Project 2 - TDL

Work and presentation by Henry Oliver-Edwards

Introduction

- Part of the SDET2020DEC cohort
- This project has a Java back-end using the spring framework and a HTML/CSS/JavaScript front-end
- The application provides a web based interface and allows the user to create generic lists and create generic tasks that go inside these lists
- These lists and tasks are then either stored in a temporary (H2 database for development/testing) or permanent (MySQL database for production)

Project flow and objectives

User Loads webpage

User chooses either List or Task option

Create Read Update Delete

Server handles this request

Data is then returned to user and displayed on the webpage

Project technologies and their usefulness

- Java and the JVM
- Maven
- JUnit
- Mockito
- SonarQube

- The project is entirely programmed in java and runs in the JVM
- Maven is a build tool which allows for the packing of java files into a war
- JUnit is a unit testing tool which individually tests the components of your code
- Mockito is a behavioral testing framework that allows us to test our codes output.
- SonarQube analyses code for bugs, security flaws and linting

Project technologies and their usefulness

- Spring framework
- Lombok
- Selenium

- This project is based entirely on the spring framework and modules from this such as the web module
- Lombok is used to generate boilerplate code such as constructors, getters, setters etc.
- Selenium provides a way to acceptance test the product emulating a browser

Testing

Overall coverage at 88%

Test methods used

- Unit testing
- Integration testing
- Acceptance testing

- Unit testing provides a way to test individual classes or objects in a project so that they can work on their own as expected
- Integration testing provides a way to test how multiple objects in a system may interface with each other
- Acceptance testing provides a way to test the business logic of an application

Demonstration time!

Sprint review

Completed

- All core CRUD functionality (for lists and tasks)
- A minimal, fast and functional front-end interface
- Over 88% test coverage

Not completed

- The ability to search/delete by name rather than ID (as IDs may be hard to remember)
- Complete acceptance test coverage

Sprint retrospective

What went well

- Completed A vast amount of work including development/UX/testing in a small amount of time
- Utilized many project technologies such as Jira/ Git/etc

What could be improved

- Prioritize testing over other features or a nice UI design

Thank you for listening any questions?