NoArgsConstructor @AllArgsConstructor
12 public class Product {
13     @Id
14     private String id;
15     private String name;
16     private double price;
17     private double quantity;
18 }
19
```

Etape 2 : Création de l'interface ProductRepository

```
package org.sid.ecomapp.repositories;

import org.sid.ecomapp.entities.Product;
import org.springframework.data.jpa.repository.JpaRepository;

public interface ProductRepository extends JpaRepository<Product,String> {
}

```

Etape 3 : Génération d'une liste des produits

```
@Bean
public CommandLineRunner start(ProductRepository productRepository){
    return args -> {
        Stream.of("Computer","Printer","SmartPhone").forEach(
            name->{
                productRepository.save(new Product(UUID.randomUUID().toString(),name, price: Math.random()*8000, quantity: Math.random()*100));
            }
        );
    };
}

```

Etape 4 : Compléter le fichier application.properties

```
1 spring.datasource.url=jdbc:h2:mem:products-db
2 spring.h2.console.enabled=true
3 server.port=8086

```

Etape 5 : Démarrer l'application sur le port 8086

localhost:8086/h2-console/login.do?sessionId=ce8b7a38d51df7a8dfb2a44a7590e09e

Auto commit: ☒ Max rows: 1000 Auto complete: ☐ Auto select: ☐

Run Run Selected Auto complete Clear SQL statement:

SELECT * FROM PRODUCT

SELECT * FROM PRODUCT;

ID	NAME	PRICE	QUANTITY
1f9fa046-c0ac-4e69-b658-948362d838e2	Computer	3193.3429420288758	75.77571220785745
21953f41-96fd-47d5-af98-711885424bba	Printer	5590.007016992583	27.14026041575488
431dabd6-6555-4333-abbb-9a0b0e172745	SmartPhone	173.57536156619435	83.21229414910113

(3 rows, 11 ms)

Edit

Etape 6 : créer un Rest Controller

```
ecom-app - ProductRestController.java
File Edit View Navigate Code Refactor Build Run Tools VCS Window Help
ecom-app - ProductRestController.java
Product.java ProductRepository.java EcomAppApplication.java application.properties ProductRestController.java
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.RestController;

import java.util.List;

@RestController
public class ProductRestController {

    @Autowired
    private ProductRepository productRepository;

    @GetMapping(path = "/Products")
    public List<Product> productsList(){
        return productRepository.findAll();
    }

    @GetMapping(path = "/products/{id}")
    public Product getProduct(@PathVariable(name = "id") String id){
        return productRepository.findById(id).get();
    }
}
```

Etape 7 : Resultats

- Consulter Liste des produits

```
localhost:8086/products/
1 // 20220506152748
2 // http://localhost:8086/products/
3
4 [
5 {
6   "id": "d66ae583-7bce-4da2-b692-e1b6d67d7920",
7   "name": "Computer",
8   "price": 3830.281092201886,
9   "quantity": 98.2543368738524
10 },
11 {
12   "id": "7854d052-1825-4046-bf5f-e9a04980ce36",
13   "name": "Printer",
14   "price": 3779.4895794661948,
15   "quantity": 91.79667108176004
16 },
17 {
18   "id": "6073390d-48bf-40e4-9f68-e1cdc05834a5",
19   "name": "SmartPhone",
20   "price": 2979.9303094356064,
21   "quantity": 18.00750706521238
22 }
23 ]
```

- Consulter un produit a l'aide de son id

```
localhost:8086/products/93ce5838-3a3b-4243-9228-78e31db86434
1 // 20220506151207
2 // http://localhost:8086/products/93ce5838-3a3b-4243-9228-78e31db86434
3
4 {
5   "id": "93ce5838-3a3b-4243-9228-78e31db86434",
6   "name": "Computer",
7   "price": 3887.7695693915202,
8   "quantity": 70.2633762342597
9 }
```

Etape 7 : Création des méthodes ajouter, modifier et supprimer un produit

```
@PostMapping(path = "/products")
public Product saveProduct(@RequestBody Product p){
    p.setId(UUID.randomUUID().toString());
    return productRepository.save(p);
}

@PutMapping(path = "/products/{id}")
public Product updateProduct(@RequestBody Product p,@PathVariable(name = "id") String id){
    p.setId(id);
    return productRepository.save(p);
}

@DeleteMapping(path = "/products/{id}")
public void deleteProduct(@PathVariable(name = "id") String id){
    productRepository.deleteById(id);
}
```

Etape 8 : Resultats

- Consulter Liste des produits / un seul produit

Overview GET http://localhost:8086/ No Environment

http://localhost:8086/products Save

GET http://localhost:8086/products Send

Params Authorization Headers (7) Body Pre-request Script Tests Settings Cookies

Body Cookies Headers (5) Test Results 200 OK 370 ms 524 B Save Response

Pretty Raw Preview Visualize JSON

```
1 [
2   {
3     "id": "d66ae583-7bce-4da2-b692-e1b6d67d7920",
4     "name": "Computer",
5     "price": 3830.281092201886,
6     "quantity": 98.2543368738524
7   },
8   {
9     "id": "7854d052-1825-4046-bf5f-e9a04980ce36",
10    "name": "Printer",
11    "price": 3779.4895794661948,
12    "quantity": 91.79667108176004
13  },
14  {
15    "id": "6073390d-48bf-40e4-9f68-e1cdc05834a5",
16    "name": "SmartPhone",
17    "price": 2979.9202004256064
```

GET http://localhost:8086/products/d66ae583-7bce-4da2-b692-e1b6d67d7920 Send

Params Authorization Headers (7) Body Pre-request Script Tests Settings Cookies

Body Cookies Headers (5) Test Results 200 OK 368 ms 281 B Save Response

Pretty Raw Preview Visualize JSON

```
1 {
2   "id": "d66ae583-7bce-4da2-b692-e1b6d67d7920",
3   "name": "Computer",
4   "price": 3830.281092201886,
5   "quantity": 98.2543368738524
6 }
```

- Ajouter un produit

POST

http://localhost:8086/products

Send

ParamsAuthorizationHeaders (11)BodyPre-request ScriptTestsSettingsCookies

noneform-datax-www-form-urlencodedorawbinaryGraphQLJSONBeautify

```
1 {
2   ... "name": "Laptop",
3   ... "price": 38300,
4   ... "quantity": 9
5 }
```

BodyCookiesHeaders (5)Test Results200 OK501 ms256 BSave Response

PrettyRawPreviewVisualizeJSON

```
1 {
2   "id": "e9f7330f-8751-4a7e-8b66-668af266b040",
3   "name": "Laptop",
4   "price": 38300.0,
5   "quantity": 9.0
6 }
```

- Modifier un produit

PUT

http://localhost:8086/products/e9f7330f-8751-4a7e-8b66-668af266b040

Send

ParamsAuthorizationHeaders (11)BodyPre-request ScriptTestsSettingsCookies

noneform-datax-www-form-urlencodedorawbinaryGraphQLJSONBeautify

```
1 {
2   ... "name": "Laptop 777",
3   ... "price": 377300,
4   ... "quantity": 6
5 }
```

BodyCookiesHeaders (5)Test Results200 OK36 ms261 BSave Response

PrettyRawPreviewVisualizeJSON

```
1 {
2   "id": "e9f7330f-8751-4a7e-8b66-668af266b040",
3   "name": "Laptop 777",
4   "price": 377300.0,
5   "quantity": 6.0
6 }
```

- Supprimer un produit

DELETE

http://localhost:8086/products/e9f7330f-8751-4a7e-8b66-668af266b040

Send

ParamsAuthorizationHeaders (11)BodyPre-request ScriptTestsSettingsCookies

noneform-datax-www-form-urlencodedorawbinaryGraphQLJSONBeautify

```
1 {
2   ... "name": "Laptop 777",
3   ... "price": 377300,
4   ... "quantity": 6
5 }
```

BodyCookiesHeaders (4)Test Results200 OK60 ms123 BSave Response

Etape 9 : Création de l'entité Category

```
@Entity
@Data @NoArgsConstructor @AllArgsConstructor
public class Category {
    @Id @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long id;
    private String name;
    @OneToMany(mappedBy = "category")
    private List<Product> products = new ArrayList<>();
}
```

Etape 10 : Création du repository CategoryRepository

```
1 package org.sid.ecomapp.repositories;
2
3 import ...
4
5
6
7 public interface CategoryRepository extends JpaRepository<Category, Long> {
8 }
9
```

Etape 11 : Création des produits et categories

```
@Bean
public CommandLineRunner start(ProductRepository productRepository, CategoryRepository categoryRepository){
    return args -> {
        Stream.of("Computer", "Printer", "SmartPhone").forEach(
            name->{
                Category c = new Category();
                c.setName(name);
                categoryRepository.save(c);
            }
        );
        categoryRepository.findAll().forEach(cat->{
            for (int i = 1; i <= 5; i++){
                Product product = new Product();
                product.setId(UUID.randomUUID().toString());
                product.setPrice(Math.random()*9000);
                product.setQuantity(1+Math.random()*50);
                product.setName(cat.getName()+"-"+i);
                product.setCategory(cat);
                productRepository.save(product);
            }
        });
    };
}
```

Etape 12 : Resultats

Run Run Selected Auto complete Clear SQL statement:

```
SELECT * FROM CATEGORY|
```

SELECT * FROM CATEGORY;

ID	NAME
1	Computer
2	Printer
3	SmartPhone

(3 rows, 5 ms)

Edit

Run Run Selected Auto complete Clear SQL statement:

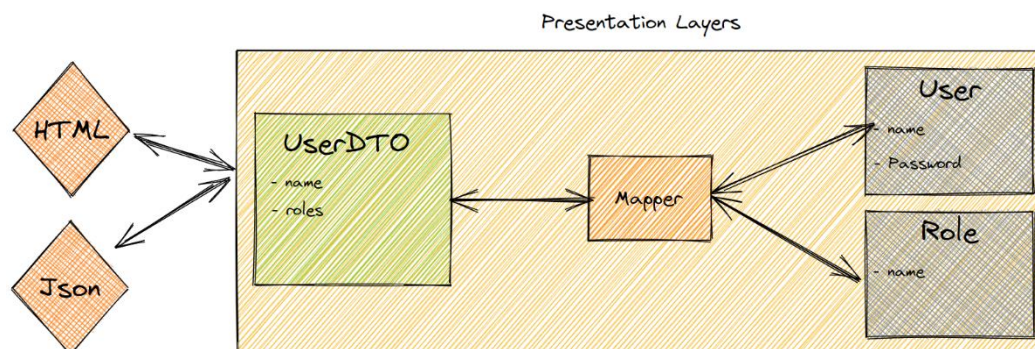
```
SELECT * FROM PRODUCT
```

SELECT * FROM PRODUCT;

ID	NAME	PRICE	QUANTITY	CATEGORY_ID
e11b32f7-4ae8-4ef0-b18e-7261c8f5ebd6	Computer-1	8690.369622665423	27.835907078367278	1
af40866e-bd04-4cef-b73f-bb7e862b59ab	Computer-2	3923.2621200083927	42.34470091642437	1
bd93eb3-fa5c-45e1-8b34-94f89eb0ede4	Computer-3	2632.2677621007715	45.29745959716875	1
416f5e65-4261-4227-aac3-cb262a6024c7	Computer-4	6965.283708734053	48.706114414954065	1
26e46c78-382d-4c9d-a5db-927d03c49f8a	Computer-5	4139.458036526757	44.95812141822144	1
ea1d912c-21c1-4e1b-b74c-ed4191ca4305	Printer-1	4656.072134112546	34.02476313188289	2
094c0fe9-843c-4db4-8c06-f0bd8c440d7e	Printer-2	6346.829095895288	23.153481209347532	2
2d441081-62df-4250-93bc-24eb4848586d	Printer-3	1105.5035024424124	11.732040830526202	2
b2b0449b-770b-4c28-bcc8-11ba13c40dee	Printer-4	4138.005551834449	3.4543258990348926	2
404b3853-e51a-4252-bf7a-ec6b6377547e	Printer-5	6902.932038232382	18.461311529391104	2
bd91c073-7646-4f7d-97f8-f0e38ad9689b	SmartPhone-1	6731.794046346893	20.371802571967017	3
b5b1fbfb-818c-44f4-b272-88358274ff10	SmartPhone-2	172.0979852288874	44.683030156864625	3
51982b7d-b251-434e-b278-ed0b66f342b6	SmartPhone-3	2036.7356100998454	3.0691148643116772	3
e29807a7-617a-4127-b07e-26a5d69f5572	SmartPhone-4	7759.755813583914	11.825430665622854	3
03c785e9-c157-4565-8f6b-98196d5a3e7b	SmartPhone-5	8132.190886647338	15.819562097553673	3

Etape 13 : Création du modèle DTO (objet de transfert de données)

- Comment l'utiliser ?



<https://www.baeldung.com/java-dto-pattern#how-to-use-it>

- **Quand l'utiliser :**

Les DTO sont utiles dans les systèmes avec des appels à distance, car ils aident à en réduire le nombre.

Les DTO sont également utiles lorsque le modèle de domaine est composé de nombreux objets différents et que le modèle de présentation a besoin de toutes leurs données en même temps, ou ils peuvent même réduire les allers-retours entre le client et le serveur.

Avec les DTO, nous pouvons créer différentes vues à partir de nos modèles de domaine, ce qui nous permet de créer d'autres représentations du même domaine mais en les optimisant en fonction des besoins des clients sans affecter notre conception de domaine. Une telle flexibilité est un outil puissant pour résoudre des problèmes complexes.

<https://www.baeldung.com/java-dto-pattern#how-to-use-it>

➤ Classe ProductDTO

```
ProductDTO.java × Product.java × IProductService.java × IProductServiceImpl.java × ProductRestController.java × CatalogMappe
1 package org.sid.ecomapp.dtos;
2
3
4 import lombok.AllArgsConstructor;
5 import lombok.Data;
6 import lombok.NoArgsConstructor;
7
8 @Data @NoArgsConstructor @AllArgsConstructor
9 public class ProductDTO {
10     private String id;
11     private String name;
12     private double price;
13     private double quantity;
14     private CategoryDTO categoryDTO;
15 }
```

➤ Classe CategoryDTO

```
1 package org.sid.ecomapp.dtos;
2
3
4 import lombok.AllArgsConstructor;
5 import lombok.Data;
6 import lombok.NoArgsConstructor;
7 import javax.persistence.*;
8
9 @Data @NoArgsConstructor @AllArgsConstructor
10 public class CategoryDTO {
11     @Id @GeneratedValue(strategy = GenerationType.IDENTITY)
12     private Long id;
13     private String name;
14 }
15
```


Etape 14 : Création de la couche service : partie métier de l'application

➤ Interface IProductService

```
public interface IProductService {  
    ProductDTO save(ProductDTO productDTO);  
    List<ProductDTO> productsList();  
    ProductDTO getProduct(String id);  
    ProductDTO updateProduct(ProductDTO productDTO);  
    void deleteProduct(String id);  
}
```

➤ Implémentation de l'interface IProductService

```
@Service  
@Transactional  
public class IProductServiceImpl implements IProductService {  
    @Autowired  
    private ProductRepository productRepository;  
    @Autowired  
    private CatalogMappers catalogMappers;  
    @Override  
    public ProductDTO save(ProductDTO productDTO) {  
        Product product = catalogMappers.fromProductDTO(productDTO);  
        product.setId(UUID.randomUUID().toString());  
        Product savedProduct = productRepository.save(product);  
        return catalogMappers.fromProduct(savedProduct);  
    }  
  
    @Override  
    public List<ProductDTO> productsList() {  
        List<Product> products = productRepository.findAll();  
        List<ProductDTO> productDTOS = products.stream().map(  
            p->catalogMappers.fromProduct(p)).collect(Collectors.toList());  
        return productDTOS;  
    }  
}
```

```
@Autowired  
private IProductService iProductService;  
@GetMapping(path = "/products" )  
public List<ProductDTO> productsList(){  
    return iProductService.productsList();  
}  
  
@GetMapping(path = "/products/{id}")  
public Product getProduct(@PathVariable(name = "id") String id){  
    return productRepository.findById(id).get();  
}  
  
@PostMapping(path = "/products")  
public ProductDTO saveProduct(@RequestBody ProductDTO p){  
    return iProductService.save(p);  
}
```

Etape 15 : Resultats

- Ajouter un produit

POST

http://localhost:8086/products

Send

Params

Authorization

Headers (11)

Body

Pre-request Script

Tests

Settings

Cookies

none

form-data

x-www-form-urlencoded

raw

binary

GraphQL

JSON

Beautify

```
1 {
2   .... "name": "HP Smart 999",
3   .... "price": 43000,
4   .... "quantity": 5,
5   .... "categoryDTO": {
6     .... "id": 1
7   }
8 }
```

Body

Cookies

Headers (5)

Test Results

200 OK

141 ms

303 B

Save Response

Pretty

Raw

Preview

Visualize

```
{
  "id": "a91e9833-b26c-4468-9093-c9acff897f8d",
  "name": "HP Smart 999",
  "price": 43000.0,
  "quantity": 5.0,
  "categoryDTO": {
    "id": 1,
    "name": "Computer"
  }
}
```

Run	Run Selected	Auto complete	Clear	SQL statement:
SELECT * FROM PRODUCT				
71323eef-6bda-4615-be07-a6fc3ce33889	Computer-3	756.7807305058826	34.12235772857406	1
ed84824b-ba9e-4206-bf86-f5621c402bdb	Computer-4	2181.9143542519964	27.58128691062118	1
c584170f-90f0-4479-9da3-7812ea8d84a8	Computer-5	1539.190963231758	5.484767676228168	1
8f24ee59-6526-4ae8-bc9c-7c155be0b799	Printer-1	4554.990768911559	12.494807282089015	2
02143c22-b1ba-4724-b99d-a06d56db9b6f	Printer-2	7331.221654005137	44.505587273225636	2
ae0de449-9d67-4103-b7cf-f184c7462c3e	Printer-3	6764.580773640632	7.212428232712636	2
0b2f2280-f49b-4a02-bfe8-d656babb8c37	Printer-4	5022.935241185435	41.69267043276204	2
affd5d0c-9887-4c86-8de3-a0af6cec4cd1	Printer-5	1229.359694788638	31.960759432983863	2
51bff9e-c8fe-48d2-be51-6d936626f77b	SmartPhone-1	6270.890678283063	41.45621041785973	3
92011903-7461-4d8c-8dd9-cf8c0e3b3c2f	SmartPhone-2	3295.994912324504	20.48399733193976	3
04eddfaa-3c47-4d20-b262-3b54cbf96c06	SmartPhone-3	8188.423743927855	24.254390898484843	3
af176e03-0ae2-4d5c-8951-35fdf26b7d7f	SmartPhone-4	2192.4204158242414	40.39646427636488	3
c3f44777-dfea-4edb-aa91-1fc3eb5edd6a	SmartPhone-5	2142.732841216112	36.71079815226233	3
a91e9833-b26c-4468-9093-c9acff897f8d	HP Smart 999	43000.0	5.0	1
(16 rows, 7 ms)				

http://localhost:8086/products

GET http://localhost:8086/products Send

Params Authorization Headers (11) Body ● Pre-request Script Tests Settings Cookies

Body Cookies Headers (5) Test Results 200 OK 425 ms 2.55 KB Save Response

Pretty Raw Preview Visualize JSON

```

1  [
2    {
3      "id": "bc23bd73-5678-43d5-b4ed-67109ebc9aab",
4      "name": "Computer-1",
5      "price": 7923.25698044296,
6      "quantity": 33.56270482805317,
7      "categoryDTO": {
8        "id": 1,
9        "name": "Computer"
10     }
11  },
12  {
13    "id": "664e268a-f109-4ac8-8d74-a3117b3a7f8e",
14    "name": "Computer-2",
15    "price": 3624.4655789856192,
16    "quantity": 1.0505031107365828,

```

• Modifier un produit

http://localhost:8086/products/bc23bd73-5678-43d5-b4ed-67109ebc9aab

PUT http://localhost:8086/products/bc23bd73-5678-43d5-b4ed-67109ebc9aab Send

Params Authorization Headers (11) Body ● Pre-request Script Tests Settings Cookies

● none ● form-data ● x-www-form-urlencoded ● raw ● binary ● GraphQL JSON Beautify

```

1  {
2    "name": "HP Smart 999",
3    "price": 5000,
4    "quantity": 5,
5    "categoryDTO": {
6      "id": 2

```

Body Cookies Headers (5) Test Results 200 OK 166 ms 301 B Save Response

Pretty Raw Preview Visualize JSON

```

2    "id": "bc23bd73-5678-43d5-b4ed-67109ebc9aab",
3    "name": "HP Smart 999",
4    "price": 5000.0,
5    "quantity": 5.0,
6    "categoryDTO": {
7      "id": 2,
8      "name": "Printer"

```

SELECT * FROM PRODUCT;

ID	NAME	PRICE	QUANTITY	CATEGORY_ID
bc23bd73-5678-43d5-b4ed-67109ebc9aab	HP Smart 999	5000.0	5.0	2

- Supprimer un produit

DELETE

http://localhost:8086/products/bc23bd73-5678-43d5-b4ed-67109ebc9aab

Send

Params

Authorization

Headers (11)

Body

Pre-request Script

Tests

Settings

Cookies

none

form-data

x-www-form-urlencoded

raw

binary

GraphQL

JSON

Beautify

1 {

2 "name": "HP Smart 999",

3 "price": 5000,

4 "quantity": 5,

5 "categoryDT0":

6 "id": 2

Body

Cookies

Headers (4)

Test Results

200 OK

57 ms

123 B

Save Response

Pretty

Raw

Preview

Visualize

Text

1