Final Project Work

Points possible: 100

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 that were committed to for the given week.	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:

Continue contributing to your final project.

URL to GitHub Repository: https://github.com/danielleyokley/finalApi

Screenshots of Code: Finished up all classes in my remaining packages (util, service and controller)

Screenshots below :

finalProjectApi | MembershipLevel | OrderService.ja | SustainabilityP

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

6 references
public enum SustainabilityProceeds {

    DONATIONFIVEPERCENT(.05),
    DONATIONTENPERCENT(.10),
    DONATIONFIFTEENPERCENT(.15),
    DONATIONTWENTYPERCENT(.20);

    2 references
    private double proceed;

    4 references
    SustainabilityProceeds(double proceed) {
        this.proceed = proceed;
    }

    1 reference
    public double getProceed() {
        return proceed;
    }


}
```

SustainabilityP | OrderStatus.jav | AddressReposito | CustomerService | 11

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

import org.apache.logging.log4j.LogManager;

@Service
4 references
public class CustomerService {

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

    @Autowired
    6 references
    private CustomerRepository repo;

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

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

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

    1 reference
    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.setReviews(customer.getReviews());
            return repo.save(oldCustomer);
        } catch (Exception e) {
            logger.error("Exception occured while trying to update customer: " + id, e);
            throw new Exception("Unable to update customer.");
        }
    }

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

PROMINEO TECH

OrderStatus.jav | AddressReposito | CustomerService | OrderService.ja | 12

```
50          Order order = repo.findOne(orderId);
51          order.setStatus(OrderStatus.CANCELED);
52          return repo.save(order);
53      } catch (Exception e) {
54          logger.error("Exception occured while trying to cancel order: " + orderId, e);
55          throw new Exception("Unable to update order.");
56      }
57  }
58
    1 reference
59  public Order completeOrder(Long orderId) throws Exception {
60      try {
61          Order order = repo.findOne(orderId);
62          order.setStatus(OrderStatus.DELIVERED);
63          return repo.save(order);
64      } catch (Exception e) {
65          logger.error("Exception occured while trying to complete order :" + orderId, e);
66          throw new Exception("Unable to update order.");
67      }
68  }
69
    1 reference
70  private Order initializeNewOrder(Set<Long> productIds, Customer customer) {
71      Order order = new Order();
72      order.setProducts(convertToProductSet(productRepo.findAll(productIds)));
73      order.setOrdered(LocalDate.now());
74      order.setEstimatedDelivery(LocalDate.now().plusDays(DELIVERY_DAYS));
75      order.setCustomer(customer);
76      order.setInvoiceAmount(calculateOrderTotal(order.getProducts(), customer.getProceeds()));
77      order.setStatus(OrderStatus.ORDERED);
78      addOrderToProducts(order);
79      return order;
80  }
    1 reference
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
    1 reference
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
    1 reference
96  private double calculateOrderTotal(Set<Product> products, SustainabilityProceeds proceeds) {
97      double total = 0;
98      for (Product product : products) {
99          total += product.getPrice();
100     }
101     return total + total * proceeds.getProceed();
102 }
103
```

PROMINEO TECH

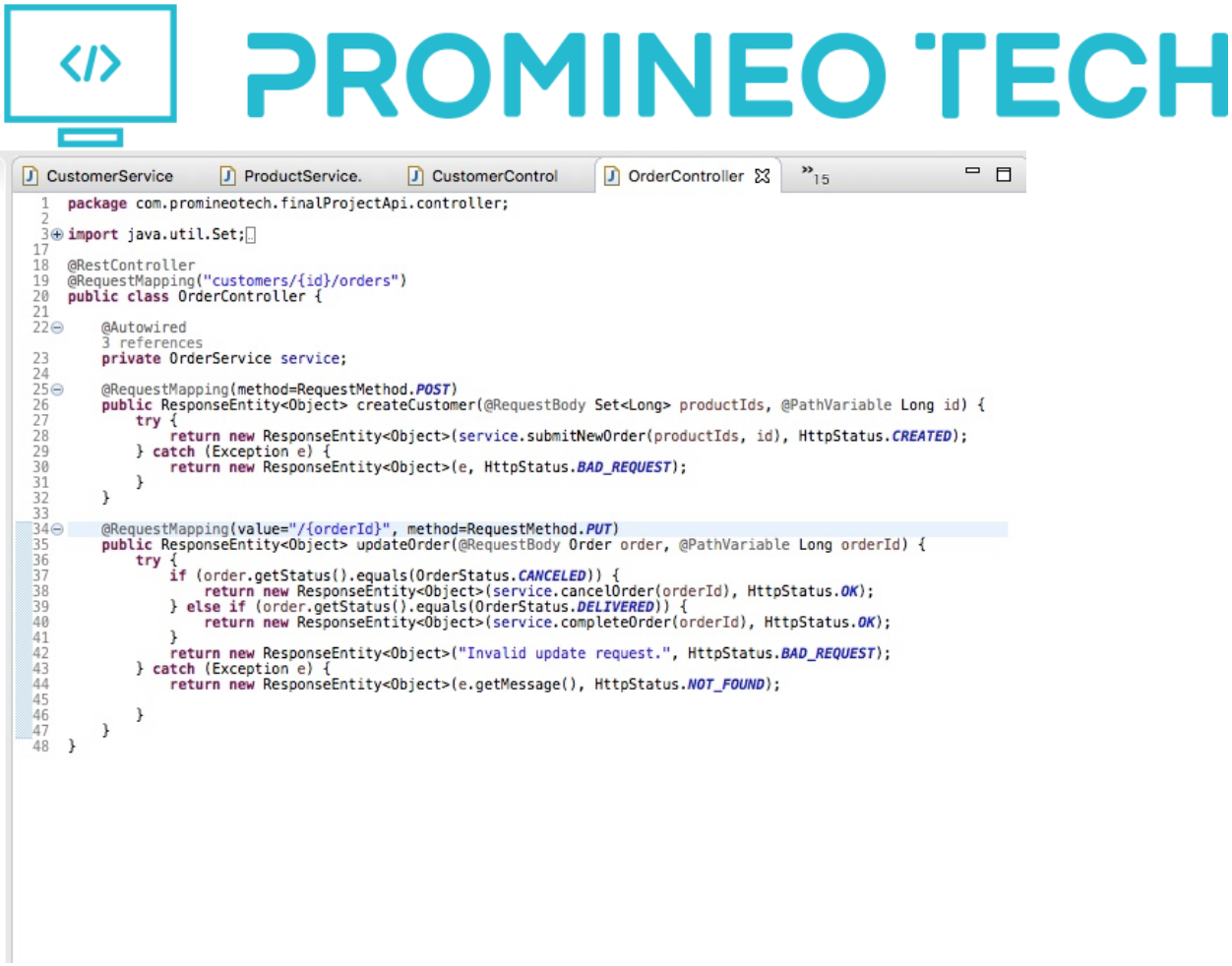
AddressReposito | CustomerService | OrderService.ja | ProductService. | »13

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

import org.apache.logging.log4j.LogManager;

@Service
4 references
public class ProductService {

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

    @Autowired
    5 references
    private ProductRepository repo;

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

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

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

    1 reference
    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 update product.");
        }
    }
}
```

PROMINEO TECH

CustomerService | OrderService.ja | ProductService. | CustomerControl | »14

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

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

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



    @Autowired
    5 references
    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);
        }
    }
}

```


PROMINEO TECH
CustomerService
ProductService.
CustomerControl
OrderController
15
package com.promineotech.finalProjectApi.controller;
import java.util.Set;
@RestController
@RequestMapping("customers/{id}/orders")
public class OrderController {
@Autowired
3 references
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 {
if (order.getStatus().equals(OrderStatus.CANCELED)) {
return new ResponseEntity<Object>(service.cancelOrder(orderId), HttpStatus.OK);
} else if (order.getStatus().equals(OrderStatus.DELIVERED)) {
return new ResponseEntity<Object>(service.completeOrder(orderId), HttpStatus.OK);
}
return new ResponseEntity<Object>("Invalid update request.", HttpStatus.BAD_REQUEST);
} catch (Exception e) {
return new ResponseEntity<Object>(e.getMessage(), HttpStatus.NOT_FOUND);
}
}
}

PROMINEO TECH

CustomerService | CustomerControl | OrderController | ProductControll | 16

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

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

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

    @Autowired
    4 references
    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> createProduct(@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>("Successfully deleted product with id: " + id, HttpStatus.OK
        } catch (Exception e) {
            return new ResponseEntity<Object>("Unable to delete product.", HttpStatus.BAD_REQUEST);
        }
    }
}
```

PROMINEO TECH