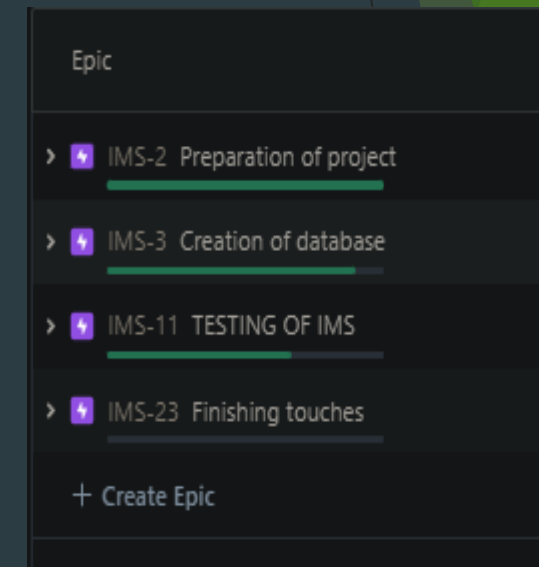
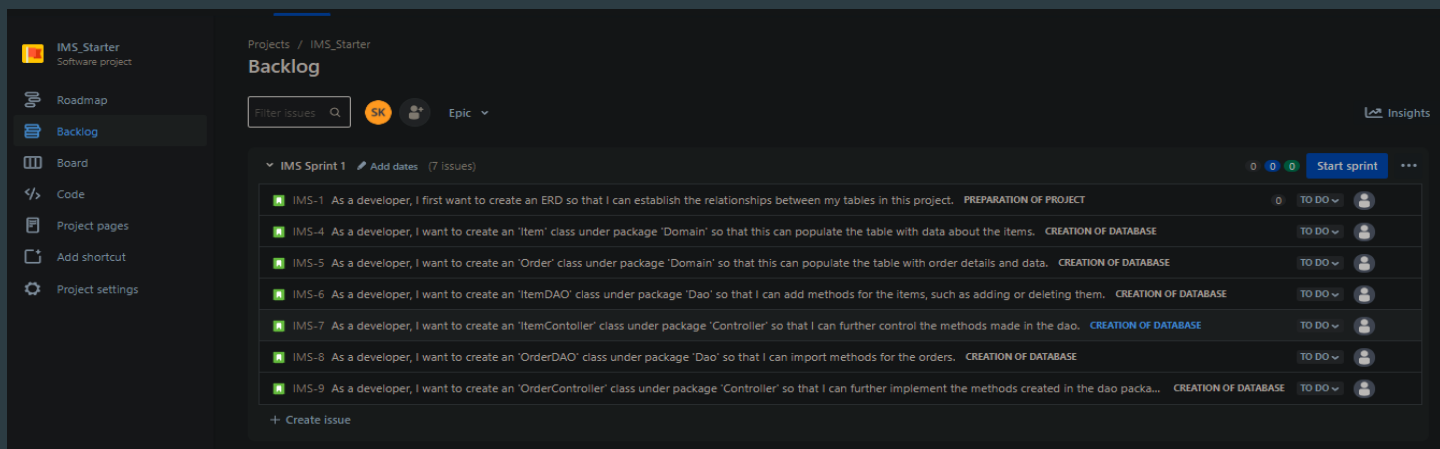
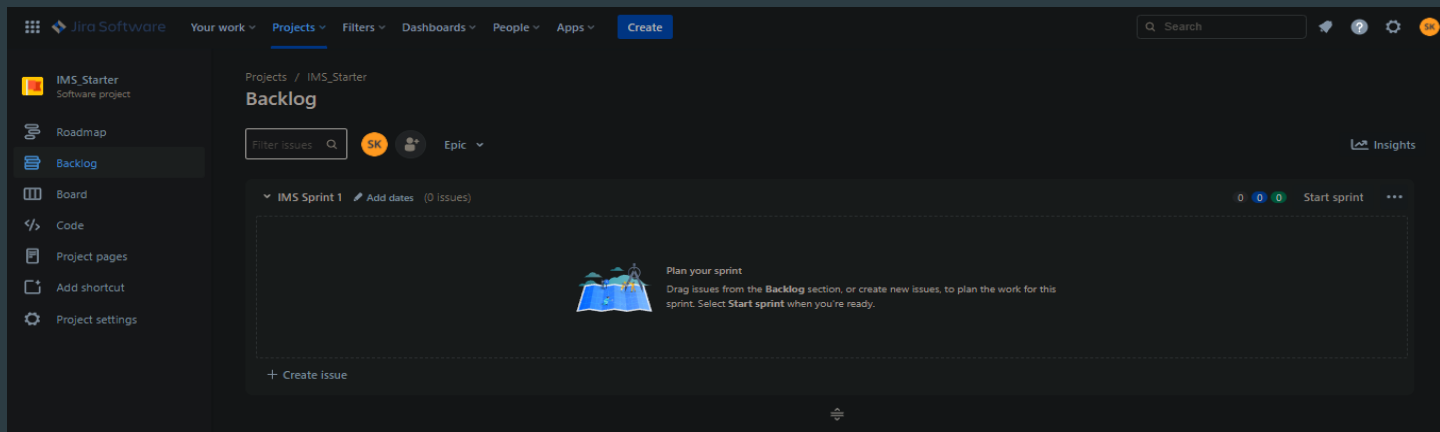


IMS Starter project

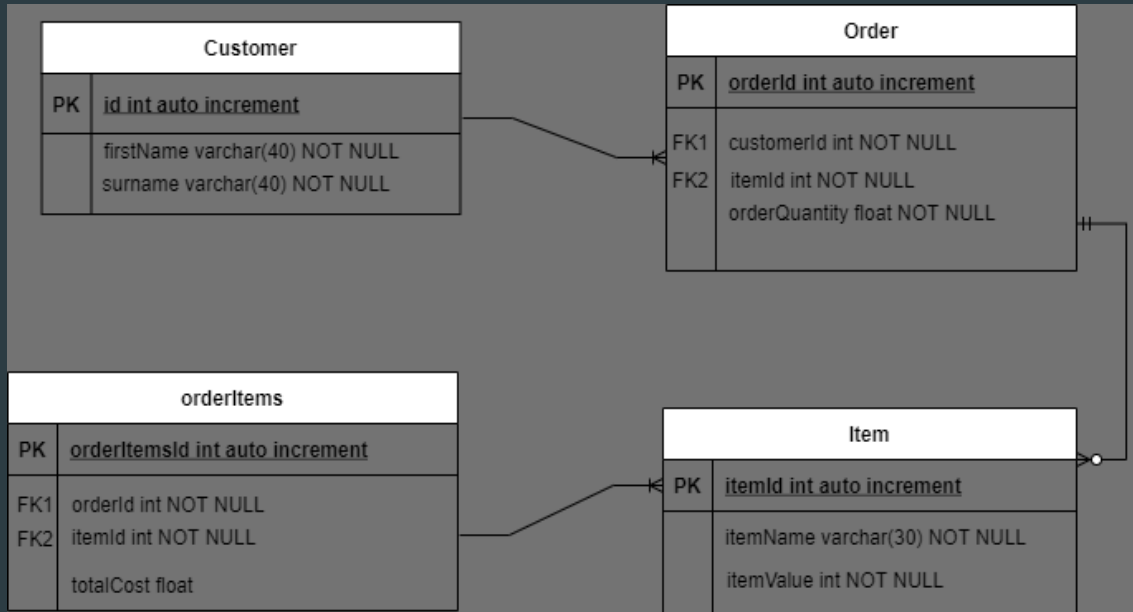
Sarah Khan

Introduction

- For this project, I approached the specification by clearly setting user stories for each task, and thoroughly looking through the scope required.

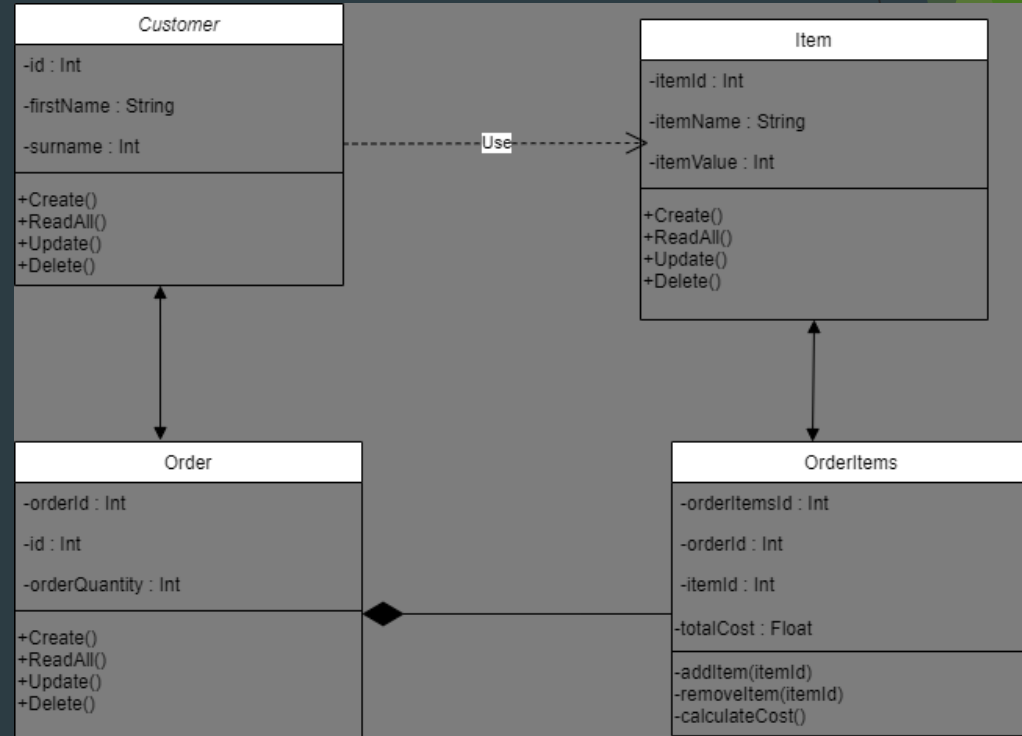


- ▶ Alongside user stories, I prepared for the project by creating ERD and UML diagrams. Both were used in ways to visual the IMS.



ERD Diagram

UML Diagram



- Using GitHub, code could be backed up with ease, within my own repository, and changes can be accessed at any time (version control). This process was done through using the same commands on GitBash.

```
MINGW64/c:/Users/Sarah/OneDrive/Documents/QA/IMS/IMS-Starter

Sarah@LAPTOP-NV4VFBK MINGW64 ~/OneDrive/Documents/QA/IMS
$ git clone https://github.com/sarahkgh/IMS-Starter.git
Cloning into 'IMS-Starter'...
remote: Enumerating objects: 267, done.
Receiving objectremote: Total 267 (delta 0), reused 0 (delta 0), pack-reused 267
Receiving objects: 100% (267/267), 41.16 KiB | 569.00 KiB/s, done.
Resolving deltas: 100% (56/56), done.

Sarah@LAPTOP-NV4VFBK MINGW64 ~/OneDrive/Documents/QA/IMS
$ ls
IMS-Starter/


Sarah@LAPTOP-NV4VFBK MINGW64 ~/OneDrive/Documents/QA/IMS
$ cd IMS-Starter

Sarah@LAPTOP-NV4VFBK MINGW64 ~/OneDrive/Documents/QA/IMS/IMS-Starter (master)
$ git branch dev




Sarah@LAPTOP-NV4VFBK MINGW64 ~/OneDrive/Documents/QA/IMS/IMS-Starter (master)
$ git checkout
Your branch is up to date with 'origin/master'.


Sarah@LAPTOP-NV4VFBK MINGW64 ~/OneDrive/Documents/QA/IMS/IMS-Starter (master)
$ git checkout dev
Switched to branch 'dev'

Sarah@LAPTOP-NV4VFBK MINGW64 ~/OneDrive/Documents/QA/IMS/IMS-Starter (dev)
$ A|
```



[Pull requests](#) [Issues](#) [Marketplace](#) [Explore](#)

 **sarahkgh / IMS-Starter** Public template


[Watch](#) 0 [Star](#) 0 [Fork](#) 137

[Code](#) [Pull requests](#) [Actions](#) [Projects](#) [Wiki](#) [Security](#) [Insights](#) [Settings](#)

master 1 branch 0 tags

[Go to file](#) [Add file](#) [Code](#) [Use this template](#)

This branch is up to date with QACTrainers:master. [Contribute](#) [Fetch upstream](#)

 JHarry444 Merge pull request JHarry444#4 from JHarry444/dev ... 716c545 on 20 Jan 21 commits

src	added try-with-resources to close ResultSet	9 months ago
.gitignore	init commit	15 months ago
LICENSE.md	init commit	15 months ago
README.md	init commit	15 months ago
pom.xml	bumped junit version	9 months ago

About

No description, website, or topics provided.

[Readme](#)

[MIT License](#)

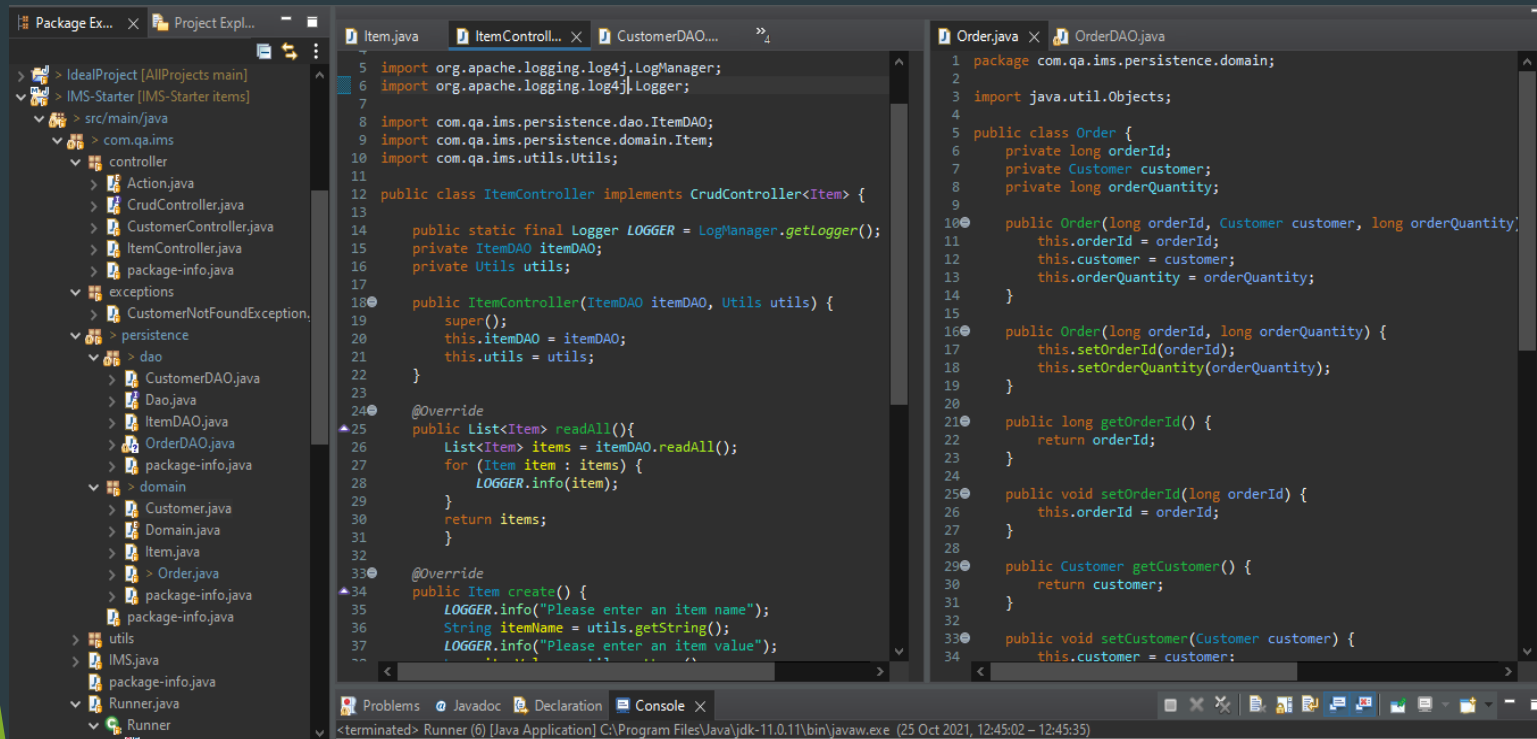
Releases

No releases published [Create a new release](#)

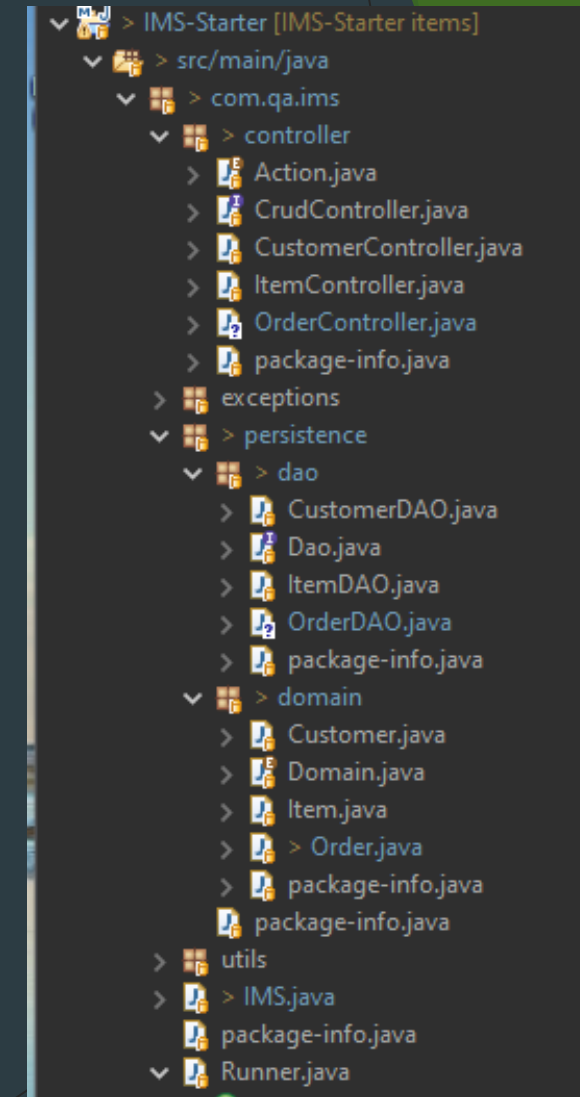
Packages

Lessons learnt

- ▶ During my consultant journey, I learnt and overall became more familiar with creating DAO's and controller classes, using Jira as a project board, creating methods for JUnit testing, and finally, creating project diagrams.



The screenshot shows an IDE with two open Java files. The left file is `ItemController.java` and the right file is `OrderDAO.java`. The `ItemController` class implements `CrudController<Item>` and uses `ItemDAO` and `Utils`. The `OrderDAO` class is a `Dao` for the `Order` entity, implementing methods like `readAll()`, `create()`, `getById()`, and `setCustomer()`. The IDE interface includes a Package Explorer on the left, a Console at the bottom, and a status bar.

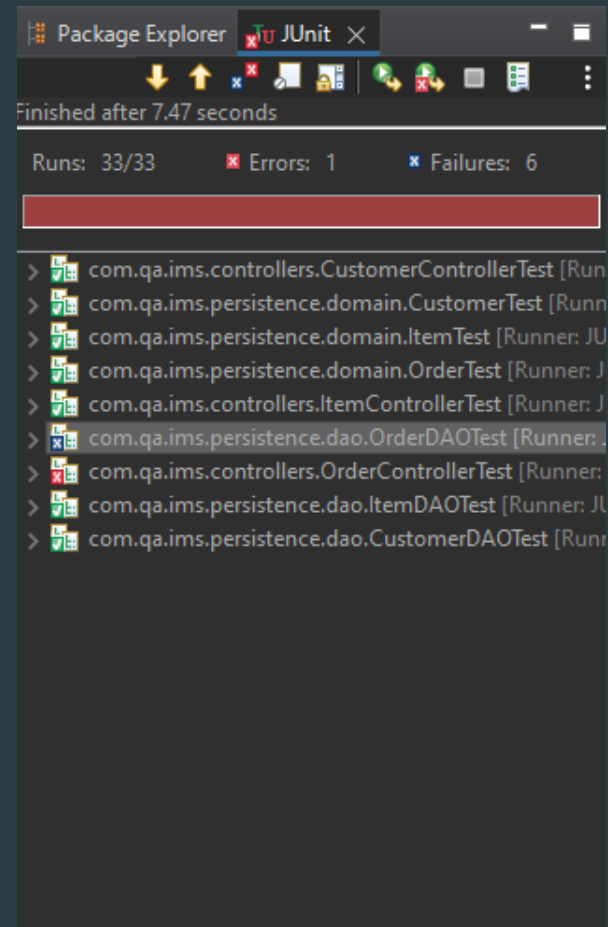
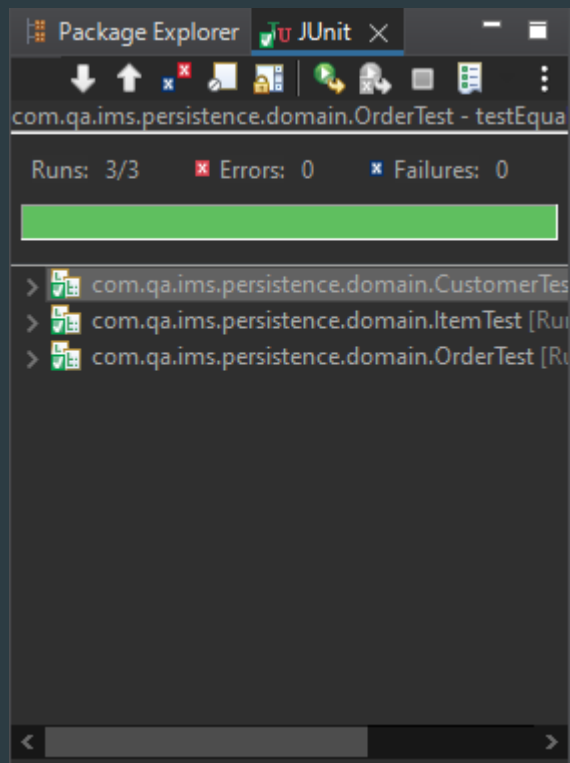


The screenshot shows a Package Explorer view of a project named 'IMS-Starter [IMS-Starter items]'. The structure is as follows:

- IMS-Starter [IMS-Starter items]
 - src/main/java
 - com.qa.ims
 - controller
 - Action.java
 - CrudController.java
 - CustomerController.java
 - ItemController.java
 - OrderController.java
 - package-info.java
 - exceptions
 - persistence
 - dao
 - CustomerDAO.java
 - Dao.java
 - ItemDAO.java
 - OrderDAO.java
 - package-info.java
 - domain
 - Customer.java
 - Domain.java
 - Item.java
 - Order.java
 - package-info.java
 - utils
 - IMS.java
 - package-info.java
 - Runner.java

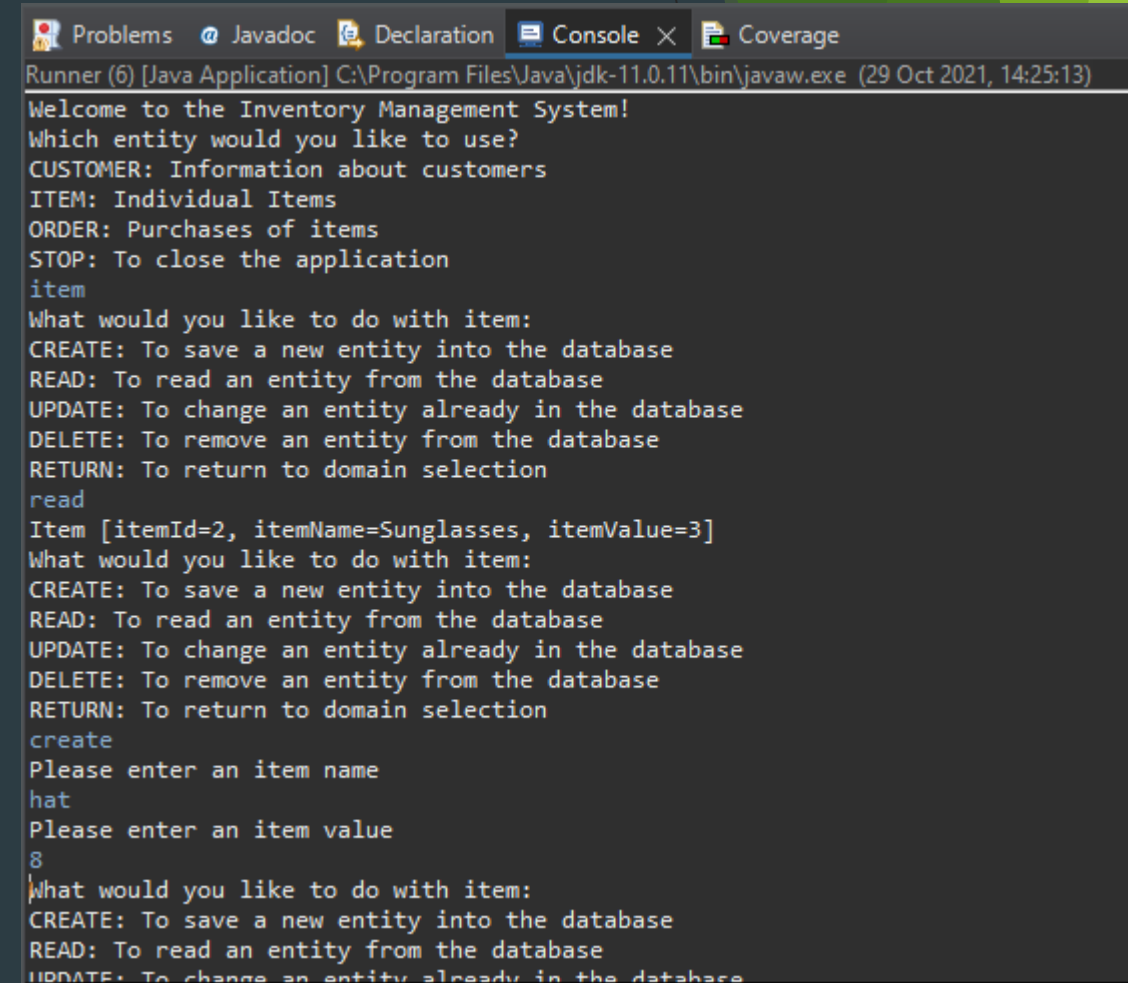
Testing

- Methods within each class, particularly create, read, update and delete methods, were created per class and then tested. This ensures that each of these methods are passing in tests so that they function as normal.

[illegible]

Sprint review

- ▶ Completed: order, item and customer entities, JDBC connections, git repository, Jira storyboard, project documentation.
- ▶ Incomplete: 80% test coverage (only reached 71%).
- ▶ The IMS project was generally a success in terms of being able to document the process and have no errors. It could be improved with refinement of full Junit testing, and extra methods or Loggers to the console.
- ▶ The project overall has prepared me for future Java projects.



```
Runner (6) [Java Application] C:\Program Files\Java\jdk-11.0.11\bin\javaw.exe (29 Oct 2021, 14:25:13)
Welcome to the Inventory Management System!
Which entity would you like to use?
CUSTOMER: Information about customers
ITEM: Individual Items
ORDER: Purchases of items
STOP: To close the application
item
What would you like to do with item:
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
read
Item [itemId=2, itemName=Sunglasses, itemValue=3]
What would you like to do with item:
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
create
Please enter an item name
hat
Please enter an item value
8
What would you like to do with item:
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
```