Inventory Management system

Anthony Papasavva - 22JuneEnable2

Presentation Contents

1) The Brief
2) Deliverable Checklist (MVP)
3) Project Management Board (Trello)
4) Risk Assessment
5) Entity Relationship Diagram (ERD)
6) GitHub Network Graph
7) Testing + Coverage
8) Demonstration
9) Questions
10) Reflections

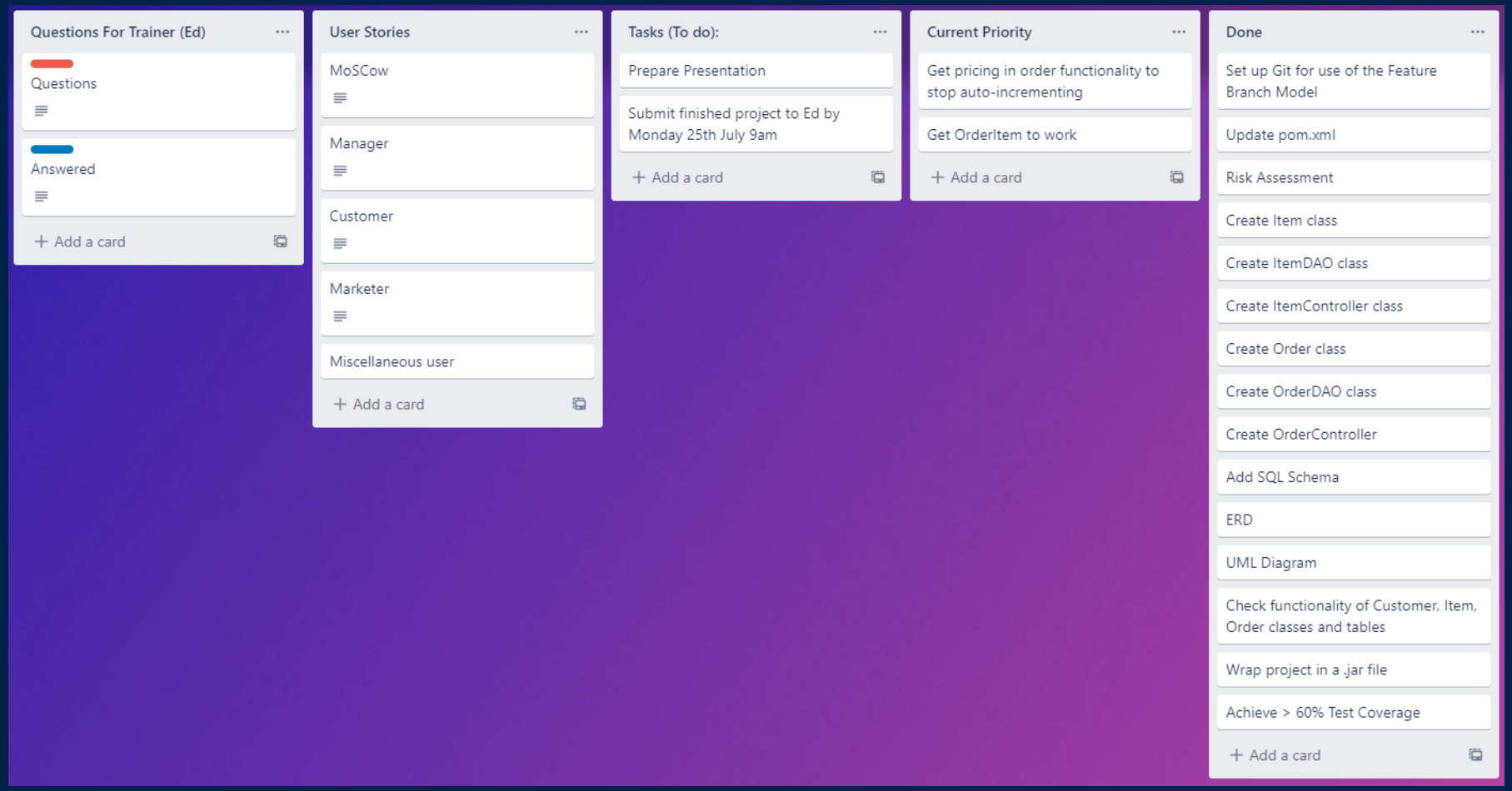
The Brief

My task was to create an Inventory Management System (IMS), in order to enable a third-party user to interact with and manage various functionality of the system through the Command Line.

Deliverable Checklist (MVP)

- CRUD functionality for customers, items, and orders entities.
- Connect via JDBC to a local MySQL instance.
- Unit test coverage - aim for industry standard of 80%.
- Git repository utilising the Feature-Branch-Model.
- Appropriate documentation including: Project Management Board, README.md, Risk Assessment, Entity Relationship Diagram, and a Presentation + recording.

Project Management Board (Trello)



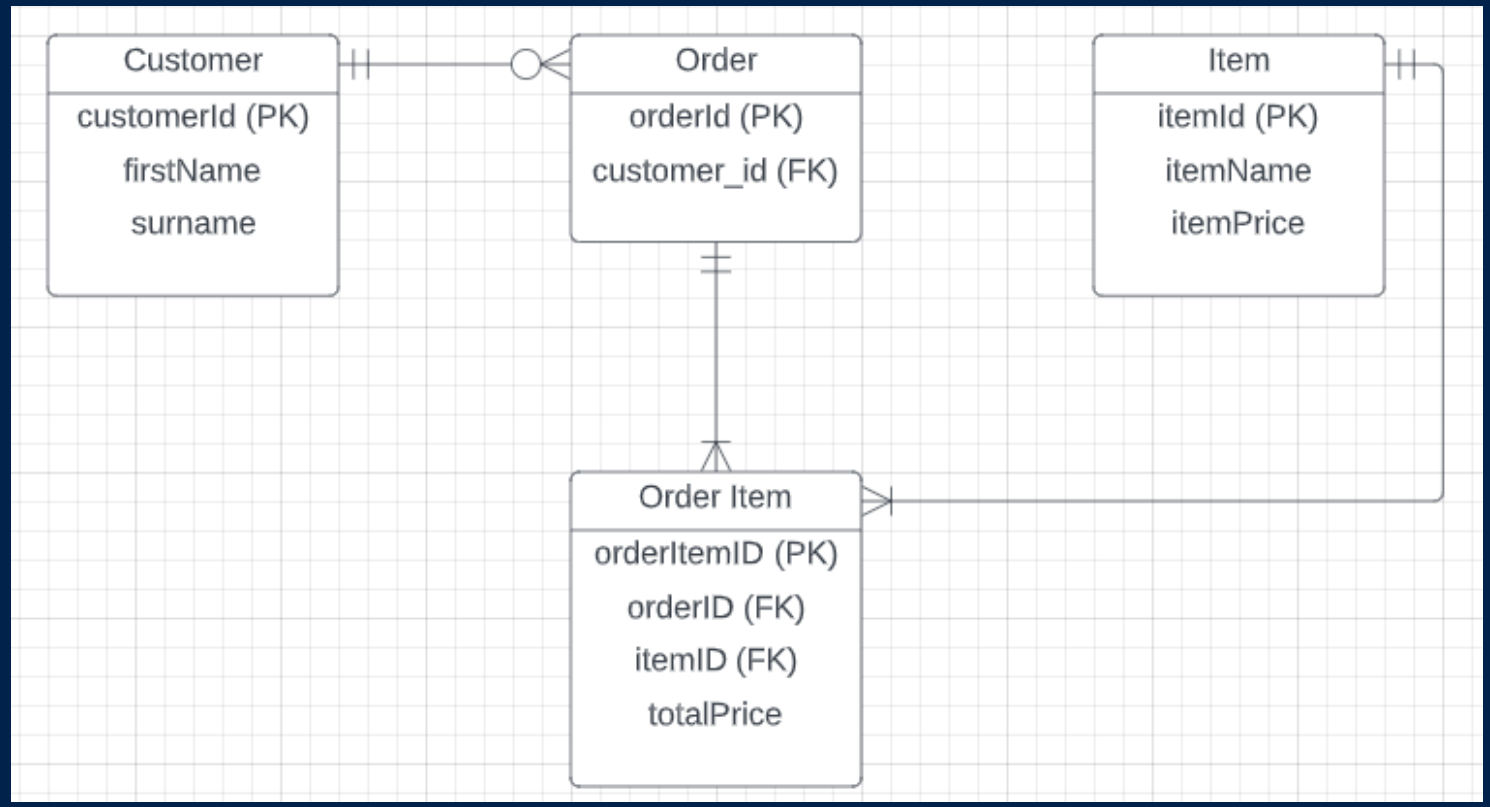
https://trello.com/b/0ReHXEuq/ims-project-management

Risk Assessment

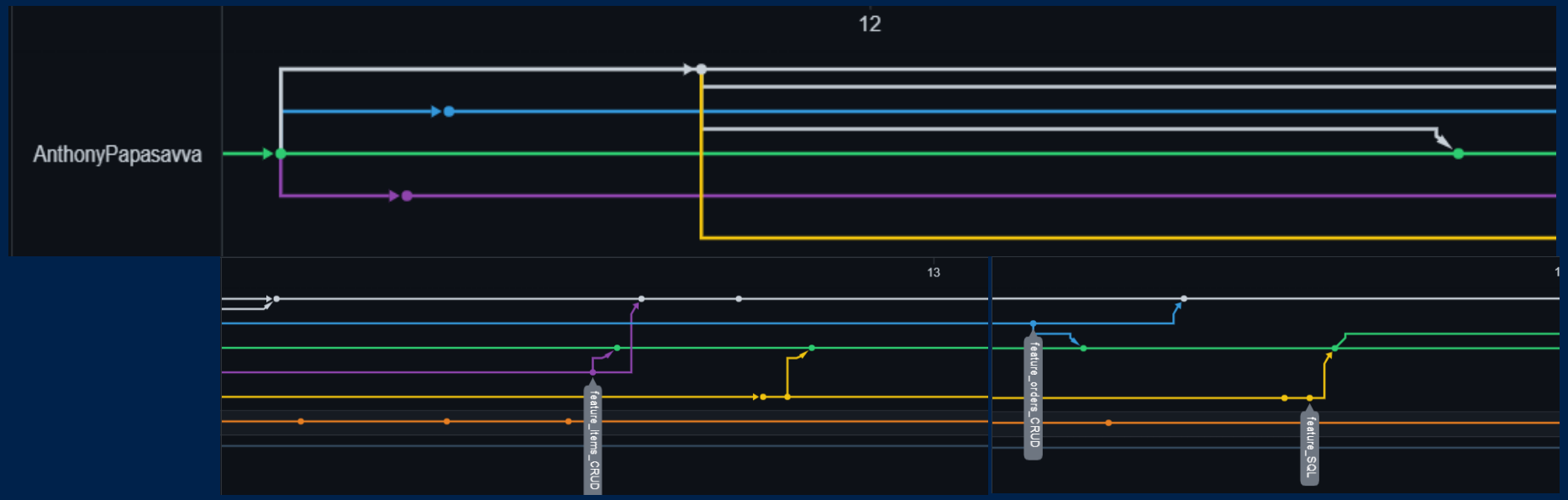
	Risk	Statement	Response	Objective	Likelihood	Impact	Risk Level
1	Ease of access for user - Local repository versus RDS	Using a local repo and neglecting that user will be using remotely can create barriers	Ensure correct steps and details are provided so that external users can access and use your code	Update readme file and check functionality	Possible	Moderate	6
2	Time Management and Project Organisation	Adding unnecessary functionality can waste time and time estimates of tasks may be unrealistic or inaccurate	Use project management board and stick to schedule and follow project specification	Produce the Minimum Viable Product as per the project specification to the best of your ability	Likely	Major	12
3	Problems associated with data transfer from old, or legacy, system(s)	Forking from a 2 year old repository; code, IDEs, etc. may malfunction	Check integration; software is backwards compatible and correct versions are used	Both developer and future users can interact with all code seamlessly	Unlikely	Major	4
4	Reliability of the system	Inexperienced developer working with unfamiliar code and/or concepts	Review and update documentation, code, and apply theory regularly	Maximise reliability for demonstration	Expected	Major	16
5	Communicating functionality to non-technical users (or from an inexperienced developer)	Improper use of terminology and/or limited understanding of concepts and project material causing confusion	Familiarise yourself with relevant, project specific, material throughout project	Explain processes, development decisions, and answer questions adequately	Likely	Moderate	9
6	Low test coverage	Low percentage test coverage increases chances of containing undetected software bugs	Relatively simple to assess and fix through repeated testing throughout development	Aim for 60-80%+ to increase confidence in and simplicity of code, and improve overall functionality	Possible	Moderate	6
7	Merge conflicts with Git repository	Conflicting edits and/or deletion of multiple branches in same project causing project disruption	Unlikely to occur in a non-teamwork environment, however not impossible	Understand the functionality of the feature branch model and use appropriately throughout project	Unlikely	Minor	2
8	Dehydration - specific to forecasted heatwave	Working without proper hydration in high heat causes lightheadedness, feeling weaker, and having less energy	Hydrate regularly, rest periodically, and use a fan or A/C (if baller) throughout hottest intervals of the day	Stay hydrated to maintain productive and in good health	Expected	Moderate	12

		IMPACT				
		Minimal	Minor	Moderate	Major	Severe
LIKELIHOOD	Guaranteed	5	10	15	20	25
	Expected	4	8	12	16	20
	Likely	3	6	9	12	15
	Possible	2	4	6	8	10
	Unlikely	1	2	3	4	5

Entity Relationship Diagram (ERD)



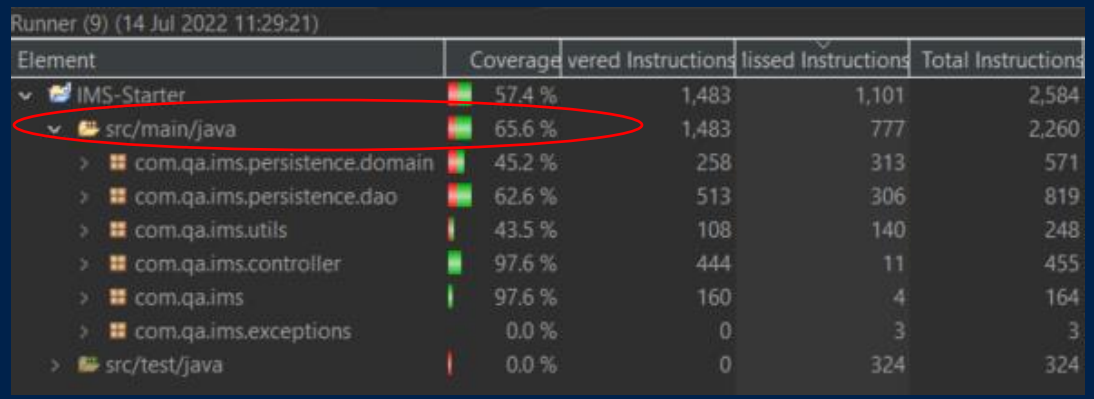
GitHub Network Graph



Testing + Coverage

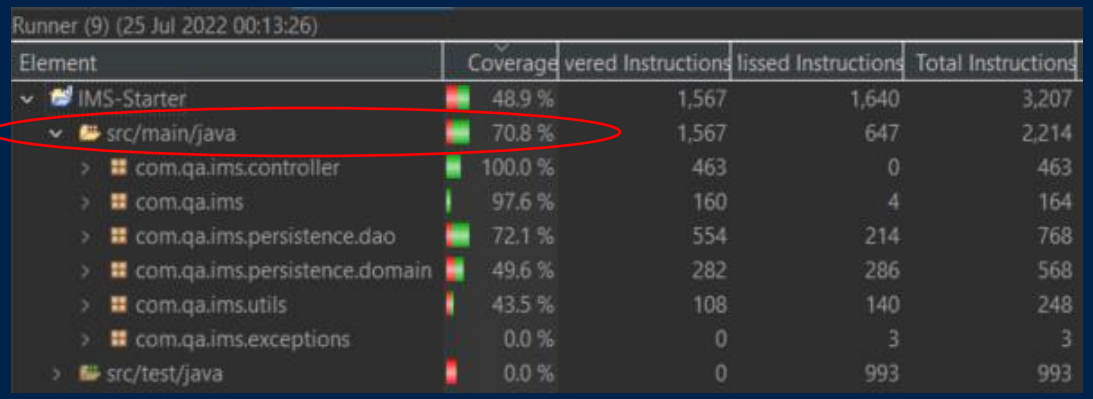
Initial testing resulted in coverage of 65.6%, as shown below.

Increased testing coverage to 70.8%, as shown below.



Runner (9) (14 Jul 2022 11:29:21)

Element	Coverage	vered Instructions	lissed Instructions	Total Instructions
IMS-Starter	57.4 %	1,483	1,101	2,584
src/main/java	65.6 %	1,483	777	2,260
com.qa.ims.persistence.domain	45.2 %	258	313	571
com.qa.ims.persistence.dao	62.6 %	513	306	819
com.qa.ims.utils	43.5 %	108	140	248
com.qa.ims.controller	97.6 %	444	11	455
com.qa.ims	97.6 %	160	4	164
com.qa.ims.exceptions	0.0 %	0	3	3
src/test/java	0.0 %	0	324	324



Runner (9) (25 Jul 2022 00:13:26)

Element	Coverage	vered Instructions	lissed Instructions	Total Instructions
IMS-Starter	48.9 %	1,567	1,640	3,207
src/main/java	70.8 %	1,567	647	2,214
com.qa.ims.controller	100.0 %	463	0	463
com.qa.ims	97.6 %	160	4	164
com.qa.ims.persistence.dao	72.1 %	554	214	768
com.qa.ims.persistence.domain	49.6 %	282	286	568
com.qa.ims.utils	43.5 %	108	140	248
com.qa.ims.exceptions	0.0 %	0	3	3
src/test/java	0.0 %	0	993	993

Demonstration

Now for a demonstration of the IMS functionality through Java with reference to MySQL to visualise what we are doing.

Questions?

- What was the most difficult part of the project?
- Is there anything given the time you'd immediately refactor?
- Was there starter code you felt you could improve?
- How does the customer "readAll" test work?
  a. Define the process in steps
- Where could I potentially automate the creation of the IMS database in the application?

Reflection

What went well?

- Enjoyed the challenge of the task.
- Used Feature-branch-model throughout.
- Planning and documentation.
- Made considerable progress despite multiple setbacks.

Improvements?

- 80%+ test coverage, up from 70.8%.
- Aim for full functionality of MVP before additional functionality.
- UML Diagram