IMS Project

Presentation

Quick intro

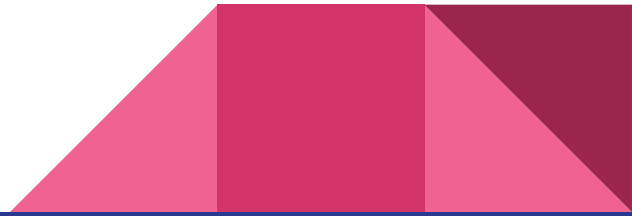
Name: Daniel Li

Role: Software Developer

Company: QA Ltd

Product: Create a functional back-end application with CRUD capabilities.

Approach: Implement the MoSCOW method with regards to the MVP condition.



MoSCoW

Must have

A functional back-end application which connects to a database and has ‘CRUD’ capabilities with respect to the following three entities (customer, items, orders).

Should have

Users can view all items (including associated info such as item_name) for a particular order.

Could have

A refund entity with CRUD’ capabilities.

A quantity field for the item entity.

Won’t have

A front-end user interface.

Tech stack

Version control system: Git

Source code management: GitHub

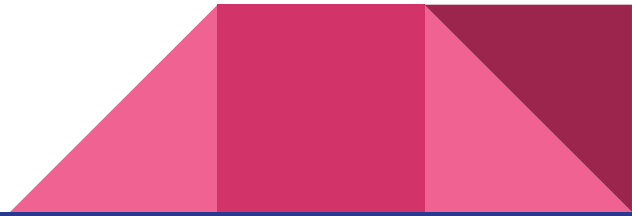
Scrum board: Jira

Database management system: MySQL

Programming language: Java

Build tool: Maven

Unit testing: JUnit and Mockito Framework



Testing

Overall testing coverage: 79%

Unit Tests

The unit test files test whether each 'unit' i.e. method of the application is functioning as expected.

Unit tests have been written to cover the following file types:

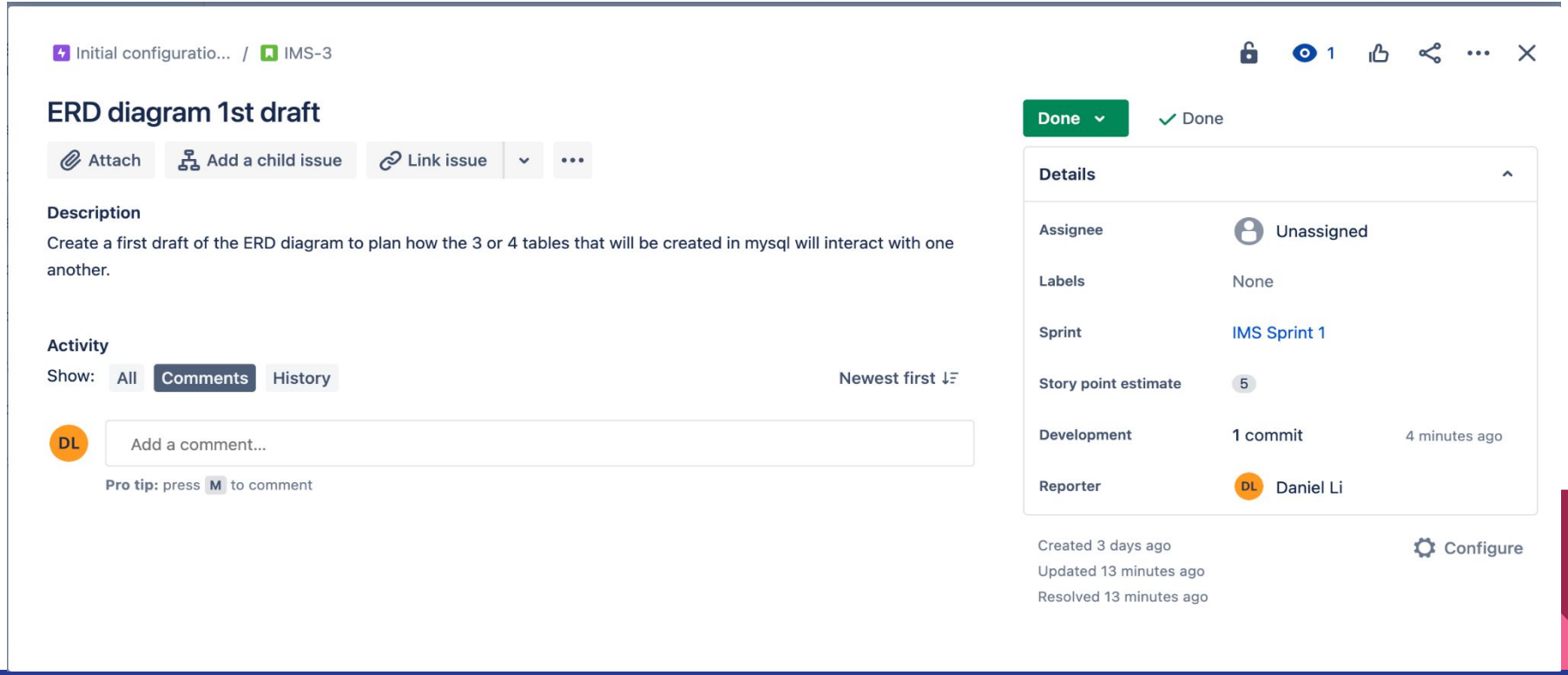
1. Classes e.g. Customer, Item etc.
2. Data Access Object (DAO) files e.g. CustomerDAO, ItemDAO etc.

Integration Tests

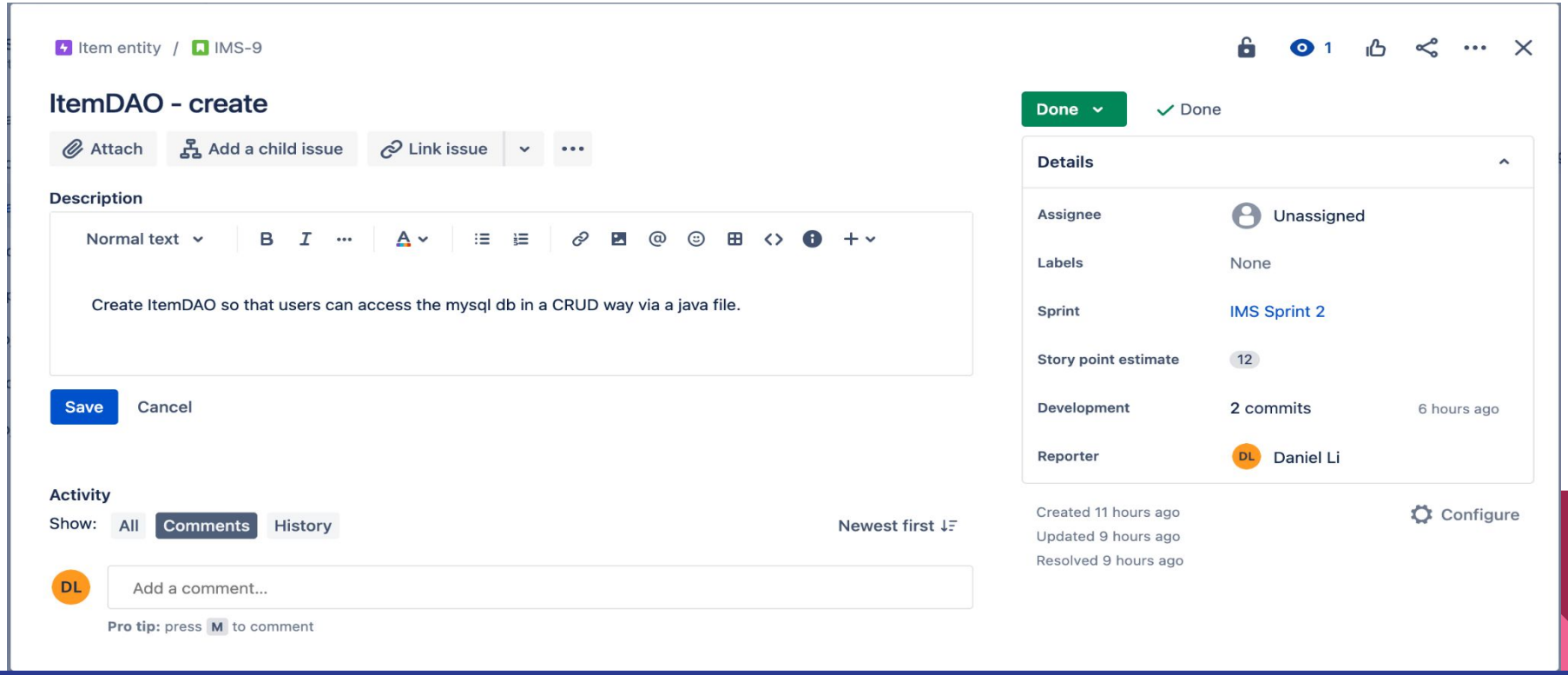
Integration tests have been written using the Mockito framework, the purpose of which is to test the functioning of the software when modules are combined.

Integration tests have been written to test how the controller files (e.g. CustomerController, ItemController etc.) interacts with the corresponding DAO files and Logger methods found within the utilities file.

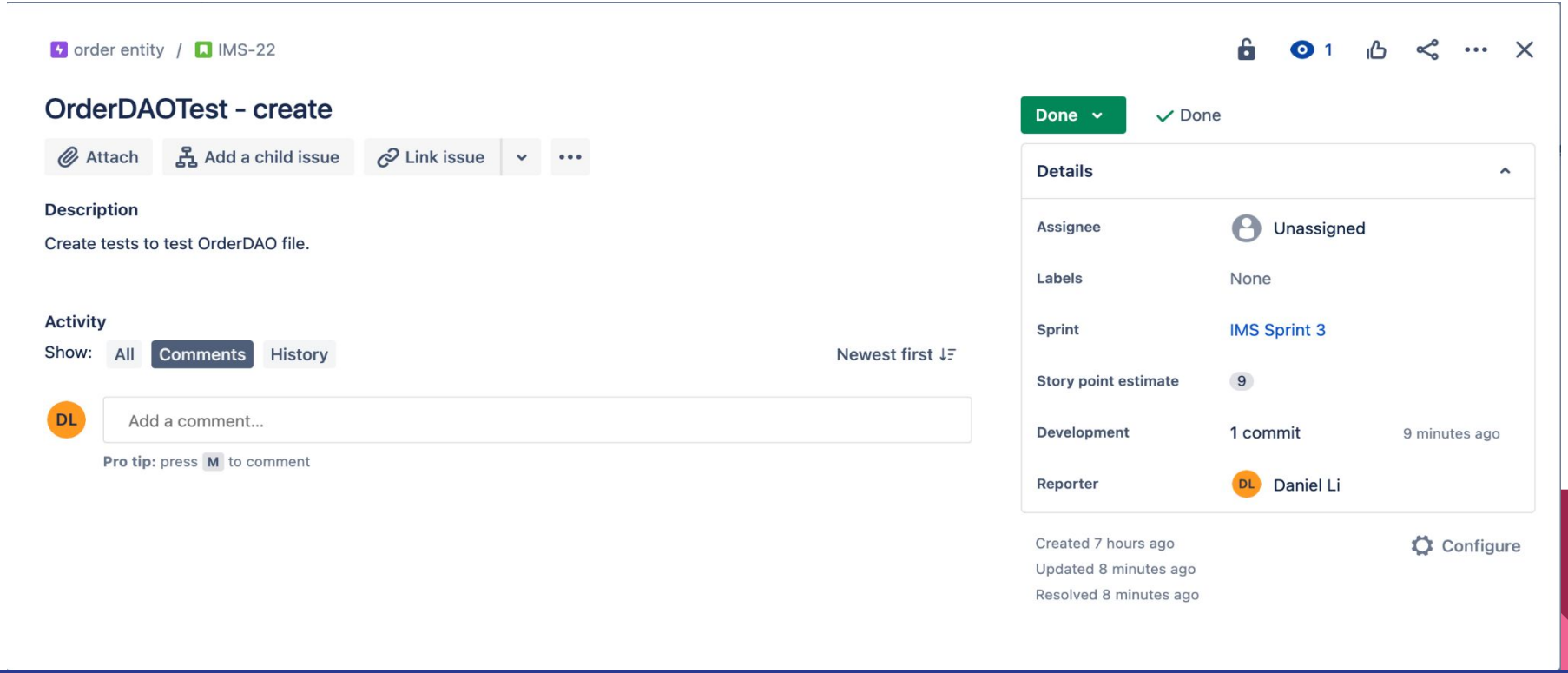
Jira - user stories



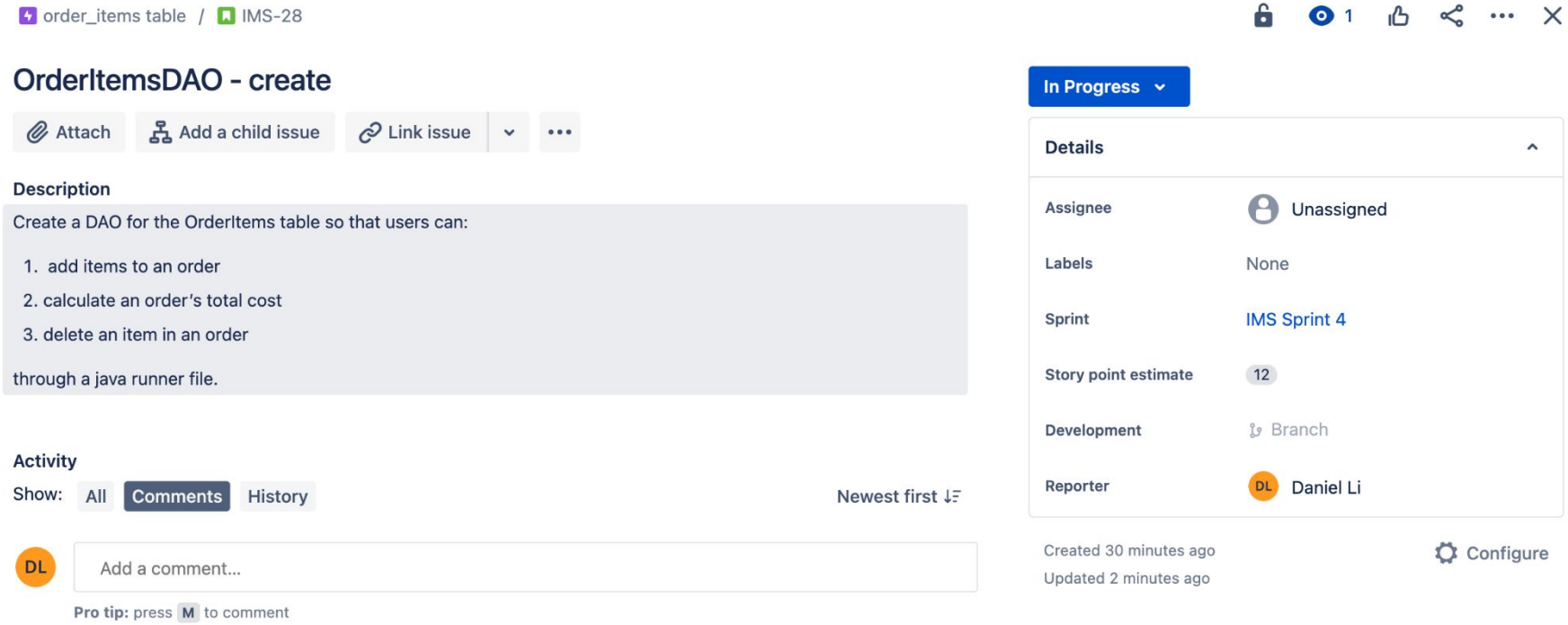
Jira - user stories



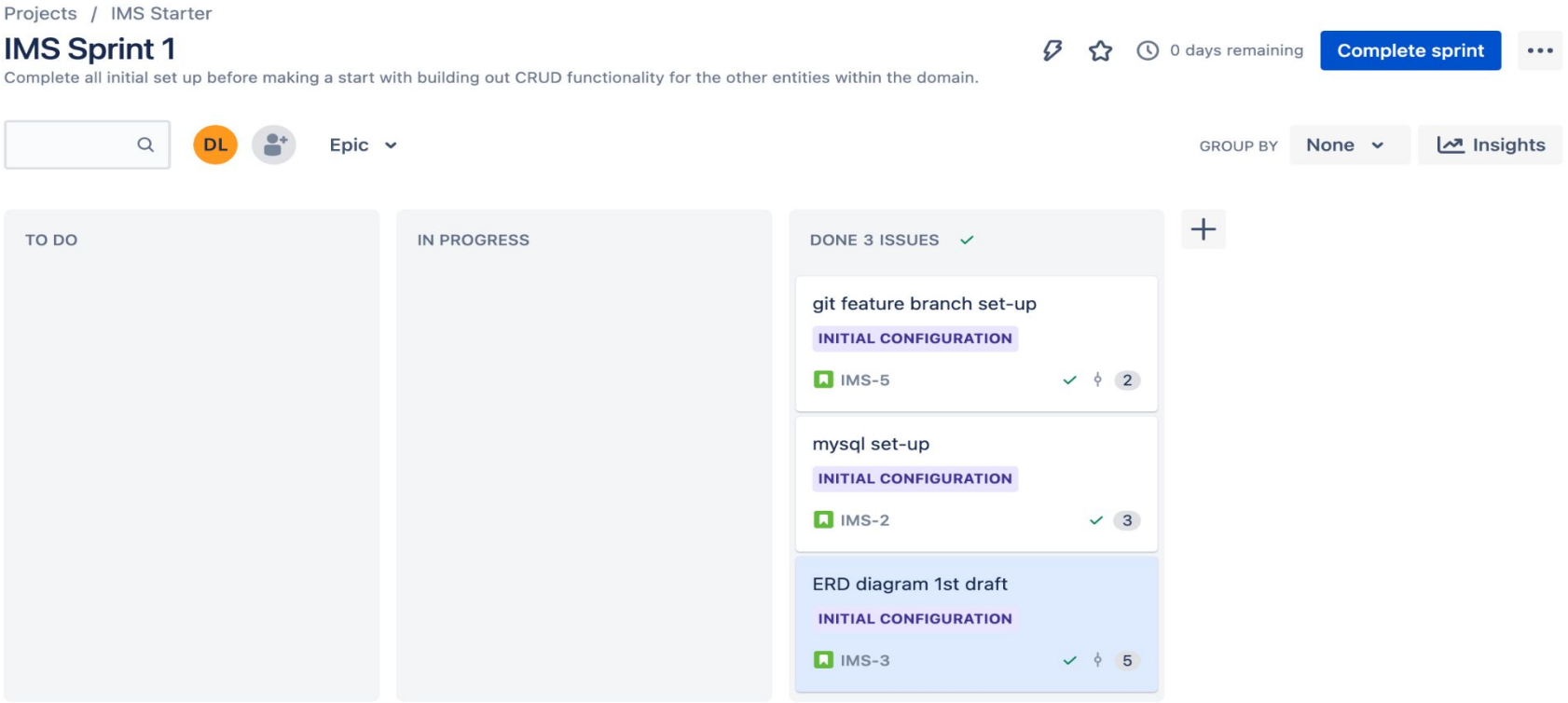
Jira - user stories



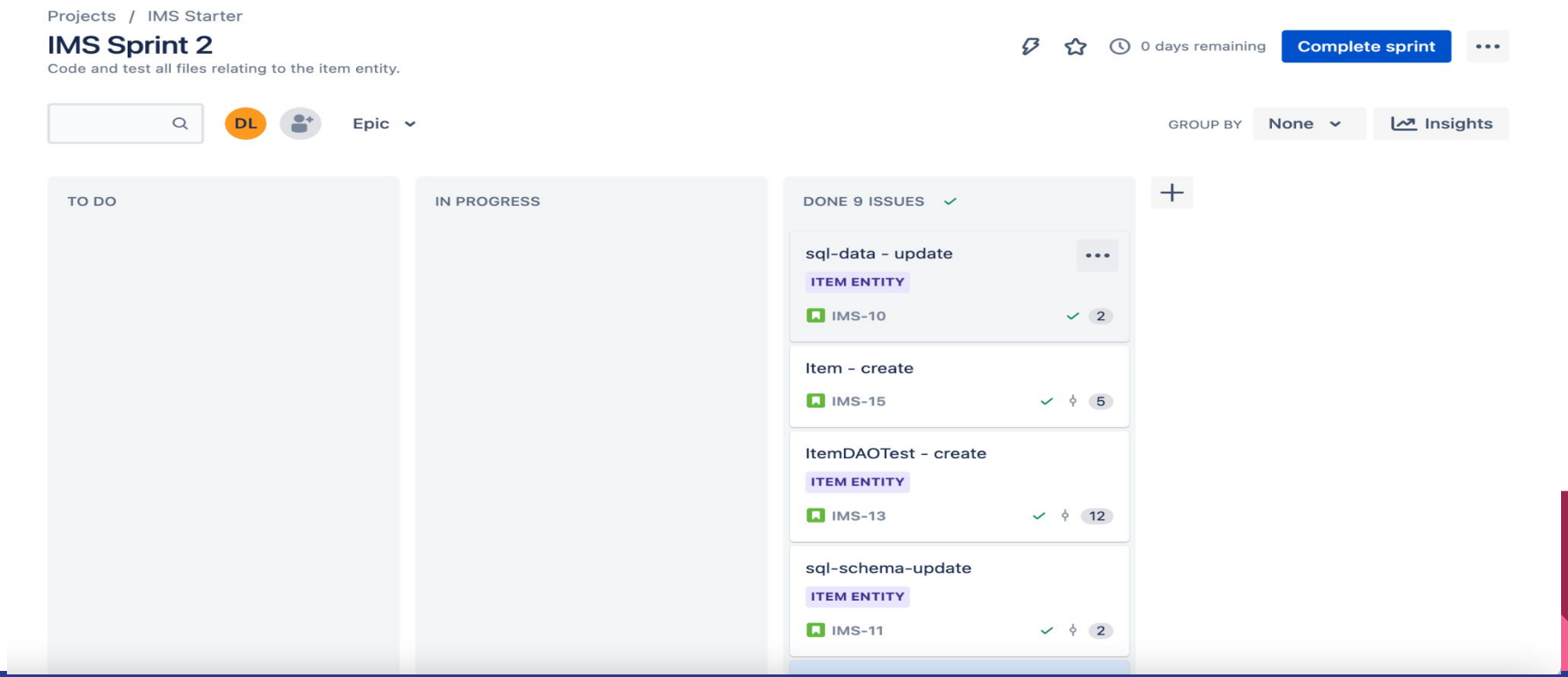
Jira - user stories



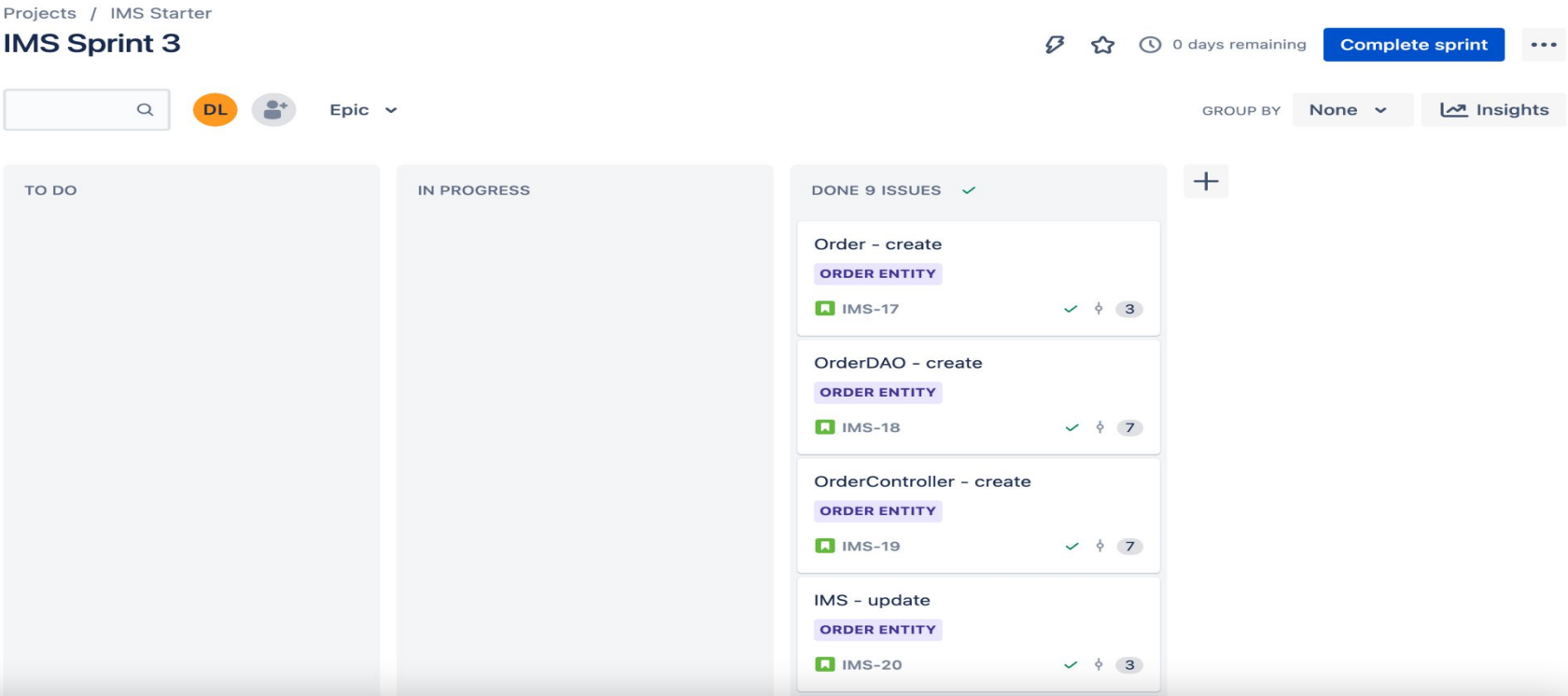
Sprint 1


Projects / IMS Starter
IMS Sprint 1
Complete all initial set up before making a start with building out CRUD functionality for the other entities within the domain.
0 days remaining
Complete sprint
DL
Epic
GROUP BY
None
Insights
TO DO
IN PROGRESS
DONE 3 ISSUES
git feature branch set-up
INITIAL CONFIGURATION
IMS-5
2
mysql set-up
INITIAL CONFIGURATION
IMS-2
3
ERD diagram 1st draft
INITIAL CONFIGURATION
IMS-3
5

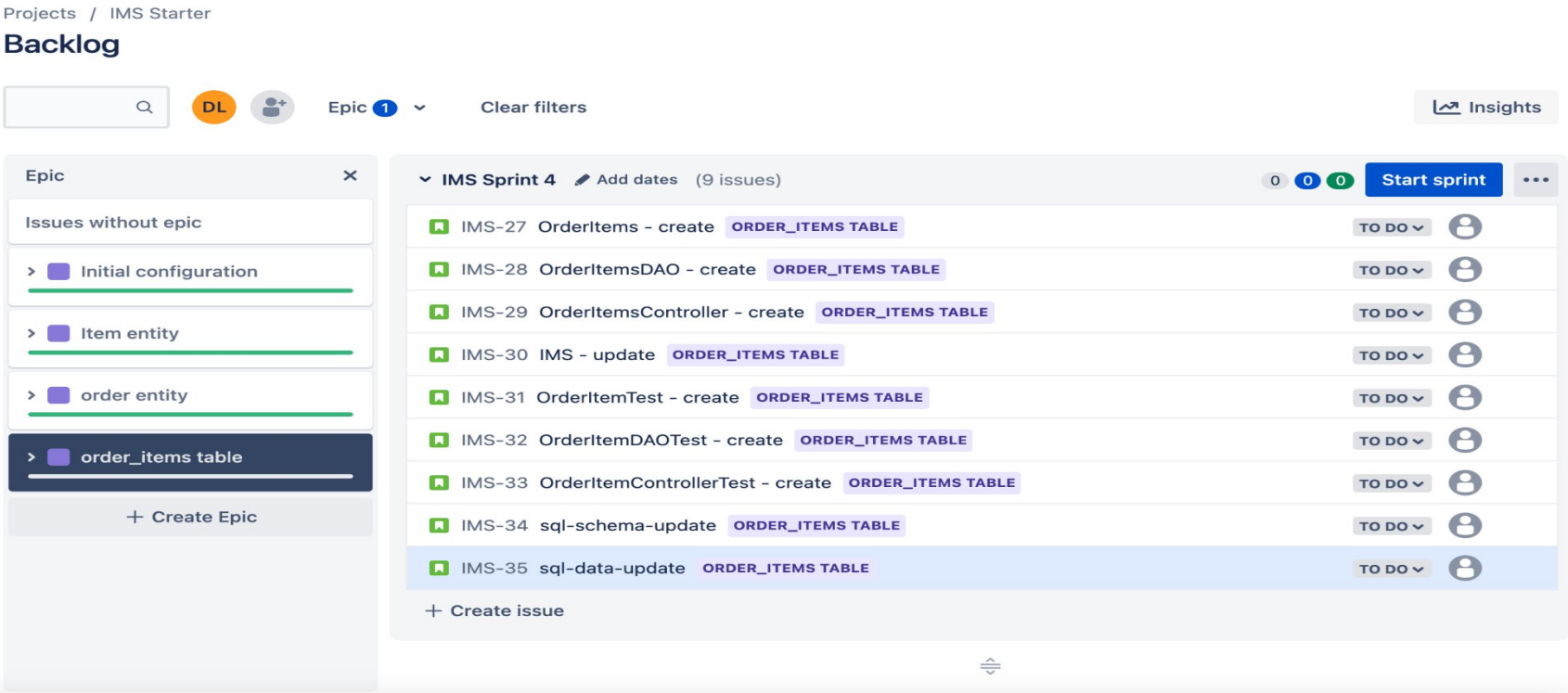
Sprint 2


Projects / IMS Starter
IMS Sprint 2
Code and test all files relating to the item entity.
0 days remaining
Complete sprint
DL
Epic
GROUP BY
None
Insights
TO DO
IN PROGRESS
DONE 9 ISSUES
sql-data - update
ITEM ENTITY
IMS-10
2
Item - create
IMS-15
5
ItemDAOTest - create
ITEM ENTITY
IMS-13
12
sql-schema-update
ITEM ENTITY
IMS-11
2

Sprint 3


Projects / IMS Starter
IMS Sprint 3
0 days remaining
Complete sprint
DL
Epic
GROUP BY
None
Insights
TO DO
IN PROGRESS
DONE 9 ISSUES
Order - create
ORDER ENTITY
IMS-17
3
OrderDAO - create
ORDER ENTITY
IMS-18
7
OrderController - create
ORDER ENTITY
IMS-19
7
IMS - update
ORDER ENTITY
IMS-20
3

Sprint 4


Projects / IMS Starter
Backlog
DL
Epic 1
Clear filters
Insights
Epic
Issues without epic
Initial configuration
Item entity
order entity
order_items table
Create Epic
IMS Sprint 4
Add dates
(9 issues)
0 0 0
Start sprint
IMS-27 OrderItems - create ORDER_ITEMS TABLE TO DO
IMS-28 OrderItemsDAO - create ORDER_ITEMS TABLE TO DO
IMS-29 OrderItemsController - create ORDER_ITEMS TABLE TO DO
IMS-30 IMS - update ORDER_ITEMS TABLE TO DO
IMS-31 OrderItemTest - create ORDER_ITEMS TABLE TO DO
IMS-32 OrderItemDAOTest - create ORDER_ITEMS TABLE TO DO
IMS-33 OrderItemControllerTest - create ORDER_ITEMS TABLE TO DO
IMS-34 sql-schema-update ORDER_ITEMS TABLE TO DO
IMS-35 sql-data-update ORDER_ITEMS TABLE TO DO
Create issue

Sprint retrospective

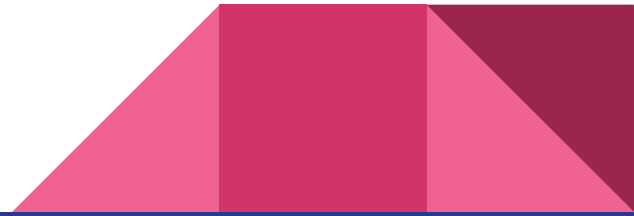
Pros

- Having clearly defined and distinct user stories helped me with problem solving and debugging.
- Using smart commits helped keep my git commits atomic and specific.

Cons

- Underestimated the difficulty and time-required with regards to the initial set-up, in particular - creating the ERD diagram which would meet the MVP.
- Similarly, some story point estimates were either significantly overestimated or underestimated.

Live demo



Conclusion

Summary

Created a functioning backend which satisfies the MVP conditions.

Adopted the SOLID principles when designing the software.

43 tests passing with overall coverage of 79%

Areas of improvement

Refactoring unrelated files - go back into original branch or create a specific branch to handle refactors.

Configure the Action enums - so more appropriate messages are flagged to the users.

Q&A

Your questions please.