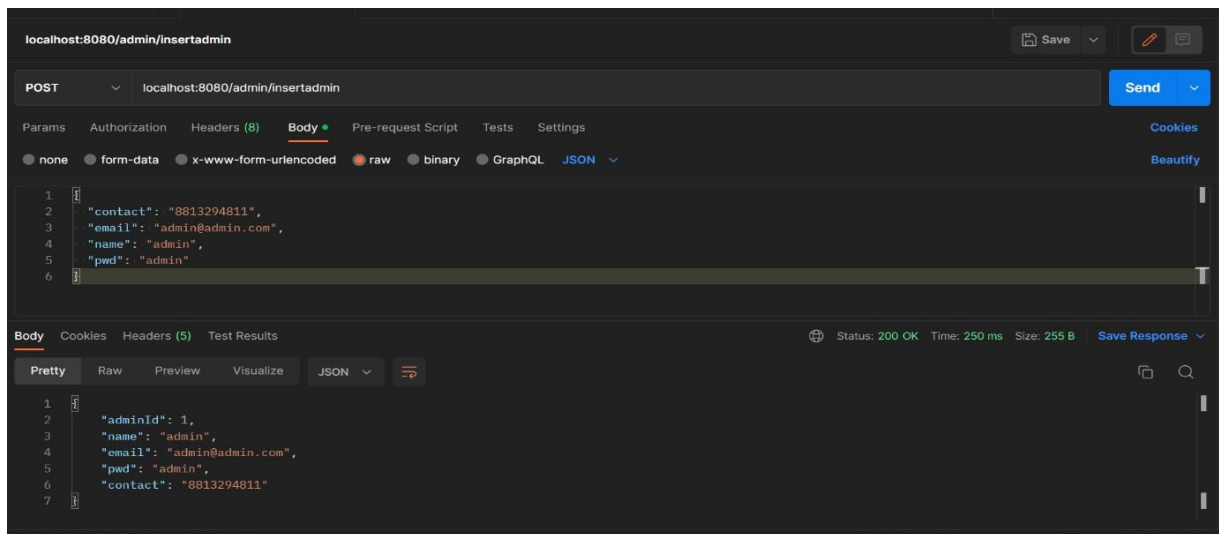
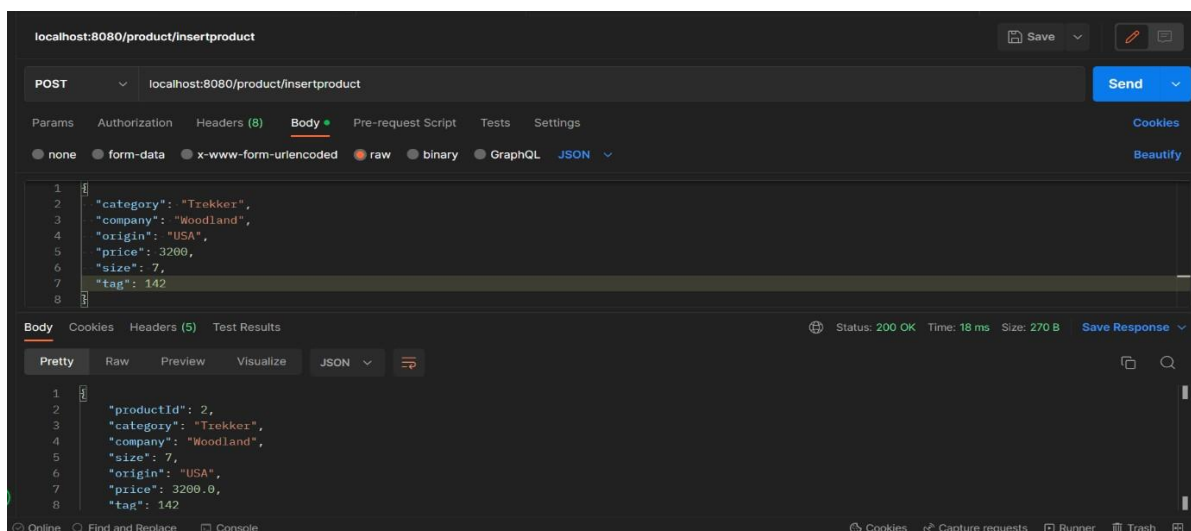
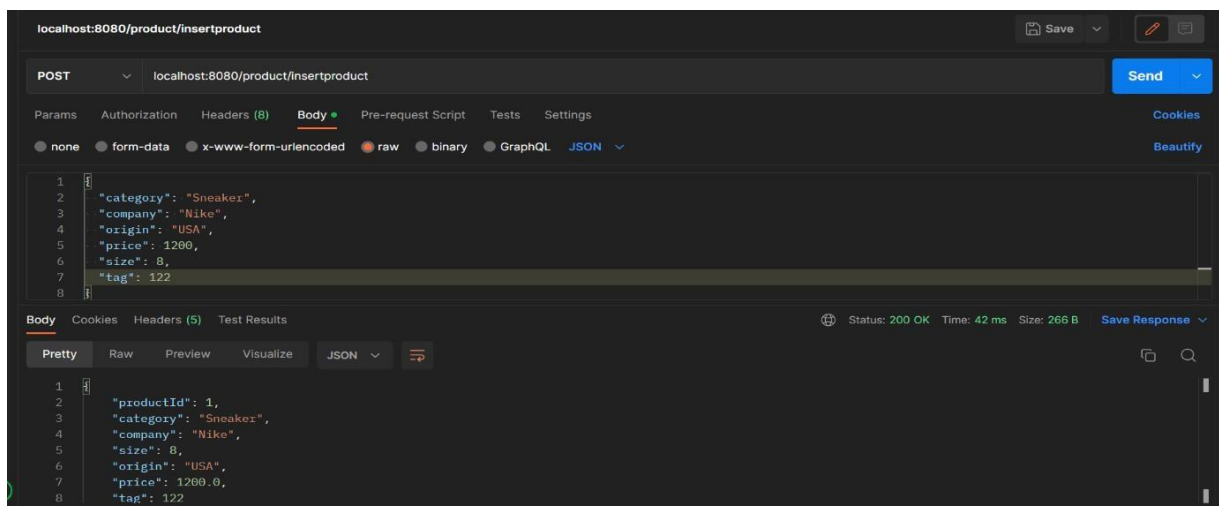


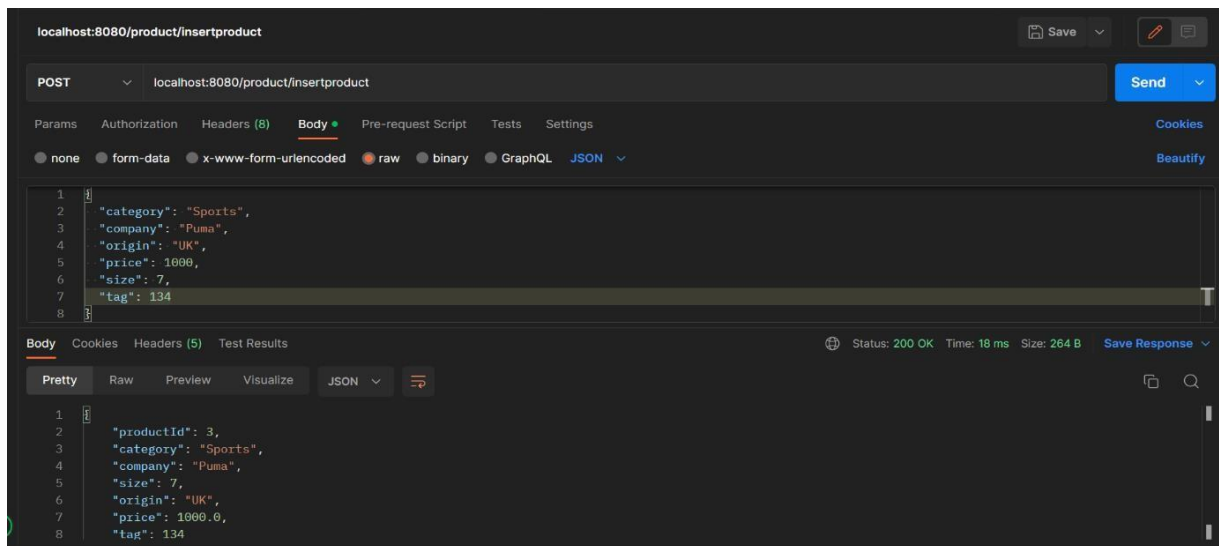
Screenshots

1: Login And Register (Admin Section)

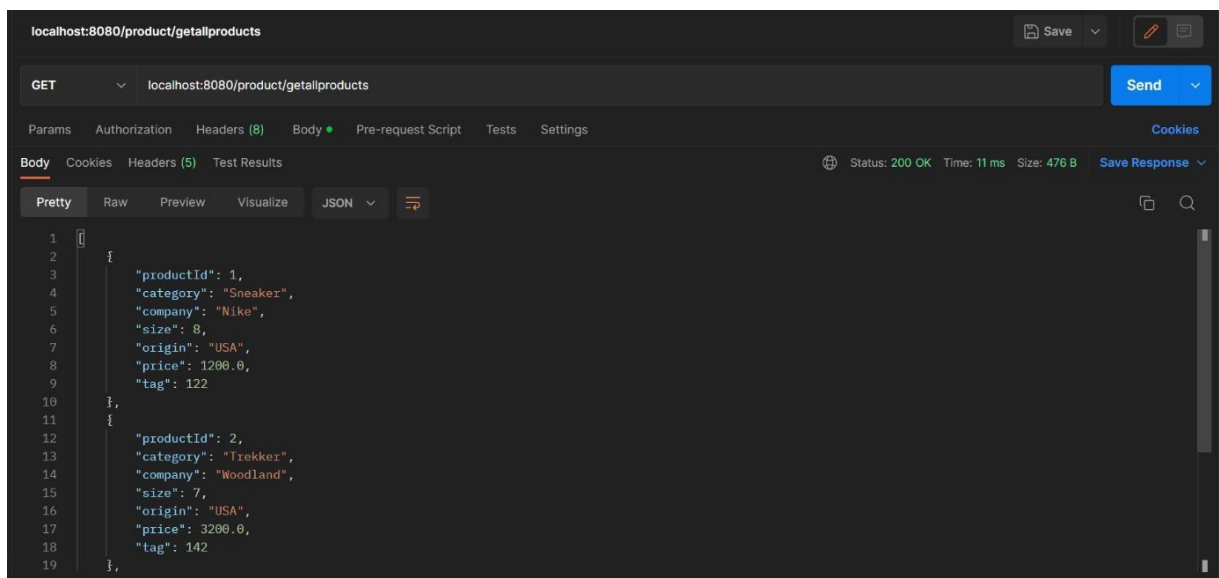


2: Inserting Product (Admin Section)





3: Display All Available Products (Admin Section)



4: Purchasing Product through productid (Customer Section)

POST /customer/insertcustomer/{id} insertCustomer

Parameters Cancel

Name	Description
id * required integer(\$int64) (path)	id <input type="text" value="1"/>
newUser * required object (body)	newUser Edit Value Model <pre> { "contact": 1391391312, "customerName": "John", "date": "20-01-2023", "email": "john@gmail.com", "list": [{ "category": "string", "company": "string", "origin": "string", "price": 0, "productId": 0, "size": 0, "tag": 0 }], "totalPrice": 0 } </pre>

http://localhost:9080/customer/insertcustomer/1

Server response

Code	Details
200	<div><p>Response body</p><pre>{ "customerId": 1, "customerName": "John", "email": "john@gmail.com", "contact": "2391391312", "date": "20-01-2023", "totalPrice": 1200, "list": [{ "productId": 1, "category": "Sneaker", "company": "Nike", "size": 8, "origin": "USA", "price": 1200, "tag": 122 }] }</pre>Download</div> <p>Response headers</p> <pre>connection: keep-alive content-type: application/json date: Sun29 Jan 2023 12:31:08 GMT keep-alive: timeout=60 transfer-encoding: chunked</pre>

Responses

5: Adding more Products through customerId and Product Id (Customer Section)

custid * required
integer(int64)
(path)

prodid * required
integer(int64)
(path)

user * required
object
(body)

custid

prodid

user

Edit Value | Model

```
{
  "contact": 0,
  "customerId": 0,
  "customerName": "string",
  "date": "20-01-2023",
  "email": "string",
  "list": [
    {
      "category": "string",
      "company": "string",
      "origin": "string",
      "price": 0,
      "productId": 0,
      "size": 0,
      "tag": 0
    }
  ],
  "totalPrice": 0
}
```

200

Response body

```
[
  {
    "customerId": 2,
    "customerName": "John",
    "email": "john@gmail.com",
    "contact": "2391391312",
    "date": "20-01-2023",
    "totalPrice": 4400,
    "list": [
      {
        "productId": 3,
        "category": "Trekker",
        "company": "Woodland",
        "size": 7,
        "origin": "USA",
        "price": 3200,
        "tag": 142
      },
      {
        "productId": 2,
        "category": "Sneaker",
        "company": "Nike",
        "size": 8,
        "origin": "USA",
        "price": 1200,
        "tag": 122
      }
    ]
  }
]
```

Response headers

```
connection: keep-alive
content-type: application/json
date: Sun29 Jan 2023 12:32:42 GMT
keep-alive: timeout=60
transfer-encoding: chunked
```

Responses

6: Register And Login (User Section)

POST /user/signup insertUser

Parameters Cancel

Name	Description
newUser * required	newUser
object	
(body)	
	<div>Edit Value Model</div> <div><pre>{ "userContact": 2345910910, "userEmail": "jack@gmail.com", "userName": "Jack", "userPwd": "12345" }</pre></div> <div>Cancel</div>

7: Inserting Product through userId and productId (User Section)

http://localhost:8080/user/insertproduct/user/1/1

Server response

Code	Details
200	<div>Response body</div> <div><pre>{ "customerId": 3, "customerName": "Jack", "email": "jack@gmail.com", "contact": 2345910910, "date": "string", "totalPrice": 1200, "list": [{ "productId": 4, "category": "Sneaker", "company": "Nike", "size": 8, "origin": "USA", "price": 1200, "tag": 122 }] }</pre></div> <div>Download</div> <div>Response headers</div> <div><pre>connection: keep-alive content-type: application/json date: Sun 29 Jan 2023 12:04:33 GMT keep-alive: timeout=60 transfer-encoding: chunked</pre></div>

Responses

8: Adding more products in same userId/customerId (User Section)

custid * required
integer(\$int64)
(path)
3

prodId * required
integer(\$int64)
(path)
2

User * required
object
(body)
User
Edit Value | Model

```
{
  "contact": 0,
  "customerId": 0,
  "customerName": "string",
  "date": "22-01-2023",
  "email": "string",
  "list": [
    {
      "category": "string",
      "company": "string",
      "origin": "string",
      "price": 0,
      "productId": 0,
      "size": 0,
      "tag": 0
    }
  ],
  "totalPrice": 0
}
```

Cancel

9: Other Functions in Admin Section

admin-controller Admin Controller			⌵
GET	/admin/getallcustomerbydate/{date}	getAllCustomersByDateAndCategory	
GET	/admin/getallcustomerbyname/{custname}	findByCustomerName	
GET	/admin/getallproductsbycategory/{category}	getAllProductssByCategory	
GET	/admin/getallusers	getAllUsers1	
GET	/admin/getallvalidcustomers	getAllUsers	
POST	/admin/insertadmin	InsertAdmin	
PUT	/admin/updateadminbyid/{adminid}	updateAdmin	

product-controller Product Controller			⌵
DELETE	/product/deleteproduct/{productid}	deleteProduct	
GET	/product/getallproducts	getAllProducts	
GET	/product/getproduct/{productid}	getProduct	
POST	/product/insertproduct	InsertProduct	
PUT	/product/updateproductbyid/{productid}	updateProduct	

10: Other Functions in Customer Section

customer-controller Customer Controller			⌵
PUT	/customer/addcustomerproducts/{custid}/{prodid}	updateProductInCustomer	
DELETE	/customer/deletecustomer/{userid}	deleteCustomer	
GET	/customer/getallcustomers	getAllCustomers	
GET	/customer/getcustomer/{userid}	getCustomer	
POST	/customer/insertcustomer/{id}	InsertCustomer	
PUT	/customer/updatecustomerbyid/{userid}	updateCustomer	

purchased-product-controller Purchased Product Controller			⌵
DELETE	/purchased/deletepurchasedproduct/{productid}	deletePurchasedProduct	
GET	/purchased/getallpurchasedproducts	getAllPurchasedProducts	
GET	/purchased/getpurchasedproduct/{productid}	getProduct	
POST	/purchased/insertpurchasedproduct	InsertProduct	
PUT	/purchased/updatepurchasedproductbyid/{productid}	updatePurchasedProduct	

11: Other Functions in User Section

user-controller User Controller			⌵
PUT	/user/addmoreuserproducts/{custid}/{prodid}	updateProductInCustomer	
GET	/user/getproducts/{userid}	getproducts	
POST	/user/insertproductinuser/{userid}/{prodid}	InsertProductUserInDB	
POST	/user/signup	insertUser	
PUT	/user/updateuserbyid/{userid}	updateUser	

12: Database (MySQL)

```
MySQL 8.0 Command Line Client
Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> use ciscophase2;
Database changed
mysql> select * from product;
+-----+-----+-----+-----+-----+-----+-----+
| product_id | product_category | product_company | product_origin | product_price | product_size | product_tag |
+-----+-----+-----+-----+-----+-----+-----+
| 1 | sneaker | nike | usa | 1000 | 8 | 123 |
+-----+-----+-----+-----+-----+-----+-----+
1 row in set (0.00 sec)

mysql> select * from customer;
+-----+-----+-----+-----+-----+-----+
| customer_id | customer_contact | customer_name | purchased_date | customer_email | total_price |
+-----+-----+-----+-----+-----+-----+
| 1 | 1234 | amit | 20-02-2020 | t@gmail.com | 1000 |
+-----+-----+-----+-----+-----+-----+
1 row in set (0.00 sec)

mysql> select * from purchased_product;
+-----+-----+-----+-----+-----+-----+-----+-----+
| product_id | product_category | product_company | product_origin | product_price | product_size | product_tag | customer_id |
+-----+-----+-----+-----+-----+-----+-----+-----+
| 1 | sneaker | nike | usa | 1000 | 8 | 123 | 1 |
+-----+-----+-----+-----+-----+-----+-----+-----+
1 row in set (0.00 sec)

mysql>
```

```
Database changed
mysql> select * from product;
+-----+-----+-----+-----+-----+-----+-----+
| product_id | product_category | product_company | product_origin | product_price | product_size | product_tag |
+-----+-----+-----+-----+-----+-----+-----+
| 1 | Sneaker | Nike | USA | 1200 | 8 | 122 |
| 2 | Trekker | Woodland | USA | 3200 | 7 | 142 |
| 3 | Sports | Puma | UK | 1000 | 7 | 134 |
+-----+-----+-----+-----+-----+-----+-----+
3 rows in set (0.01 sec)

mysql>
```

```
mysql> select * from customer;
+-----+-----+-----+-----+-----+-----+
| customer_id | customer_contact | customer_name | purchased_date | customer_email | total_price |
+-----+-----+-----+-----+-----+-----+
| 2 | 2391391312 | John | 20-01-2023 | john@gmail.com | 4400 |
| 4 | 2345910910 | Jack | 22-01-2023 | jack@gmail.com | 4400 |
+-----+-----+-----+-----+-----+-----+
2 rows in set (0.00 sec)

mysql>
```

5 rows in set (0.01 sec)

mysql> select * from customer;

customer_id	customer_contact	customer_name	purchased_date	customer_email	total_price
2	2391391312	John	20-01-2023	john@gmail.com	4400
4	2345910910	Jack	22-01-2023	jack@gmail.com	4400

2 rows in set (0.00 sec)

mysql> select * from user;

user_id	user_contact	user_email	user_name	user_pwd
1	2345910910	jack@gmail.com	Jack	12345

1 row in set (0.00 sec)

mysql>

mysql> select * from user;

user_id	user_contact	user_email	user_name	user_pwd
1	2345910910	jack@gmail.com	Jack	12345

1 row in set (0.00 sec)

mysql> select * from admin;

admin_id	admin_contact	admin_email	admin_name	admin_password
1	8813294811	admin@admin.com	admin	admin

1 row in set (0.00 sec)

mysql>

MySQL 8.0 Command Line Client

2 rows in set (0.00 sec)

mysql> select * from user;

user_id	user_contact	user_email	user_name	user_pwd
1	2345910910	jack@gmail.com	Jack	12345

1 row in set (0.00 sec)

mysql> select * from admin;

admin_id	admin_contact	admin_email	admin_name	admin_password
1	8813294811	admin@admin.com	admin	admin

1 row in set (0.00 sec)

mysql> select * from purchased_product;

product_id	product_category	product_company	product_origin	product_price	product_size	product_tag	customer_id
2	Sneaker	Nike	USA	1200	8	122	2
3	Trekker	Woodland	USA	3200	7	142	2
5	Trekker	Woodland	USA	3200	7	142	4
6	Sneaker	Nike	USA	1200	8	122	4

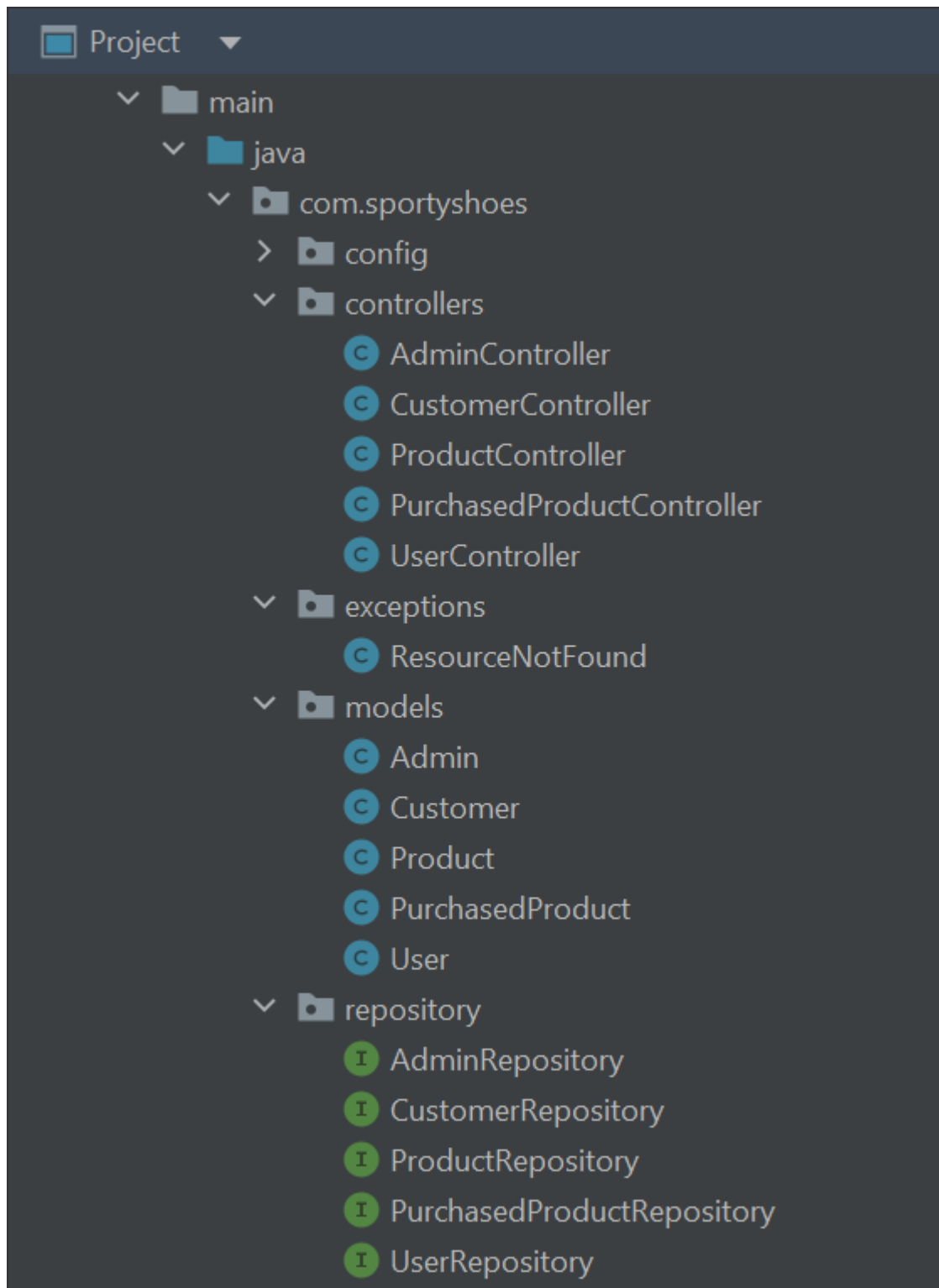
4 rows in set (0.00 sec)

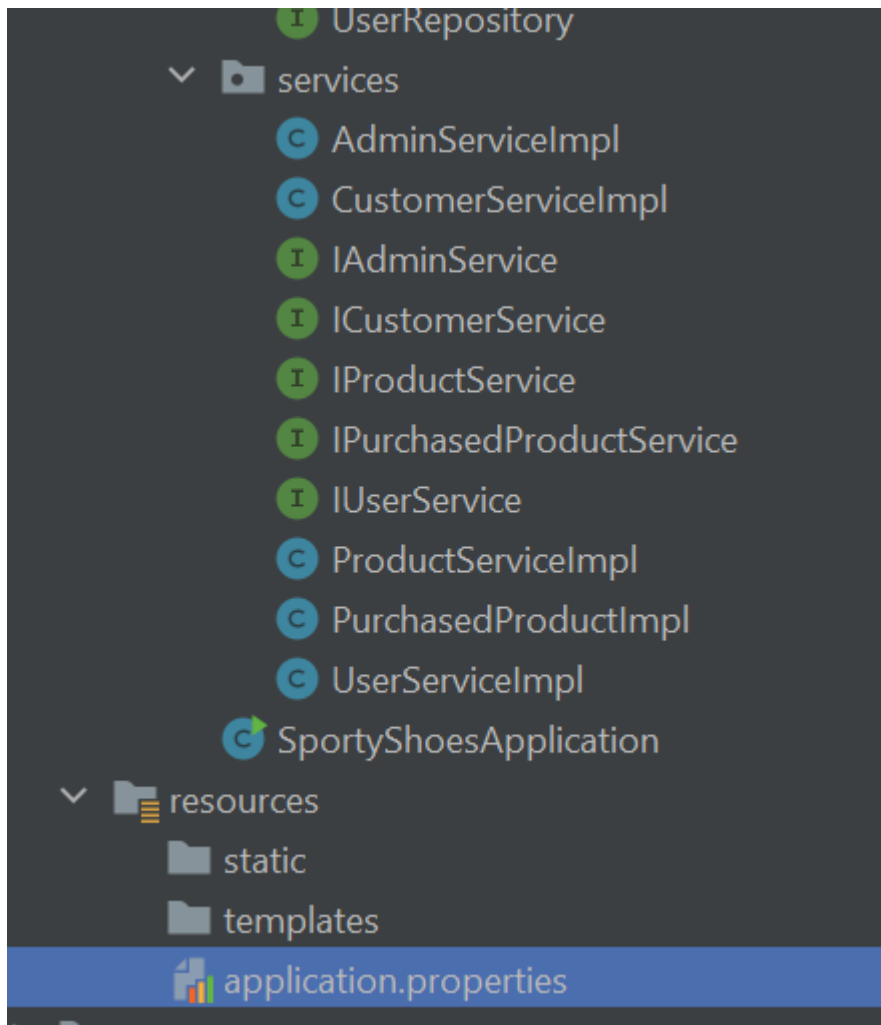
mysql>

Source Code:

Here is some of the source code.

1> Project Structure





2> Controller

a) Admin Controller

```
AdminController.java
29  @Autowired
30  private IPurchasedProductService prodService;
31
32  @Autowired
33  private IUserService userService;
34
35  @PostMapping("/insertadmin")
36  public Admin insertAdmin(@RequestBody Admin admin) { return adminService.insertAdminInDB(admin); }
37
38  @PostMapping("/updateadminbyid/{adminid}")
39  public void updateAdmin(@PathVariable("adminid") Long adminId, @RequestBody Admin admin) throws ResourceNotFoundException {
40      adminService.updateAdminInDB(admin, adminId);
41  }
42
43  @GetMapping("/getallvalidcustomers")
44  public List<Customer> getAllUsers() { return custService.getAllCustomers(); }
45
46  @GetMapping("/getallcustomerbyname/{custname}")
47  public List<Customer> findByCustomerName(@PathVariable("custname") String customerName){
48      return custService.findByCustomerName(customerName);
49  }
50
51  }
```

b) Customer Controller

```

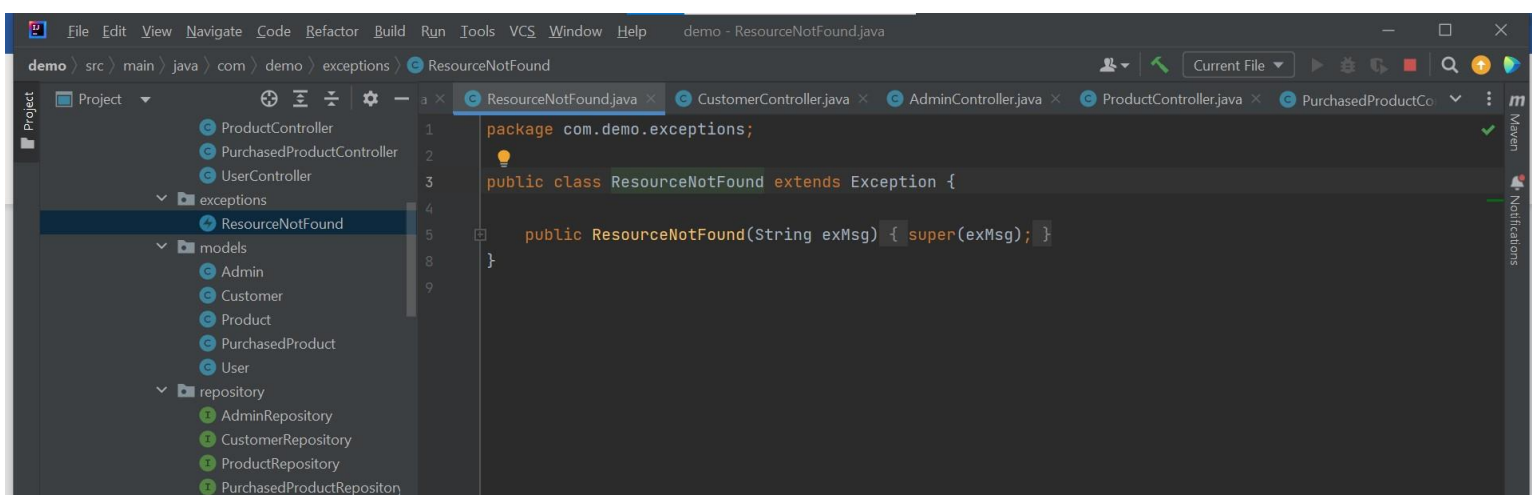
35
36 no usages
37 @GetMapping("/getcustomer/{userid}")
38 public Customer getCustomer(@PathVariable("userid") Long customerId) throws ResourceNotFound {
39     return customerService.getCustomerInDB(customerId);
40 }
41
42 no usages
43 @DeleteMapping("/deletecustomer/{userid}")
44 public void deleteCustomer(@PathVariable("userid") Long customerId) {
45     customerService.deleteCustomerInDB(customerId);
46 }
47
48 no usages
49 @PutMapping("/updatecustomerbyid/{userid}")
50 public void updateCustomer(@PathVariable("userid") Long userId, @RequestBody Customer user) throws ResourceNotFound {
51     customerService.updateCustomerInDB(user, userId);
52 }
53
54 no usages
55 @PutMapping("/addcustomerproducts/{custid}/{prodid}")
56 public void updateProductInCustomer(@PathVariable("custid") Long userId, @PathVariable("prodid") Long prodid, @RequestBody Product product) throws ResourceNotFound {
57     customerService.insertProductInExistingCustomerInDB(user, userId, prodid, product);
58 }
59

```

c) Product Controller

```
AdminController.java × CustomerController.java × ProductController.java ×
19 @RestController
20 @RequestMapping("/product")
21 public class ProductController {
22
23     5 usages
24     @Autowired
25     private IProductService productService;
26
27     no usages
28     @PostMapping("/insertproduct")
29     public Product insertProduct(@RequestBody Product newProduct) {
30         return productService.insertProductInDB(newProduct);
31     }
32
33     no usages
34     @GetMapping("/getallproducts")
35     public List<Product> getAllProducts() { return productService.getAllProducts(); }
36
37     no usages
38     @GetMapping("/getproduct/{productId}")
39     public Product getProduct(@PathVariable("productId") Long productId) throws ResourceNotFoundException {
40         return productService.getProductInDB(productId);
41     }
42 }
```

3> Resource Not Found Exception



4> Models

a) Admin

```
AdminController.java x CustomerController.java x ProductController.java x ResourceNotFound.java x Admin.java x
11 @Table(name = "admin")
12 public class Admin {
13
14     2 usages
15     @Id
16     @GeneratedValue(strategy = GenerationType.IDENTITY)
17     @Column(name = "admin_id")
18     private Long adminId;
19
20     3 usages
21     @Column(name = "admin_name")
22     private String name;
23
24     3 usages
25     @Column(name = "admin_email")
26     private String email;
27
28     3 usages
29     @Column(name = "admin_password")
30     private String pwd;
31
32     3 usages
33     @Column(name = "admin_contact")
34     private String contact;
35
36     no usages
37     public Admin() {
38     }
39 }
```

b) Customer

```
AdminController.java x CustomerController.java x ProductController.java x ResourceNotFound.java x Admin.java x Customer.java x
25 private Float totalPrice;
26
27     2 usages
28     @OneToMany(cascade = CascadeType.ALL)
29     @JoinColumn(name = "customer_id")
30     private Set<PurchasedProduct> list = new HashSet<>();
31
32     public Customer() { totalPrice = (float) 0; }
33
34     public Customer(String customerName, String email, Long contact, String date, Float totalPrice) {
35         this.customerName = customerName;
36         this.email = email;
37         this.contact = contact;
38         this.date = date;
39         this.totalPrice = totalPrice;
40     }
41
42     no usages
43     public Long getCustomerId() { return customerId; }
44
45     no usages
46     public void setCustomerId(Long customerId) { this.customerId = customerId; }
47
48     1 usage
49     public String getCustomerName() { return customerName; }
50
51 }
```

c) Product

```
AdminController.java x CustomerController.java x ProductController.java x ResourceNotFound.java x Admin.java x Customer.java x Product.java x
62
63     3 usages
64     public int getSize() { return size; }
65
66     no usages
67     public void setSize(int size) { this.size = size; }
68
69     4 usages
70     public String getOrigin() { return origin; }
71
72     1 usage
73     public void setOrigin(String origin) { this.origin = origin; }
74
75     7 usages
76     public Float getPrice() { return price; }
77
78     2 usages
79     public void setPrice(Float price) { this.price = price; }
80
81     4 usages
82     public int getTag() {
83         return tag;
84     }
85
86     1 usage
87     public void setTag(int tag) { this.tag = tag; }
88
89 }
```

d) PurchasedProduct

```
11  @Table(name = "purchased_product")
12  public class PurchasedProduct {
13
14      no usages
15      public int getSize() { return size; }
16
17      3 usages
18      public void setSize(int size) { this.size = size; }
19
20
21      1 usage
22      public String getOrigin() { return origin; }
23
24      4 usages
25      public void setOrigin(String origin) { this.origin = origin; }
26
27
28      3 usages
29      public Float getPrice() { return price; }
30
31      5 usages
32      public void setPrice(Float price) { this.price = price; }
33
34
35      1 usage
36      public int getTag() {
37          return tag;
38      }
39
40      4 usages
41      public void setTag(int tag) { this.tag = tag; }
42  }
```

e) User

```
52  public void setUserEmail(String userEmail) {
53      this.userEmail = userEmail;
54  }
55
56      1 usage
57      public String getUserPwd() {
58          return userPwd;
59      }
60
61      1 usage
62      public void setUserPwd(String userPwd) {
63          this.userPwd = userPwd;
64      }
65
66      2 usages
67      public Long getUserContact() {
68          return userContact;
69      }
70
71      1 usage
72      public void setUserContact(Long userContact) {
73          this.userContact = userContact;
74      }
75  }
```

5> Repository

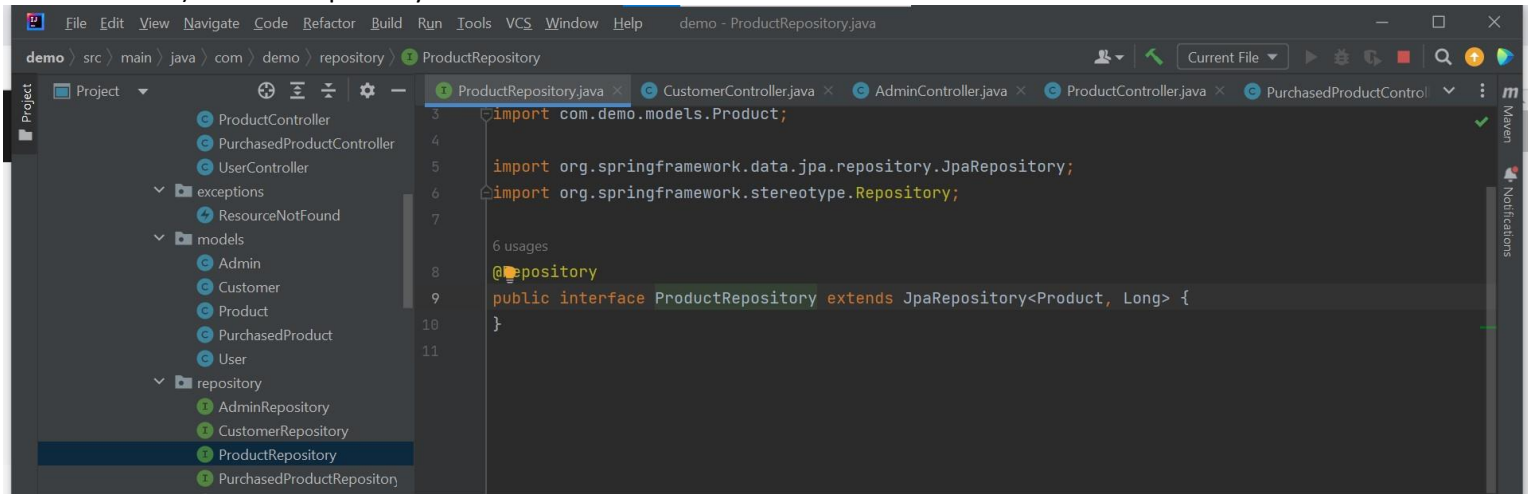
a) Admin Repository

```
demo - AdminRepository.java
demo > src > main > java > com > demo > repository > AdminRepository

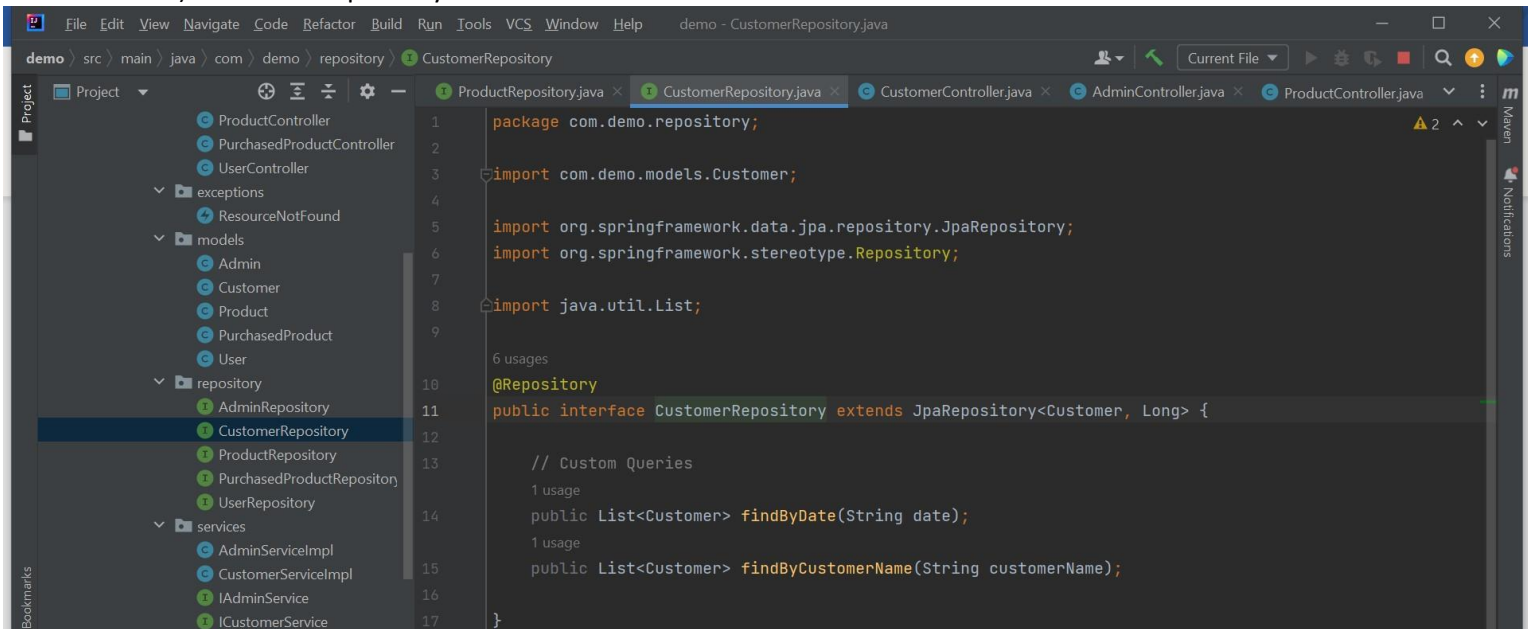
Project
├── Project
│   ├── ProductController
│   ├── PurchasedProductController
│   ├── UserController
│   ├── exceptions
│   │   ├── ResourceNotFound
│   ├── models
│   │   ├── Admin
│   │   ├── Customer
│   │   ├── Product
│   │   ├── PurchasedProduct
│   │   ├── User
│   └── repository
│       ├── AdminRepository
│       ├── CustomerRepository
│       ├── ProductRepository
│       └── PurchasedProductRepository
└── repository
    ├── AdminRepository
    ├── CustomerRepository
    ├── ProductRepository
    └── PurchasedProductRepository

1  package com.demo.repository;
2
3  import com.demo.models.Admin;
4
5  import org.springframework.data.jpa.repository.JpaRepository;
6  import org.springframework.stereotype.Repository;
7
8  2 usages
9  @Repository
10 public interface AdminRepository extends JpaRepository<Admin, Long> {
11 }
```

b) Product Repository



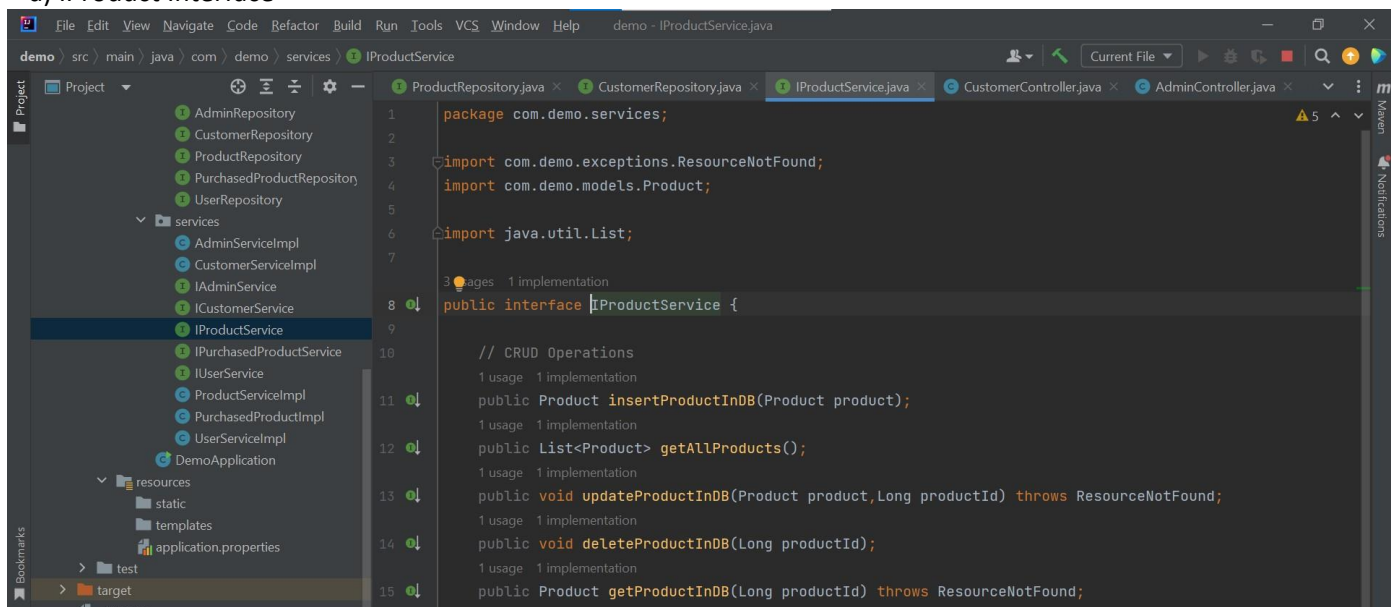
c) Customer Repository



6> Services

=> Interfaces

a) IProduct Interface



=> Classes

a) ProductServiceImpl

```
ProductRepository.java x ProductRepository.java x CustomerRepository.java x IAdminService.java x IProductService.java x ProductServiceImpl.java x
15 private ProductRepository productRepo;
16
17
18 1 usage
19 @Override
20 public Product insertProductInDB(Product product) { return productRepo.save(product); }
21
22
23 1 usage
24 @Override
25 public List<Product> getAllProducts() { return productRepo.findAll(); }
26
27
28 1 usage
29 @Override
30 public void updateProductInDB(Product product, Long productId) throws ResourceNotFound {
31     Product existingProduct = productRepo.findById(productId).get();
32     if(product != null) {
33         // Update existing Product Details with new Details.
34         existingProduct.setCategory(product.getCategory());
35         existingProduct.setCompany(product.getCompany());
36         existingProduct.setOrigin(product.getOrigin());
37         existingProduct.setPrice(product.getPrice());
38         existingProduct.setTag(product.getTag());
39         productRepo.save(existingProduct);
40     } else {
41         throw new ResourceNotFound("Product not found");
42     }
43 }
```

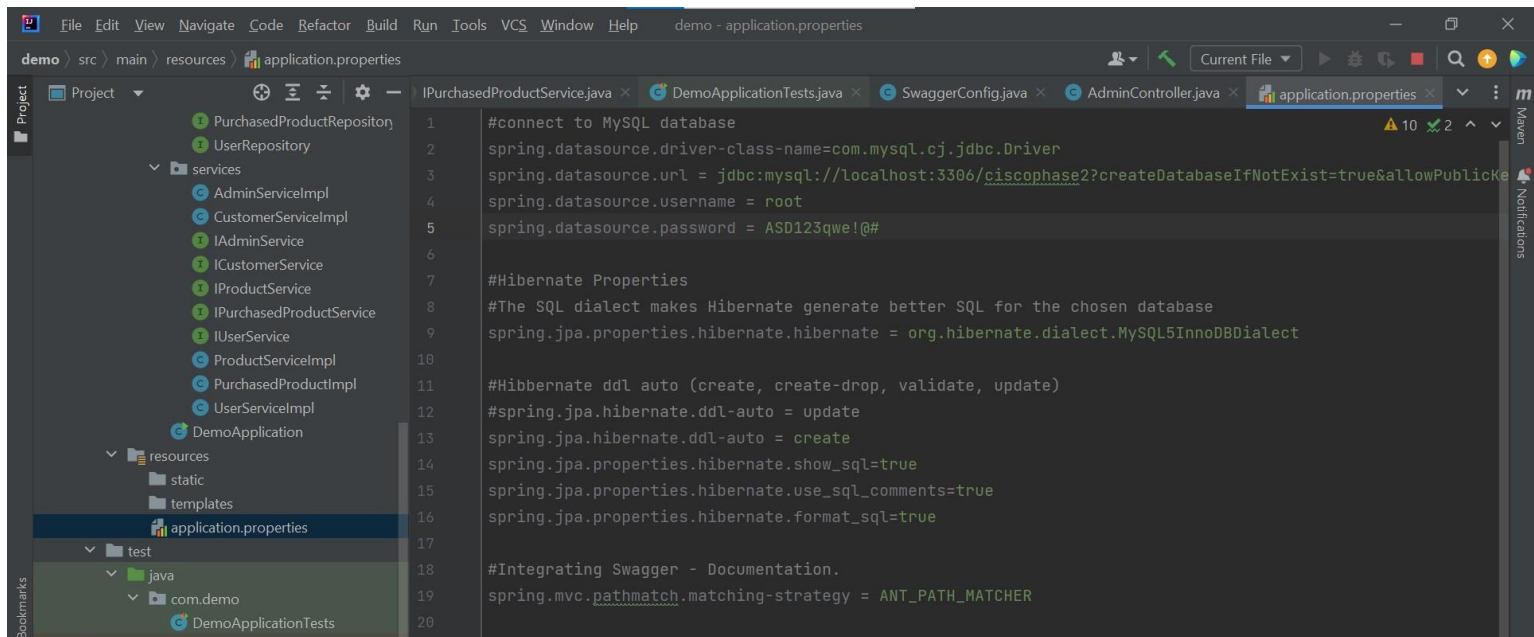
b) AdminServiceImpl

```
ProductRepository.java x CustomerRepository.java x IAdminService.java x IProductService.java x ProductServiceImpl.java x AdminServiceImpl.java x
11 @Service
12 public class AdminServiceImpl implements IAdminService {
13
14     3 usages
15     @Autowired
16     private AdminRepository adminRepo;
17
18     no usages
19     @Autowired
20     private CustomerRepository custRepo;
21
22
23     1 usage
24     @Override
25     public Admin insertAdminInDB(Admin admin) { return adminRepo.save(admin); }
26
27
28     1 usage
29     @Override
30     public void updateAdminInDB(Admin admin, Long adminId) throws ResourceNotFound {
31         Admin existingAdmin = adminRepo.findById(adminId).get();
32         if(admin != null) {
33             // Update existing Admin with Password.
34             existingAdmin.setPwd(admin.getPwd());
35             adminRepo.save(existingAdmin);
36         } else {
37             throw new ResourceNotFound("Customer not found");
38         }
39     }
40 }
```

c) UserServiceImpl

```
CustomerRepository.java x IAdminService.java x IProductService.java x ProductServiceImpl.java x AdminServiceImpl.java x UserServiceImpl.java x
12 import org.springframework.beans.factory.annotation.Autowired;
13 import org.springframework.stereotype.Service;
14
15 import java.util.List;
16 import java.util.Set;
17
18 no usages
19 @Service
20 public class UserServiceImpl implements IUserService {
21
22     5 usages
23     @Autowired
24     UserRepository userRepo;
25
26     2 usages
27     @Autowired
28     ProductRepository productRepo;
29
30     1 usage
31     @Autowired
32     CustomerRepository customerRepo;
33
34     3 usages
35     Float total = (float) 0;
36
37     1 usage
38     @Override
39 }
```

7> application.properties



The screenshot shows an IDE window titled "demo - application.properties". The left sidebar displays a project tree with the following structure:

- demo
 - src
 - main
 - resources
 - application.properties (selected)
 - test
 - java
 - com.demo
 - DemoApplicationTests

The main editor area shows the content of `application.properties` with line numbers 1 through 20:

```
1 #connect to MySQL database
2 spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver
3 spring.datasource.url = jdbc:mysql://localhost:3306/ciscophase2?createDatabaseIfNotExist=true&allowPublicKe
4 spring.datasource.username = root
5 spring.datasource.password = ASD123qwe!@#
6
7 #Hibernate Properties
8 #The SQL dialect makes Hibernate generate better SQL for the chosen database
9 spring.jpa.properties.hibernate.hibernate = org.hibernate.dialect.MySQL5InnoDBDialect
10
11 #Hibernate ddl auto (create, create-drop, validate, update)
12 #spring.jpa.hibernate.ddl-auto = update
13 spring.jpa.hibernate.ddl-auto = create
14 spring.jpa.properties.hibernate.show_sql=true
15 spring.jpa.properties.hibernate.use_sql_comments=true
16 spring.jpa.properties.hibernate.format_sql=true
17
18 #Integrating Swagger - Documentation.
19 spring.mvc.pathmatch.matching-strategy = ANT_PATH_MATCHER
20
```

The right sidebar shows the Maven and Notifications panels.

