



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

```
1 package com.promineotech.inventoryManagement;
2
3 import org.springframework.boot.SpringApplication;
4 import org.springframework.boot.autoconfigure.SpringBootApplication;
5 import org.springframework.context.annotation.ComponentScan;
6
7 @ComponentScan("com.promineotech.inventoryManagement")
8 @SpringBootApplication
9 public class App {
10     public static void main(String[] args) {
11         SpringApplication.run(App.class, args);
12     }
13 }
14
15 package com.promineotech.inventoryManagement.controller;
16
17 import org.springframework.beans.factory.annotation.Autowired;
18 import org.springframework.http.HttpStatus;
19 import org.springframework.http.ResponseEntity;
20 import org.springframework.web.bind.annotation.PathVariable;
21 import org.springframework.web.bind.annotation.RequestBody;
22 import org.springframework.web.bind.annotation.RequestMapping;
23 import org.springframework.web.bind.annotation.RequestMethod;
24 import org.springframework.web.bind.annotation.RestController;
25
26 import com.promineotech.inventoryManagement.entity.Customer;
27 import com.promineotech.inventoryManagement.service.CustomerService;
28
29 @RestController
30 @RequestMapping("/customers")
31 public class CustomerController {
32
33     @Autowired
34     private CustomerService service;
35
36     @RequestMapping(value =("/{id}", method = RequestMethod.GET)
37     public ResponseEntity<Object> getCustomer(@PathVariable Long id) {
38         try {
39             return new ResponseEntity<Object>(service.getCustomerById(id), HttpStatus.OK);
40         } catch (Exception e) {
41             return new ResponseEntity<Object>(e.getMessage(), HttpStatus.NOT_FOUND);
42         }
43     }
44
45     @RequestMapping(method = RequestMethod.GET)
46     public ResponseEntity<Object> getCustomers() {
47         return new ResponseEntity<Object>(service.getCustomers(), HttpStatus.OK);
48     }
49
50     @RequestMapping(method = RequestMethod.POST)
51     public ResponseEntity<Object> createCustomer(@RequestBody Customer customer) {
52         return new ResponseEntity<Object>(service.createCustomer(customer), HttpStatus.CREATED);
53     }
54
55     @RequestMapping(value =("/{id}", method = RequestMethod.PUT)
56     public ResponseEntity<Object> updateCustomer(@RequestBody Customer customer, @PathVariable Long id) {
57         try {
58             return new ResponseEntity<Object>(service.updateCustomer(customer, id), HttpStatus.OK);
59         } catch (Exception e) {
60             return new ResponseEntity<Object>(e.getMessage(), HttpStatus.NOT_FOUND);
61         }
62     }
63
64     @RequestMapping(value =("/{id}", method = RequestMethod.DELETE)
65     public ResponseEntity<Object> deleteCustomer(@PathVariable Long id) {
66         try {
67             service.deleteCustomer(id);
68             return new ResponseEntity<Object>("Successfully deleted customer with id: " + id, HttpStatus.OK);
69         } catch (Exception e) {
70             return new ResponseEntity<Object>(e.getMessage(), HttpStatus.NOT_FOUND);
71         }
72     }
73 }
```



PROMINEO TECH

```
1 package com.promineotech.inventoryManagement.controller;
2
3 import java.util.Set;
4
5 import org.springframework.beans.factory.annotation.Autowired;
6 import org.springframework.http.HttpStatus;
7 import org.springframework.http.ResponseEntity;
8 import org.springframework.web.bind.annotation.PathVariable;
9 import org.springframework.web.bind.annotation.RequestBody;
10 import org.springframework.web.bind.annotation.RequestMapping;
11 import org.springframework.web.bind.annotation.RequestMethod;
12 import org.springframework.web.bind.annotation.RestController;
13
14 import com.promineotech.inventoryManagement.entity.Order;
15 import com.promineotech.inventoryManagement.service.OrderService;
16 import com.promineotech.inventoryManagement.util.OrderStatus;
17
18 @RestController
19 @RequestMapping("customers/{id}/orders")
20 public class OrderController {
21
22     @Autowired
23     private OrderService service;
24
25     @RequestMapping(method = RequestMethod.POST)
26     public ResponseEntity<Object> createCustomer(@RequestBody Set<Long> productIds, @PathVariable Long id) {
27         try {
28             return new ResponseEntity<Object>(service.submitNewOrder(productIds, id), HttpStatus.CREATED);
29         } catch (Exception e) {
30             return new ResponseEntity<Object>(e, HttpStatus.BAD_REQUEST);
31         }
32     }
33
34     @RequestMapping(value = "/{orderId}", method = RequestMethod.PUT)
35     public ResponseEntity<Object> updateOrder(@RequestBody Order order, @PathVariable Long orderId) {
36         try {
37
38 package com.promineotech.inventoryManagement.controller;
39
40 import org.springframework.beans.factory.annotation.Autowired;
41 import org.springframework.http.HttpStatus;
42 import org.springframework.http.ResponseEntity;
43 import org.springframework.web.bind.annotation.PathVariable;
44 import org.springframework.web.bind.annotation.RequestBody;
45 import org.springframework.web.bind.annotation.RequestMapping;
46 import org.springframework.web.bind.annotation.RequestMethod;
47 import org.springframework.web.bind.annotation.RestController;
48
49 import com.promineotech.inventoryManagement.entity.Product;
50 import com.promineotech.inventoryManagement.service.ProductService;
51
52 @RestController
53 @RequestMapping("/products")
54 public class ProductController {
55
56     @Autowired
57     private ProductService service;
58
59     @RequestMapping(method = RequestMethod.GET)
60     public ResponseEntity<Object> getProducts() {
61         return new ResponseEntity<Object>(service.getProducts(), HttpStatus.OK);
62     }
63
64     @RequestMapping(method = RequestMethod.POST)
65     public ResponseEntity<Object> createProducts(@RequestBody Product product) {
66         return new ResponseEntity<Object>(service.createProduct(product), HttpStatus.CREATED);
67     }
68
69     @RequestMapping(value = "/{id}", method = RequestMethod.PUT)
70     public ResponseEntity<Object> updateProduct(@RequestBody Product product, @PathVariable Long id) {
71         try {
72             return new ResponseEntity<Object>(service.updateProduct(product, id), HttpStatus.OK);
73         } catch (Exception e) {
74             return new ResponseEntity<Object>("Unable to update product. ", HttpStatus.BAD_REQUEST);
75         }
76     }
77
78     @RequestMapping(value = "/{id}", method = RequestMethod.DELETE)
79     public ResponseEntity<Object> deleteProduct(@PathVariable Long id) {
80         try {
81             service.removeProduct(id);
82             return new ResponseEntity<Object>("Successfully deleted product with id: " + id, HttpStatus.OK);
83         } catch (Exception e) {
84             return new ResponseEntity<Object>("Unable to delete product.", HttpStatus.BAD_REQUEST);
85         }
86     }
87 }
88
89 }
```



PROMINEO TECH

```
3 import javax.persistence.Entity;
4 import javax.persistence.GeneratedValue;
5 import javax.persistence.GenerationType;
6 import javax.persistence.Id;
7 import javax.persistence.OneToOne;
8
9 import com.fasterxml.jackson.annotation.JsonIgnore;
10
11 @Entity
12 public class Address {
13
14     private Long id;
15     private String street;
16     private String city;
17     private String state;
18     private String zip;
19
20     @JsonIgnore
21     private Customer customer;
22
23     @Id
24     @GeneratedValue(strategy = GenerationType.AUTO)
25     public Long getId() {
26         return id;
27     }
28
29     public void setId(Long id) {
30         this.id = id;
31     }
32
33     public String getStreet() {
34         return street;
35     }
36
37     public void setStreet(String street) {
38         this.street = street;
39     }
40
41     public String getCity() {
42         return city;
43     }
44
45     public void setCity(String city) {
46         this.city = city;
47     }
48
49     public String getState() {
50         return state;
51     }
52
53     public void setState(String state) {
54         this.state = state;
55     }
56
57     public String getZip() {
58         return zip;
59     }
60
```

```
60
61     public void setZip(String zip) {
62         this.zip = zip;
63     }
64
65     @OneToOne(mappedBy = "address")
66     public Customer getCustomer() {
67         return customer;
68     }
69
70     public void setCustomer(Customer customer) {
71         this.customer = customer;
72     }
73 }
```

```

3 import java.util.Set;
4
5 import javax.persistence.CascadeType;
6 import javax.persistence.Entity;
7 import javax.persistence.GeneratedValue;
8 import javax.persistence.GenerationType;
9 import javax.persistence.Id;
10 import javax.persistence.JoinColumn;
11 import javax.persistence.OneToMany;
12 import javax.persistence.OneToOne;
13
14 import com.promineotech.inventoryManagement.util.MembershipLevel;
15
16 @Entity
17 public class Customer {
18
19     private Long id;
20     private String firstName;
21     private String lastName;
22     private Address address;
23     private MembershipLevel level;
24     private Set<Order> orders;
25
26     @Id
27     @GeneratedValue(strategy = GenerationType.AUTO)
28     public Long getId() {
29         return id;
30     }
31
32     public void setId(Long id) {
33         this.id = id;
34     }
35
36     public String getFirstName() {
37         return firstName;
38     }
39
40     public void setFirstName(String firstName) {
41         this.firstName = firstName;
42     }
43
44     public String getLastName() {
45         return lastName;
46     }
47
48     public void setLastName(String lastName) {
49         this.lastName = lastName;
50     }
51
52     @OneToOne(cascade = CascadeType.ALL)
53     @JoinColumn(name = "id")
54     public Address getAddress() {
55         return address;
56     }
57
58     public void setAddress(Address address) {
59         this.address = address;
60
61
62     public MembershipLevel getLevel() {
63         return level;
64     }
65
66     public void setLevel(MembershipLevel level) {
67         this.level = level;
68     }
69
70     @OneToMany(mappedBy = "customer")
71     public Set<Order> getOrders() {
72         return orders;
73     }
74
75     public void setOrders(Set<Order> orders) {
76         this.orders = orders;
77     }
78
79 }

```



PROMINEO TECH

```
3 import java.time.LocalDate;
4 import java.util.Set;
5
6 import javax.persistence.Entity;
7 import javax.persistence.GeneratedValue;
8 import javax.persistence.GenerationType;
9 import javax.persistence.Id;
10 import javax.persistence.JoinColumn;
11 import javax.persistence.ManyToMany;
12 import javax.persistence.ManyToOne;
13
14 import com.fasterxml.jackson.annotation.JsonIgnore;
15 import com.promineotech.inventoryManagement.util.OrderStatus;
16
17
18 @Entity
19 public class Order {
20
21     private Long id;
22     private LocalDate ordered;
23     private LocalDate estimatedDelivery;
24     private LocalDate delivered;
25     private double invoiceAmount;
26     private OrderStatus status;
27     private Set<Product> products;
28
29     @JsonIgnore
30     private Customer customer;
31
32     @Id
33     @GeneratedValue(strategy = GenerationType.AUTO)
34     public Long getId() {
35         return id;
36     }
37
38     public void setId(Long id) {
39         this.id = id;
40     }
41
42     public LocalDate getOrdered() {
43         return ordered;
44     }
45
46     public void setOrdered(LocalDate ordered) {
47         this.ordered = ordered;
48     }
49
50     public LocalDate getEstimatedDelivery() {
51         return estimatedDelivery;
52     }
53
54     public void setEstimatedDelivery(LocalDate estimatedDelivery) {
55         this.estimatedDelivery = estimatedDelivery;
56     }
57
58     public LocalDate getDelivere() {
59         return delivered;
```




PROMINEO TECH

```
61
62 public void setDelivere(LocalDate deliver) {
63     this.delivered = deliver;
64 }
65
66 public double getInvoiceAmount() {
67     return invoiceAmount;
68 }
69
70 public void setInvoiceAmount(double invoiceAmount) {
71     this.invoiceAmount = invoiceAmount;
72 }
73
74 public OrderStatus getStatus() {
75     return status;
76 }
77
78 public void setStatus(OrderStatus status) {
79     this.status = status;
80 }
81
82 @ManyToMany(mappedBy = "orders")
83 public Set<Product> getProducts() {
84     return products;
85 }
86
87 public void setProducts(Set<Product> products) {
88     this.products = products;
89 }
90
91 @ManyToOne
92 @JoinColumn(name = "customerId")
93 public Customer getCustomer() {
94     return customer;
95 }
96
97 public void setCustomer(Customer customer) {
98     this.customer = customer;
99 }
100 }
101
102
```



PROMINEO TECH

```
3 import java.util.Set;
4
5 import javax.persistence.CascadeType;
6 import javax.persistence.Entity;
7 import javax.persistence.GeneratedValue;
8 import javax.persistence.GenerationType;
9 import javax.persistence.Id;
10 import javax.persistence.JoinColumn;
11 import javax.persistence.JoinTable;
12 import javax.persistence.ManyToMany;
13
14 import com.fasterxml.jackson.annotation.JsonIgnore;
15
16 @Entity
17 public class Product {
18
19     private Long id;
20     private String name;
21     private String description;
22     private double price;
23
24     @JsonIgnore
25     private Set<Order> orders;
26
27     @Id
28     @GeneratedValue(strategy = GenerationType.AUTO)
29     public Long getId() {
30         return id;
31     }
32
33     public void setId(Long id) {
34         this.id = id;
35     }
36
37     public String getName() {
38         return name;
39     }
40
41     public void setName(String name) {
42         this.name = name;
43     }
44
45     public String getDescription() {
46         return description;
47     }
48
49     public void setDescription(String description) {
50         this.description = description;
51     }
52
53     public double getPrice() {
54         return price;
55     }
56
57     public void setPrice(double price) {
58         this.price = price;
59     }
60
61     @ManyToMany(cascade = CascadeType.ALL)
62     @JoinTable(name = "product_order", joinColumns = @JoinColumn(name = "orderId", referencedColumnName = "id"), inverseJoinColumns = @Join
63     public Set<Order> getOrders() {
64         return orders;
65     }
66
67     public void setOrders(Set<Order> orders) {
68         this.orders = orders;
69     }
70
71 }
72
```

```
1 package com.promineotech.inventoryManagement.repository;
2
3 import org.springframework.data.repository.CrudRepository;
4
5 import com.promineotech.inventoryManagement.entity.Address;
6
7 public interface AddressRepository extends CrudRepository<Address, Long> {
8
9 }
10
```




PROMINEO TECH

```
1 package com.promineotech.inventoryManagement.repository;
2
3 import org.springframework.data.repository.CrudRepository;
4
5 import com.promineotech.inventoryManagement.entity.Customer;
6
7 public interface CustomerRepository extends CrudRepository<Customer, Long> {
8
9 }
10
11 package com.promineotech.inventoryManagement.repository;
12
13 import org.springframework.data.repository.CrudRepository;
14
15 import com.promineotech.inventoryManagement.entity.Order;
16
17 public interface OrderRepository extends CrudRepository<Order, Long>{
18
19 }
20
21 package com.promineotech.inventoryManagement.repository;
22
23 import org.springframework.data.repository.CrudRepository;
24
25 import com.promineotech.inventoryManagement.entity.Order;
26
27 public interface OrderRepository extends CrudRepository<Order, Long> {
28
29 }
30
31 package com.promineotech.inventoryManagement.repository;
32
33 import org.springframework.data.repository.CrudRepository;
34
35 import com.promineotech.inventoryManagement.entity.Product;
36
37 public interface ProductRepository extends CrudRepository<Product, Long> {
38
39 }
40
```



PROMINEO TECH

```
3 import org.apache.logging.log4j.LogManager;
4 import org.apache.logging.log4j.Logger;
5
6 import org.springframework.beans.factory.annotation.Autowired;
7 import org.springframework.stereotype.Service;
8
9 import com.promineotech.inventoryManagement.entity.Customer;
10 import com.promineotech.inventoryManagement.repository.CustomerRepository;
11
12 @Service
13 public class CustomerService {
14
15     private static final Logger logger = LogManager.getLogger(CustomerService.class);
16
17     @Autowired
18     private CustomerRepository repo;
19
20     public Customer getCustomerById(Long id) throws Exception {
21         try {
22             return repo.findOne(id);
23         } catch (Exception e) {
24             logger.error("Exception occurred while trying to retrieve customer: " + id, e);
25             throw e;
26         }
27     }
28
29     public Iterable<Customer> getCustomers() {
30         return repo.findAll();
31     }
32
33     public Customer createCustomer(Customer customer) {
34         return repo.save(customer);
35     }
36
37     public Customer updateCustomer(Customer customer, Long id) throws Exception {
38         try {
39             Customer oldCustomer = repo.findOne(id);
40             oldCustomer.setAddress(customer.getAddress());
41             oldCustomer.setFirstName(customer.getFirstName());
42             oldCustomer.setLastName(customer.getLastName());
43             oldCustomer.setLevel(customer.getLevel());
44             return repo.save(oldCustomer);
45         } catch (Exception e) {
46             logger.error("Exception occurred while trying to update customer: " + id, e);
47             throw new Exception("Unable to update customer.");
48         }
49     }
50
51     public void deleteCustomer(Long id) throws Exception {
52         try {
53             repo.delete(id);
54         } catch (Exception e) {
55             logger.error("Exception occurred while trying to delete customer: " + id, e);
56             throw new Exception("Unable to delete customer.");
57         }
58     }
59 }
60 }
```



PROMINEO TECH

```
3 import org.apache.logging.log4j.LogManager;
4 import org.apache.logging.log4j.Logger;
5
6 import org.springframework.beans.factory.annotation.Autowired;
7 import org.springframework.stereotype.Service;
8
9 import com.promineotech.inventoryManagement.entity.Customer;
10 import com.promineotech.inventoryManagement.repository.CustomerRepository;
11
12 @Service
13 public class CustomerService {
14
15     private static final Logger logger = LogManager.getLogger(CustomerService.class);
16
17     @Autowired
18     private CustomerRepository repo;
19
20     public Customer getCustomerById(Long id) throws Exception {
21         try {
22             return repo.findOne(id);
23         } catch (Exception e) {
24             logger.error("Exception occurred while trying to retrieve customer: " + id, e);
25             throw e;
26         }
27     }
28
29     public Iterable<Customer> getCustomers() {
30         return repo.findAll();
31     }
32
33     public Customer createCustomer(Customer customer) {
34         return repo.save(customer);
35     }
36
37     public Customer updateCustomer(Customer customer, Long id) throws Exception {
38         try {
39             Customer oldCustomer = repo.findOne(id);
40             oldCustomer.setAddress(customer.getAddress());
41             oldCustomer.setFirstName(customer.getFirstName());
42             oldCustomer.setLastName(customer.getLastName());
43             oldCustomer.setLevel(customer.getLevel());
44             return repo.save(oldCustomer);
45         } catch (Exception e) {
46             logger.error("Exception occurred while trying to update customer: " + id, e);
47             throw new Exception("Unable to update customer.");
48         }
49     }
50
51     public void deleteCustomer(Long id) throws Exception {
52         try {
53             repo.delete(id);
54         } catch (Exception e) {
55             logger.error("Exception occurred while trying to delete customer: " + id, e);
56             throw new Exception("Unable to delete customer.");
57         }
58     }
59
60 }
```



PROMINEO TECH

```
3 import java.time.LocalDate;
4 import java.util.HashSet;
5 import java.util.Set;
6
7 import org.apache.logging.log4j.LogManager;
8 import org.apache.logging.log4j.Logger;
9 import org.springframework.beans.factory.annotation.Autowired;
10 import org.springframework.stereotype.Service;
11
12 import com.promineotech.inventoryManagement.entity.Customer;
13 import com.promineotech.inventoryManagement.entity.Order;
14 import com.promineotech.inventoryManagement.entity.Product;
15 import com.promineotech.inventoryManagement.repository.CustomerRepository;
16 import com.promineotech.inventoryManagement.repository.OrderRepository;
17 import com.promineotech.inventoryManagement.repository.ProductRepository;
18 import com.promineotech.inventoryManagement.util.MembershipLevel;
19 import com.promineotech.inventoryManagement.util.OrderStatus;
20
21 @Service
22 public class OrderService {
23
24     private static final Logger logger = LogManager.getLogger(OrderService.class);
25     private final int DELIVERY_DAYS = 7;
26
27     @Autowired
28     private OrderRepository repo;
29
30     @Autowired
31     private CustomerRepository customerRepo;
32
33     @Autowired
34     private ProductRepository productRepo;
35
36     public Order submitNewOrder(Set<Long> productIds, Long customerId) throws Exception {
37         try {
38             Customer customer = customerRepo.findOne(customerId);
39             Order order = initializeNewOrder(productIds, customer);
40             return repo.save(order);
41         } catch (Exception e) {
42             logger.error("Exception occurred while trying to create new order for customer: " + customerId, e);
43             throw e;
44         }
45     }
46
47     public Order cancelOrder(Long orderId) throws Exception {
48         try {
49             Order order = repo.findOne(orderId);
50             order.setStatus(OrderStatus.CANCELED);
51             return repo.save(order);
52         } catch (Exception e) {
53             logger.error("Exception occurred while trying to cancel order: " + orderId, e);
54             throw new Exception("Unable to update order.");
55         }
56     }
57 }
```



PROMINEO TECH

```
58 public Order completeOrder(Long orderId) throws Exception {
59     try {
60         Order order = repo.findOne(orderId);
61         order.setStatus(OrderStatus.DELIVERED);
62         return repo.save(order);
63     } catch (Exception e) {
64         logger.error("Exception occurred while trying to complete order: " + orderId, e);
65         throw new Exception("Unable to update order.");
66     }
67 }
68
69 private Order initializeNewOrder(Set<Long> productIds, Customer customer) {
70     Order order = new Order();
71     order.setProducts(convertToProductSet(productRepo.findAll(productIds)));
72     order.setOrdered(LocalDate.now());
73     order.setEstimatedDelivery(LocalDate.now().plusDays(DELIVERY_DAYS));
74     order.setCustomer(customer);
75     order.setInvoiceAmount(calculateOrderTotal(order.getProducts(), customer.getLevel()));
76     order.setStatus(OrderStatus.ORDERED);
77     addOrderToProducts(order);
78     return order;
79 }
80
81 private void addOrderToProducts(Order order) {
82     Set<Product> products = order.getProducts();
83     for (Product product : products) {
84         product.getOrders().add(order);
85     }
86 }
87
88 private Set<Product> convertToProductSet(Iterable<Product> iterable) {
89     Set<Product> set = new HashSet<Product>();
90     for (Product product : iterable) {
91         set.add(product);
92     }
93     return set;
94 }
95
96 private double calculateOrderTotal(Set<Product> products, MembershipLevel level) {
97     double total = 0;
98     for (Product product : products) {
99         total += product.getPrice();
100     }
101     return total - total * level.getDiscount();
102 }
103
104 }
105
```



PROMINEO TECH

```
1 package com.promineotech.inventoryManagement.service;
2
3 import org.apache.logging.log4j.LogManager;
4 import org.apache.logging.log4j.Logger;
5 import org.springframework.beans.factory.annotation.Autowired;
6 import org.springframework.stereotype.Service;
7
8 import com.promineotech.inventoryManagement.entity.Product;
9 import com.promineotech.inventoryManagement.repository.ProductRepository;
10
11 @Service
12 public class ProductService {
13
14     private static final Logger logger = LogManager.getLogger(ProductService.class);
15
16     @Autowired
17     private ProductRepository repo;
18
19     public Iterable<Product> getProducts() {
20         return repo.findAll();
21     }
22
23     public Product createProduct(Product product) {
24         return repo.save(product);
25     }
26
27     public Product updateProduct(Product product, Long id) throws Exception {
28         try {
29             Product oldProduct = repo.findOne(id);
30             oldProduct.setDescription(product.getDescription());
31             oldProduct.setName(product.getName());
32             oldProduct.setPrice(product.getPrice());
33             return repo.save(oldProduct);
34         } catch (Exception e) {
35             logger.error("Exception occurred while trying to update product: " + id, e);
36             throw new Exception("Unable to update product.");
37         }
38     }
39
40     public void removeProduct(Long id) throws Exception {
41         try {
42             repo.delete(id);
43         } catch (Exception e) {
44             logger.error("Exception occurred while trying to delete product: " + id, e);
45             throw new Exception("Unable to delete product.");
46         }
47     }
48 }
49
50
```



PROMINEO TECH

```
1 package com.promineotech.inventoryManagement.util;
2
3 public enum MembershipLevel {
4     SILVER(.05), GOLD(.10), DIAMOND(.15), PLATINUM(.20);
5
6     private double discount;
7
8     MembershipLevel(double discount) {
9         this.discount = discount;
10    }
11
12    public double getDiscount() {
13        return discount;
14    }
15 }
16
17 }
```

```
1 package com.promineotech.inventoryManagement.util;
2
3 public enum OrderStatus {
4     ORDERED, DELIVERED, CANCELED;
5
6 }
7
8 }
```

```
1 spring.datasource.url=jdbc:mysql://localhost/inventory_api
2 spring.datasource.username=root
3 spring.datasource.password=root1234
4 spring.datasource.driver-class-name=com.mysql.jdbc.Driver
5
6 spring.jpa.hibernate.ddl-auto=create
7 spring.jpa.show-sql=true
8 spring.jpa.properties.hibernate.dialect=org.hibernate.dialect.MySQL5Dialect
9 spring.jpa.properties.hibernate.globally_quoted_identifiers=true
```




PROMINEO TECH

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <Configuration status="WARN" monitorIntervals="30">
3   <Properties>
4     <Property name="LOG_PATTERN">
5       %d{yyyy-MM-dd HH:mm:ss.SSS} %5p ${hostName}
6       --- [%15.15t] %-40.40c{1.} : %m%n%ex
7     </Property>
8   </Properties>
9   <Appenders>
10    <Console name="ConsoleAppender" target="SYSTEM_OUT"
11      follows="true">
12      <PatternLayout pattern="${LOG_PATTERN}" />
13    </Console>
14    <RollingFile name="FileAppender"
15      fileName="logs/log4j2-example.log"
16      filePattern="logs/log4j2-example-%d{yyyy-MM-dd}-%i.log"
17    <PatternLayout>
18      <Pattern>${LOG_PATTERN}</Pattern>
19    </PatternLayout>
20    <Policies>
21      <SizeBasedTriggeringPolicy size="10MB" />
22    </Policies>
23    <DefaultRolloverStrategy max="10" />
24  </RollingFile>
25 </Appenders>
26 <Loggers>
27   <Logger name="com.example.log4j2example" level="debug"
28     additivity="false">
29     <AppenderRef ref="ConsoleAppender" />
30   </Logger>
31   <Root level="info">
32     <AppenderRef ref="ConsoleAppender" />
33     <AppenderRef ref="FileAppender" />
34   </Root>
35 </Loggers>
36 </Configuration>
```



PROMINEO TECH

Launchpad GET http://localhost:8080/customers + ...

GET http://localhost:8080/customers

Key	Value
-----	-------

Body Cookies Headers (4) Test Results

Pretty Raw Preview Visualize JSON

1

POST http://localhost:8080/customers/

Params Authorization Headers (8) Body Pre-request Script Tests Settings

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

```
1 {
2   "firstName": "Dez",
3   "lastName": "Young",
4   "level": "GOLD"
5 }
```

Body Cookies Headers (4) Test Results

Pretty Raw Preview Visualize JSON

```
1 {
2   "id": 1,
3   "firstName": "Dez",
4   "lastName": "Young",
5   "address": null,
6   "level": "GOLD",
7   "orders": null
8 }
```



PROMINEO TECH

POST

http://localhost:8080/customers/

Params

Authorization

Headers (8)

Body

Pre-request Script

none

form-data

x-www-form-urlencoded

raw

binary

1

2

3

4

5

"firstName": "Nick",

"lastName": "Johnson",

"level": "SILVER"

Untitled Request

GET

http://localhost:8080/customers/

Params

Authorization

Headers (6)

Body

Pre-request Script

Tests

Settings

none

form-data

x-www-form-urlencoded

raw

binary

GraphQL

JS

1

Body

Cookies

Headers (4)

Test Results

Pretty

Raw

Preview

Visualize

JSON

1

2

3

4

5

6

7

8

"id": 2,

"firstName": "Nick",

"lastName": "Johnson",

"address": null,

"level": "SILVER",

"orders": null

Body

Cookies

Headers (4)

Test Results

Pretty

Raw

Preview

Visualize

JSON

1

2

3

4

5

6

7

8

9

10

11

12

13

14

15

16

17

18

{

"id": 1,

"firstName": "Dez",

"lastName": "Young",

"address": null,

"level": "GOLD",

"orders": []

},

{

"id": 2,

"firstName": "Nick",

"lastName": "Johnson",

"address": null,

"level": "SILVER",

"orders": []

}



PROMINEO TECH

Launchpad

Untitled Request

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

Params Authorization Headers (6) Body Pre-requests

none form-data x-www-form-urlencoded raw

1

Untitled Request

GET http://localhost:8080/customers/

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

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

1

Body Cookies Headers (4) Test Results

Pretty Raw Preview Visualize JSON

1 {
2 "id": 1,
3 "firstName": "Dez",
4 "lastName": "Young",
5 "address": null,
6 "level": "GOLD",
7 "orders": []
8 }
9
10

Body Cookies Headers (4) Test Results

Pretty Raw Preview Visualize Text

1 Successfully deleted customer with id: 2



PROMINEO TECH

Launchpad

GET http://localhost:8080/products/ + ...

Untitled Request

GET

http://localhost:8080/products/

Params

Authorization

Headers (6)

Body

Pre-request Script

Tests

Settings

☐ none

☐ form-data

☐ x-www-form-urlencoded

☒ raw

☐ binary

☐ GraphQL

JSON

1

Body

Cookies

Headers (4)

Test Results

Pretty

Raw

Preview

Visualize

JSON

1



PROMINEO TECH

Launchpad POST http://localhost:8080/products/ + ...

Untitled Request

POST http://localhost:8080/products/

Params Authorization Headers (8) Body Pre-request Script Tests Settings

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

```
1 {
2   "name": "Penny Skatboard",
3   "description": "This skateborad is the best skateboard that you can own and it fits in your backpack so you can ride it in college",
4   "price": "99.95"
5 }
6
```

Body Cookies Headers (4) Test Results

Pretty Raw Preview Visualize JSON

```
1 {
2   "id": 1,
3   "name": "Penny Skatboard",
4   "description": "This skateborad is the best skateboard that you can own and it fits in your backpack so you can ride it in college",
5   "price": 99.95
6 }
```

Launchpad PUT http://localhost:8080/products/1 + ...

Untitled Request

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

Params Authorization Headers (8) Body Pre-request Script Tests Settings

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

```
1 {
2   "name": "Penny Skatboard",
3   "description": "This skateborad is the best skateboard that you can own and it fits in your backpack so you can ride it in college",
4   "price": "10000.76"
5 }
6
```

Body Cookies Headers (4) Test Results

Pretty Raw Preview Visualize JSON

```
1 {
2   "id": 1,
3   "name": "Penny Skatboard",
4   "description": "This skateborad is the best skateboard that you can own and it fits in your backpack so you can ride it in college",
5   "price": 10000.76
6 }
```



PROMINEO TECH

```
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 your backpack so you can ride it in college	Penny Skatboard	10000.76

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