



IMS Project - Richard de Young

Presentation



1) - Kanban Board

IMS-Project

Software project

PLANNING

Roadmap

Backlog

Board

DEVELOPMENT

Code

Project pages

Add shortcut

Project settings

You're in a team-managed project

Learn more

Projects / IMS-Project

Roadmap

RY

Status category

Epic

PR

IP-1 Customer Section

IP-11 As a user, I need to be ... TO DO

IP-7 As a user, I need to be a... TO DO

IP-2 As a user, I need to be a... TO DO

IP-8 As a user, I need to be a... TO DO

IP-3 Item Section

IP-12 As a user, I need to be ... TO DO

IP-4 As a user, I need to be a... TO DO

IP-9 As a user, I need to be a... TO DO

IP-10 As a user, I need to be ... TO DO

IP-5 Orders Section

IP-17 As a user, I need to be ... TO DO

IMS-Project

Software project

PLANNING

Roadmap

Backlog

Board

DEVELOPMENT

Code

Project pages

Add shortcut

Project settings

You're in a team-managed project

Learn more

Projects / IMS-Project

IP Sprint 1

To complete project

RY

Epic

TO DO 10 ISSUES

As a user, I need to be able to calculate the cost of an order in order to know how much to pay.

ORDERS SECTION

IP-17

As a user, I need to be able to delete orders in order to keep the database up to date.

ORDERS SECTION

IP-15

As a user, I need to be able to delete items from an order in order to amend an existing order.

IN PROGRESS 4 ISSUES

As a user, I need to be able to view all customers so I can check the status of the database.

CUSTOMER SECTION

IP-11

As a user, I need to be able to update customer details in database in order to keep the database up to date.

CUSTOMER SECTION

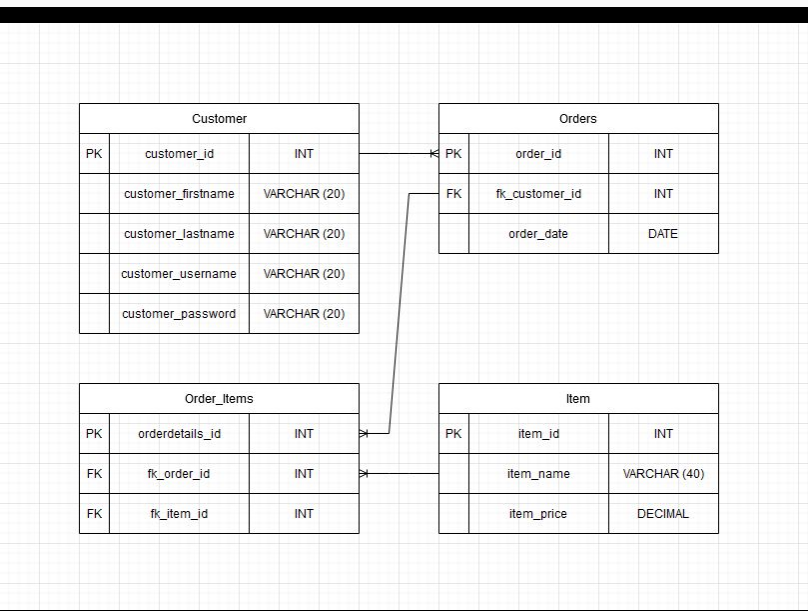
IP-7

As a user, I need to be able to add customers to database so that the database remains up to

DONE



1) - ERD Creation & SQL Table Creation



MySQL Workbench interface showing SQL queries for database and table creation.

Navigation: Schemas, Filter objects, customers, gameshop, ims-project, Tables, Views, Stored Procedures, Functions, improject, imstarterproject, Tables, customer, item, order_items, orders, Views, Stored Procedures, Functions, info, managers, person, skills, sys, Administration, Schemas, Information.

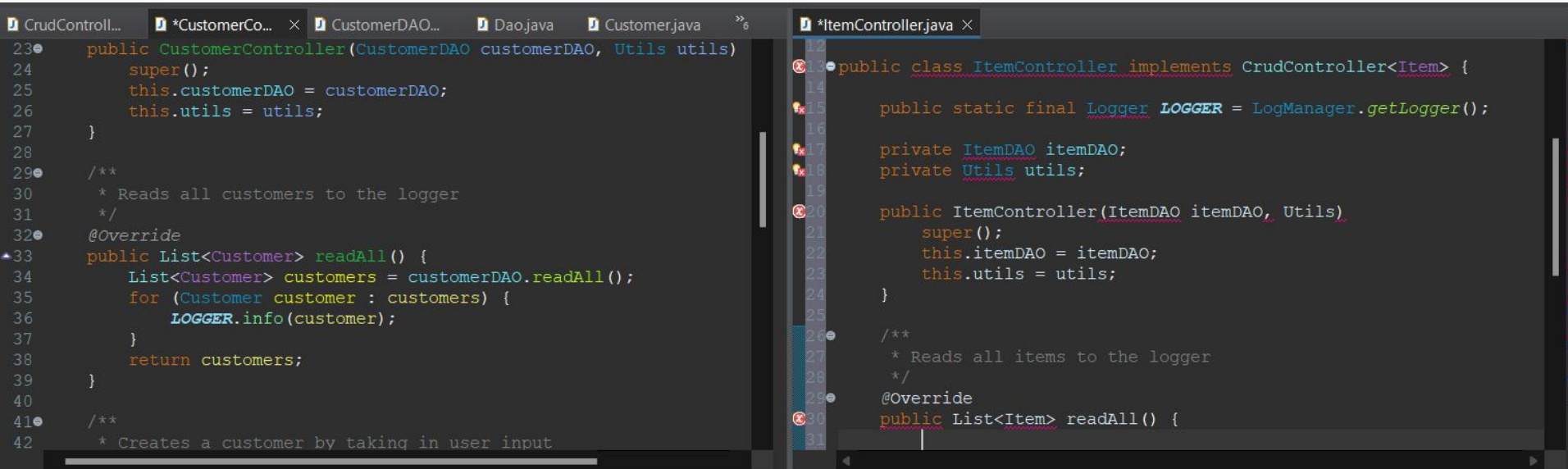
Schema: ims-project

```
1 CREATE DATABASE IF NOT EXISTS imstarterproject;
2 USE imstarterproject;
3
4 DROP TABLE IF EXISTS customers;
5 DROP TABLE IF EXISTS orders;
6 DROP TABLE IF EXISTS item;
7 DROP TABLE IF EXISTS order_items;
8
9 CREATE TABLE IF NOT EXISTS customer (
10     customer_id INT PRIMARY KEY AUTO_INCREMENT,
11     customer_firstname VARCHAR (20) NOT NULL,
12     customer_surname VARCHAR (20) NOT NULL,
13     customer_username VARCHAR (16) UNIQUE NOT NULL,
14     customer_password VARCHAR (20) NOT NULL
15 );
16
17 CREATE TABLE IF NOT EXISTS item (
18     item_id INT PRIMARY KEY AUTO_INCREMENT,
19     item_name VARCHAR (40) NOT NULL,
20     item_price DECIMAL NOT NULL
21 );
22
23 CREATE TABLE IF NOT EXISTS orders (
24     order_id INT PRIMARY KEY AUTO_INCREMENT,
25     fk_customer_id INT NOT NULL,
26     order_date DATE NOT NULL,
27     FOREIGN KEY (fk_customer_id) REFERENCES customer(customer_id),
28     FOREIGN KEY (fk_orderdetails_id) REFERENCES order_items(orderdetails_id)
29 );
30
31 CREATE TABLE IF NOT EXISTS order_items (
32     orderdetails_id INT PRIMARY KEY AUTO_INCREMENT,
33     fk_order_id INT NOT NULL,
34     fk_item_id INT NOT NULL,
35     FOREIGN KEY (fk_order_id) REFERENCES orders(order_id),
36     FOREIGN KEY (fk_item_id) REFERENCES item(item_id)
37 );
```

Action Output:

#	Time	Action	Duration / Fetch
47	10:59:44	CREATE TABLE IF NOT EXISTS item (item_id INT PRIMARY KEY AUTO_INCREMENT, item_name VARCHAR (40) NOT NULL, item_price DECIMAL NOT NULL)	0.016 sec
48	10:59:44	CREATE TABLE IF NOT EXISTS order_items (orderdetails_id INT PRIMARY KEY AUTO_INCREMENT, fk_order_id INT NOT NULL, fk_item_id INT NOT NULL, FOREIGN KEY (fk_order_id) REFERENCES orders(order_id), FOREIGN KEY (fk_item_id) REFERENCES item(item_id))	0.015 sec
49	10:59:44	CREATE TABLE IF NOT EXISTS orders (order_id INT PRIMARY KEY AUTO_INCREMENT, fk_customer_id INT NOT NULL, order_date DATE NOT NULL, FOREIGN KEY (fk_customer_id) REFERENCES customer(customer_id), FOREIGN KEY (fk_orderdetails_id) REFERENCES order_items(orderdetails_id))	0.031 sec

2) - Item Classes Creation



The screenshot shows an IDE with two Java files open. The left pane shows `CustomerController.java` and the right pane shows `ItemController.java`.

```
23 public CustomerController(CustomerDAO customerDAO, Utils utils)
24     super();
25     this.customerDAO = customerDAO;
26     this.utils = utils;
27 }
28
29 /**
30  * Reads all customers to the logger
31  */
32 @Override
33 public List<Customer> readAll() {
34     List<Customer> customers = customerDAO.readAll();
35     for (Customer customer : customers) {
36         LOGGER.info(customer);
37     }
38     return customers;
39 }
40
41 /**
42  * Creates a customer by taking in user input
```

```
12 public class ItemController implements CrudController<Item> {
13
14     public static final Logger LOGGER = LogManager.getLogger();
15
16     private ItemDAO itemDAO;
17     private Utils utils;
18
19     public ItemController(ItemDAO itemDAO, Utils utils)
20     {
21         super();
22         this.itemDAO = itemDAO;
23         this.utils = utils;
24     }
25
26     /**
27      * Reads all items to the logger
28      */
29     @Override
30     public List<Item> readAll() {
31
```

2) - Item Classes Completion

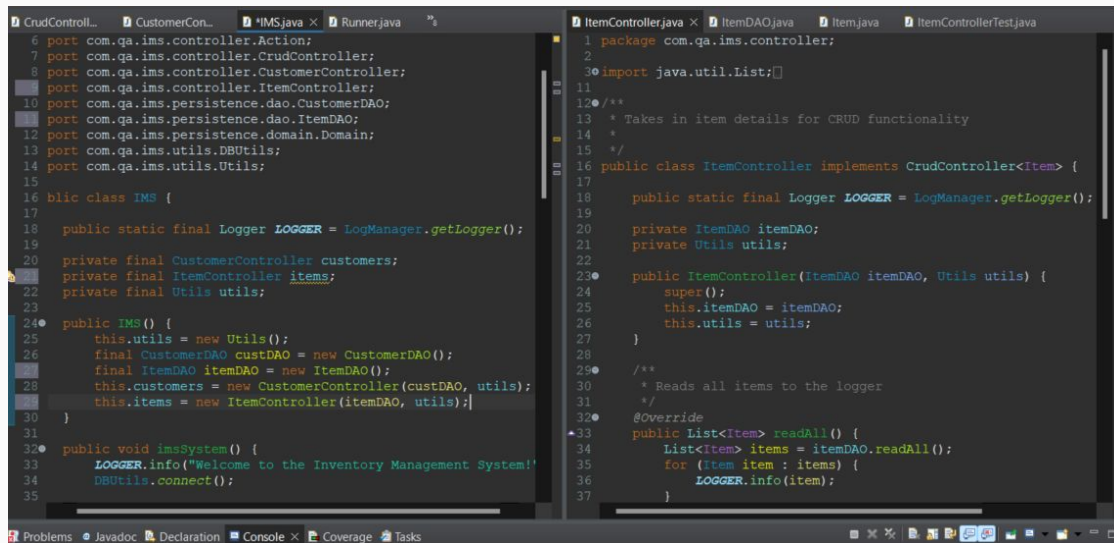
```
MINGW64/c/Users/redy/example/22AprilEnable2/IMS-Starter-master/IMS-Sta...
redy@LAPTOP-T17TIMTH MINGW64 ~/example/22AprilEnable2/IMS-Starter-master/IMS-Sta
rter-master (devbranch)
$ git checkout multiplefeatures
Switched to branch 'multiplefeatures'

redy@LAPTOP-T17TIMTH MINGW64 ~/example/22AprilEnable2/IMS-Starter-master/IMS-Sta
rter-master (multiplefeatures)
$ git merge devbranch
Updating 1866921..0daf4b8
Fast-forward
 src/main/java/com/qa/ims/Runner.java          | 2 +-
 .../java/com/qa/ims/controller/ItemController.java | 81 ++++++++
 .../com/qa/ims/persistence/dao/CustomerDAO.java | 18 +--
 .../java/com/qa/ims/persistence/dao/ItemDAO.java | 143 ++++++++
 .../java/com/qa/ims/persistence/domain/Item.java | 86 ++++++++
 src/main/resources/db.properties              | 4 +-
 src/main/resources/sql-schema.sql              | 8 +-
 .../com/qa/ims/controllers/ItemControllerTest.java | 91 ++++++++
 8 files changed, 417 insertions(+), 16 deletions(-)
 create mode 100644 src/main/java/com/qa/ims/controller/ItemController.java
 create mode 100644 src/main/java/com/qa/ims/persistence/dao/ItemDAO.java
 create mode 100644 src/main/java/com/qa/ims/persistence/domain/Item.java
 create mode 100644 src/test/java/com/qa/ims/controllers/ItemControllerTest.java

redy@LAPTOP-T17TIMTH MINGW64 ~/example/22AprilEnable2/IMS-Starter-master/IMS-Sta
rter-master (multiplefeatures)
$ git branch -M devbranch

redy@LAPTOP-T17TIMTH MINGW64 ~/example/22AprilEnable2/IMS-Starter-master/IMS-Sta
rter-master (devbranch)
$ |
```

3) - Item Controller Issue & Fix



```
6 port com.qa.ims.controller.Action;
7 port com.qa.ims.controller.CrudController;
8 port com.qa.ims.controller.CustomerController;
9 port com.qa.ims.controller.ItemController;
10 port com.qa.ims.persistence.dao.CustomerDAO;
11 port com.qa.ims.persistence.dao.ItemDAO;
12 port com.qa.ims.persistence.domain.Domain;
13 port com.qa.ims.utils.DBUtils;
14 port com.qa.ims.utils.Utils;
15
16 public class IMS {
17
18     public static final Logger LOGGER = LogManager.getLogger();
19
20     private final CustomerController customers;
21     private final ItemController items;
22     private final Utils utils;
23
24     public IMS() {
25         this.utils = new Utils();
26         final CustomerDAO custDAO = new CustomerDAO();
27         final ItemDAO itemDAO = new ItemDAO();
28         this.customers = new CustomerController(custDAO, utils);
29         this.items = new ItemController(itemDAO, utils);
30     }
31
32     public void imsSystem() {
33         LOGGER.info("Welcome to the Inventory Management System!");
34         DBUtils.connect();
35     }
36 }
```

```
1 package com.qa.ims.controller;
2
3 import java.util.List;
4
5 /**
6  * Takes in item details for CRUD functionality
7  */
8 public class ItemController implements CrudController<Item> {
9
10     public static final Logger LOGGER = LogManager.getLogger();
11
12     private ItemDAO itemDAO;
13     private Utils utils;
14
15     public ItemController(ItemDAO itemDAO, Utils utils) {
16         super();
17         this.itemDAO = itemDAO;
18         this.utils = utils;
19     }
20
21     /**
22      * Reads all items to the logger
23      */
24     @Override
25     public List<Item> readAll() {
26         List<Item> items = itemDAO.readAll();
27         for (Item item : items) {
28             LOGGER.info(item);
29         }
30     }
31 }
```

```
redy@LAPTOP-T17TIMTH MINGW64 ~/example/22AprilEnable2/IMS-Starter-master/IMS-Starter-master (itemfeatures)
$ git commit -m "Updated item features to fix bug"
[itemfeatures a185ea8] Updated item features to fix bug
3 files changed, 12 insertions(+), 4 deletions(-)

redy@LAPTOP-T17TIMTH MINGW64 ~/example/22AprilEnable2/IMS-Starter-master/IMS-Starter-master (itemfeatures)
$ git checkout devbranch
Switched to branch 'devbranch'

redy@LAPTOP-T17TIMTH MINGW64 ~/example/22AprilEnable2/IMS-Starter-master/IMS-Starter-master (devbranch)
$ git merge itemfeatures
Updating 0daf4b8..a185ea8
Fast-forward
 src/main/java/com/qa/ims/IMS.java | 6 ++++++
 src/main/java/com/qa/ims/controller/ItemController.java | 2 ++
 src/test/java/com/qa/ims/controllers/CustomerControllerTest.java | 8 ++++++
 3 files changed, 12 insertions(+), 4 deletions(-)

redy@LAPTOP-T17TIMTH MINGW64 ~/example/22AprilEnable2/IMS-Starter-master/IMS-Starter-master (devbranch)
$
```


4) - Order Classes Creation

```
Action.java  CrudControll...  CustomerCon...  CustomerDAO...  Dao.java  Customer.java  Domain.java
46     return "id:" + id + " first name:" + firstName + " surname:" + surname;
47
48
49
50     @Override
51     public int hashCode() {
52         final int prime = 31;
53         int result = 1;
54         result = prime * result + ((firstName == null) ? 0 : firstName.hashCode());
55         result = prime * result + ((id == null) ? 0 : id.hashCode());
56         result = prime * result + ((surname == null) ? 0 : surname.hashCode());
57     }
58
59     @Override
60     public boolean equals(Object obj) {
61         if (this == obj)
62             return true;
63         if (obj == null)
```

```
OrderController.java  Orders.java  Car.java
@Override
public String toString() {
    return "Order id: " + orderId + " Order Date: " + orderDate;
}

@Override
public int hashCode() {
    final int prime = 31;
    int result = 1;
    result = prime * result + ((orderId == null) ? 0 : orderId.hashCode());
    result = prime * result + ((orderDate == null) ? 0 : orderDate.hashCode());
    return result;
}
```

```
Action.java  CrudControll...  CustomerCon...  CustomerDAO...  Dao.java  Customer.java  Domain.java
36     LOGGER.info(customer);
37
38     return customers;
39
40
41     /**
42      * Creates a customer by taking in user input
43      */
44     @Override
45     public Customer create() {
46         LOGGER.info("Please enter a first name");
47         String firstName = utils.getString();
48         LOGGER.info("Please enter a surname");
49         String surname = utils.getString();
50         Customer customer = customerDAO.create(new Customer(firstName, surname));
51         LOGGER.info("Customer created");
52         return customer;
53     }
54
55     /**
56      * Reads all customers
57      */
58     @Override
59     public List<Customer> readAll() {
60         List<Customer> customers = customerDAO.readAll();
61         for (Customer customer : customers) {
62             LOGGER.info(customer);
63         }
64         return customers;
65     }
66
67     /**
68      * Creates an order by taking in user input
69      */
70     @Override
71     public Orders create() {
72         LOGGER.info("Please enter an order id");
73         int orderId = utils.getInt();
74         LOGGER.info("Please enter an order date");
75         Date orderDate = utils.getDate();
76         Orders order = ordersDAO.create(new Orders(orderId, orderDate));
77         LOGGER.info("Order created");
78         return order;
79     }
80
81     /**
82      * Reads all orders
83      */
84     @Override
85     public List<Orders> readAll() {
86         List<Orders> orders = ordersDAO.readAll();
87         for (Orders order : orders) {
88             LOGGER.info(order);
89         }
90         return orders;
91     }
92
93     /**
94      * Updates an order
95      */
96     @Override
97     public Orders update(Orders order) {
98         Orders updatedOrder = ordersDAO.update(order);
99         LOGGER.info("Order updated");
100        return updatedOrder;
101    }
102
103    /**
104     * Deletes an order
105     */
106    @Override
107    public boolean delete(Orders order) {
108        boolean deleted = ordersDAO.delete(order);
109        LOGGER.info("Order deleted");
110        return deleted;
111    }
112}
```

4) - Order Classes Creation (cont.)

```
OrdersController.java x Orders.java OrdersDAO.java
54 • @Override
55 public Orders update() {
56     LOGGER.info("Please enter the id of the order you wish to update");
57     Long orderId = utils.getLong();
58     LOGGER.info("Would you like to add to, or remove an item from an order?");
59     String addOrDelete = utils.getString();
60     if (addOrDelete=="add") {
61         LOGGER.info("Please enter the id of the item you wish to add to the order");
62         Long itemId = utils.getLong();
63         Orders orders = ordersDAO.addItem(new Orders(orderId, itemId));
64         LOGGER.info("Order Updated");
65         return orders;
66     } else if (addOrDelete=="remove") {
67         LOGGER.info("Please enter the id of the item you wish to remove from the order");
68         Long itemId = utils.getLong();
69         Orders orders = ordersDAO.deleteItem(new Orders(orderId, itemId));
70         LOGGER.info("Order updated");
71         return orders;
72     } else {
73         System.out.println("Please type either 'add' or 'remove'");
74     }
75     return ordersDAO.read(orderId);
}
```

```
OrdersController.java Orders.java OrdersDAO.java x
104 //
105 • @Override
106 public Orders update(Orders orders) {
107     try (Connection connection = DBUtils.getInstance().getConnection(); PreparedStatement statement = connection
108         .prepareStatement("INSERT INTO order_items SET fk_item_id = ? WHERE order_id = ?");) {
109         statement.setLong(1, orders.getOrderItems());
110         statement.setLong(2, orders.getOrderID());
111         statement.executeUpdate();
112         return read(orders.getOrderID());
113     } catch (Exception e) {
114         LOGGER.debug(e);
115         LOGGER.error(e.getMessage());
116     }
117     return null;
}
```

```
*OrdersController.java *OrdersDAO.java x Orders.java
142 • public Orders deleteItem(Long orderId, Long itemId) {
143     try (Connection connection = DBUtils.getInstance().getConnection();
144         PreparedStatement statement = connection
145             .prepareStatement("DELETE FROM order_items (fk_order_id, fk_item_id) VALUES (?,?)")) {
146         statement.setLong(1, orderId);
147         statement.setLong(2, itemId);
148         statement.executeUpdate();
149     } catch (Exception e) {
150         LOGGER.debug(e);
151         LOGGER.error(e.getMessage());
152     }
153     return null;
}
```




5) - Showing Order Items when called

```
public List<Item> getItems(Long orderId) {  
    List<Long> itemIds = new ArrayList<>();  
    try (Connection connection = DBUtils.getInstance().getConnection();  
        Statement statement = connection.createStatement();  
        ResultSet resultSet = statement.executeQuery("SELECT * FROM order_items WHERE fk_order_id = " + orderId);) {  
        while (resultSet.next()) {  
            itemIds.add(resultSet.getLong("fk_item_id"));  
        }  
    } catch (Exception e) {  
        LOGGER.debug(e);  
        LOGGER.error(e.getMessage());  
    }  
    List<Item> itemList = new ArrayList<>();  
    for (Long i : itemIds) {  
        itemList.add(itemDAO.read(i));  
    }  
    return itemList;  
}
```

6) - Order Calculator Creation

OrdersController.java *OrdersDAO.java × Orders.java

```
56     LOGGER.debug(e);
57     LOGGER.error(e.getMessage());
58 }
59 List<Item> itemList = new ArrayList<>();
60 for (Long i : itemIds) {
61     itemList.add(itemDAO.read(i));
62 }
63 return itemList;
64 }
65
66 public List<Item> totalPrice(Long orderId) {
67     List<Long> itemPrices = new ArrayList<>();
68     try (Connection connection = DBUtils.getInstance().getConnection();
69         Statement statement = connection.createStatement();
70         ResultSet resultSet = statement.executeQuery("SELECT * FROM order_items WHERE fk_order_id = " + orderId);) {
71         while (resultSet.next()) {
72             itemPrices.add(resultSet.getLong("fk_item_id"));
73             try (Connection connection2 = DBUtils.getInstance().getConnection();
74                 Statement statement2 = connection2.createStatement();
75                 ResultSet resultSet2 = statement2.executeQuery("SELECT * FROM item WHERE item_id = ?");) {
76                 while (resultSet2.next()) {
77                     itemPrices.add(resultSet2.getDouble("item_price"));
78                 }
79             }
80         }
81     }
```

6) - Order Calculator Creation (cont.)

```

OrdersController.java  OrdersDAO.java  *Orders.java X
48         return orderItems;
49     }
50
51     public void setOrderItems(List<Item> orderItems) {
52         this.orderItems = orderItems;
53     }
54
55     @Override
56     public String toString() {
57         double orderPrice = 0;
58         for (Item item : orderItems) {
59             orderPrice += item.getPrice();
60         }
61         return "Orders [orderId=" + orderId + ", customerId=" + customerId + ", orderItems=" + orderItems + "]\nOrder Total: £" + orderPrice;
62     }

```

```

OrdersController.java  OrdersDAO.java  *Orders.java X
54
55     @Override
56     public String toString() {
57         double orderPrice = 0;
58         for (double total : orderItems) {
59             orderPrice += total;
60         }
61         for (double i = 0 ; i < orderItems.size() : i++) {
62             orderPrice += orderItems.get(i);
63         }
64         return "Orders [orderId=" + orderId + ", customerId=" + customerId + ", orderItems=" + orderItems + "]\nOrder Total: £" + orderPrice;
65     }
66
67     @Override
68     public int hashCode() {

```

```

OrdersController.java  OrdersDAO.java  Orders.java X
57         double orderPrice = 0;
58         for (Item i : orderItems)
59             orderPrice += i.getPrice();
60         for (double i = 0 ; i < orderItems.size() : i++) {
61             orderPrice += orderItems.get(i);
62         }
63         return "Orders [orderId=" + orderId + ", customerId=" + customerId + ", orderItems=" + orderItems + "]\nOrder Total: £" + orderPrice;
64     }
65
66     @Override
67     public int hashCode() {
68         return Objects.hash(customerId, orderId, orderItems);
69     }
70

```

```

Runner (18) [Java Application] C:\Users\redy\p2\poo\plugins\org.eclipse.jdt.launcher\org.eclipse.jdt.launcher.exe (11 May 2022, 14:53:32) [pid: 11928]
UPDATE: To change an entity already in the database
DELETE: To remove an entity from the database
RETURN: To return to domain selection
read
Orders [orderId=3, customerId=2, orderItems=[id:1 item name: Hydra Plushie. Item price: £109.99, id:3 item name: Gorgon Plushie. Item price: £25.55]]
Order Total: £135.54
Orders [orderId=4, customerId=3, orderItems=[id:4 item name: Golem Plushie. Item price: £56.55, id:4 item name: Golem Plushie. Item price: £56.55]]
Order Total: £113.1
What would you like to do with order:
CREATE: To save a new entity into the database
READ: To read an entity from the database
UPDATE: To change an entity already in the database
DELETE: To remove an entity from the database
RETURN: To return to domain selection

```

```

MINGW64~/c/Users/redy/example/22AprilEnable2/IMS-Starter-master/IMS-Sta...
.../java/com/qa/ims/persistence/domain/Orders.java | 5 +-
5 files changed, 63 insertions(+), 21 deletions(-)

redy@LAPTOP-T17TINTH MINGW64 ~/example/22AprilEnable2/IMS-Starter-master/IMS-Sta
rter-master (devbranch)
$ git push origin devbranch
Enumerating objects: 50, done.
Counting objects: 100% (50/50), done.
Delta compression using up to 8 threads
Compressing objects: 100% (29/29), done.
Writing objects: 100% (35/35), 5.48 KiB | 934.00 KiB/s, done.
Total 35 (delta 14), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (14/14), completed with 7 local objects.
To https://github.com/RicharddeYoung/IMS-Starter-Project.git
a185ea8..19b981e devbranch -> devbranch

redy@LAPTOP-T17TINTH MINGW64 ~/example/22AprilEnable2/IMS-Starter-master/IMS-Sta
rter-master (devbranch)
$ git checkout -b testingbranch
Switched to a new branch 'testingbranch'

redy@LAPTOP-T17TINTH MINGW64 ~/example/22AprilEnable2/IMS-Starter-master/IMS-Sta
rter-master (testingbranch)
$

```



7) - Test Creation

File Edit Source Refactor Navigate Search Project Run Window Help

Package ExplorerJUnit X

Finished after 4.705 seconds

Runs: 25/25 Errors: 2 Failures: 11

- com.qa.ims.controllers.CustomerController
- com.qa.ims.persistence.domain.OrdersTest
- com.qa.ims.persistence.dao.OrdersDAOTest
 - initializationError (0.006 s)
- com.qa.ims.persistence.domain.Customer
- com.qa.ims.persistence.domain.ItemTest
- com.qa.ims.controllers.ItemControllerTest
- com.qa.ims.persistence.dao.CustomerDAOTest
 - testReadLatest (0.702 s)
 - testRead (0.016 s)
 - testReadAll (0.009 s)
 - testCreate (0.010 s)
 - testDelete (0.011 s)
 - testUpdate (0.014 s)
- com.qa.ims.persistence.dao.ItemDAOTest
 - testReadLatest (0.006 s)
 - testRead (0.016 s)

Failure Trace

org.junit.runners.model.InvalidTestClassError: Invalid class: com.qa.ims.persistence.dao.OrdersDAOTest

1. Method setup() should be public

CustomerDAO.java Customer.java IMS.java Runner.java CustomerCon... ItemDAO.java ItemControll... Item.java CustomerDAO... CustomerTes...

```
59 @Override
60 public Item update() {
61     LOGGER.info("Please enter the id of the item you would like to update");
62     Long id = utils.getLong();
63     LOGGER.info("Please enter a new item name");
64     String itemName = utils.getString();
65     LOGGER.info("Please enter a new price");
66     double itemPrice = utils.getDouble();
67     Item item = itemDAO.update(new Item(id, itemName, itemPrice));
68     LOGGER.info("Item updated");
69     return item;
70 }
```

OrdersController... OrdersDAO.java Orders.java ItemControllerT... OrdersController... ItemDAOTest.java OrdersDAOTest.java ItemTest.java OrdersTest.java

```
10 import org.mockito.InjectMocks;
11 import org.mockito.Mock;
12 import org.mockito.Mockito;
13 import org.mockito.junit.MockitoJUnitRunner;
14
15 import com.qa.ims.controller.OrdersController;
16 import com.qa.ims.persistence.dao.OrdersDAO;
17 import com.qa.ims.persistence.domain.Orders;
18 import com.qa.ims.utils.Utils;
19
20 @RunWith(MockitoJUnitRunner.class)
21 public class OrdersControllerTest {
22
23     @Mock
24     private Utils utils;
25
26     @Mock
27     private OrdersDAO dao;
28
29     @InjectMocks
```

Problems Javadoc Declaration Console Coverage X Tasks

java (12 May 2022 11:32:21)

Element	Coverage	Vered Instructions	Issued Instructions	Total Instructions
src/main/java	43.8 %	1,029	1,321	2,350
src/test/java	70.7 %	575	238	813
com.qa.ims.persistence.dao	36.4 %	119	208	327
OrdersDAOTest.java	0.0 %	0	107	107
CustomerDAOTest.java	50.9 %	56	54	110
ItemDAOTest.java	57.3 %	63	47	110



8) - Testing

```
OrdersController.java OrdersDAO.java Orders.java ItemControllerT... OrdersContro...
37 }
38
39 @Test
40 public void testRead() {
41     final long ID = 1L;
42     assertEquals(new Orders(ID, 2L), DAO.read(ID));
43 }
44
45 @Test
46 public void testAddItem() {
47     final Orders updated = new Orders(1L, 3L);
48     assertEquals(updated, DAO.addItem(0));
49 }
50
51 @Test
52 public void testDeleteItem() {
53     final Orders updated = new Orders(1L, 3L);
54     assertEquals(updated, DAO.deleteItem(updated));
55 }
56 }
```

File Edit Source Refactor Navigate Search Project Run Window Help

Project Explorer JUnit X

Runs: 35/35 Errors: 0 Failures: 7

Finished after 4.957 seconds

- com.qa.ims.controllers.CustomerControllerTest [Runner: JUnit 4] (2.967 s)
- com.qa.ims.persistence.domain.OrdersTest [Runner: JUnit 4] (0.347 s)
- com.qa.ims.persistence.dao.OrdersDAOTest [Runner: JUnit 4] (0.954 s)
 - testGetItems (0.762 s)
 - testAddItem (0.044 s)
 - testRead (0.025 s)
 - testReadAll (0.027 s)
 - testDeleteItem (0.025 s)
 - testCreate (0.022 s)
 - testDelete (0.028 s)
- com.qa.ims.persistence.domain.CustomerTest [Runner: JUnit 4] (0.028 s)
- com.qa.ims.persistence.domain.ItemTest [Runner: JUnit 4] (0.015 s)
- com.qa.ims.controllers.ItemControllerTest [Runner: JUnit 4] (0.080 s)
- com.qa.ims.persistence.dao.CustomerDAOTest [Runner: JUnit 4] (0.150 s)
 - testReadItem (0.022 s)
 - testReadAll (0.030 s)
 - testReadAll (0.003 s)
 - testCreate (0.029 s)

Failure Trace

```
java.lang.AssertionError: expected: <Orders [orderId=1, customerId=1, orderItems=[]]>
Order Total: £0.0> but was: <Orders [orderId=1, customerId=1, orderItems=[id:1 item name
Order Total: £219.0>
at com.qa.ims.persistence.dao.OrdersDAOTest.testAddItem(OrdersDAOTest.java:56)
```

Runner.java Item.java OrdersDAO.java x sql-schema.sql sql-data.sql sql-schema.sql CustomerDAOT...

```
144 public Orders addItem(Long orderId, Long itemId) {
145     try {
146         Connection connection = DBUtils.getInstance().getConnection();
147         PreparedStatement statement = connection
148             .prepareStatement("INSERT INTO order_items (fk_order_id, fk_item_id) VALUES (?, ?)");
149         statement.setLong(1, orderId);
150         statement.setLong(2, itemId);
151         statement.executeUpdate();
152     } catch (Exception e) {
153         LOGGER.debug(e);
154         LOGGER.error(e.getMessage());
155     }
156     return read(orderId);
157 }
158 /**
```

OrdersContro... OrdersDAO.java ItemControll... CustomerDAO... OrdersContro... ItemDAOTest... OrdersDAOTe...

```
49 }
50 assertEquals(new Orders(ID, 1L), DAO.read(ID));
51 }
52
53 @Test
54 public void testAddItem() {
55     final Orders updated = new Orders(1L, 1L);
56     assertEquals(updated, DAO.addItem(1L, 1L));
57 }
58
59 @Test
```

Console Problems Debug Shell Coverage x

java (12 May 2022 15:41:38)

Element	Coverage	Veried Instructions	Issed Instructions	Total Instructions
com.qa.ims.persistence.dao	76.1 %	642	202	844
OrdersDAO.java	77.5 %	268	78	346
CustomerDAO.java	75.1 %	187	62	249
ItemDAO.java	75.1 %	187	62	249
com.qa.ims	0.0 %	0	166	166
com.qa.ims.controller	73.0 %	343	127	470
com.qa.ims.persistence.domain	81.4 %	506	116	622
com.qa.ims.utils	65.7 %	163	85	248
com.qa.ims.exceptions	0.0 %	0	3	3
src/test/java	90.1 %	1,005	110	1,115

8) - Testing (cont.)

```

sql-schema.sql x
1 DROP TABLE IF EXISTS `order_items`;
2 DROP TABLE IF EXISTS `orders`;
3 DROP TABLE IF EXISTS `item`;
4 DROP TABLE IF EXISTS `customer`;
5
6
7 CREATE TABLE IF NOT EXISTS `customer` (
8   `customer_id` INT PRIMARY KEY AUTO INCREMENT,
9   `customer_firstname` VARCHAR (20) NOT NULL,
10  `customer_surname` VARCHAR (20) NOT NULL
11 );
12
13 CREATE TABLE IF NOT EXISTS `item` (
14   `item_id` INT PRIMARY KEY AUTO INCREMENT,
15   `item_name` VARCHAR (30) NOT NULL,
16   `item_price` DOUBLE NOT NULL
17 );
18
19 CREATE TABLE IF NOT EXISTS `orders` (
20   `order_id` INT PRIMARY KEY AUTO INCREMENT,
21   `fk_customer_id` INT NOT NULL,
22   FOREIGN KEY (`fk_customer_id`) REFERENCES `customer`
23 );
24
25 CREATE TABLE IF NOT EXISTS `order_items` (
26   `orderdetails_id` INT PRIMARY KEY AUTO INCREMENT,
27   `fk_order_id` INT NOT NULL,
28   `fk_item_id` INT NOT NULL,
29   FOREIGN KEY (`fk_order_id`) REFERENCES `orders`(`order_id`),
30   FOREIGN KEY (`fk_item_id`) REFERENCES `item`(`item_id`)
31 );

sql-data.sql x
1 INSERT INTO `customer` (`customer_firstname`, `customer_surname`) VALUES ('David', 'Davidson');
2
3 INSERT INTO `item` (`item_name`, `item_price`) VALUES ('Dragon Plushie', 109.50);
4
5 INSERT INTO `orders` (`fk_customer_id`) VALUES (1);
6
7 INSERT INTO `order_items` (`fk_order_id`, `fk_item_id`) VALUES (1, 1);

MINGW64/c:/Users/rdy/example/22AprilEnable2/IMS-Starter-master/IMS-Starter-master (devbranch)
src/test/resources/db.properties 2 +-
src/test/resources/sql-data.sql 8 +-
src/test/resources/sql-schema.sql 39 +++++
14 files changed, 365 insertions(+), 28 deletions(-)
create mode 100644 src/test/java/com/qa/ims/controllers/OrdersControllerTest.java
create mode 100644 src/test/java/com/qa/ims/persistence/dao/ItemDAOTest.java
create mode 100644 src/test/java/com/qa/ims/persistence/dao/OrdersDAOTest.java
create mode 100644 src/test/java/com/qa/ims/persistence/domain/ItemTest.java
create mode 100644 src/test/java/com/qa/ims/persistence/domain/OrdersTest.java

rdy@LAPTOP-T17TIMH MINGW64 ~/example/22AprilEnable2/IMS-Starter-master/IMS-Starter-master (devbranch)
$ git push origin devbranch
Enumerating objects: 59, done.
Counting objects: 100% (59/59), done.
Delta compression using up to 8 threads
Compressing objects: 100% (29/29), done.
Writing objects: 100% (35/35), 4.56 KiB | 1.14 MiB/s, done.
Total 35 (delta 12), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (12/12), completed with 7 local objects.
To https://github.com/RicharddeYoung/IMS-Starter-Project.git
19b981e..da73281 devbranch -> devbranch

```

```

Customer.java x CustomerCon... x ItemControll... x Item.java x Orders.java x ItemDAO.java x
34 final String F_NAME = "David", L_NAME = "Davidson";
35 final Customer created = new Customer(F_NAME, L_NAME);
36
37 Mockito.when(utils.getString()).thenReturn(F_NAME, L_NAME);
38 Mockito.when(dao.create(created)).thenReturn(created);
39
40 assertEquals(created, controller.create());
41
42 Mockito.verify(utils, Mockito.times(2)).getString();
43 Mockito.verify(dao, Mockito.times(1)).create(created);
44 }
45
46 @Test
47 public void testReadAll() {
48   List<Customer> customers = new ArrayList<>();
49   customers.add(new Customer("Susan", "Susannson"));
50
51 Mockito.when(dao.readAll()).thenReturn(customers);
52
53 ItemControllerTest.java x CustomerDAO.java x OrdersControllerTest.java x ItemDAOTest.java x Orders
36 final Item created = new Item(F_NAME, I_PRICE);
37
38 Mockito.when(utils.getString()).thenReturn(F_NAME);
39 Mockito.when(utils.getDouble()).thenReturn(I_PRICE);
40 Mockito.when(dao.create(created)).thenReturn(created);
41
42 assertEquals(created, controller.create());
43
44 Mockito.verify(utils, Mockito.times(1)).getString();
45 Mockito.verify(utils, Mockito.times(1)).getDouble();
46 Mockito.verify(dao, Mockito.times(1)).create(created);
47 }
48

```

Console Problems Debug Shell Coverage x

java (12 May 2022 20:32:00)

Element	Coverage	Veried Instructions	Issued Instructions	Total Instructions
IMS-Starter-master	80.4 %	2,884	705	3,589
src/main/java	70.0 %	1,648	705	2,353
src/test/java	100.0 %	1,236	0	1,236

Writable Smart

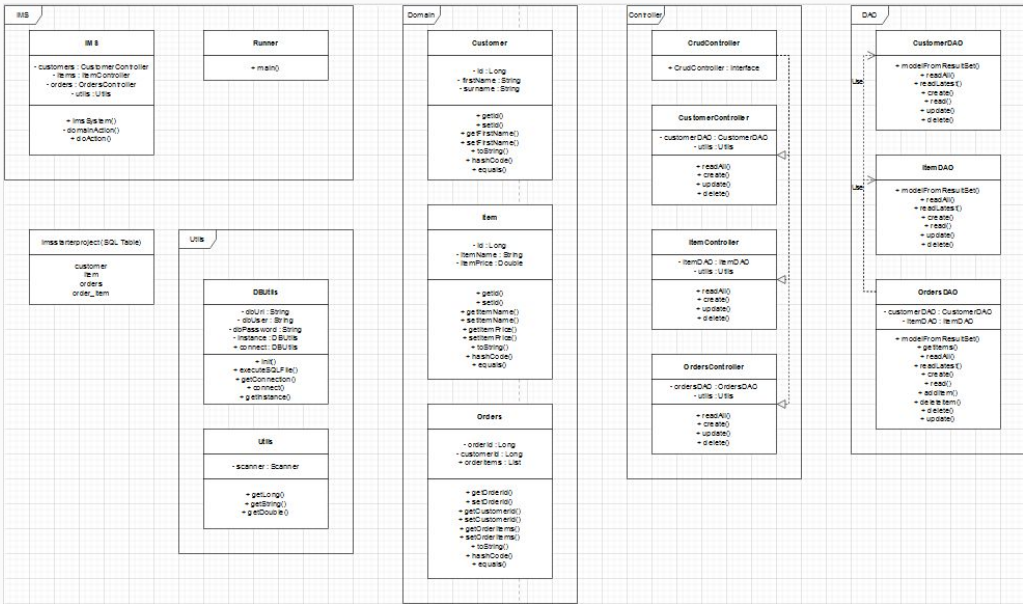


9) - Unimplemented Methods

addItem update: allow user to specify amount of item they wished to add to an order.

Item search: allow user to search for orders which contain a specific item.

UML attempt





10) - Conclusion/Demonstration

Thank you for watching!