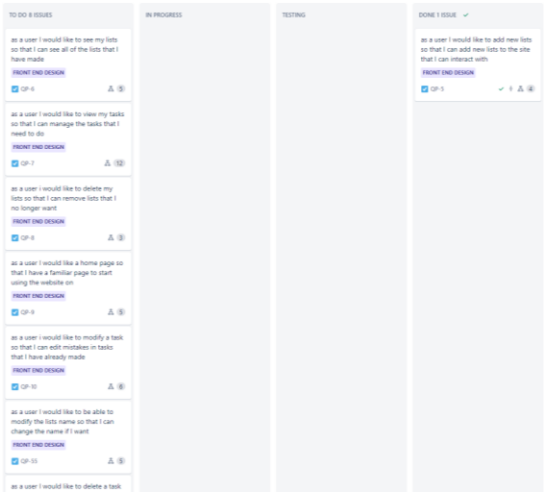
Todo List Project

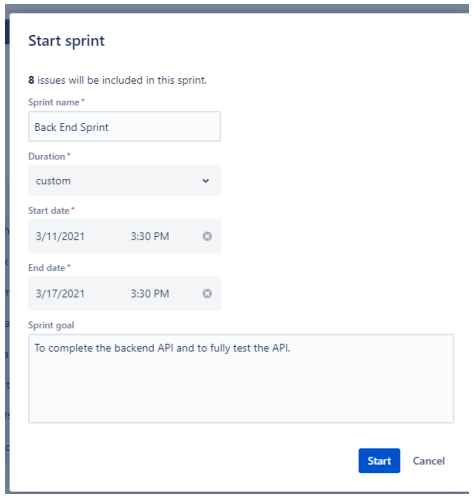
This is a presentation on my development of a Todo list application designed using enterprise architecture techniques.

Introduction

- I was tasked with creating a to do list application and implementing the knowledge that I have gained over the last few weeks.

Concept



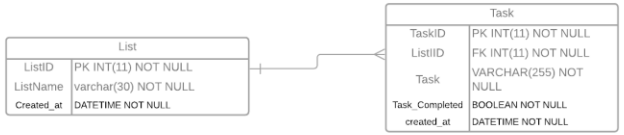


Risk Assessment

Impact vs Likelihood

	Negligible	Minor	Moderate	Significant	Severe
Very Likely					
Likely					
Possible			Internet cutting out.	Database corruption.	Complexity
Unlikely		Losing my local repository.		Computer breaking.	Missing the products deadline
Very Unlikely		Procrastination.		Misinterpreting the project requirements.	Bugs in the production code.

Potential risks are:
My computer breaking.
Internet cutting out.
Local repository being deleted or corrupted.
Database being corrupted or modified in a bad way.
bugs in code making it past test and into release.
Missing the deadline.
Misinterpreting the project specification.
Procrastination.
Complexity



- My approach to this problem was to first create the documentation such as screen designs, ERDs, UMLs, risk matrixes and to read the specification thoroughly from there I made my Kanban board and started developing.

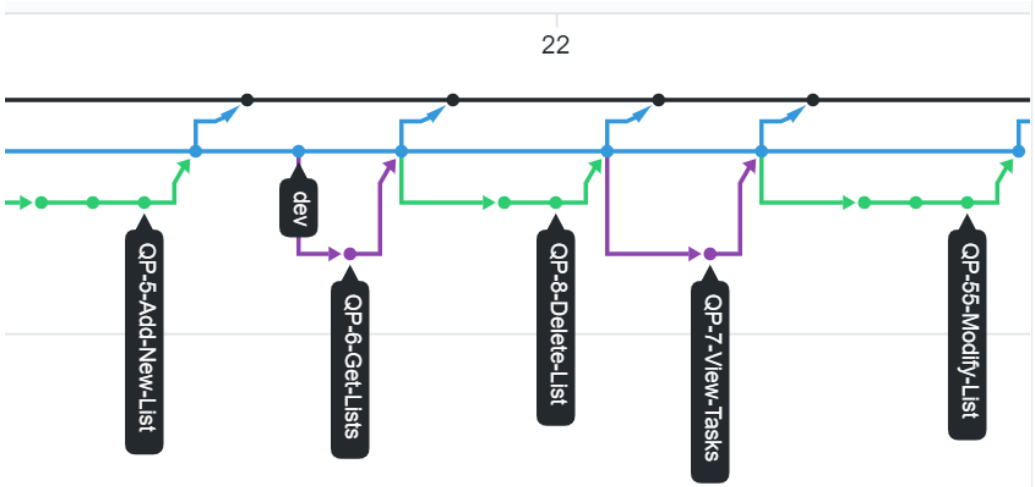
Sprint plan

- In this project I needed to implement CI using the FBM, a PM board, a relational database, a back end built using Java, a working version of my build, to run my code through unit testing, acceptance testing and integration testing, and to use tools like SonarQube, to make a front end and connect that to my backend and to reach 80% test coverage and I have managed to reach all these goals as explained throughout this presentation.
- I had planned on making a more ambitious project by adding a more complex Api, a more robust database, a better looking and feeling front end and having more thorough and robust tests but due to time and other complications these ambitions had to be removed from the project.
- I also made sure to factor in my risk matrix when making any decisions to reduce the chance of any risks affecting my project.

Consultant journey

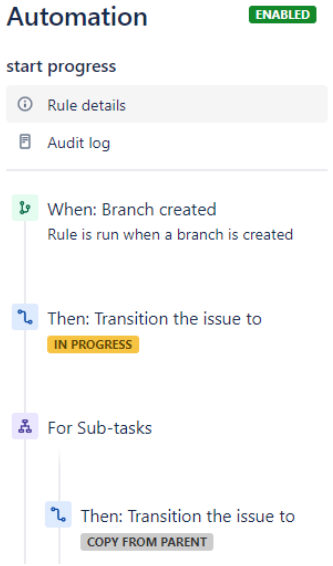
- To make this project I have had to learn:
- Spring Boot
- Html, CSS, JS
- SonarQube
- Multi-tier architecture
- Selenium
- Integration and acceptance testing

Continuous integration

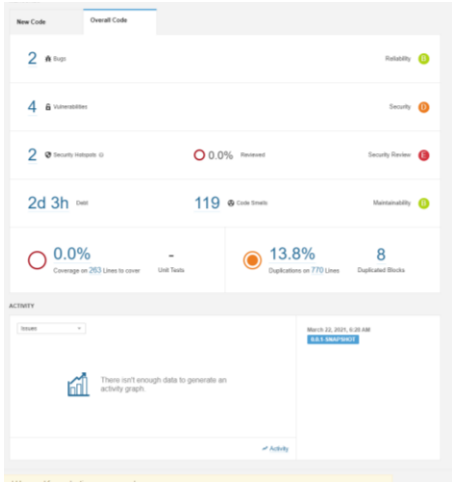


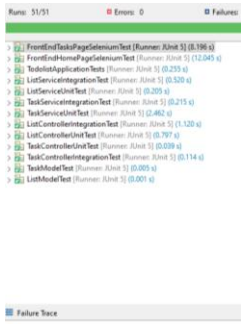
- The screenshot above shows a section of my Git branch and how I have followed the FBM whilst developing my application.

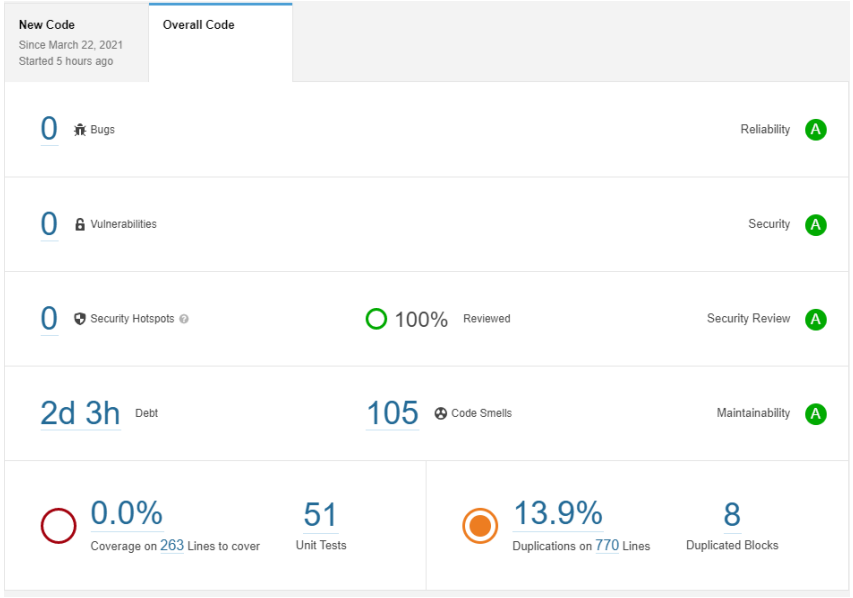
- My approach to CI was standard I used Git for my local repository and GitHub to manage my online repository. This helped with project management since I was easily able to manage my Kanban board with GitHub using Jira and Jira workflow.

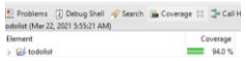


- The screenshot on the left shows Jira automation this rule states that when a branch is created the issue should be moved to in-progress and so should the child issues(tasks).









Testing

- Coverage – My overall coverage for my project was 93% this was made up of unit tests, integration tests and acceptance tests.
- Static Analysis - My static analysis started off with an average score but after going through the feedback from SonarQube I was able to reduce the problems and improve my score.

Demonstration

- I am now going to demonstrate the application and a couple stories I will focus on the front-end stories since these are easier to demonstrate.

Sprint review and retrospective

- I was able to complete the MVP and meet the specification however I didn't meet the level of quality that I would have liked this is because during development I ran into some issues the majority were fixed during development and I have learned from and I feel I have grown as a developer but some issues such as the FBM breaking for my VC, my selenium tests breaking under certain conditions and my lack of production testing and where I feel I have fallen short and these are going to be the areas that I plan to focus on for my next project and for my personal projects. On the other hand, I feel that my backend and front-end code was my strongest point these were developed the quickest and these were two areas where I exceeded my goals. I also spoke about risks that I feel would potentially pose a risk and some risks that did end up causing issues were my computer failing which was very unexpected but cost me almost 2 days of development time and poor time management due to losing those 2 days and not reorganising my time properly but like with the IMS project this is something that I plan to tackle and fix.

Conclusion and questions

- To conclude I feel have shown the development of my project throughout this presentation and how I managed to create the application and the techniques that I used, my future steps are to improve my time management and to practice my API development and testing in personal projects to improve my skills.