PROMINEO TECH

Web API Design with SpringBoot Week 2 Coding Assignment

Points possible: 70

Category	Criteria	% of Grade
Functionality	Does the code work?	25
Organization	Is the code clean and organized? Proper use of white space, syntax, and consistency are utilized. Names and comments are concise and clear.	25
Creativity	Student solved the problems presented in the assignment using creativity and out of the box thinking.	25
Completeness	All requirements of the assignment are complete.	25

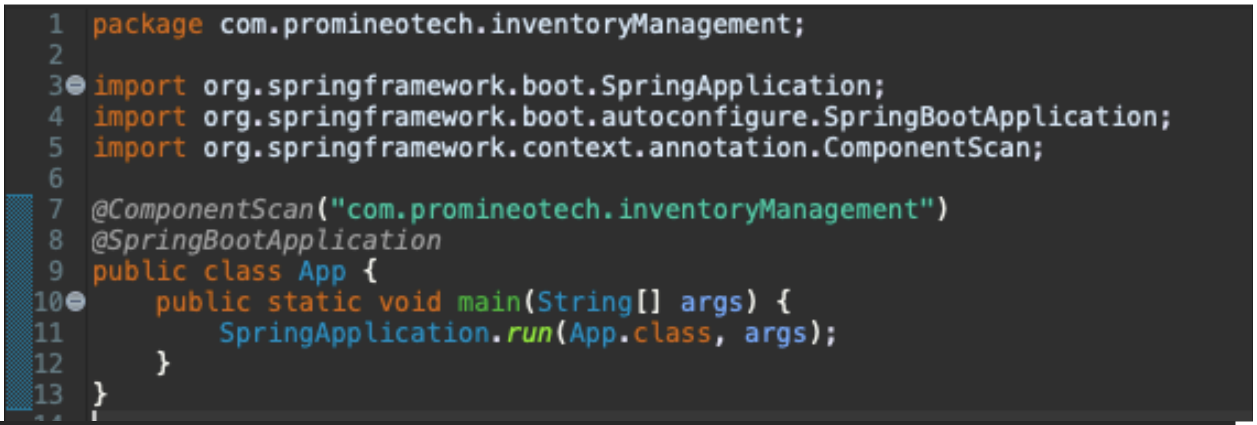
Instructions: In Eclipse, or an IDE of your choice, write the code that accomplishes the objectives listed below. Ensure that the code compiles and runs as directed. Take screenshots of the code and of the running program (make sure to get screenshots of all required functionality) and paste them in this document where instructed below. Create a new repository on GitHub for this week's assignments and push this document, with your Java project code, to the repository. Add the URL for this week's repository to this document where instructed and submit this document to your instructor when complete.

Coding Steps:

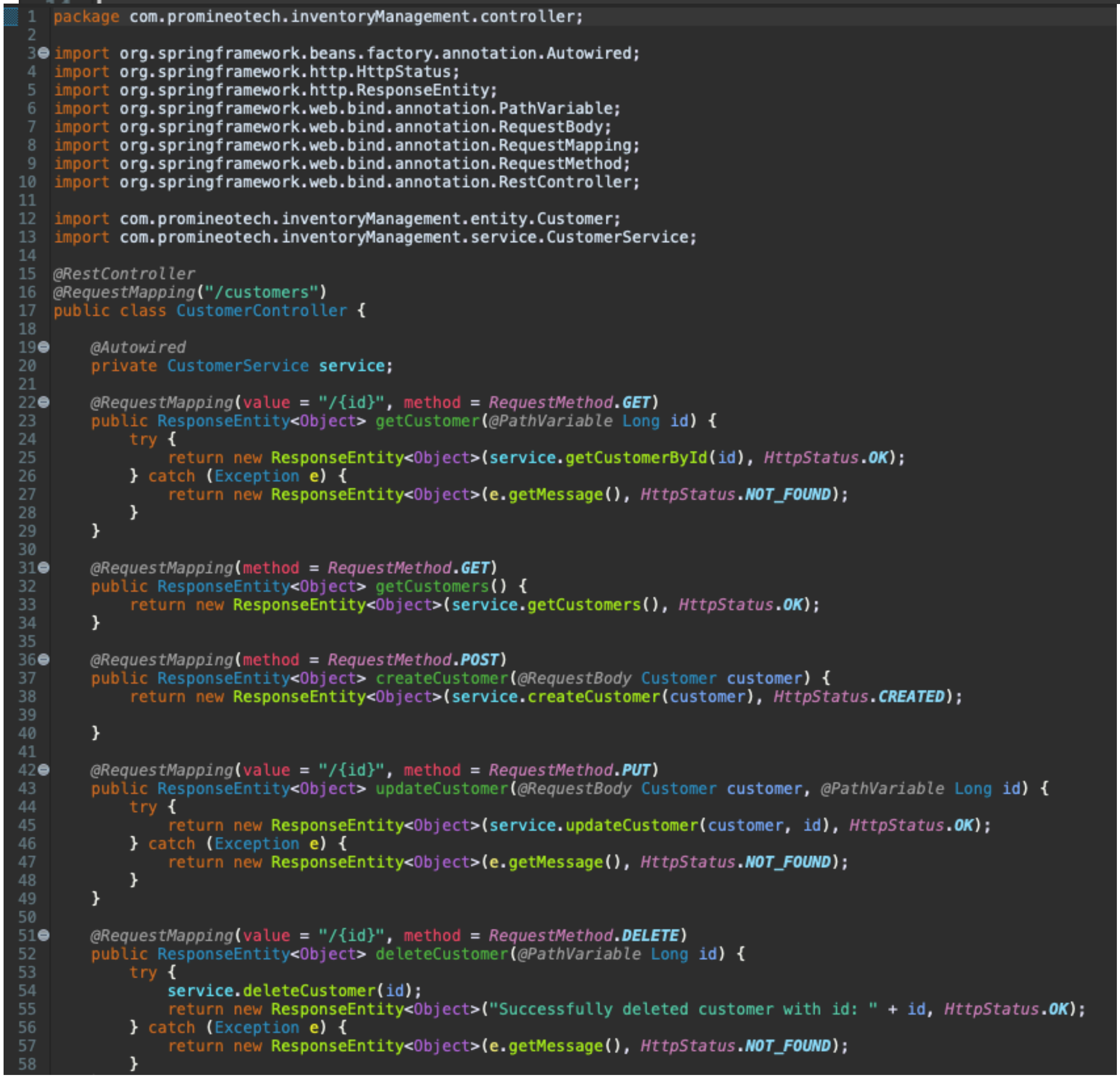
Follow the tutorial in the homework section to build an API. Paste screenshots of your code and your postman requests and responses to show the API works. Push your project to GitHub and paste the link below.

Screenshots of Code:

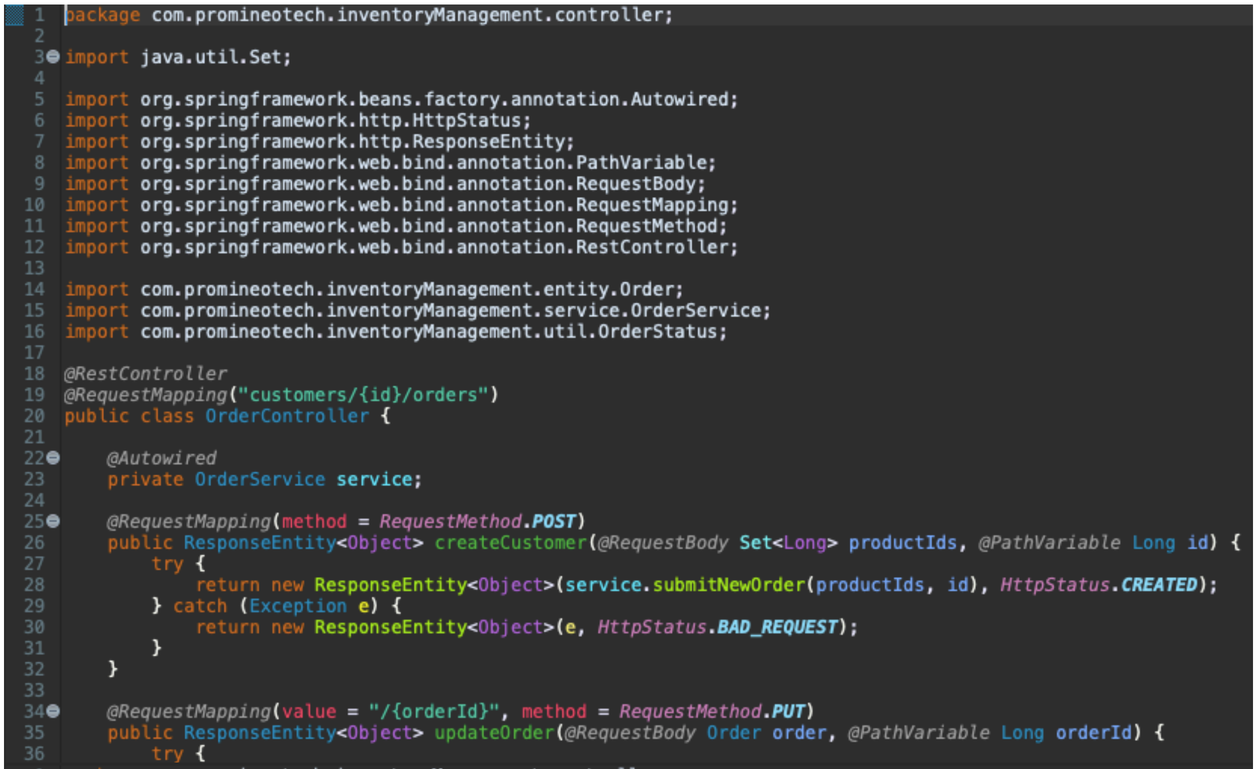
PROMINEO TECH



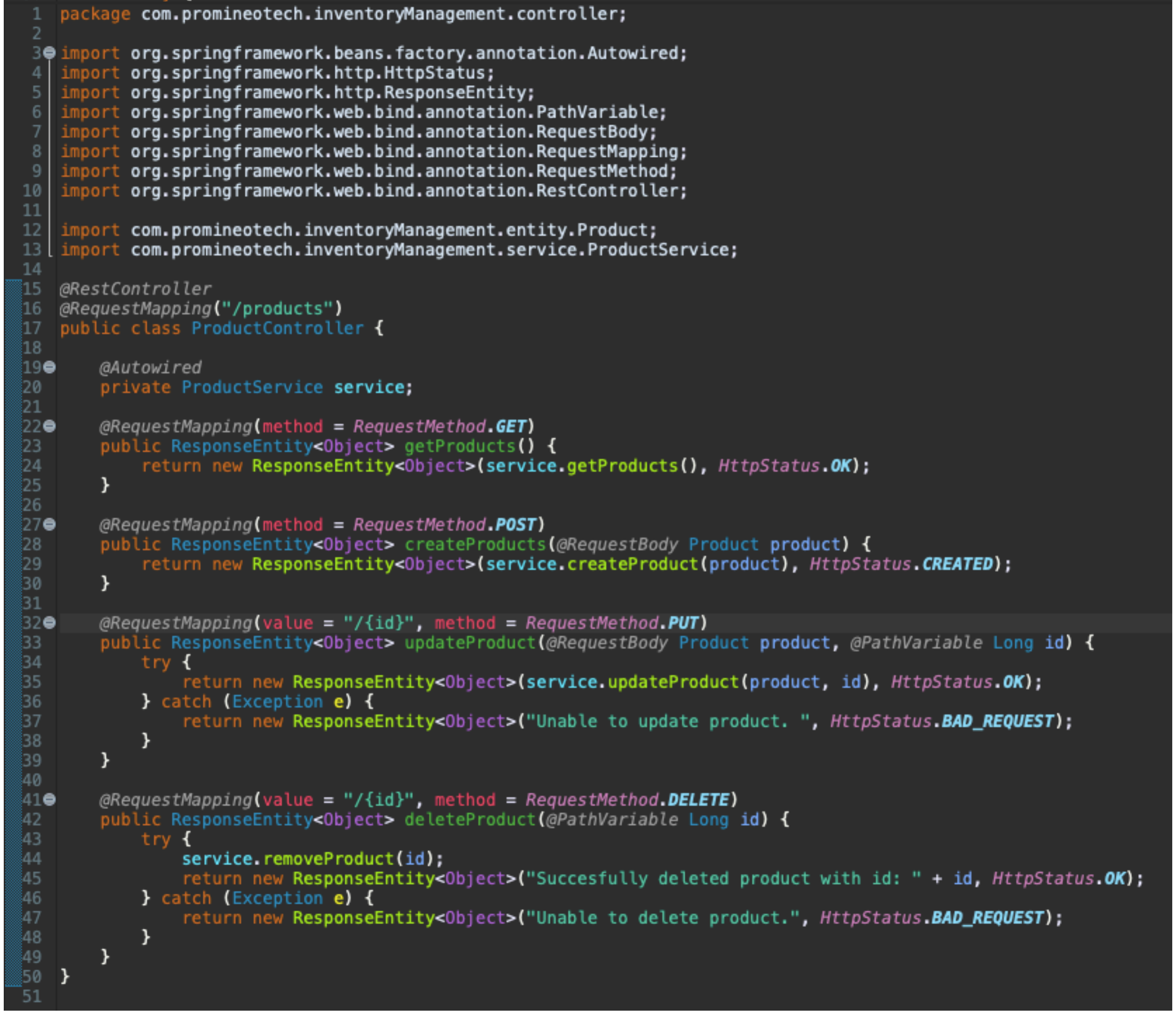
```java
package com.promineotech.inventoryManagement;

import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;
import org.springframework.context.annotation.ComponentScan;

@ComponentScan("com.promineotech.inventoryManagement")
@SpringBootApplication
public class App {
    public static void main(String[] args) {
        SpringApplication.run(App.class, args);
    }
}
```



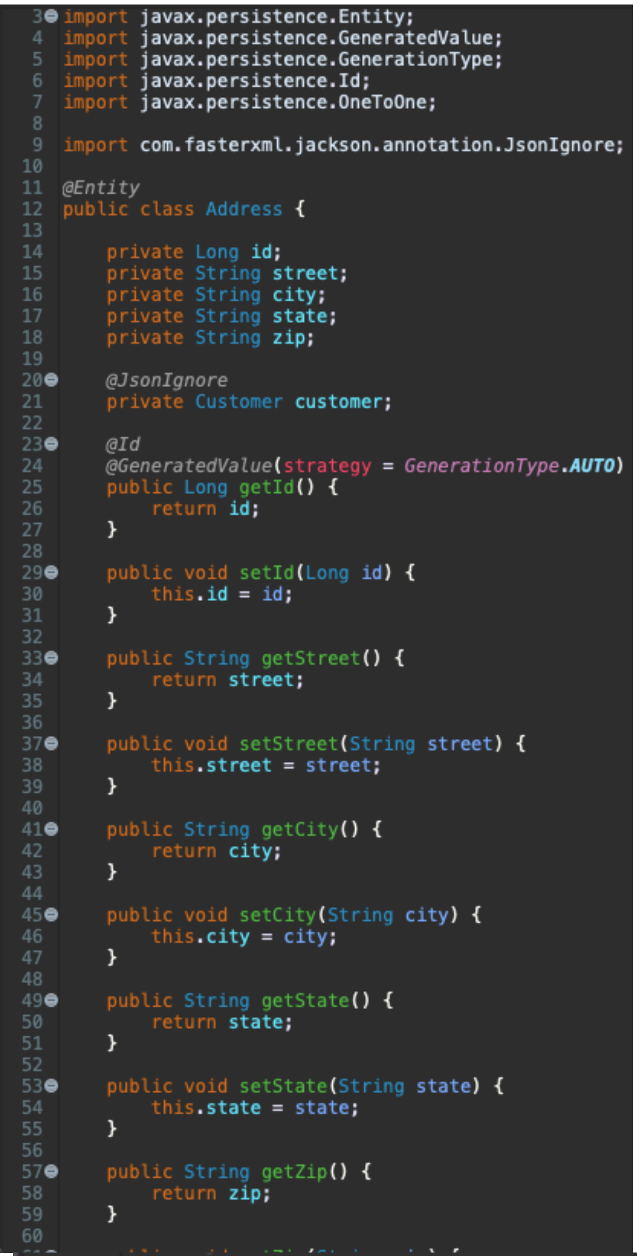
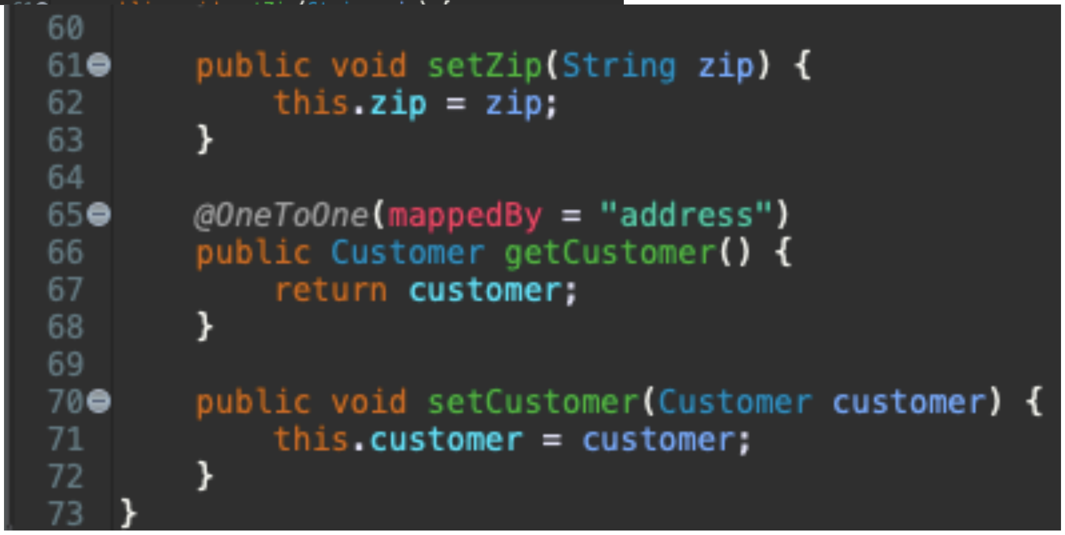
```java
package com.promineotech.inventoryManagement.controller;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.http.HttpStatus;
import org.springframework.http.ResponseEntity;
import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.RequestBody;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestMethod;
import org.springframework.web.bind.annotation.RestController;

import com.promineotech.inventoryManagement.entity.Customer;
import com.promineotech.inventoryManagement.service.CustomerService;

@RestController
@RequestMapping("/customers")
public class CustomerController {

    @Autowired
    private CustomerService service;

    @RequestMapping(value = "/{id}", method = RequestMethod.GET)
    public ResponseEntity<Object> getCustomer(@PathVariable Long id) {
        try {
            return new ResponseEntity<Object>(service.getCustomerById(id), HttpStatus.OK);
        } catch (Exception e) {
            return new ResponseEntity<Object>(e.getMessage(), HttpStatus.NOT_FOUND);
        }
    }

    @RequestMapping(method = RequestMethod.GET)
    public ResponseEntity<Object> getCustomers() {
        return new ResponseEntity<Object>(service.getCustomers(), HttpStatus.OK);
    }

    @RequestMapping(method = RequestMethod.POST)
    public ResponseEntity<Object> createCustomer(@RequestBody Customer customer) {
        return new ResponseEntity<Object>(service.createCustomer(customer), HttpStatus.CREATED);

    }

    @RequestMapping(value = "/{id}", method = RequestMethod.PUT)
    public ResponseEntity<Object> updateCustomer(@RequestBody Customer customer, @PathVariable Long id) {
        try {
            return new ResponseEntity<Object>(service.updateCustomer(customer, id), HttpStatus.OK);
        } catch (Exception e) {
            return new ResponseEntity<Object>(e.getMessage(), HttpStatus.NOT_FOUND);
        }
    }

    @RequestMapping(value = "/{id}", method = RequestMethod.DELETE)
    public ResponseEntity<Object> deleteCustomer(@PathVariable Long id) {
        try {
            service.deleteCustomer(id);
            return new ResponseEntity<Object>("Successfully deleted customer with id: " + id, HttpStatus.OK);
        } catch (Exception e) {
            return new ResponseEntity<Object>(e.getMessage(), HttpStatus.NOT_FOUND);
        }
```

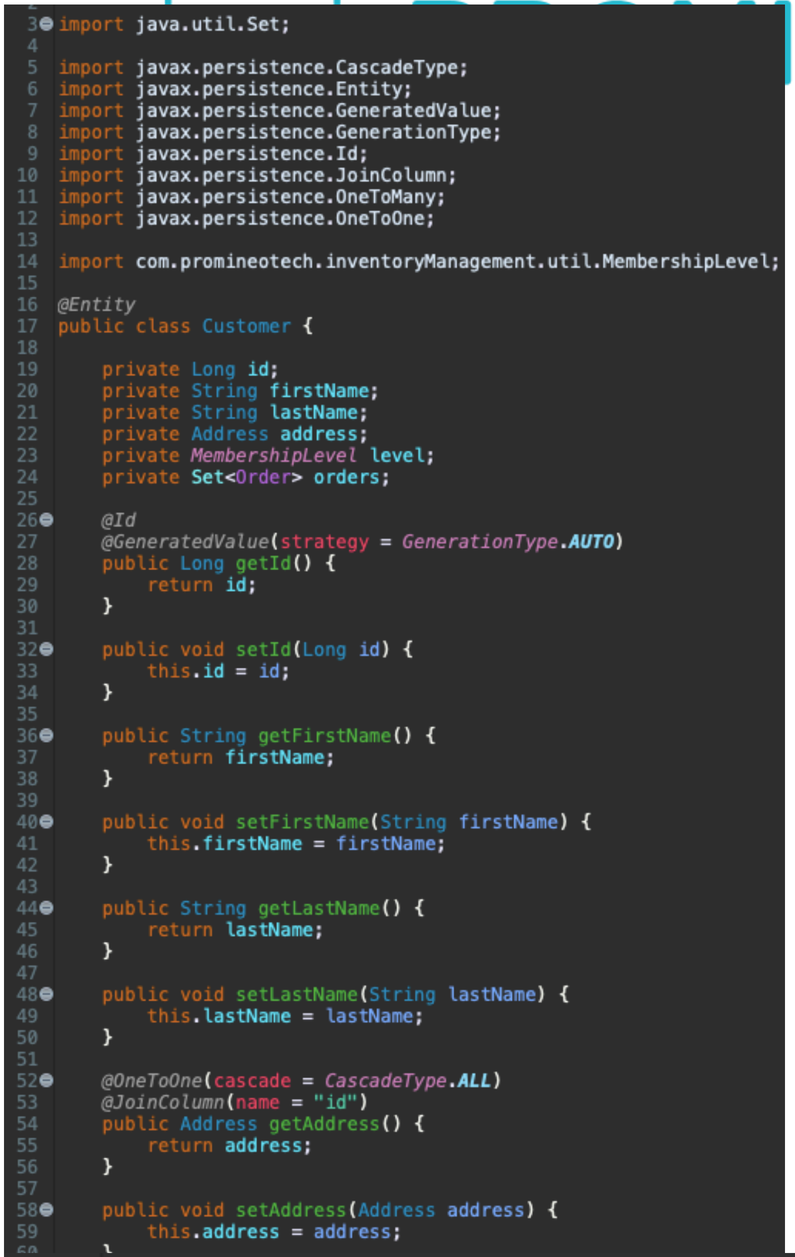


```java
package com.promineotech.inventoryManagement.controller;

import java.util.Set;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.http.HttpStatus;
import org.springframework.http.ResponseEntity;
import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.RequestBody;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestMethod;
import org.springframework.web.bind.annotation.RestController;

import com.promineotech.inventoryManagement.entity.Order;
import com.promineotech.inventoryManagement.service.OrderService;
import com.promineotech.inventoryManagement.util.OrderStatus;

@RestController
@RequestMapping("customers/{id}/orders")
public class OrderController {

    @Autowired
    private OrderService service;

    @RequestMapping(method = RequestMethod.POST)
    public ResponseEntity<Object> createCustomer(@RequestBody Set<Long> productIds, @PathVariable Long id) {
        try {
            return new ResponseEntity<Object>(service.submitNewOrder(productIds, id), HttpStatus.CREATED);
        } catch (Exception e) {
            return new ResponseEntity<Object>(e, HttpStatus.BAD_REQUEST);
        }
    }

    @RequestMapping(value = "/{orderId}", method = RequestMethod.PUT)
    public ResponseEntity<Object> updateOrder(@RequestBody Order order, @PathVariable Long orderId) {
        try {
```

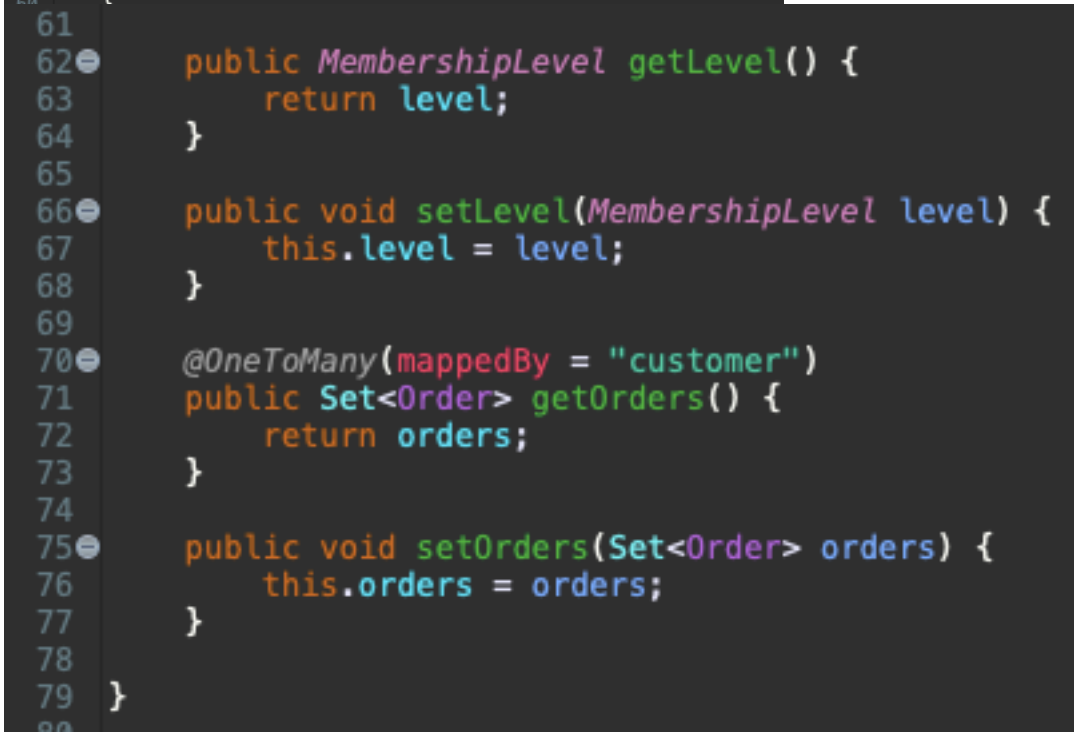


```java
package com.promineotech.inventoryManagement.controller;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.http.HttpStatus;
import org.springframework.http.ResponseEntity;
import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.RequestBody;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestMethod;
import org.springframework.web.bind.annotation.RestController;

import com.promineotech.inventoryManagement.entity.Product;
import com.promineotech.inventoryManagement.service.ProductService;

@RestController
@RequestMapping("/products")
public class ProductController {

    @Autowired
    private ProductService service;

    @RequestMapping(method = RequestMethod.GET)
    public ResponseEntity<Object> getProducts() {
        return new ResponseEntity<Object>(service.getProducts(), HttpStatus.OK);
    }

    @RequestMapping(method = RequestMethod.POST)
    public ResponseEntity<Object> createProducts(@RequestBody Product product) {
        return new ResponseEntity<Object>(service.createProduct(product), HttpStatus.CREATED);
    }

    @RequestMapping(value = "/{id}", method = RequestMethod.PUT)
    public ResponseEntity<Object> updateProduct(@RequestBody Product product, @PathVariable Long id) {
        try {
            return new ResponseEntity<Object>(service.updateProduct(product, id), HttpStatus.OK);
        } catch (Exception e) {
            return new ResponseEntity<Object>("Unable to update product. ", HttpStatus.BAD_REQUEST);
        }
    }

    @RequestMapping(value = "/{id}", method = RequestMethod.DELETE)
    public ResponseEntity<Object> deleteProduct(@PathVariable Long id) {
        try {
            service.removeProduct(id);
            return new ResponseEntity<Object>("Succesfully deleted product with id: " + id, HttpStatus.OK);
        } catch (Exception e) {
            return new ResponseEntity<Object>("Unable to delete product.", HttpStatus.BAD_REQUEST);
        }
    }
}
```

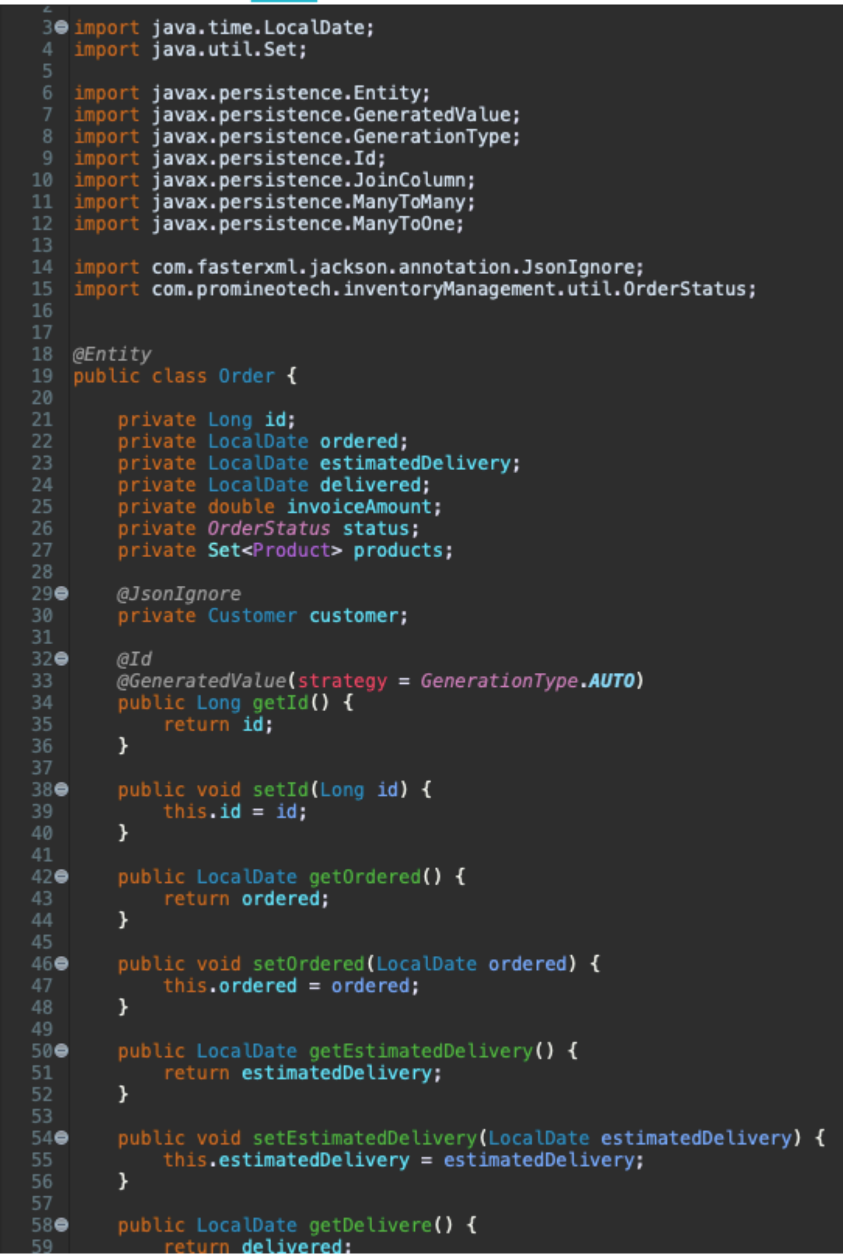

```java
import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.OneToOne;

import com.fasterxml.jackson.annotation.JsonIgnore;

@Entity
public class Address {

    private Long id;
    private String street;
    private String city;
    private String state;
    private String zip;

    @JsonIgnore
    private Customer customer;

    @Id
    @GeneratedValue(strategy = GenerationType.AUTO)
    public Long getId() {
        return id;
    }

    public void setId(Long id) {
        this.id = id;
    }

    public String getStreet() {
        return street;
    }

    public void setStreet(String street) {
        this.street = street;
    }

    public String getCity() {
        return city;
    }

    public void setCity(String city) {
        this.city = city;
    }

    public String getState() {
        return state;
    }

    public void setState(String state) {
        this.state = state;
    }

    public String getZip() {
        return zip;
    }

```

```java

    public void setZip(String zip) {
        this.zip = zip;
    }

    @OneToOne(mappedBy = "address")
    public Customer getCustomer() {
        return customer;
    }

    public void setCustomer(Customer customer) {
        this.customer = customer;
    }
}
```



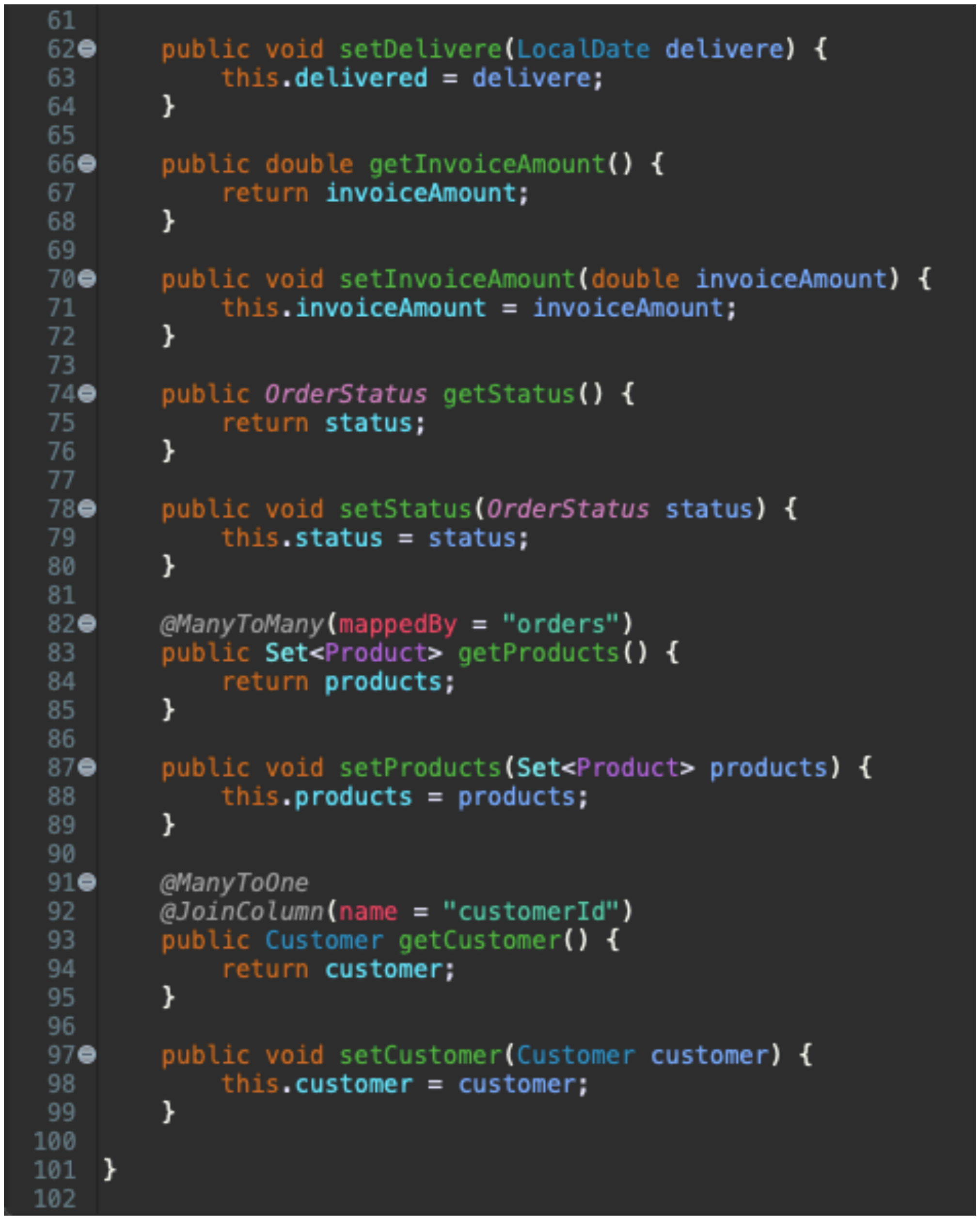
```java
import java.util.Set;

import javax.persistence.CascadeType;
import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.JoinColumn;
import javax.persistence.OneToMany;
import javax.persistence.OneToOne;

import com.promineotech.inventoryManagement.util.MembershipLevel;

@Entity
public class Customer {

    private Long id;
    private String firstName;
    private String lastName;
    private Address address;
    private MembershipLevel level;
    private Set<Order> orders;

    @Id
    @GeneratedValue(strategy = GenerationType.AUTO)
    public Long getId() {
        return id;
    }

    public void setId(Long id) {
        this.id = id;
    }

    public String getFirstName() {
        return firstName;
    }

    public void setFirstName(String firstName) {
        this.firstName = firstName;
    }

    public String getLastName() {
        return lastName;
    }

    public void setLastName(String lastName) {
        this.lastName = lastName;
    }

    @OneToOne(cascade = CascadeType.ALL)
    @JoinColumn(name = "id")
    public Address getAddress() {
        return address;
    }

    public void setAddress(Address address) {
        this.address = address;
    }
```



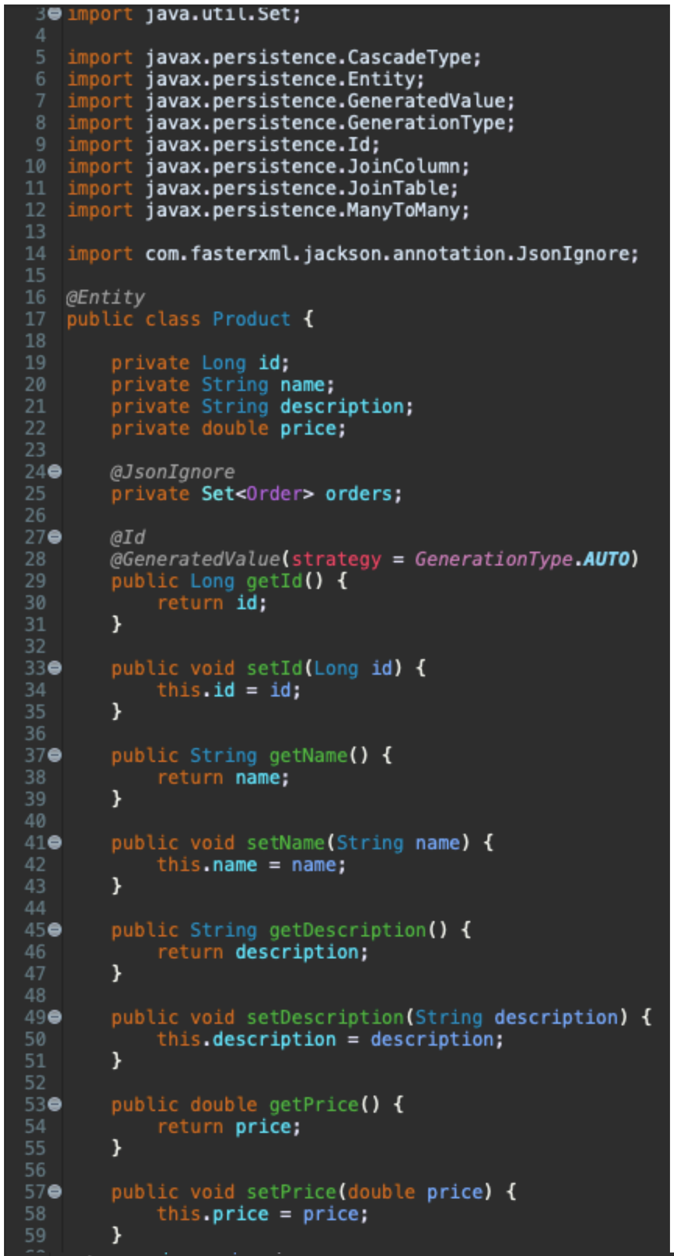
```java

    public MembershipLevel getLevel() {
        return level;
    }

    public void setLevel(MembershipLevel level) {
        this.level = level;
    }

    @OneToMany(mappedBy = "customer")
    public Set<Order> getOrders() {
        return orders;
    }

    public void setOrders(Set<Order> orders) {
        this.orders = orders;
    }

}
```



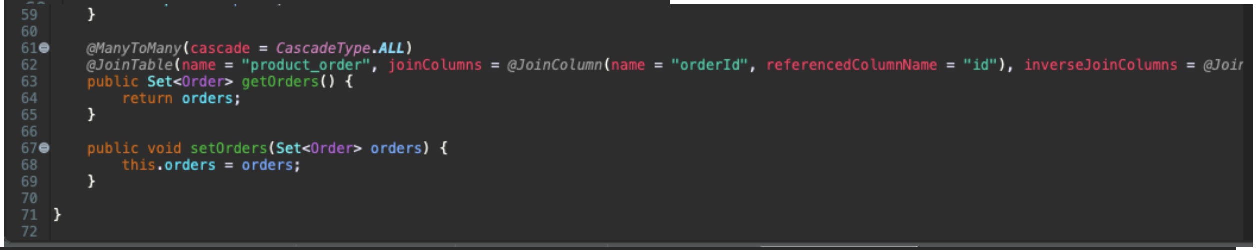
```java
import java.time.LocalDate;
import java.util.Set;

import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.JoinColumn;
import javax.persistence.ManyToMany;
import javax.persistence.ManyToOne;

import com.fasterxml.jackson.annotation.JsonIgnore;
import com.promineotech.inventoryManagement.util.OrderStatus;


@Entity
public class Order {

    private Long id;
    private LocalDate ordered;
    private LocalDate estimatedDelivery;
    private LocalDate delivered;
    private double invoiceAmount;
    private OrderStatus status;
    private Set<Product> products;

    @JsonIgnore
    private Customer customer;

    @Id
    @GeneratedValue(strategy = GenerationType.AUTO)
    public Long getId() {
        return id;
    }

    public void setId(Long id) {
        this.id = id;
    }

    public LocalDate getOrdered() {
        return ordered;
    }

    public void setOrdered(LocalDate ordered) {
        this.ordered = ordered;
    }

    public LocalDate getEstimatedDelivery() {
        return estimatedDelivery;
    }

    public void setEstimatedDelivery(LocalDate estimatedDelivery) {
        this.estimatedDelivery = estimatedDelivery;
    }

    public LocalDate getDelivere() {
        return delivered;
```



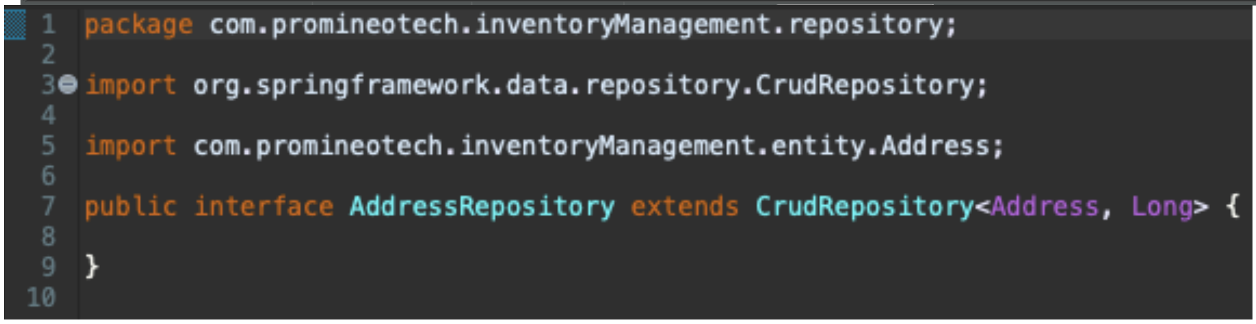
```java

    public void setDelivere(LocalDate delivere) {
        this.delivered = delivere;
    }

    public double getInvoiceAmount() {
        return invoiceAmount;
    }

    public void setInvoiceAmount(double invoiceAmount) {
        this.invoiceAmount = invoiceAmount;
    }

    public OrderStatus getStatus() {
        return status;
    }

    public void setStatus(OrderStatus status) {
        this.status = status;
    }

    @ManyToMany(mappedBy = "orders")
    public Set<Product> getProducts() {
        return products;
    }

    public void setProducts(Set<Product> products) {
        this.products = products;
    }

    @ManyToOne
    @JoinColumn(name = "customerId")
    public Customer getCustomer() {
        return customer;
    }

    public void setCustomer(Customer customer) {
        this.customer = customer;
    }

}

```



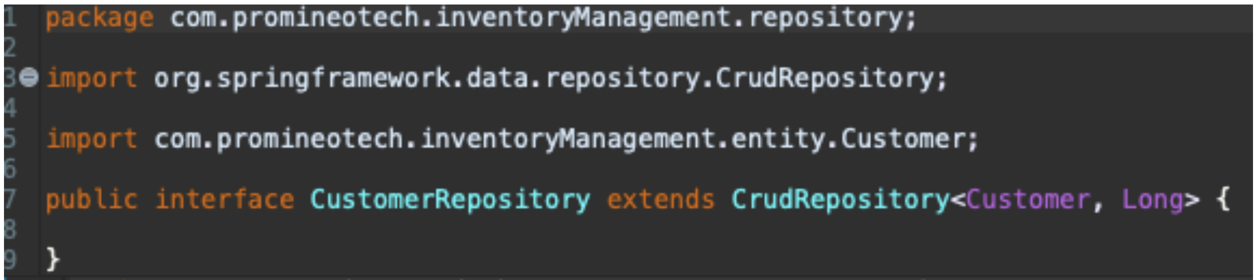
```java
import java.util.Set;

import javax.persistence.CascadeType;
import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.JoinColumn;
import javax.persistence.JoinTable;
import javax.persistence.ManyToMany;

import com.fasterxml.jackson.annotation.JsonIgnore;

@Entity
public class Product {

    private Long id;
    private String name;
    private String description;
    private double price;

    @JsonIgnore
    private Set<Order> orders;

    @Id
    @GeneratedValue(strategy = GenerationType.AUTO)
    public Long getId() {
        return id;
    }

    public void setId(Long id) {
        this.id = id;
    }

    public String getName() {
        return name;
    }

    public void setName(String name) {
        this.name = name;
    }

    public String getDescription() {
        return description;
    }

    public void setDescription(String description) {
        this.description = description;
    }

    public double getPrice() {
        return price;
    }

    public void setPrice(double price) {
        this.price = price;
    }
```



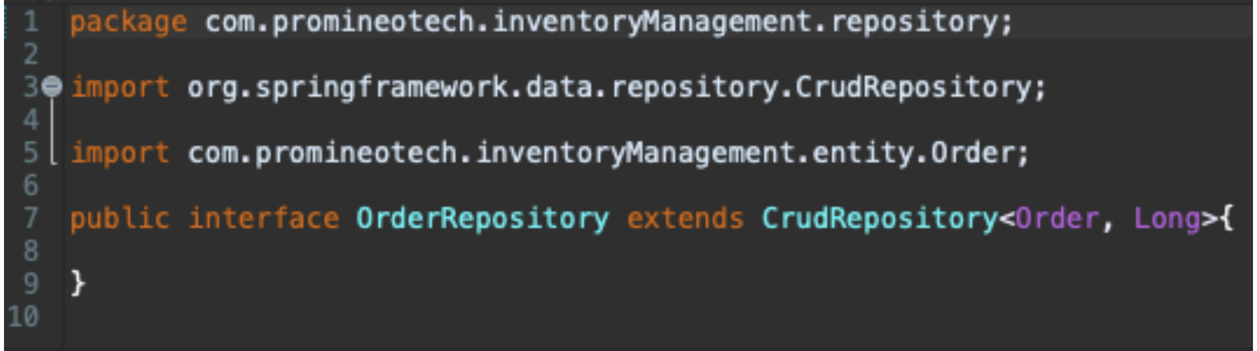
```java
    }

    @ManyToMany(cascade = CascadeType.ALL)
    @JoinTable(name = "product_order", joinColumns = @JoinColumn(name = "orderId", referencedColumnName = "id"), inverseJoinColumns = @Joi
    public Set<Order> getOrders() {
        return orders;
    }

    public void setOrders(Set<Order> orders) {
        this.orders = orders;
    }

}
```



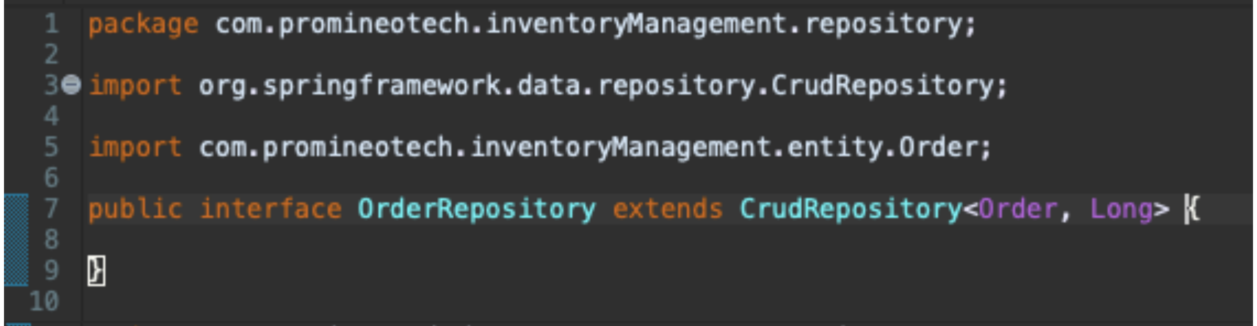
```java
package com.promineotech.inventoryManagement.repository;

import org.springframework.data.repository.CrudRepository;

import com.promineotech.inventoryManagement.entity.Address;

public interface AddressRepository extends CrudRepository<Address, Long> {

}
```



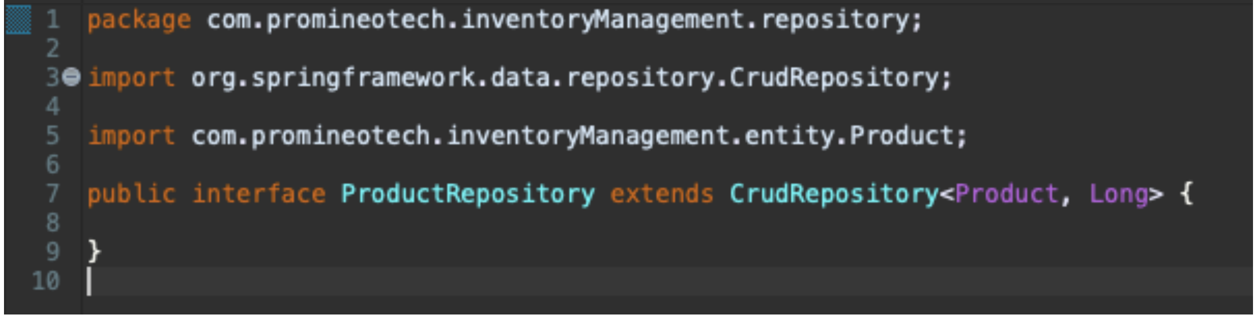
```java
package com.promineotech.inventoryManagement.repository;

import org.springframework.data.repository.CrudRepository;

import com.promineotech.inventoryManagement.entity.Customer;

public interface CustomerRepository extends CrudRepository<Customer, Long> {

}
```



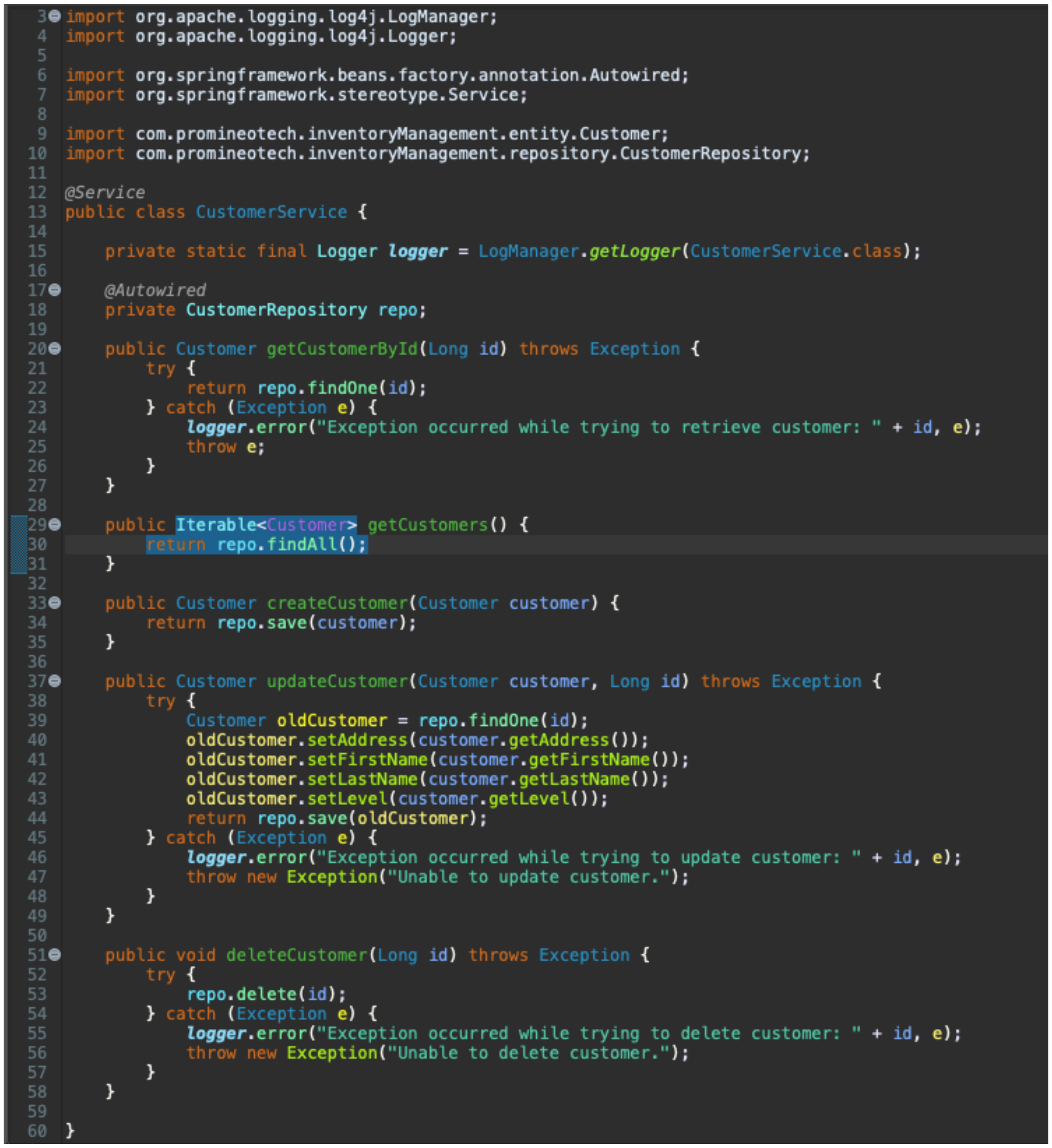
```java
package com.promineotech.inventoryManagement.repository;

import org.springframework.data.repository.CrudRepository;

import com.promineotech.inventoryManagement.entity.Order;

public interface OrderRepository extends CrudRepository<Order, Long>{

}

```



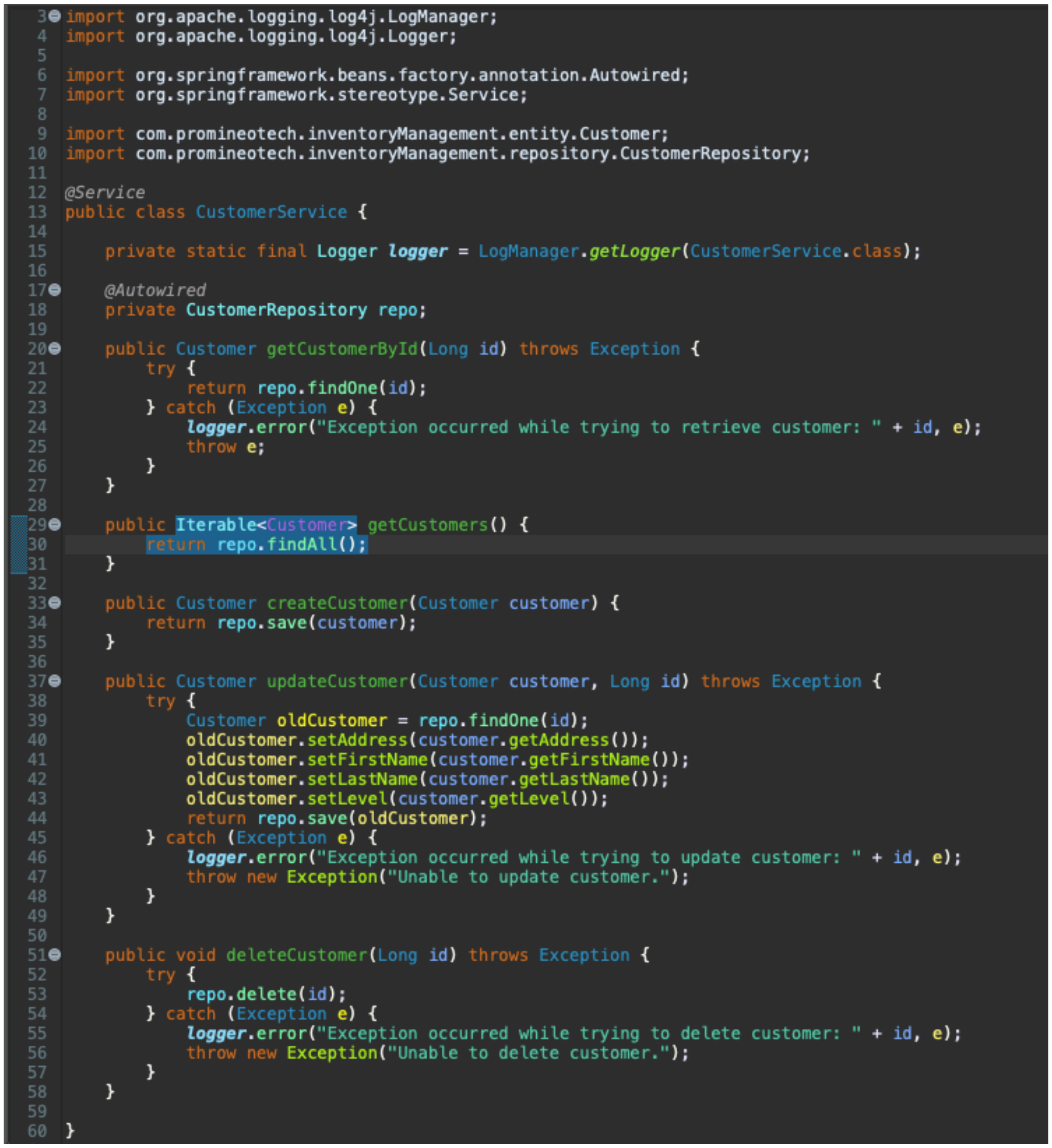
```java
package com.promineotech.inventoryManagement.repository;

import org.springframework.data.repository.CrudRepository;

import com.promineotech.inventoryManagement.entity.Order;

public interface OrderRepository extends CrudRepository<Order, Long> {

}

```



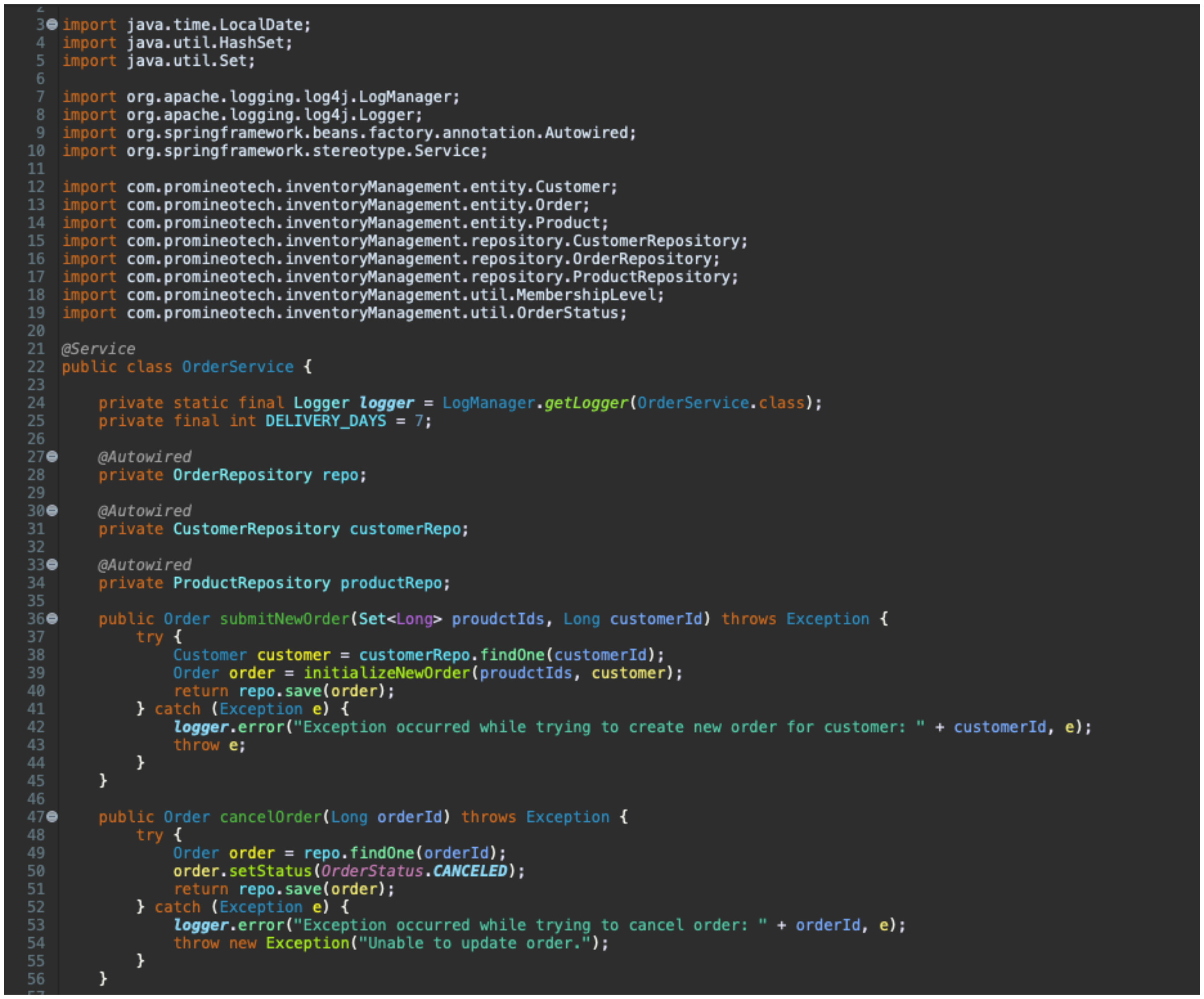
```java
package com.promineotech.inventoryManagement.repository;

import org.springframework.data.repository.CrudRepository;

import com.promineotech.inventoryManagement.entity.Product;

public interface ProductRepository extends CrudRepository<Product, Long> {

}

```



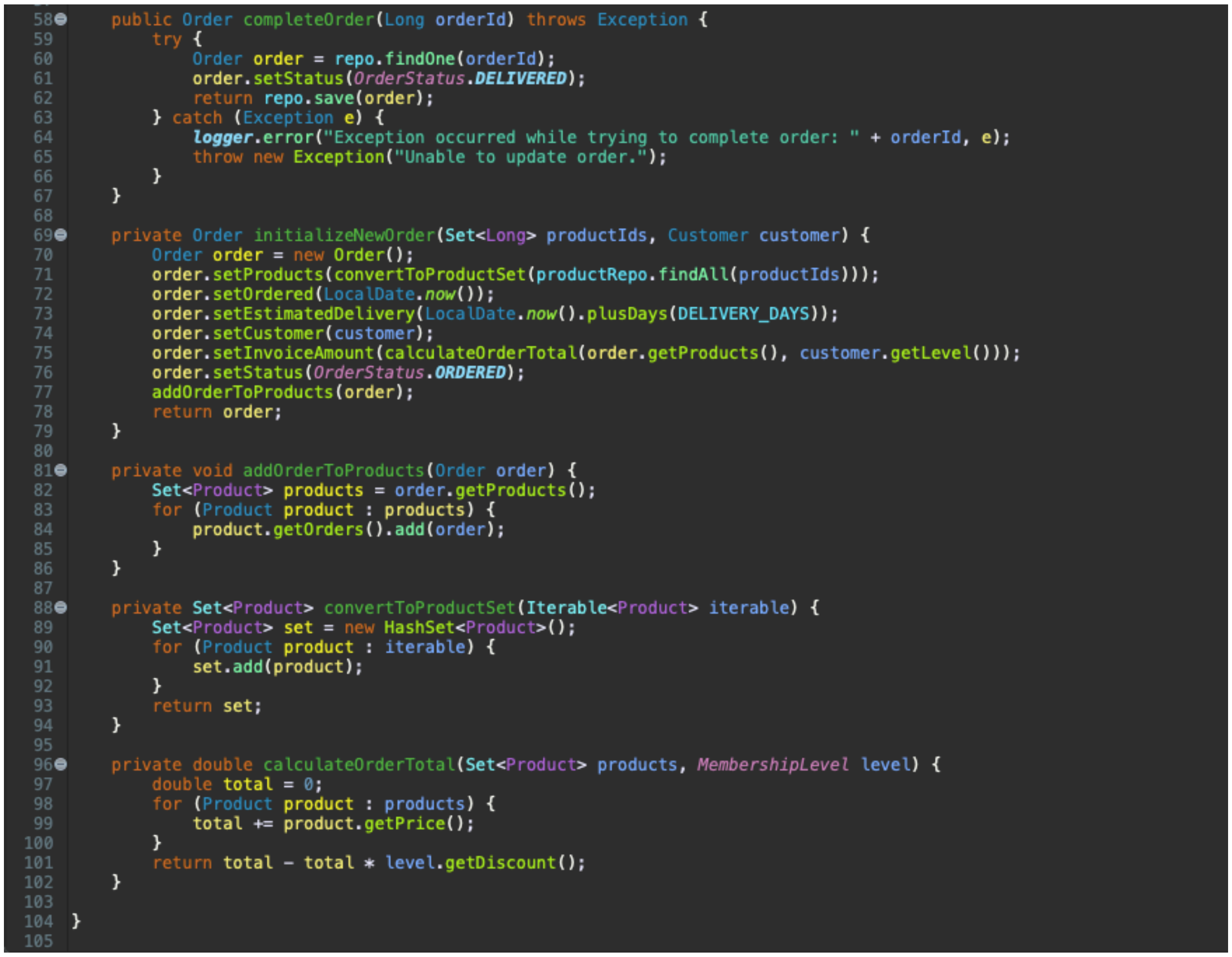
```java
import org.apache.logging.log4j.LogManager;
import org.apache.logging.log4j.Logger;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;

import com.promineotech.inventoryManagement.entity.Customer;
import com.promineotech.inventoryManagement.repository.CustomerRepository;

@Service
public class CustomerService {

    private static final Logger logger = LogManager.getLogger(CustomerService.class);

    @Autowired
    private CustomerRepository repo;

    public Customer getCustomerById(Long id) throws Exception {
        try {
            return repo.findOne(id);
        } catch (Exception e) {
            logger.error("Exception occurred while trying to retrieve customer: " + id, e);
            throw e;
        }
    }

    public Iterable<Customer> getCustomers() {
        return repo.findAll();
    }

    public Customer createCustomer(Customer customer) {
        return repo.save(customer);
    }

    public Customer updateCustomer(Customer customer, Long id) throws Exception {
        try {
            Customer oldCustomer = repo.findOne(id);
            oldCustomer.setAddress(customer.getAddress());
            oldCustomer.setFirstName(customer.getFirstName());
            oldCustomer.setLastName(customer.getLastName());
            oldCustomer.setLevel(customer.getLevel());
            return repo.save(oldCustomer);
        } catch (Exception e) {
            logger.error("Exception occurred while trying to update customer: " + id, e);
            throw new Exception("Unable to update customer.");
        }
    }

    public void deleteCustomer(Long id) throws Exception {
        try {
            repo.delete(id);
        } catch (Exception e) {
            logger.error("Exception occurred while trying to delete customer: " + id, e);
            throw new Exception("Unable to delete customer.");
        }
    }

}
```



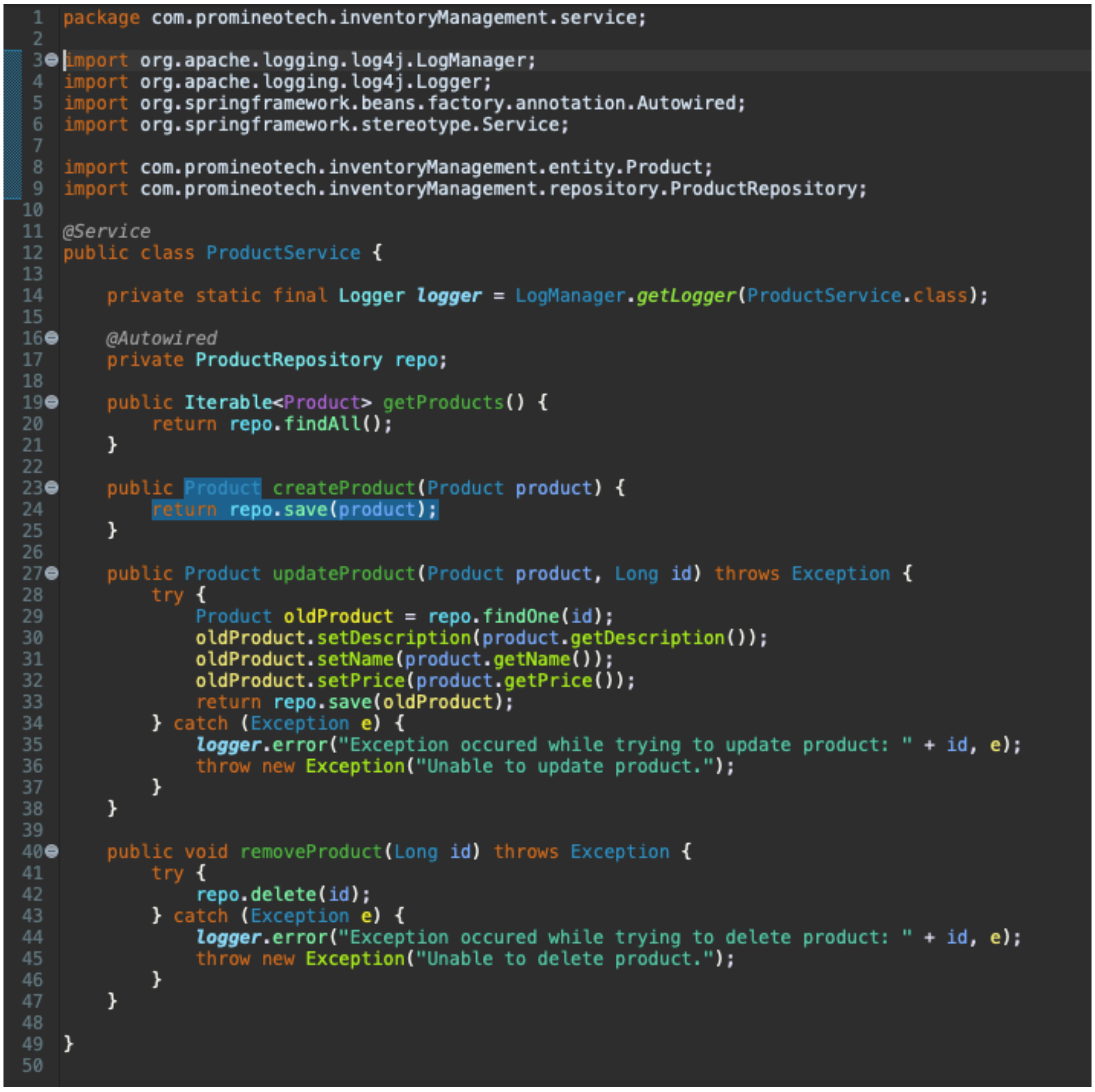
```java
import org.apache.logging.log4j.LogManager;
import org.apache.logging.log4j.Logger;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;

import com.promineotech.inventoryManagement.entity.Customer;
import com.promineotech.inventoryManagement.repository.CustomerRepository;

@Service
public class CustomerService {

    private static final Logger logger = LogManager.getLogger(CustomerService.class);

    @Autowired
    private CustomerRepository repo;

    public Customer getCustomerById(Long id) throws Exception {
        try {
            return repo.findOne(id);
        } catch (Exception e) {
            logger.error("Exception occurred while trying to retrieve customer: " + id, e);
            throw e;
        }
    }

    public Iterable<Customer> getCustomers() {
        return repo.findAll();
    }

    public Customer createCustomer(Customer customer) {
        return repo.save(customer);
    }

    public Customer updateCustomer(Customer customer, Long id) throws Exception {
        try {
            Customer oldCustomer = repo.findOne(id);
            oldCustomer.setAddress(customer.getAddress());
            oldCustomer.setFirstName(customer.getFirstName());
            oldCustomer.setLastName(customer.getLastName());
            oldCustomer.setLevel(customer.getLevel());
            return repo.save(oldCustomer);
        } catch (Exception e) {
            logger.error("Exception occurred while trying to update customer: " + id, e);
            throw new Exception("Unable to update customer.");
        }
    }

    public void deleteCustomer(Long id) throws Exception {
        try {
            repo.delete(id);
        } catch (Exception e) {
            logger.error("Exception occurred while trying to delete customer: " + id, e);
            throw new Exception("Unable to delete customer.");
        }
    }

}
```



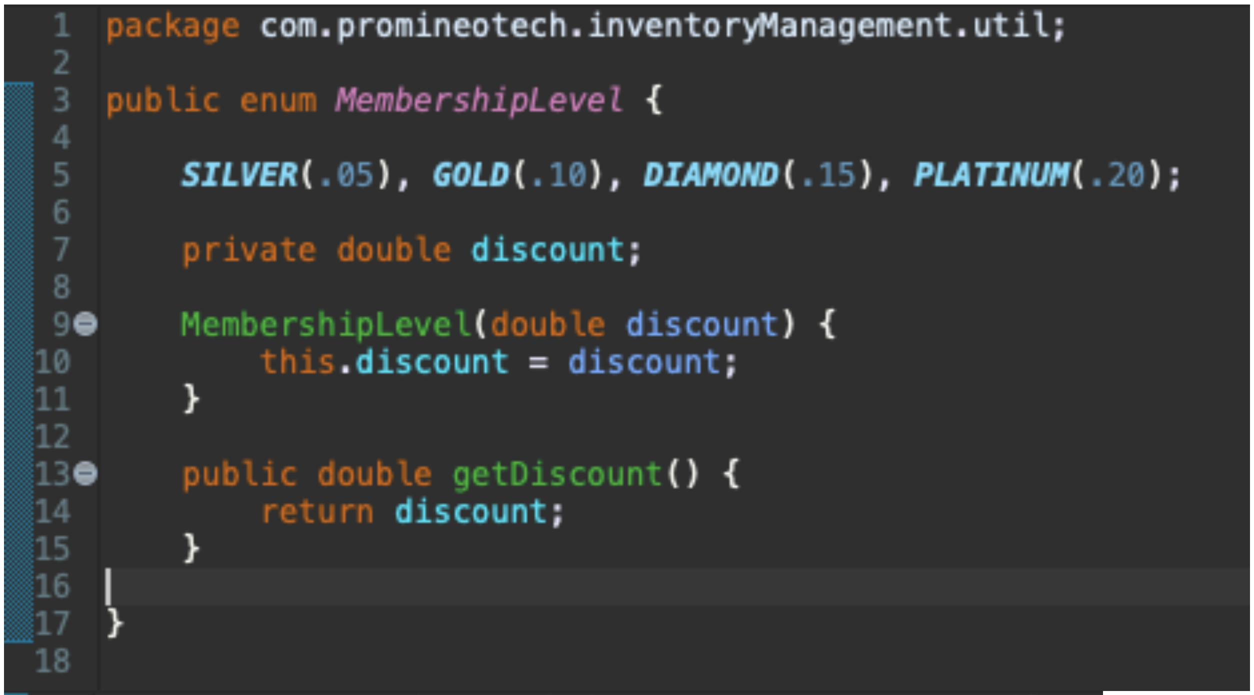
```java
import java.time.LocalDate;
import java.util.HashSet;
import java.util.Set;

import org.apache.logging.log4j.LogManager;
import org.apache.logging.log4j.Logger;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;

import com.promineotech.inventoryManagement.entity.Customer;
import com.promineotech.inventoryManagement.entity.Order;
import com.promineotech.inventoryManagement.entity.Product;
import com.promineotech.inventoryManagement.repository.CustomerRepository;
import com.promineotech.inventoryManagement.repository.OrderRepository;
import com.promineotech.inventoryManagement.repository.ProductRepository;
import com.promineotech.inventoryManagement.util.MembershipLevel;
import com.promineotech.inventoryManagement.util.OrderStatus;

@Service
public class OrderService {

    private static final Logger logger = LogManager.getLogger(OrderService.class);
    private final int DELIVERY_DAYS = 7;

    @Autowired
    private OrderRepository repo;

    @Autowired
    private CustomerRepository customerRepo;

    @Autowired
    private ProductRepository productRepo;

    public Order submitNewOrder(Set<Long> proudctIds, Long customerId) throws Exception {
        try {
            Customer customer = customerRepo.findOne(customerId);
            Order order = initializeNewOrder(proudctIds, customer);
            return repo.save(order);
        } catch (Exception e) {
            logger.error("Exception occurred while trying to create new order for customer: " + customerId, e);
            throw e;
        }
    }

    public Order cancelOrder(Long orderId) throws Exception {
        try {
            Order order = repo.findOne(orderId);
            order.setStatus(OrderStatus.CANCELED);
            return repo.save(order);
        } catch (Exception e) {
            logger.error("Exception occurred while trying to cancel order: " + orderId, e);
            throw new Exception("Unable to update order.");
        }
    }
```



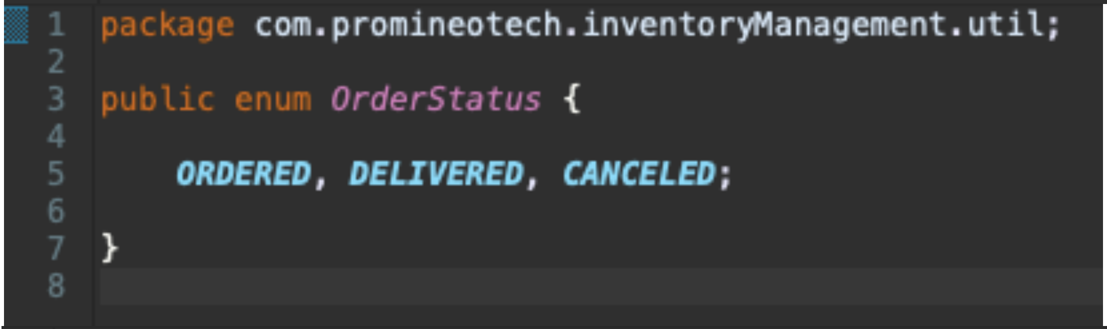
```java
    public Order completeOrder(Long orderId) throws Exception {
        try {
            Order order = repo.findOne(orderId);
            order.setStatus(OrderStatus.DELIVERED);
            return repo.save(order);
        } catch (Exception e) {
            logger.error("Exception occurred while trying to complete order: " + orderId, e);
            throw new Exception("Unable to update order.");
        }
    }

    private Order initializeNewOrder(Set<Long> productIds, Customer customer) {
        Order order = new Order();
        order.setProducts(convertToProductSet(productRepo.findAll(productIds)));
        order.setOrdered(LocalDate.now());
        order.setEstimatedDelivery(LocalDate.now().plusDays(DELIVERY_DAYS));
        order.setCustomer(customer);
        order.setInvoiceAmount(calculateOrderTotal(order.getProducts(), customer.getLevel()));
        order.setStatus(OrderStatus.ORDERED);
        addOrderToProducts(order);
        return order;
    }

    private void addOrderToProducts(Order order) {
        Set<Product> products = order.getProducts();
        for (Product product : products) {
            product.getOrders().add(order);
        }
    }

    private Set<Product> convertToProductSet(Iterable<Product> iterable) {
        Set<Product> set = new HashSet<Product>();
        for (Product product : iterable) {
            set.add(product);
        }
        return set;
    }

    private double calculateOrderTotal(Set<Product> products, MembershipLevel level) {
        double total = 0;
        for (Product product : products) {
            total += product.getPrice();
        }
        return total - total * level.getDiscount();
    }

}
```



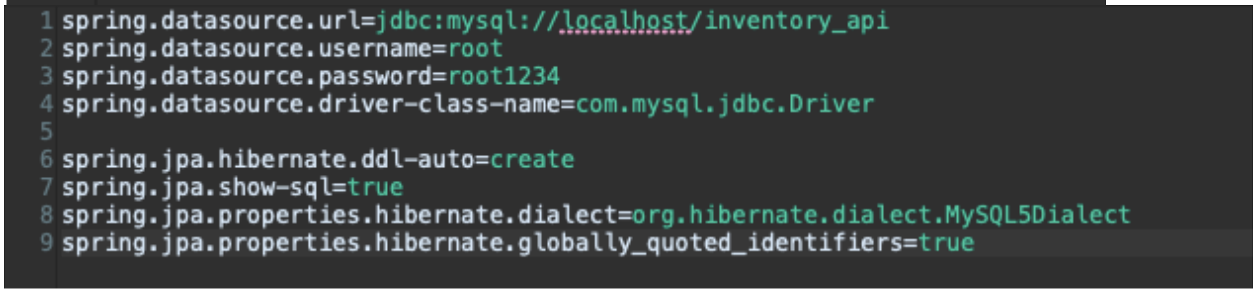
```java
package com.promineotech.inventoryManagement.service;

import org.apache.logging.log4j.LogManager;
import org.apache.logging.log4j.Logger;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;

import com.promineotech.inventoryManagement.entity.Product;
import com.promineotech.inventoryManagement.repository.ProductRepository;

@Service
public class ProductService {

    private static final Logger logger = LogManager.getLogger(ProductService.class);

    @Autowired
    private ProductRepository repo;

    public Iterable<Product> getProducts() {
        return repo.findAll();
    }

    public Product createProduct(Product product) {
        return repo.save(product);
    }

    public Product updateProduct(Product product, Long id) throws Exception {
        try {
            Product oldProduct = repo.findOne(id);
            oldProduct.setDescription(product.getDescription());
            oldProduct.setName(product.getName());
            oldProduct.setPrice(product.getPrice());
            return repo.save(oldProduct);
        } catch (Exception e) {
            logger.error("Exception occured while trying to update product: " + id, e);
            throw new Exception("Unable to update product.");
        }
    }

    public void removeProduct(Long id) throws Exception {
        try {
            repo.delete(id);
        } catch (Exception e) {
            logger.error("Exception occured while trying to delete product: " + id, e);
            throw new Exception("Unable to delete product.");
        }
    }

}
```



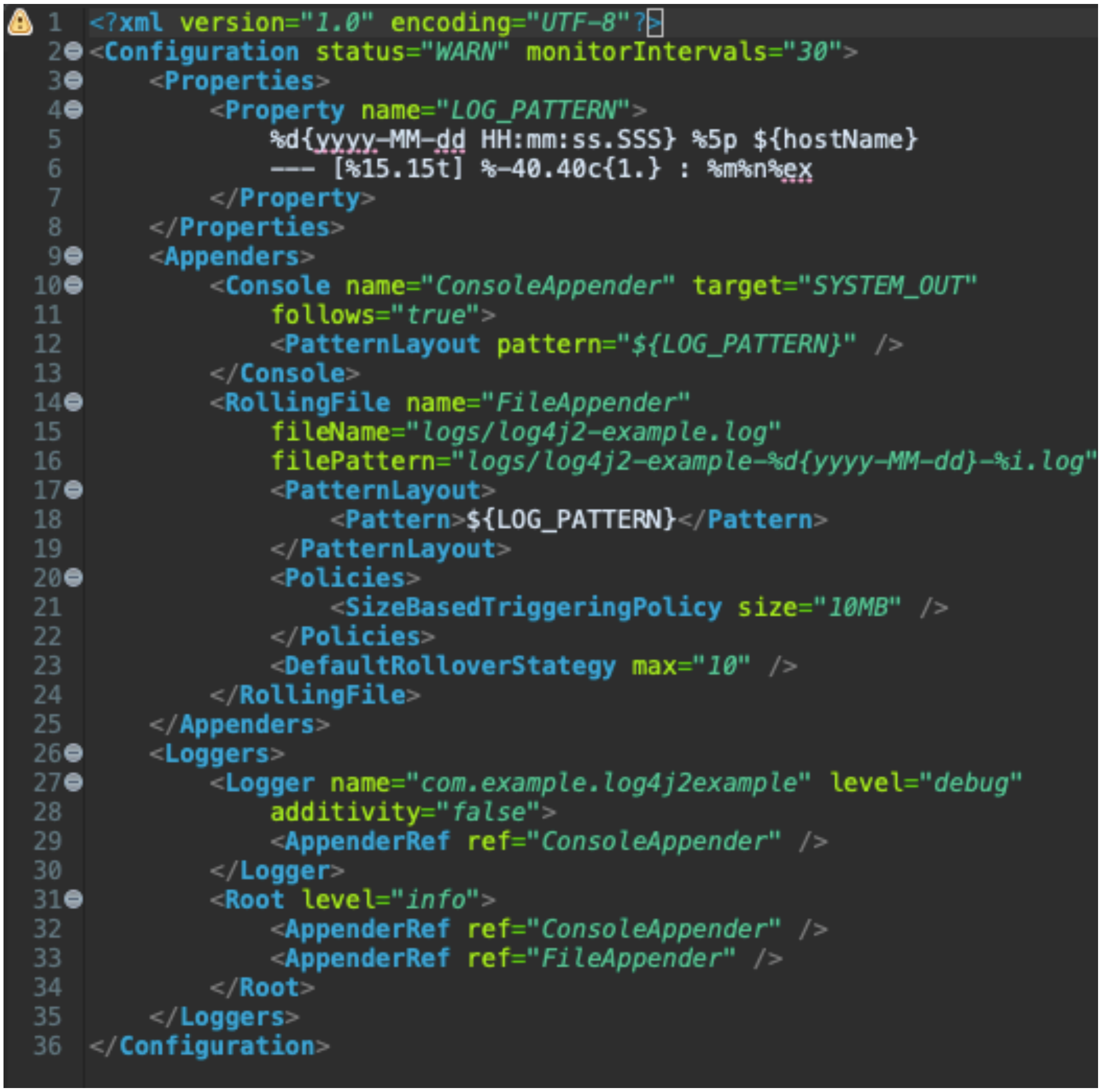
```java
package com.promineotech.inventoryManagement.util;

public enum MembershipLevel {

    SILVER(.05), GOLD(.10), DIAMOND(.15), PLATINUM(.20);

    private double discount;

    MembershipLevel(double discount) {
        this.discount = discount;
    }

    public double getDiscount() {
        return discount;
    }

}
```

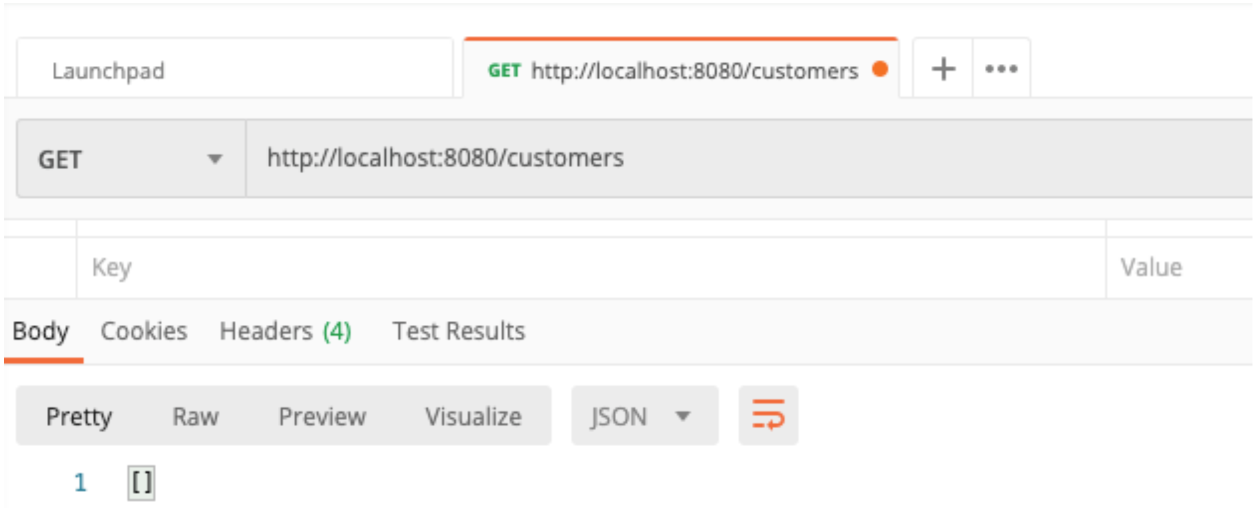


```java
package com.promineotech.inventoryManagement.util;

public enum OrderStatus {

    ORDERED, DELIVERED, CANCELED;

}
```



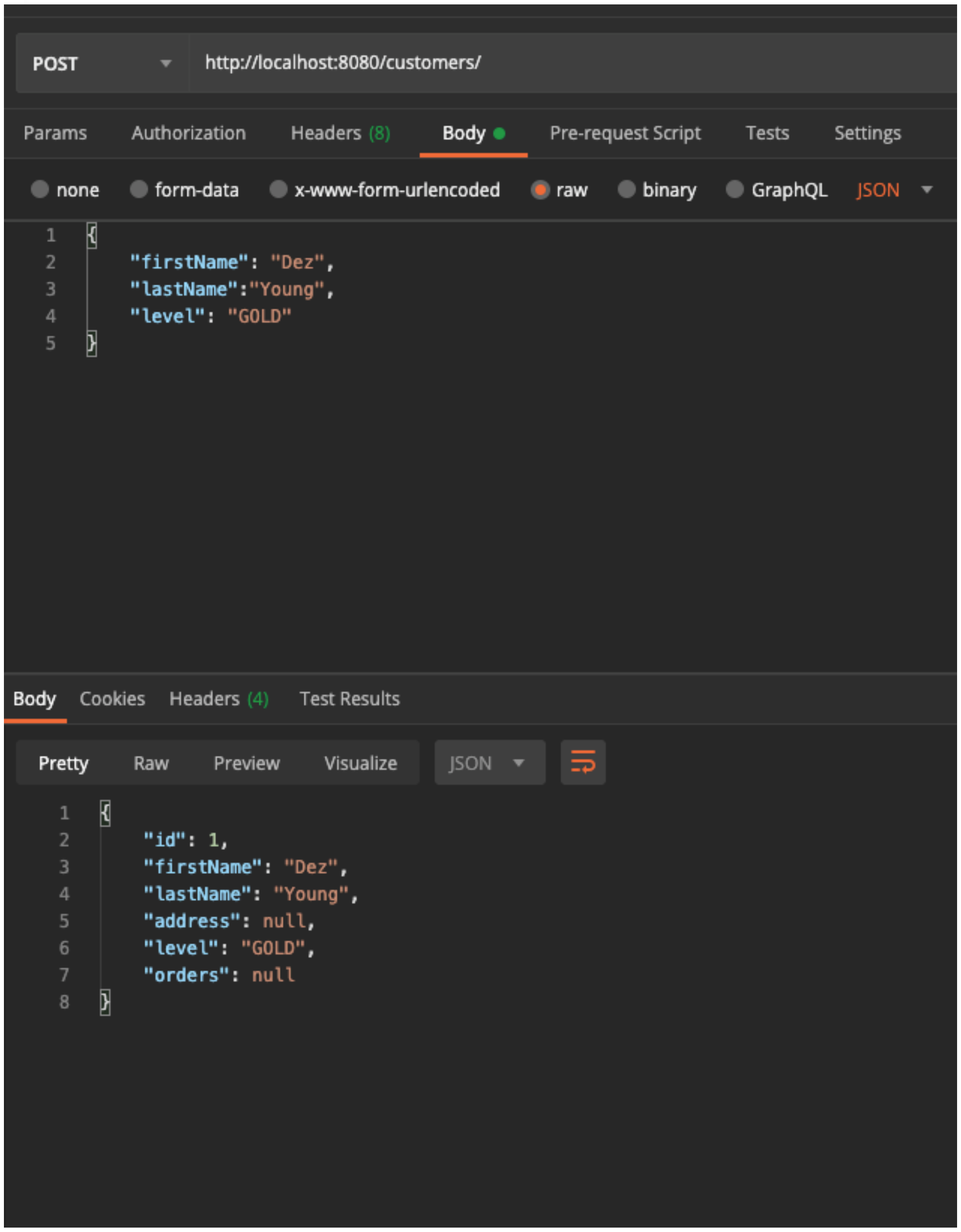
```properties
spring.datasource.url=jdbc:mysql://localhost/inventory_api
spring.datasource.username=root
spring.datasource.password=root1234
spring.datasource.driver-class-name=com.mysql.jdbc.Driver

spring.jpa.hibernate.ddl-auto=create
spring.jpa.show-sql=true
spring.jpa.properties.hibernate.dialect=org.hibernate.dialect.MySQL5Dialect
spring.jpa.properties.hibernate.globally_quoted_identifiers=true
```


```xml
<?xml version="1.0" encoding="UTF-8"?>
<Configuration status="WARN" monitorIntervals="30">
    <Properties>
        <Property name="LOG_PATTERN">
            %d{yyyy-MM-dd HH:mm:ss.SSS} %5p ${hostName}
            --- [%15.15t] %-40.40c{1.} : %m%n%ex
        </Property>
    </Properties>
    <Appenders>
        <Console name="ConsoleAppender" target="SYSTEM_OUT"
            follows="true">
            <PatternLayout pattern="${LOG_PATTERN}" />
        </Console>
        <RollingFile name="FileAppender"
            fileName="logs/log4j2-example.log"
            filePattern="logs/log4j2-example-%d{yyyy-MM-dd}-%i.log"
            <PatternLayout>
                <Pattern>${LOG_PATTERN}</Pattern>
            </PatternLayout>
            <Policies>
                <SizeBasedTriggeringPolicy size="10MB" />
            </Policies>
            <DefaultRolloverStategy max="10" />
        </RollingFile>
    </Appenders>
    <Loggers>
        <Logger name="com.example.log4j2example" level="debug"
            additivity="false">
            <AppenderRef ref="ConsoleAppender" />
        </Logger>
        <Root level="info">
            <AppenderRef ref="ConsoleAppender" />
            <AppenderRef ref="FileAppender" />
        </Root>
    </Loggers>
</Configuration>
```



GET http://localhost:8080/customers

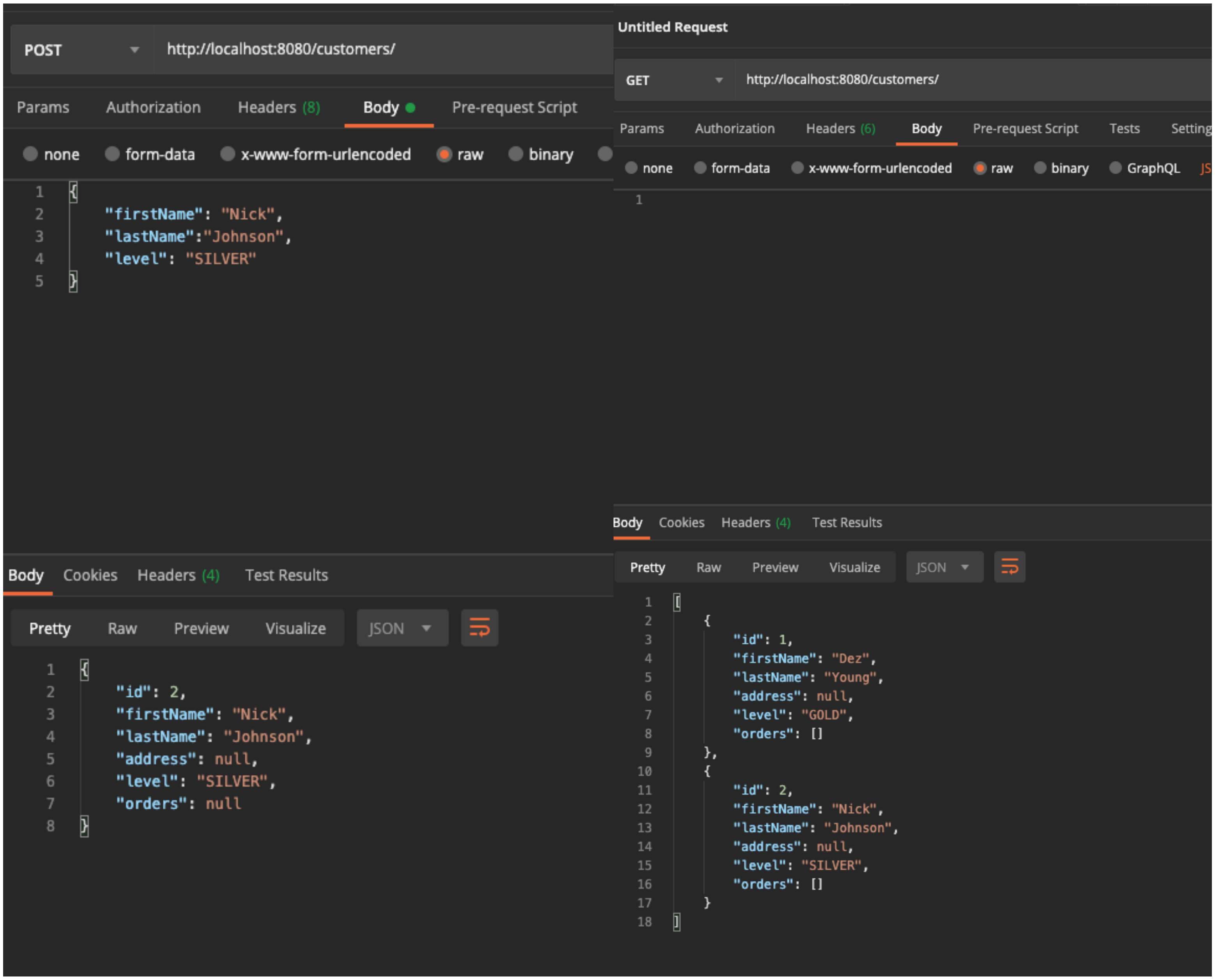
```
[]
```



POST http://localhost:8080/customers/

```
{
    "firstName": "Dez",
    "lastName":"Young",
    "level": "GOLD"
}
```

```
{
    "id": 1,
    "firstName": "Dez",
    "lastName": "Young",
    "address": null,
    "level": "GOLD",
    "orders": null
}
```



POST http://localhost:8080/customers/

Params Authorization Headers (8) Body Pre-request Script

none form-data x-www-form-urlencoded [✓ raw] binary

```
{
    "firstName": "Nick",
    "lastName":"Johnson",
    "level": "SILVER"
}
```

Body Cookies Headers (4) Test Results

Pretty Raw Preview Visualize JSON

```
{
    "id": 2,
    "firstName": "Nick",
    "lastName": "Johnson",
    "address": null,
    "level": "SILVER",
    "orders": null
}
```

Untitled Request

GET http://localhost:8080/customers/

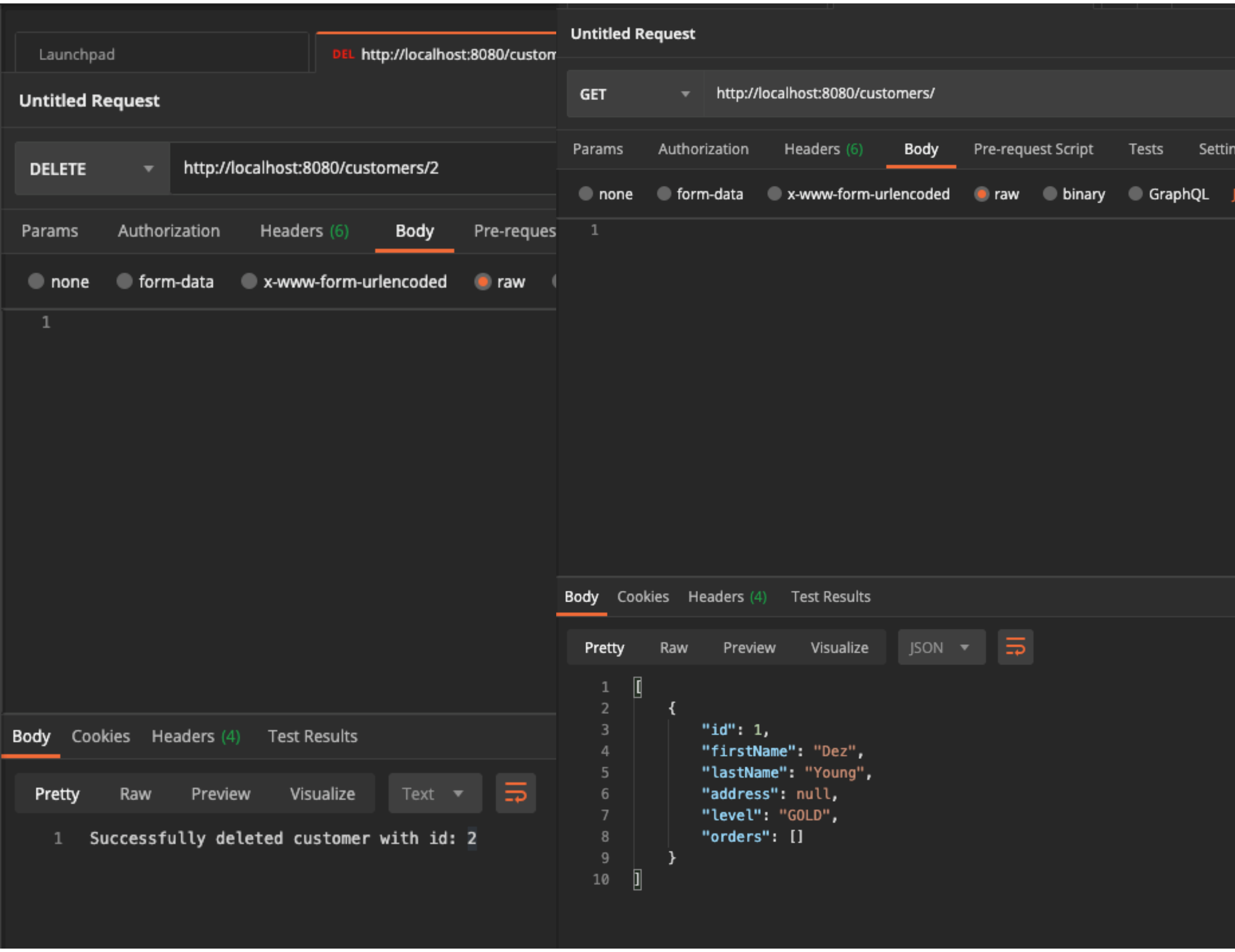
Params Authorization Headers (6) Body Pre-request Script Tests Setting

none form-data x-www-form-urlencoded [✓ raw] binary GraphQL

Body Cookies Headers (4) Test Results

Pretty Raw Preview Visualize JSON

```
[
    {
        "id": 1,
        "firstName": "Dez",
        "lastName": "Young",
        "address": null,
        "level": "GOLD",
        "orders": []
    },
    {
        "id": 2,
        "firstName": "Nick",
        "lastName": "Johnson",
        "address": null,
        "level": "SILVER",
        "orders": []
    }
]
```



Untitled Request

DELETE http://localhost:8080/customers/2

Body: raw

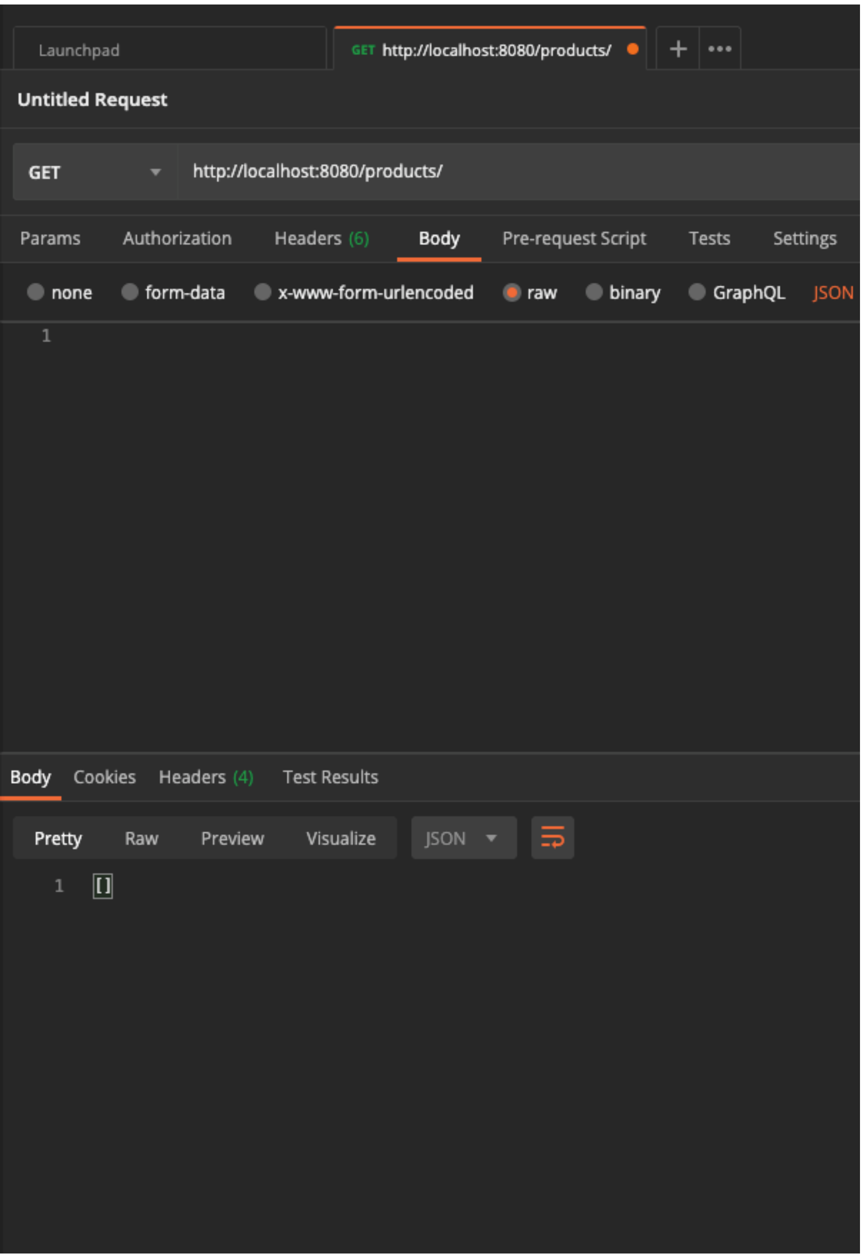
```
Successfully deleted customer with id: 2
```

Untitled Request

GET http://localhost:8080/customers/

Body: raw

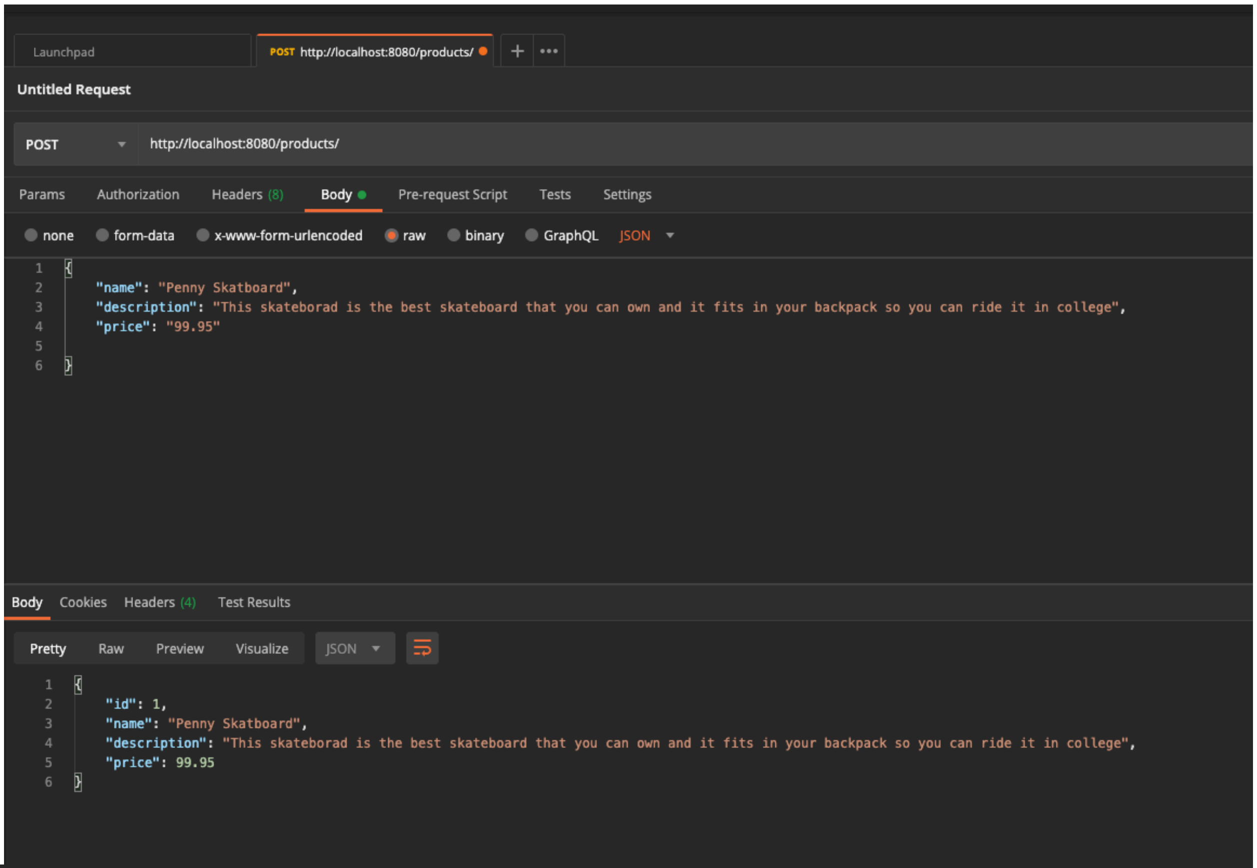
```
[
    {
        "id": 1,
        "firstName": "Dez",
        "lastName": "Young",
        "address": null,
        "level": "GOLD",
        "orders": []
    }
]
```



Untitled Request

GET http://localhost:8080/products/

```
[]
```

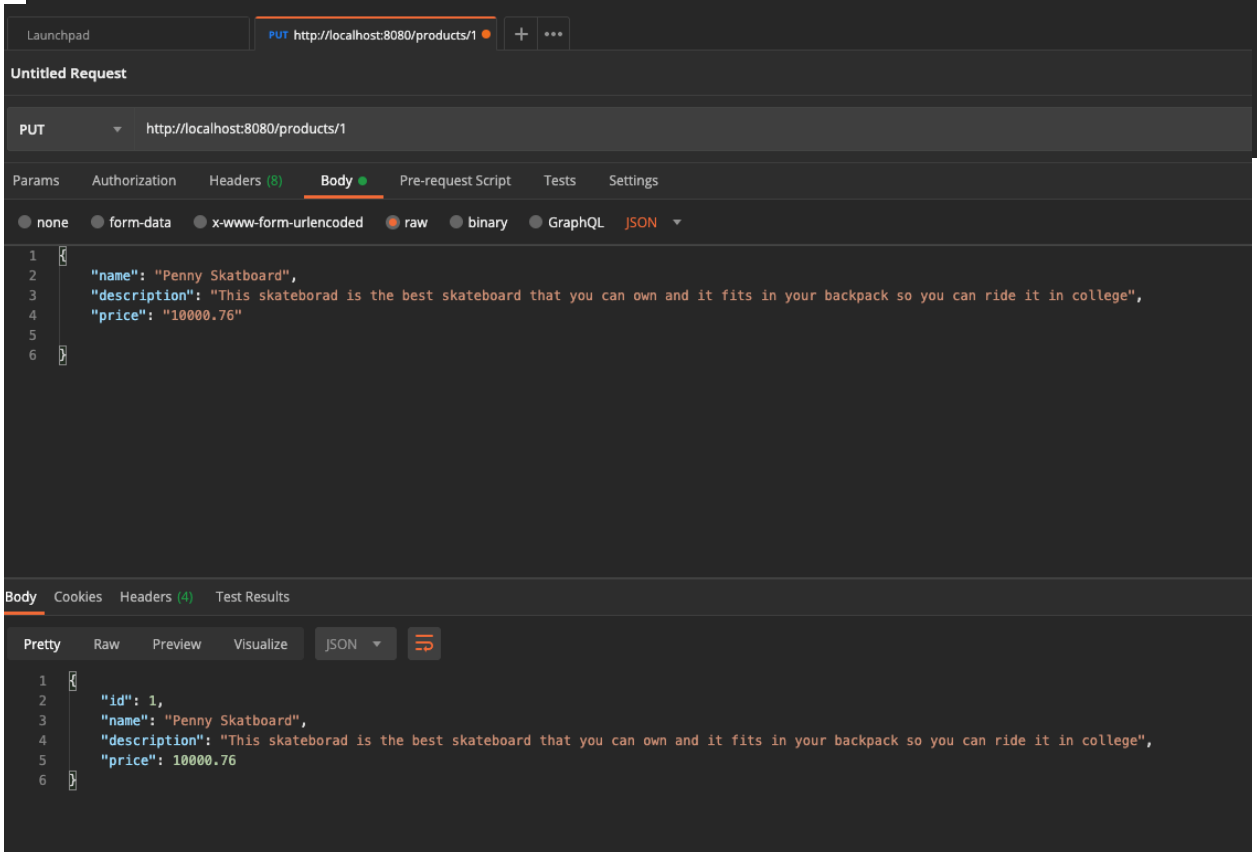

Launchpad
POST http://localhost:8080/products/
Untitled Request
Params Authorization Headers (8) Body Pre-request Script Tests Settings
none form-data x-www-form-urlencoded raw binary GraphQL JSON

POST http://localhost:8080/products/

```
{
    "name": "Penny Skatboard",
    "description": "This skateborad is the best skateboard that you can own and it fits in your backpack so you can ride it in college",
    "price": "99.95"

}
```

Body Cookies Headers (4) Test Results
Pretty Raw Preview Visualize JSON

```
{
    "id": 1,
    "name": "Penny Skatboard",
    "description": "This skateborad is the best skateboard that you can own and it fits in your backpack so you can ride it in college",
    "price": 99.95
}
```

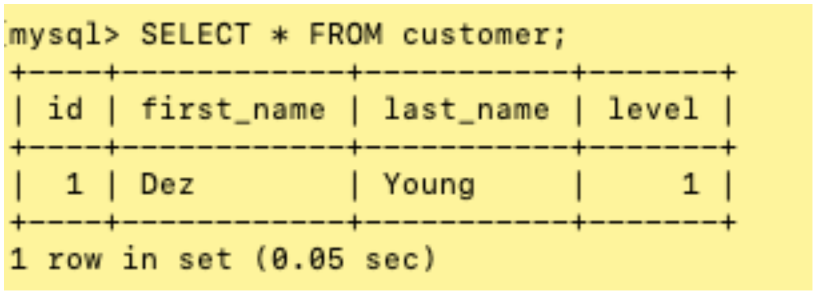

Launchpad
PUT http://localhost:8080/products/1
Untitled Request
Params Authorization Headers (8) Body Pre-request Script Tests Settings
none form-data x-www-form-urlencoded raw binary GraphQL JSON

PUT http://localhost:8080/products/1

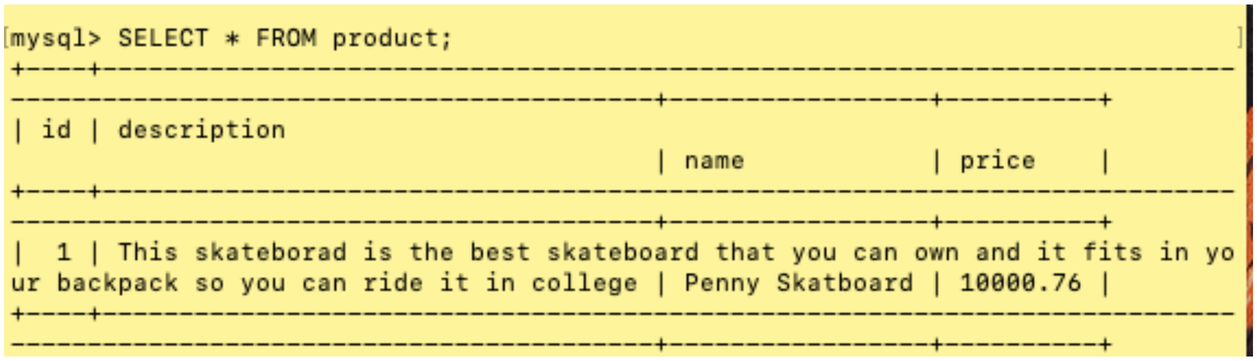
```
{
    "name": "Penny Skatboard",
    "description": "This skateborad is the best skateboard that you can own and it fits in your backpack so you can ride it in college",
    "price": "10000.76"

}
```

Body Cookies Headers (4) Test Results
Pretty Raw Preview Visualize JSON

```
{
    "id": 1,
    "name": "Penny Skatboard",
    "description": "This skateborad is the best skateboard that you can own and it fits in your backpack so you can ride it in college",
    "price": 10000.76
}
```



```
mysql> SELECT * FROM customer;
+----+------------+-----------+-------+
| id | first_name | last_name | level |
+----+------------+-----------+-------+
|  1 | Dez        | Young     |     1 |
+----+------------+-----------+-------+
1 row in set (0.05 sec)
```



```
mysql> SELECT * FROM product;
+----+--------------------------------------------------------------------------
------------------------------------------+-----------------+----------+
| id | description
                                          | name            | price    |
+----+--------------------------------------------------------------------------
------------------------------------------+-----------------+----------+
|  1 | This skateborad is the best skateboard that you can own and it fits in yo
ur backpack so you can ride it in college | Penny Skatboard | 10000.76 |
+----+--------------------------------------------------------------------------
------------------------------------------+-----------------+----------+
```

URL to GitHub Repository: https://github.com/DesmondYo/InventoryManagmentAPI