Inventory Management System- Project

Katie McCalla

12/02/2021

Introduction & Consultant Journey

- The technologies used for this product include:
- **Version Control System**: Git

Git Bash

- **Source Code Management:**

GitHub (https://github.com/k-mccalla/Project-work)

- **Kanban Board**:

Jira

- **Database Management System**: SQL

MySQL Server 5.7: MySQL Workbench 8.0

- **Back-End Programming Language**: Java

Eclipse version 4.18.0

- **Build Tool:** Maven
- **Unit Testing:** Junit
- **Code Quality testing:** SonarCube

Specification

Specification asks for:

- Code fully integrated into a Version Control System using the feature-branch model: **master/dev/multiple features**.
- A project management board with full expansion on user stories, acceptance criteria and tasks needed to complete the project.
- A risk assessment which outlines the issues and risks faced during the project timeframe.
- A relational database used to persist data for the project, containing the **customers, products, orders, and orders_items** tables. Relationships should be modelled using an ERD.
- A functional application ‘back-end’, following best practices and design principles, in the language that you have covered during training, meeting the requirements set on your project management board.
- A build of your application, including any dependencies it might need, produced using an integrated build tool.
- Unit tests for validation of the application. You should aim to reach the industry standard of **80%** test coverage.

Requirements of project

- Followed these requirements to create prioritization documents, epics and user stories on Jira.

Domain

You are required to build an application that an end user can interact with via a Command-Line Interface (CLI). The application required is an inventory management system, that needs to be able to:

- Add a **customer** to the system
- View all **customers** in the system
- Update a **customer** in the system
- Delete a **customer** in the system
- Add an **item** to the system
- View all **items** in the system
- Update an **item** in the system
- Delete an **item** in the system
- Create an **order** in the system
- View all **orders** in the system
- Delete an **order** in the system
- Add an **item** to an **order**
- Calculate a cost for an **order**
- Delete an **item** in an **order**

When considering the entities in this domain:

- A **customer** needs to have a name
- An **item** needs to have a name and a value
- An **order** needs to have a **customer** and contains **items**
- You will need to create an intermediary **orders_items** table to handle the many-to-many relationship, as MySQL does not natively support this type of relationship.

Jira Board- MoSCow Priortisation

MoSCoW Prioritization.

Must have

- Ability to add customers into the system (including their basic details)
- Customers must have First and last names as well as unique customer IDs.
- Must be able to update customer details.
- Must be able to add items to the system.
- All items must have names, prices and unique item IDs.
- Must be able to add orders to the system.
- All orders must have unique order IDs.
- Must be able to delete customers, items and orders from the system.
- The ability to update customer details.
- The ability to update item details.
- The ability to update orders.
- Create, read, update, delete (CRUD) Functionality.

Should have

- Adding one or many items to an order.
- Removing one or many items in an order.
- Ability to calculate total costs of an order.

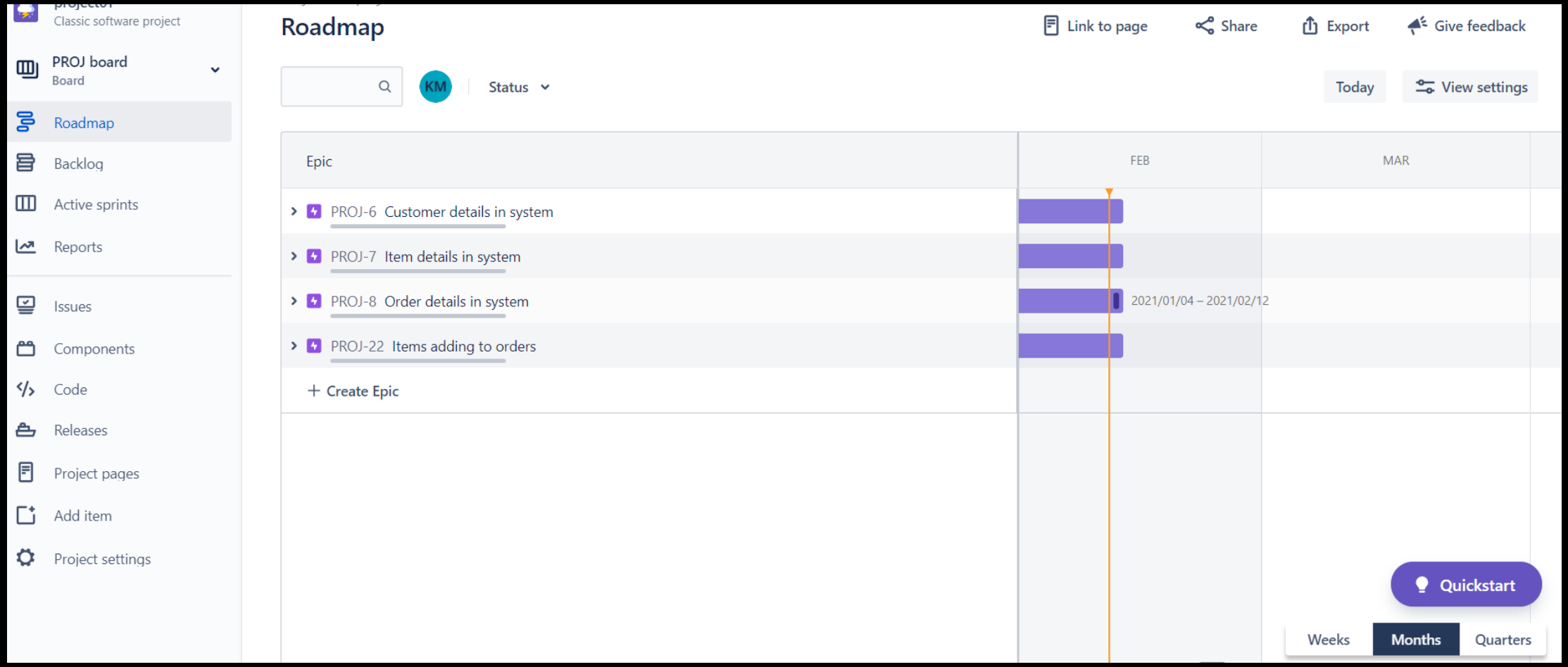
Could have

- Additional customer identification information (for example: address, postcode, date of birth)
- Additional item information (for example: weight, quantity in stock, category of item)
- Additional order information (for example: date order was placed, whether order is refundable).

Will not have

- Sensitive customer information (for example: payment details, log in details)

Jira Board- Epics



Jira Board- User stories

- For example <u>“As a user I must be able to add customers into the system“</u>
- Written with the perspective of the system user.
- What they want to achieve and what information they wish to add and view in the system.
- They were updated during the course of the project as user needs were evaluated.

User stories- Customer and Item

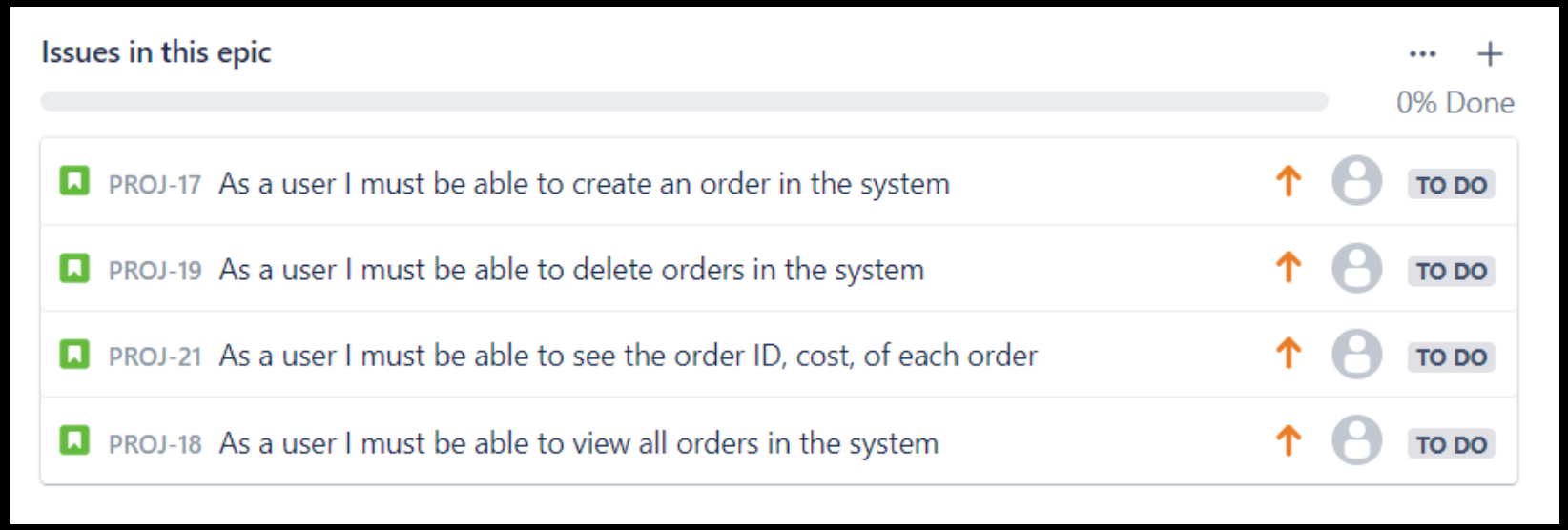
Epic	FEB
PROJ-6 Customer details in system	
PROJ-9 As a user I must be able to add customers to the system	TO DO
PROJ-10 As a user I must be able to view all customers in the system	TO DO
PROJ-27 As a user I must be able to add customer details to the system including name, customer ID,	TO DO
PROJ-11 As a user I must be able to delete customers in the system	TO DO
PROJ-12 As a user I must be able to update customer details in the system	TO DO
PROJ-7 Item details in system	
PROJ-13 As a user I must be able to add items to the system	TO DO
PROJ-28 As a user I must be able to view the item name, cost and item ID	TO DO
PROJ-14 As a user I must be able to view all items in the system	TO DO
PROJ-15 As a user I must be able to delete items from the system	TO DO
PROJ-16 As I user I must be able to update item details in the system	TO DO

User stories- Order and Items

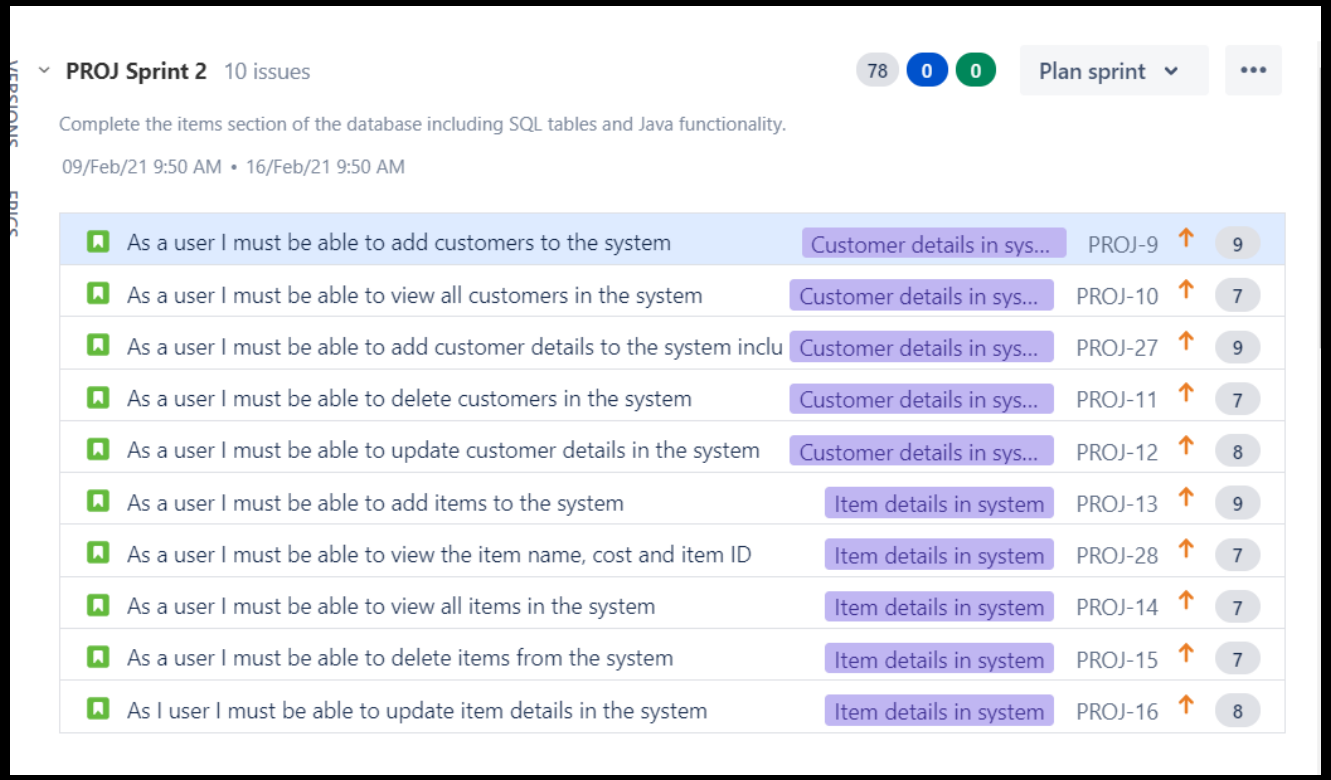
PROJ-8 Order details in system	
PROJ-17 As a user I must be able to create an order in the system	TO DO
PROJ-19 As a user I must be able to delete orders in the system	TO DO
PROJ-21 As a user I must be able to see the order ID, cost, of each order	TO DO
PROJ-18 As a user I must be able to view all orders in the system	TO DO
PROJ-22 Items adding to orders	
PROJ-23 As a user I must be able to add an item to an order	TO DO
PROJ-25 As a user I must be able to delete an item from an order	TO DO
PROJ-24 As a user I must be able to calculate the cost of an order	TO DO
PROJ-26 As a user I must be able to cancel an order	TO DO

User stories updates.

- Updated user stories for what information about an order a user should see.



Jira Board- Sprints



PROJ Sprint 2 10 issues

78 0 0 Plan sprint

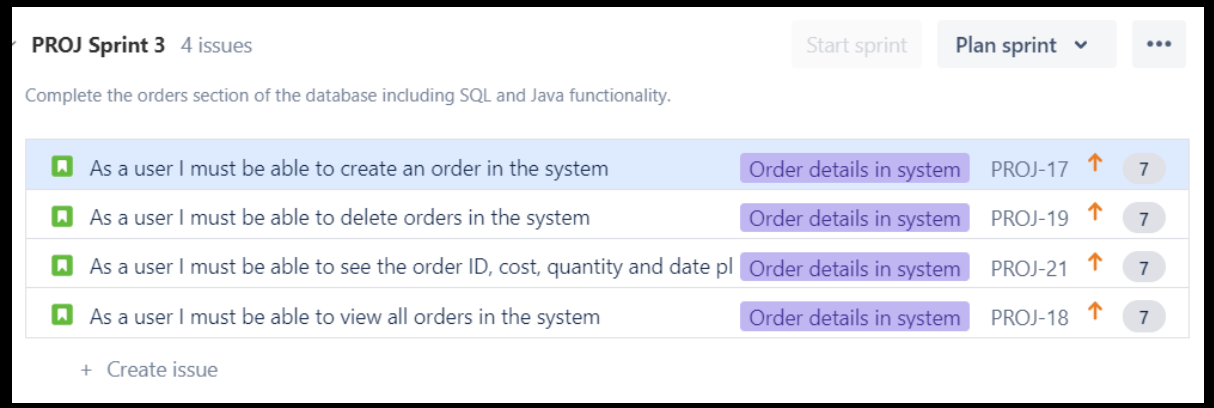
Complete the items section of the database including SQL tables and Java functionality.

09/Feb/21 9:50 AM • 16/Feb/21 9:50 AM

As a user I must be able to add customers to the system	Customer details in sys...	PROJ-9	9
As a user I must be able to view all customers in the system	Customer details in sys...	PROJ-10	7
As a user I must be able to add customer details to the system inclu	Customer details in sys...	PROJ-27	9
As a user I must be able to delete customers in the system	Customer details in sys...	PROJ-11	7
As a user I must be able to update customer details in the system	Customer details in sys...	PROJ-12	8
As a user I must be able to add items to the system	Item details in system	PROJ-13	9
As a user I must be able to view the item name, cost and item ID	Item details in system	PROJ-28	7
As a user I must be able to view all items in the system	Item details in system	PROJ-14	7
As a user I must be able to delete items from the system	Item details in system	PROJ-15	7
As I user I must be able to update item details in the system	Item details in system	PROJ-16	8

- Sprint 1
- Aim was to complete the Customer and item details functionality. This included creating, reading, updating and deleting details.
- Aim was to write the SQL tables and create the Java functionality.
- Assigned story points based on difficulty and necessity to complete
- Adding customers and adding items were assigned the most story points after consideration.

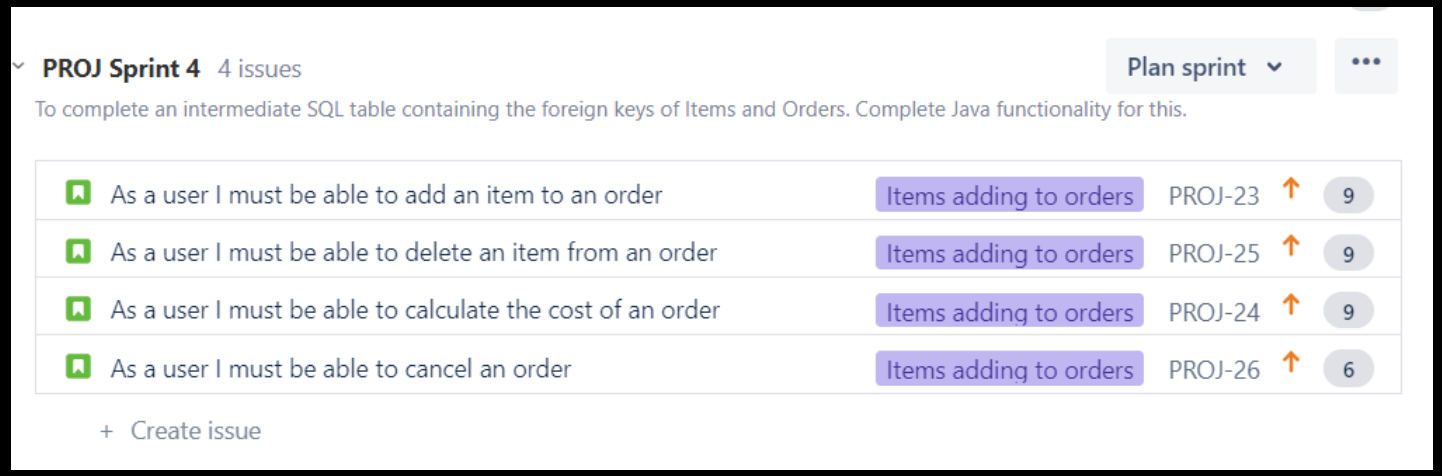
Jira Board- Sprints



- Sprint 2

- Involved focusing on orders. This included all CRUD functionality and the creation of an SQL table.
- Assigned story points and predicted that this would be a faster sprint than the first sprint.

Jira Board- Sprints



- Sprint 3
- Involved creating an intermediate SQL table (order_items) with foreign keys for both items and orders tables. For many to many relationships.
- Implementing this into the orders class (in java).
- Updating the OrderDAO and CRUD controller methods.
- Predicted that this sprint would take the longest and be the most challenging.

Risk assessment

- Was conducted prior to the start of the project on 29/01/2021.
- Considered the **risk, where and who it will impact, action taken, who is responsible, the risk level and the likelihood of it occurring.**
- Quantified the risk level using a risk matrix. With values from 1-9 from lowest to highest risk severity and likelihood of it occurring.
- Ensured that all risks were considered and an action was taken for each.

Risk assessment

- Risk assessment doc.pdf

Risk assessment- Inventory management system

Company name: QA

Assessment carried out by: Katie McCalla

Date assessment was carried out: 29/01/2021

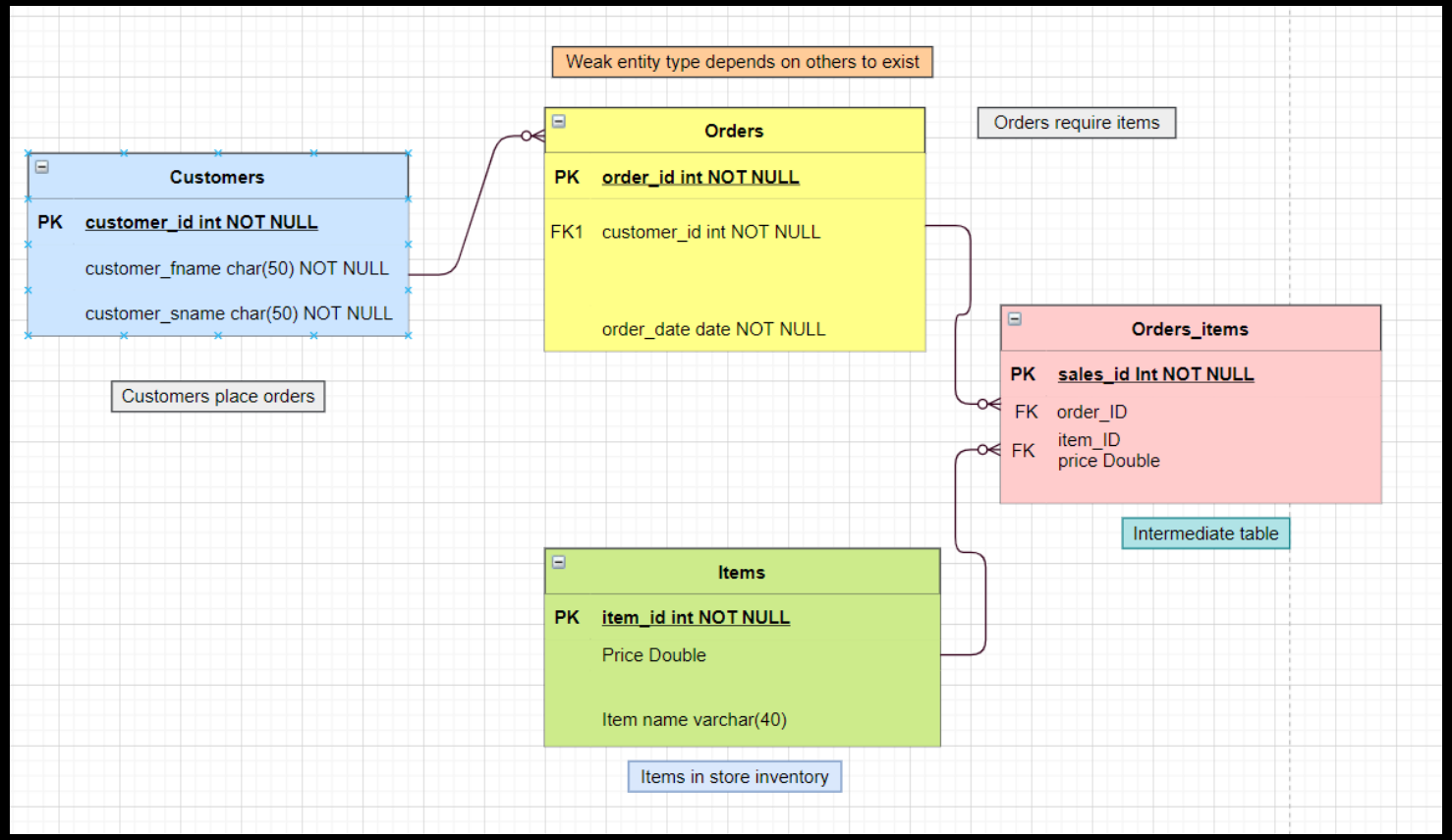
Date of next review: 15/02/2021

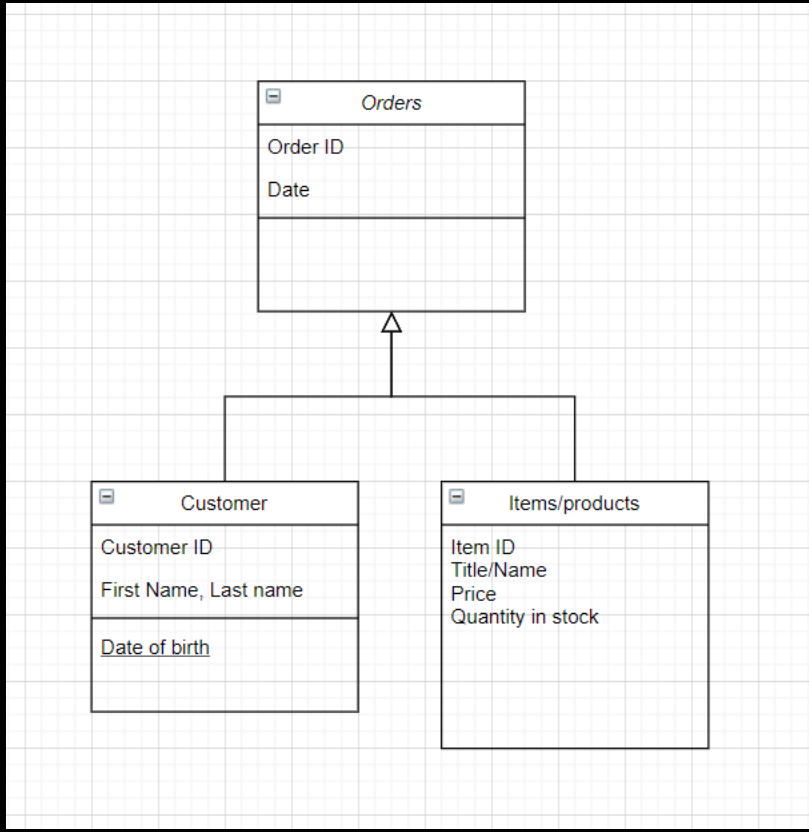
Who might be harmed and where is the impact?	Risk description and impact	Action taken	Objective	Likelihood	Impact	Risk Level	Who is responsible?
Customers, GitHub, SQL database	Vulnerability of customer information and unauthorized access to database. The database contains sensitive information relating to customers. Additionally, the source code may contain login details for databases. The impact of this could be unauthorised access, malicious alterations, and data leaks.	Use stronger passwords and usernames keep them regularly updated. Store customer information securely,	Reduce the likelihood of hacking and data leaks.	Medium	6 High	Medium-to-High	System developers
Customers, SQL Database	Loss of customer data. An example of this includes human error and accidental File deletions. Additionally, customer data can be lost when the project fails to connect to the SQL Database. This impact of this is data loss and an inaccurate database.	All customer data is backed up to allow for a full restore. Back-ups include online cloud platforms and local repositories. Furthermore, ensuring continual update and monitoring of back-up.	Reduce the likelihood of customer data loss.	Low	6 High	High	System developers

Risk assessment matrix

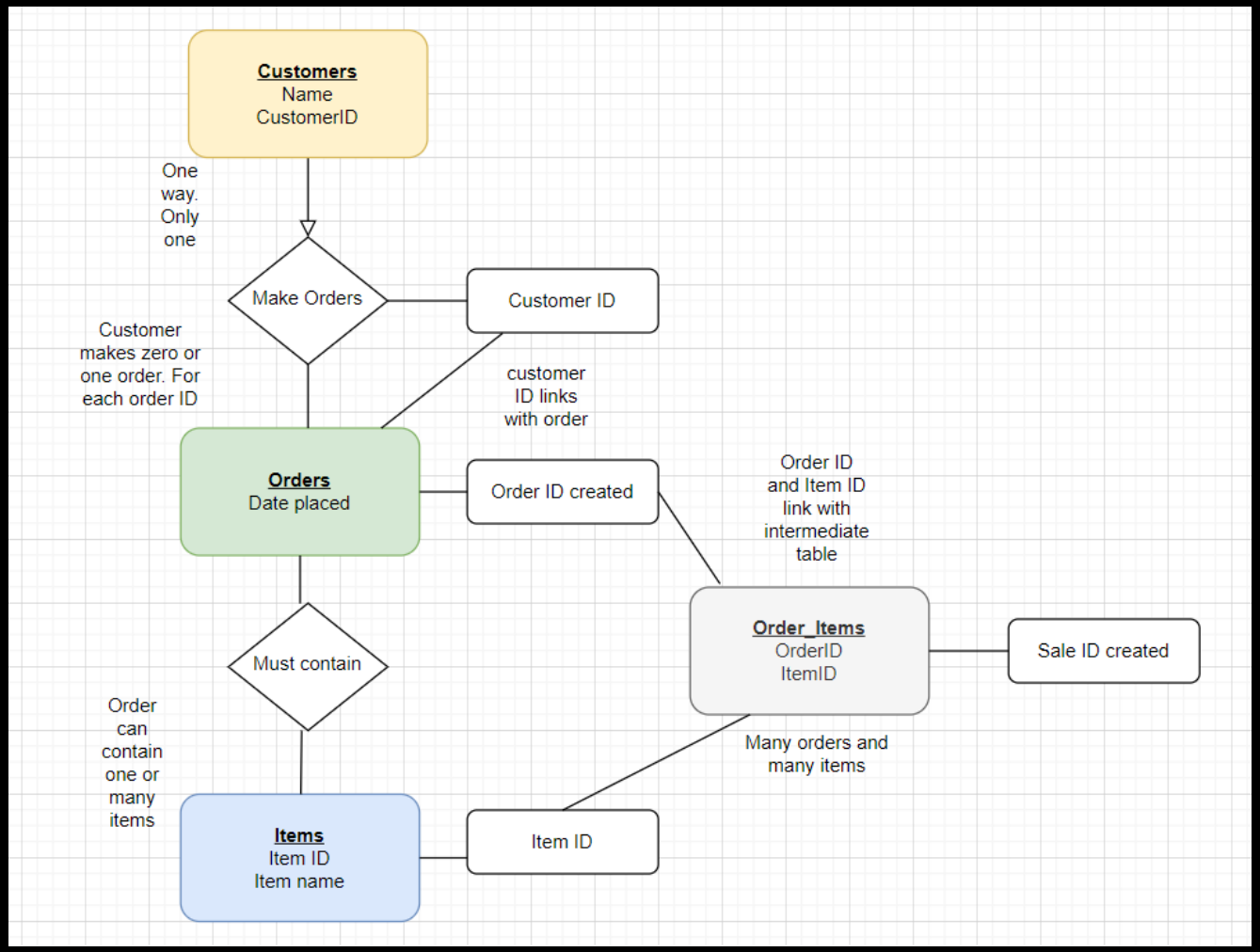
Risk Matrix

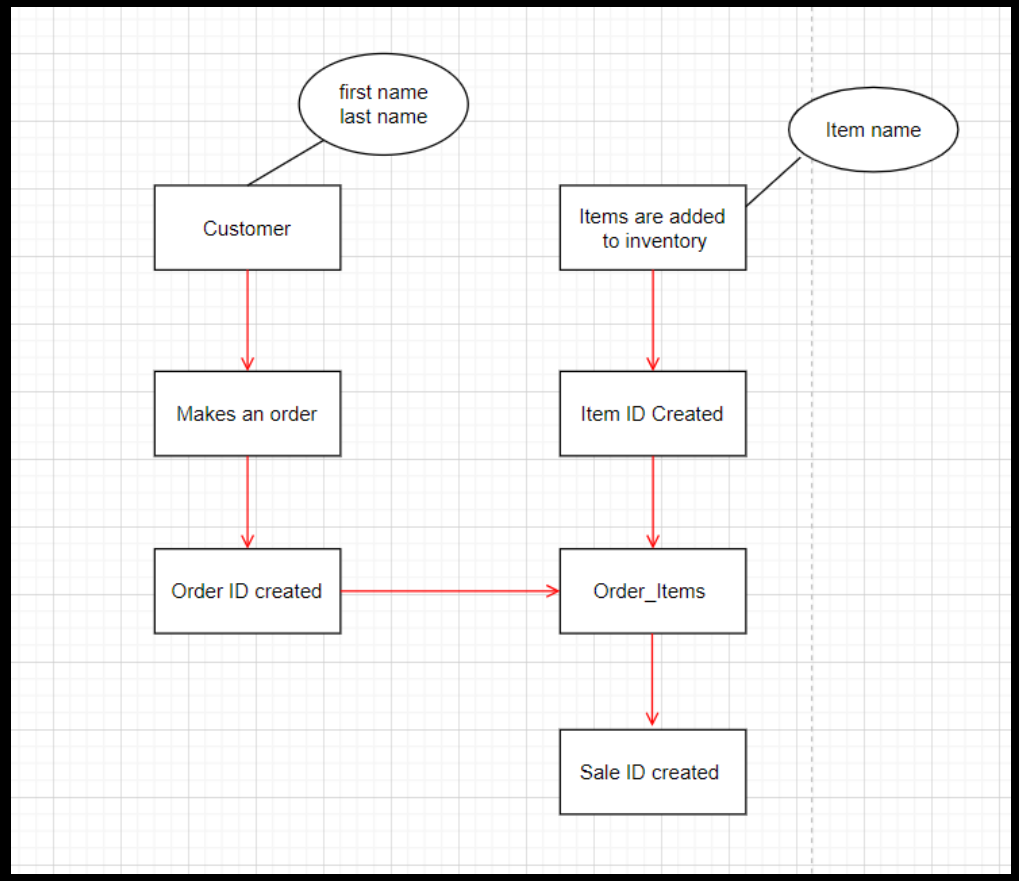
	SEVERITY		
LIKELIHOOD	1	2	3
1	LOW 1	LOW 2 - COVID-19	MEDIUM 3 - Hardware Failure - Hardware and software compatibility
2	LOW 2 - Forgotten usernames and passwords	MEDIUM 4 -Internet connectivity issues	HIGH 6 - Customer data loss - Data leaks
3	MEDIUM 3	HIGH 6	HIGH 9


Weak entity type depends on others to exist
Orders require items
Customers
PK customer_id int NOT NULL
customer_fname char(50) NOT NULL
customer_sname char(50) NOT NULL
Orders
PK order_id int NOT NULL
FK1 customer_id int NOT NULL
order_date date NOT NULL
Orders_items
PK sales_id Int NOT NULL
FK order_ID
FK item_ID
price Double
Customers place orders
Intermediate table
Items
PK item_id int NOT NULL
Price Double
Item name varchar(40)
Items in store inventory

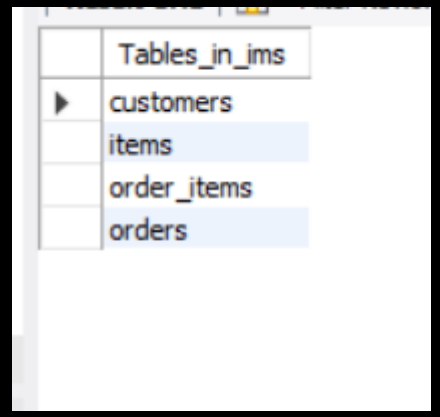

Orders
Order ID
Date
Customer
Customer ID
First Name, Last name
Date of birth
Items/products
Item ID
Title/Name
Price
Quantity in stock

UML


Customers
Name
CustomerID
One way. Only one
Make Orders
Customer ID
Customer makes zero or one order. For each order ID
customer ID links with order
Orders
Date placed
Order ID created
Order ID and Item ID link with intermediate table
Order_Items
OrderID
ItemID
Sale ID created
Must contain
Order can contain one or many items
Many orders and many items
Items
Item ID
Item name
Item ID


first name
last name
Item name
Customer
Items are added to inventory
Makes an order
Item ID Created
Order ID created
Order_Items
Sale ID created

SQL Table creation



Tables_in_ims
customers
items
order_items
orders

- Created a database called **IMS.**
- Created all tables in MySQL Workbench and added the details to the SQL schema text files on eclipse.
- Used ERD diagrams to help build the tables.
- Ensured the database was password and username protected and details of this were added to eclipse.
- Updated tables based on user requirements adjustments.
- 4 tables.

1. Customers
2. Items
3. Order items
4. Orders

SQL

- Customers

Field	Type	Null	Key	Default	Extra
id	int(11)	NO	PRI	NULL	auto_increment
first_name	varchar(40)	YES		NULL	
surname	varchar(40)	YES		NULL	

- Items

Field	Type	Null	Key	Default	Extra
item_id	int(11)	NO	PRI	NULL	auto_increment
item_name	varchar(40)	YES		NULL	
price	double	YES		NULL	

- Orders

Field	Type	Null	Key	Default	Extra
order_id	int(11)	NO	PRI	NULL	auto_increment
price	double	YES		NULL	
fk_id	int(11)	NO	MUL	NULL	

SQL

- Order items

(Intermediate relationship table)

Field	Type	Null	Key	Default	Extra
sale_id	int(11)	NO	PRI	NULL	auto_increment
order_id	int(11)	NO	MUL	NULL	
item_id	int(11)	NO	MUL	NULL	

1) Creating the Customer features

- 3 Classes for Customer in 3 packages.
- 1) Domain – Customer.java
- 2) DAO - CustomerDAO.java
- 3) Controller – CustomerController.java

2) Creating the item features

- 3 Classes for Item in 3 packages.
- 1) Domain – Item.java
- 2) DAO - ItemDAO.java
- 3) Controller – ItemController.java

3) Creating the Order features

- 3 Classes for Order in 3 packages.
- 1) Domain – Order.java
- 2) DAO - OrderDAO.java
- 3) Controller – OrderController.java

Classes

- **Domain classes:**

- Constructors- to initialise the objects.
- Getters and setters.
- Hash Code and equals.

- **DAO Classes:**
- Connects to SQL database.
- Connects with interface DAO.
- Executes SQL queries.

- **Controller classes:**
- Connects with interface CRUDController.
- Logger statements and links to DAO.

4) Creating the order_items feature and relating them

- Order_items variables were added to the Order.java domain class. (including Sales ID, Item ID and Order ID)
- Created three new methods in OrderDAO (add Item, remove item and read item)
- These methods referenced the SQL table order_items.

Order Controller. Adding and deleting Items in an order

- User stories required items to be added or deleted from an order.
- Many to many relationship. Requiring the order_items table.

Testing

- Testing was completed using Junit.

- Testing for Controllers – 30.4% coverage
- Testing for Domain – 13.3% coverage
- Testing for DAO – 28.4% coverage
- 71.4% total

- Testing for main – 58.7%

ava (2) (12 Feb 2021 12:42:39)

Element	Coverage	vered Instructions	lissed Instructions	Total Instructions
New	60.9 %	2,235	1,435	3,670
src/main/java	58.7 %	1,430	1,007	2,437
com.qa.ims	0.0 %	0	165	165
com.qa.ims.controller	65.8 %	311	162	473
com.qa.ims.exceptions	0.0 %	0	3	3
com.qa.ims.persistence.dao	49.8 %	443	447	890
com.qa.ims.persistence.domain	77.2 %	514	152	666
com.qa.ims.utils	67.5 %	162	78	240
src/test/java	65.3 %	805	428	1,233
com.qa.ims.controllers	81.3 %	503	116	619
com.qa.ims.persistence.dao	47.3 %	211	235	446
com.qa.ims.persistence.domain	54.2 %	91	77	168

GitHub- Feature Branch Model

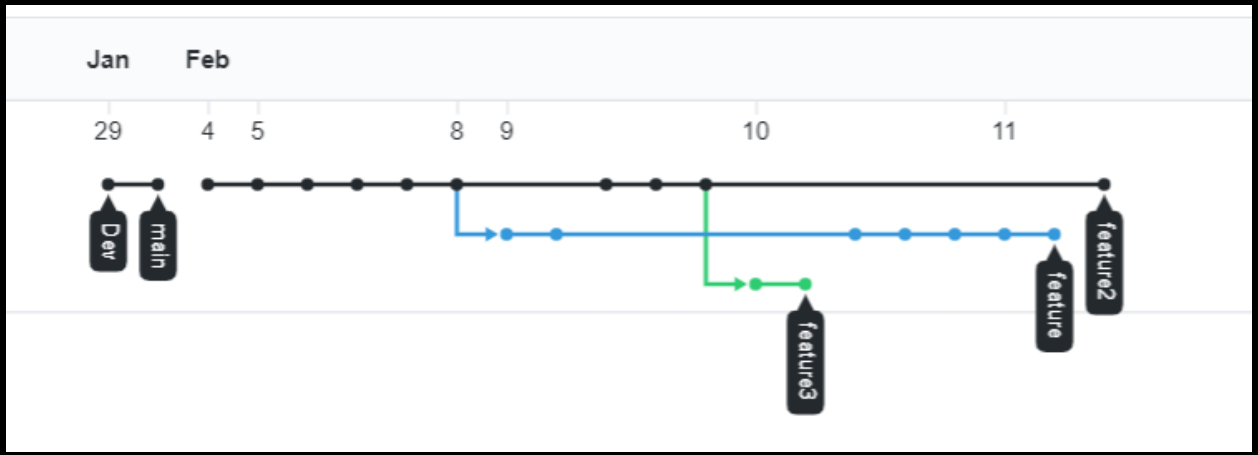
- https://github.com/k-mccalla/Project-work

- My project included a **Main, Dev and 3 feature branches.**

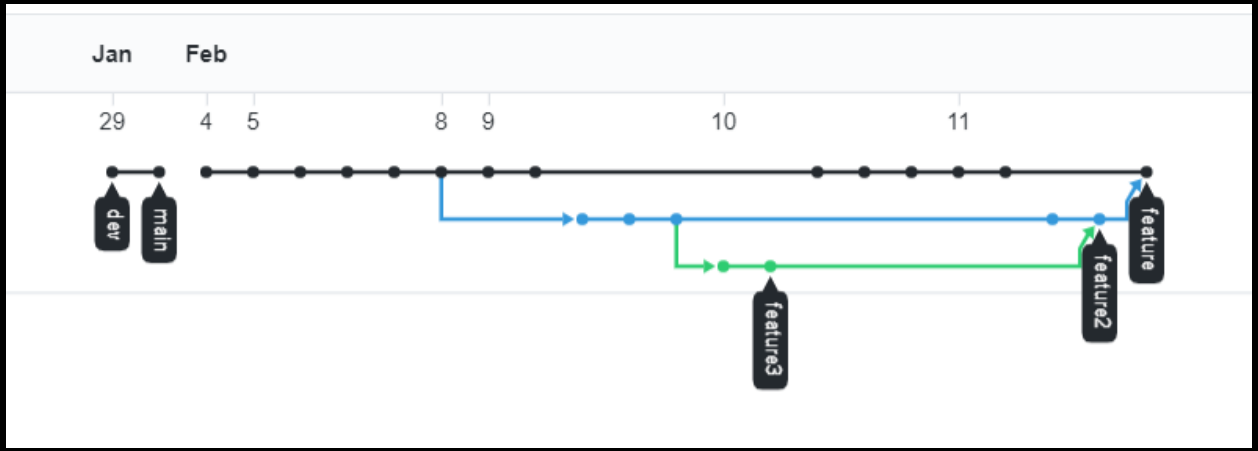
- Feature 1 – Initial CRUD Functionality.
- Feature 2- Implementing order_items table and new methods.
- Feature 3- Testing.

Branch model

- Made commits to feature branches. Image references before merging.



- After merging.



Sprint Review

- <u>What was completed:</u>

1. Able to complete the user stories for customers, items and orders. With a few adjustments.
2. Created all SQL tables.
3. Able to successfully connect database with Java application.
4. Able to complete documentation required.

Sprint Review

- <u>What was not completed?</u>

1. Not able to read all items in an Order.
2. Did not complete the user story for calculating the cost of an order.
3. Did not successfully create a fat Jar file.
4. Did not successfully utilise the Dev branch.
5. Did not reach 80% coverage of testing aim.

Sprints

PROJ Sprint 2 10 issues

78 0 0 Plan sprint

Complete the items section of the database including SQL tables and Java functionality.

09/Feb/21 9:50 AM • 16/Feb/21 9:50 AM

As a user I must be able to add customers to the system	Customer details in sys...	PROJ-9	9
As a user I must be able to view all customers in the system	Customer details in sys...	PROJ-10	7
As a user I must be able to add customer details to the system inclu	Customer details in sys...	PROJ-27	9
As a user I must be able to delete customers in the system	Customer details in sys...	PROJ-11	7
As a user I must be able to update customer details in the system	Customer details in sys...	PROJ-12	8
As a user I must be able to add items to the system	Item details in system	PROJ-13	9
As a user I must be able to view the item name, cost and item ID	Item details in system	PROJ-28	7
As a user I must be able to view all items in the system	Item details in system	PROJ-14	7
As a user I must be able to delete items from the system	Item details in system	PROJ-15	7
As I user I must be able to update item details in the system	Item details in system	PROJ-16	8

PROJ Sprint 3 4 issues

Start sprint Plan sprint

Complete the orders section of the database including SQL and Java functionality.

As a user I must be able to create an order in the system	Order details in system	PROJ-17	7
As a user I must be able to delete orders in the system	Order details in system	PROJ-19	7
As a user I must be able to see the order ID, cost, quantity and date pl	Order details in system	PROJ-21	7
As a user I must be able to view all orders in the system	Order details in system	PROJ-18	7

+ Create issue

PROJ Sprint 4 4 issues

Plan sprint

To complete an intermediate SQL table containing the foreign keys of Items and Orders. Complete Java functionality for this.

As a user I must be able to add an item to an order	Items adding to orders	PROJ-23	9
As a user I must be able to delete an item from an order	Items adding to orders	PROJ-25	9
As a user I must be able to calculate the cost of an order	Items adding to orders	PROJ-24	9
As a user I must be able to cancel an order	Items adding to orders	PROJ-26	6

+ Create issue

Sprint retrospective.

- <u>What will I do better next time?</u>

1) Give more time and planning to the hardest sprint.
2) Predict more errors and complications beforehand and factor these into my time management and sprint preparation.
3) Better assignment of story points. Perhaps using story point poker.
4) More pushes to GitHub during the project. Using Dev branch.

- <u>What did I do well?</u>

1. Documentation.
2. My use of classes and packages in Java.
3. Following my user stories and sprint goals to create the application.
4. Following the project requirements.

Thank you

Questions