To Do List Web Application(TDL)

By: Raimonds Mezals

Introduction

The project:

- Documentation
- A To Do List Web Application that would be able to create a to do lists with CRUD functionality and have an ability to add or remove items from specific lists.
- Junit and integration testing to test the back-end of the system
- Selenium testing to test the front end of the system

L Workbench 8.

Consultant Journey

Learned skills Week 5:

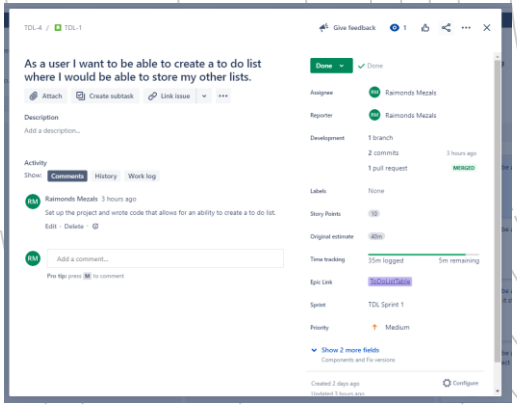
- Java
- Eclipse
- Git and GitHub
- MySQL and H2
- Agile scrum
- Junit
- Maven
- SonarQube

Learned skills Week 8:

- Spring
- Visual Studio Code
- Postman
- Integration Testing
- Mockito
- Selenium
- HTML
- CSS
- JavaScript

Visual Studio Code

Continuous Integration Version Control: Jira



TDL Sprint 1

Create the necessary documentation and the back end of the system.

0 days remaining | Complete sprint

Only My Issues | Recently Updated

TO DO

- Create a UML diagram in order to understand the functionality of the code. — Documentation — TDL-19

IN PROGRESS

- As a user I want to be able to create an item that I could add to my shopping list. — ItemsTable — 10m — TDL-2
- As a user I want to be able to read all items so i know the contents of my items table. — ItemsTable — 3m — TDL-9
- As a user I want to be able to read an items table by id so i can see the contents of specific object that it stores. — ItemsTable — 3m — TDL-10
- As a user I want to be able to update an items table by id so i can update the contents of the object that it stores. — ItemsTable — 0m — TDL-11

DONE

- Create a MoSCoW diagram so that I am able to see the most important features of the project. — Documentation — 0m — ~~TDL-17~~
- Create an ERD diagram so that I can understand the functionality of the database. — Documentation — 0m — ~~TDL-18~~
- Create a Risk Assessment and assess any potential risks that could arise during or after the project. — Documentation — 0m — ~~TDL-20~~
- Create a Risk Matrix to show all the potential risks on a graph. — Documentation — 0m — ~~TDL-21~~

TDL Sprint 2

Complete the project

1 day remaining | Complete sprint

Only My Issues | Recently Updated

TO DO

- Use Selenium to test the front end of the Project — Testing — 2h — TDL-31
- Create a Presentation to present the project. — Documentation — 1h — TDL-29

IN PROGRESS

- Perform unit and integration testing for the back end. — Testing — 5h — TDL-30

DONE

- As a user I want to have a pleasant and clear interface that I can use in order to choose what functionality I want to perform with my to do list. — Front-End — 0m — ~~TDL-13~~
- As a user I want to be able to enter data on the web page and have it successfully transferred to a database. — Front-End — 0m — ~~TDL-14~~
- Create main page using HTML and Bootstrap. — Front-End — 20m — ~~TDL-25~~
- Create a page for to do list (Bag) using HTML and Bootstrap. — Front-End — 30m — ~~TDL-26~~
- Create a page for items table using HTML and Bootstrap. — Front-End — 0m

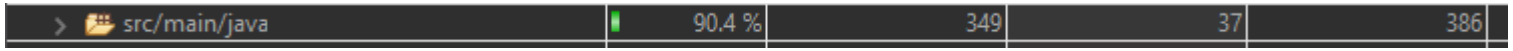
Quickstart

Testing

Selenium Test Create Bag

2021-02-20 22:22:07 | 2021-02-20 22:22:08 | 0h 0m 0s+460ms

STATUS	TIMESTAMP	DETAILS
ⓘ	22:22:07	Given - I can access http://localhost:9090/index.html
ⓘ	22:22:07	When - I navigate to the Bag Modification table
ⓘ	22:22:07	When - I enter values for a bag and crate a bag
ⓘ	22:22:08	Then - I should see Successfully Created a (createTest) List
✔	22:22:08	Successfully created a list!



Junit and integration testing

- BagController
- BagService

Selenium testing

- Front-end

Demonstration

Sprint Review

Minimum Viable Product

User Stories

Specific Tasks

Bugs

Sprint Retrospect

What Went Well

- Product meets a minimum viable product
- Junit and Integration tests covered over 80% of the project code.

What Could Be Improved

- More time was needed in order to refactor the code, time management could be improved.
- Selenium testing could be improved given more time.
- More time could be spent to improve the design and general aesthetics of the front-end of the application.

Conclusion

- The overall project and its functionality was a success.
- Junit and integration tests covered over 80% of the main code for the back-end
- Relevant documentation was created on time

Questions?