To-Do List Project

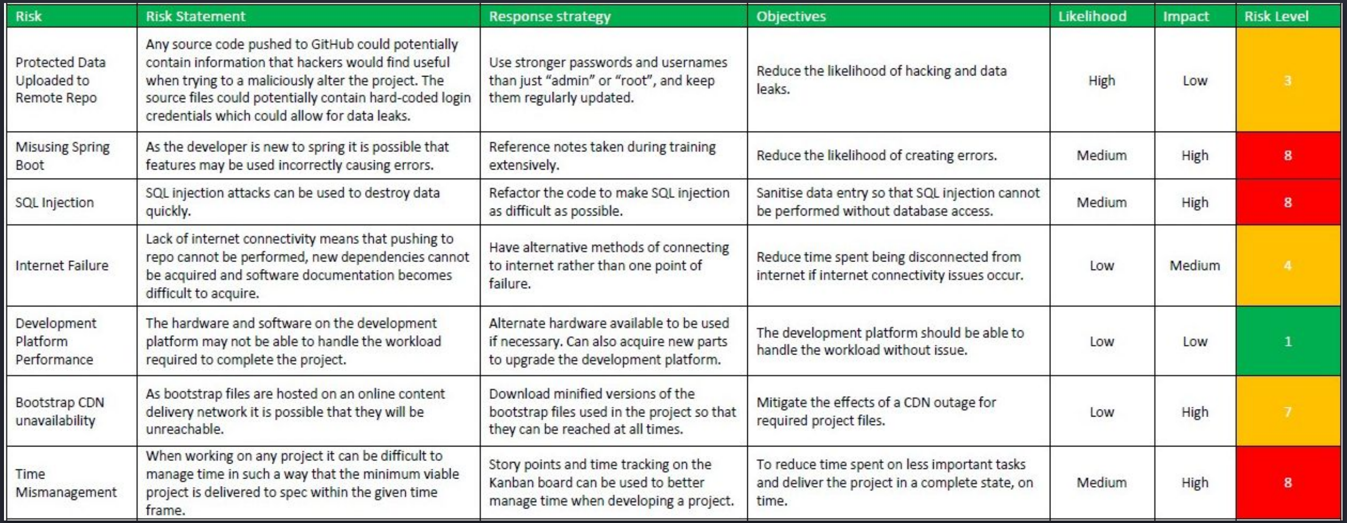
20DecSDET2 // Cameron Guthrie

Introduction

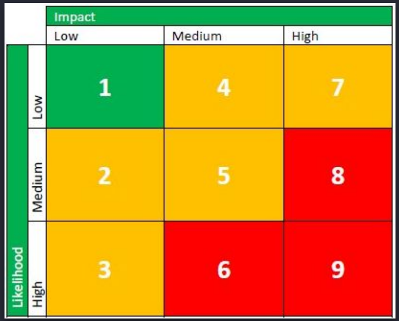
How did I approach the specification?

- ➔ **Fundamentals**
- ➔ **Minimum Viable Product**
- ➔ **The Scope**

Risk



Risk	Risk Statement	Response strategy	Objectives	Likelihood	Impact	Risk Level
Protected Data Uploaded to Remote Repo	Any source code pushed to GitHub could potentially contain information that hackers would find useful when trying to a maliciously alter the project. The source files could potentially contain hard-coded login credentials which could allow for data leaks.	Use stronger passwords and usernames than just “admin” or “root”, and keep them regularly updated.	Reduce the likelihood of hacking and data leaks.	High	Low	3
Misusing Spring Boot	As the developer is new to spring it is possible that features may be used incorrectly causing errors.	Reference notes taken during training extensively.	Reduce the likelihood of creating errors.	Medium	High	8
SQL Injection	SQL injection attacks can be used to destroy data quickly.	Refactor the code to make SQL injection as difficult as possible.	Sanitise data entry so that SQL injection cannot be performed without database access.	Medium	High	8
Internet Failure	Lack of internet connectivity means that pushing to repo cannot be performed, new dependencies cannot be acquired and software documentation becomes difficult to acquire.	Have alternative methods of connecting to internet rather than one point of failure.	Reduce time spent being disconnected from internet if internet connectivity issues occur.	Low	Medium	4
Development Platform Performance	The hardware and software on the development platform may not be able to handle the workload required to complete the project.	Alternate hardware available to be used if necessary. Can also acquire new parts to upgrade the development platform.	The development platform should be able to handle the workload without issue.	Low	Low	1
Bootstrap CDN unavailability	As bootstrap files are hosted on an online content delivery network it is possible that they will be unreachable.	Download minified versions of the bootstrap files used in the project so that they can be reached at all times.	Mitigate the effects of a CDN outage for required project files.	Low	High	7
Time Mismanagement	When working on any project it can be difficult to manage time in such a way that the minimum viable project is delivered to spec within the given time frame.	Story points and time tracking on the Kanban board can be used to better manage time when developing a project.	To reduce time spent on less important tasks and deliver the project in a complete state, on time.	Medium	High	8



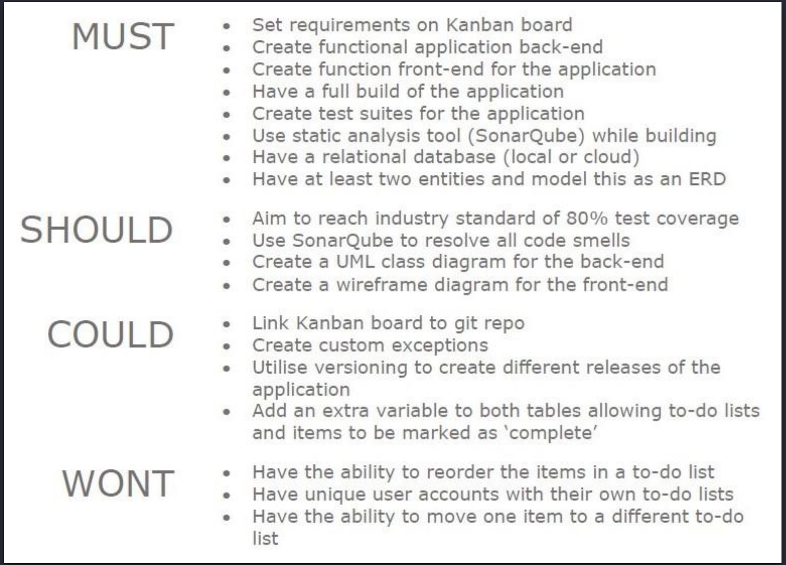
		Impact		
		Low	Medium	High
Likelihood	Low	1	4	7
	Medium	2	5	8
	High	3	6	9

MoSCoW

An useful method to help organise project goals.

➔ **Must and Should**
Cover the Domain and Scope

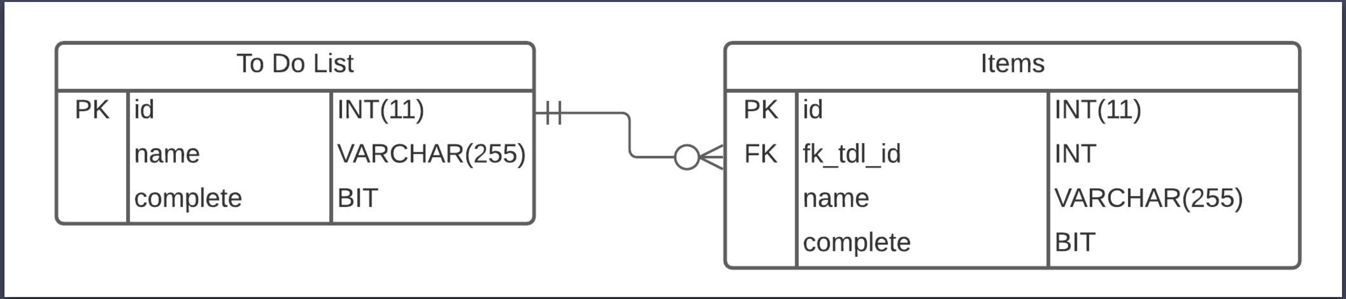
➔ **Could and Wont**
Are for stretch goals and extras.



MUST	• Set requirements on Kanban board • Create functional application back-end • Create function front-end for the application • Have a full build of the application • Create test suites for the application • Use static analysis tool (SonarQube) while building • Have a relational database (local or cloud) • Have at least two entities and model this as an ERD
SHOULD	• Aim to reach industry standard of 80% test coverage • Use SonarQube to resolve all code smells • Create a UML class diagram for the back-end • Create a wireframe diagram for the front-end
COULD	• Link Kanban board to git repo • Create custom exceptions • Utilise versioning to create different releases of the application • Add an extra variable to both tables allowing to-do lists and items to be marked as 'complete'
WONT	• Have the ability to reorder the items in a to-do list • Have unique user accounts with their own to-do lists • Have the ability to move one item to a different to-do list

Entity Relationship Diagram

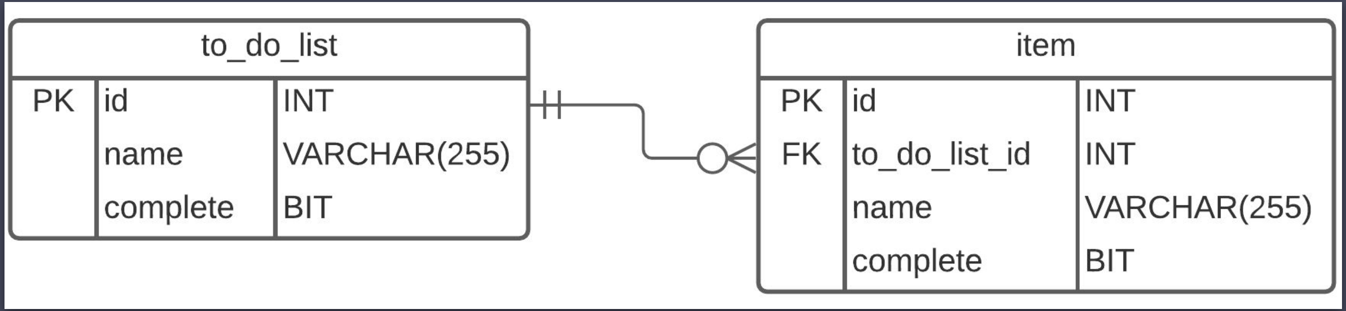
Predicted



To Do List		
PK	id	INT(11)
	name	VARCHAR(255)
	complete	BIT

Items		
PK	id	INT(11)
FK	fk_tdl_id	INT
	name	VARCHAR(255)
	complete	BIT

v0.1 / Final

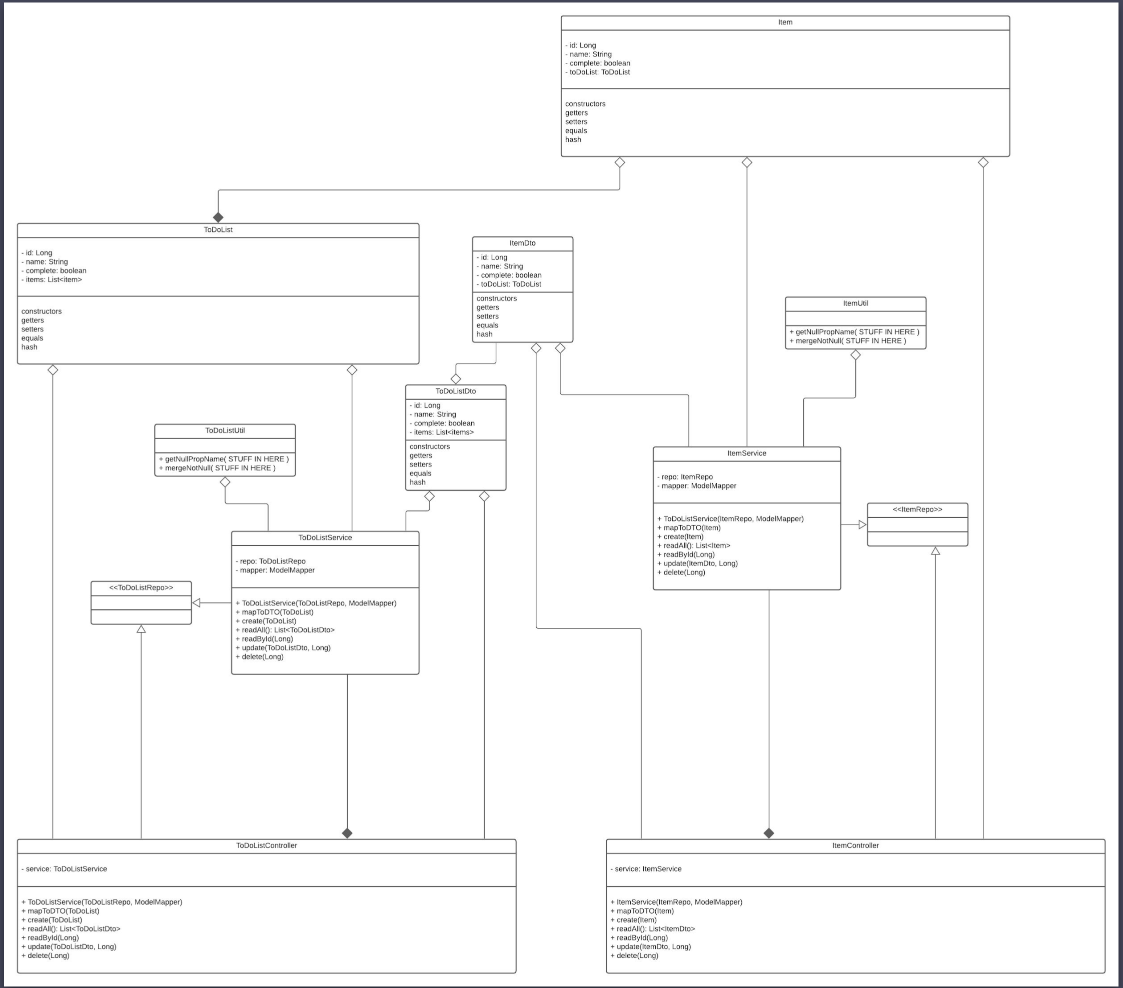


to_do_list		
PK	id	INT
	name	VARCHAR(255)
	complete	BIT

item		
PK	id	INT
FK	to_do_list_id	INT
	name	VARCHAR(255)
	complete	BIT

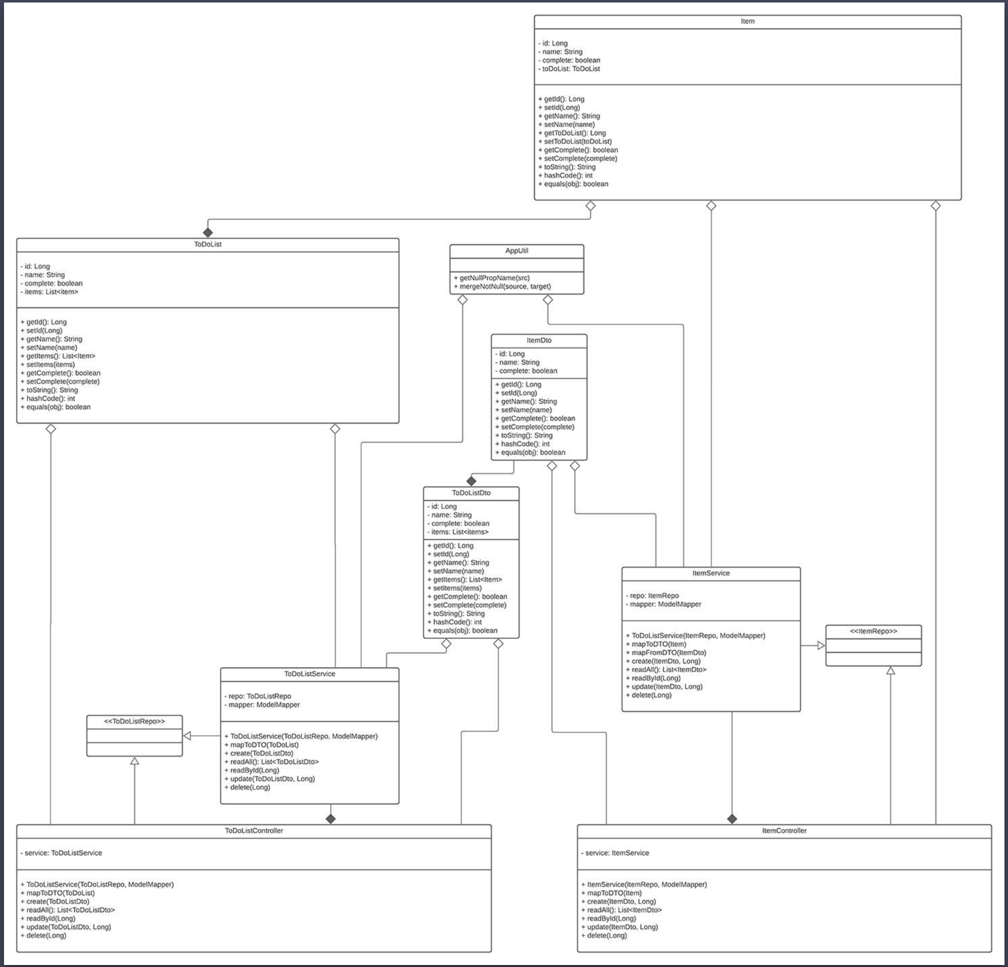
Unified Modeling Language Class Diagram

Predicted



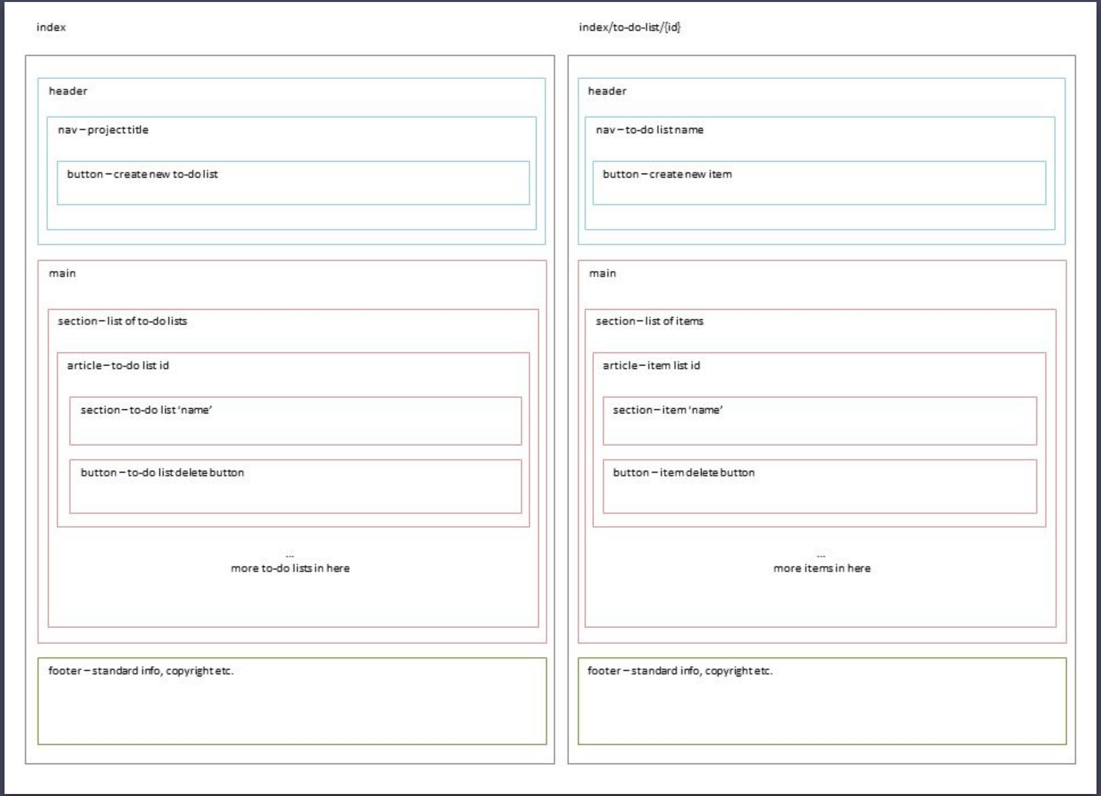
Unified Modeling Language Class Diagram

v0.1 / Final



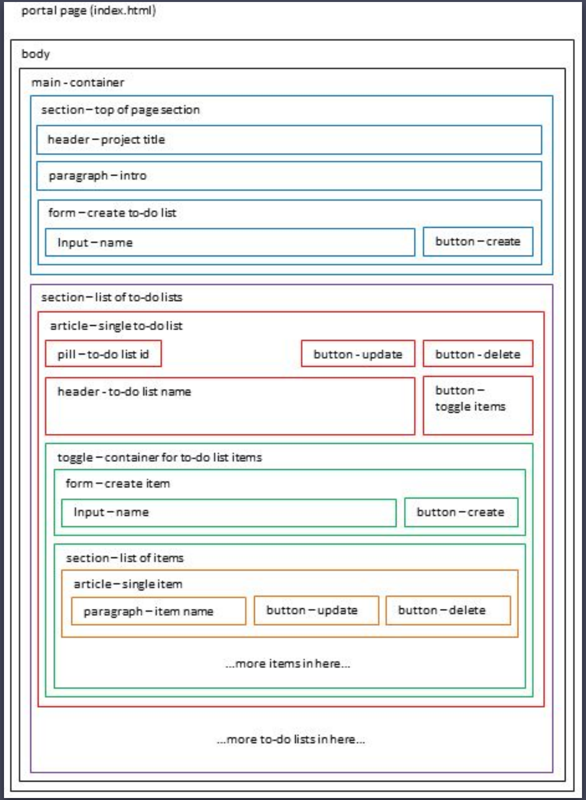
Wireframe

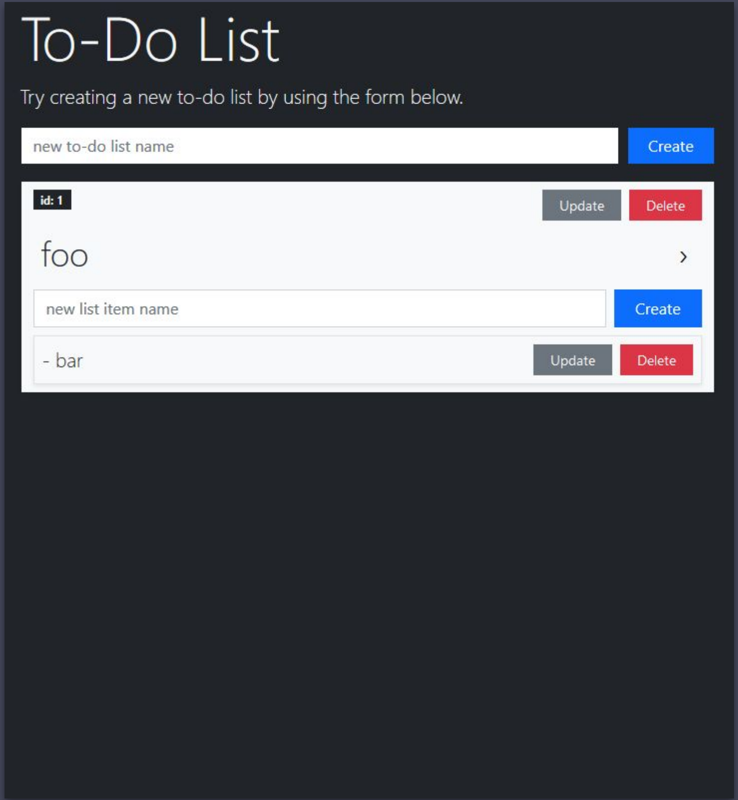
Predicted



Wireframe

v0.1 / Final





To-Do List

Try creating a new to-do list by using the form below.

Consultant Journey

QA Academy has been a joy to learn at and has provided me with all of the skills that I needed to complete this project.

➔ **HTML, CSS, JS**
Front-end web development to create responsive user interfaces

➔ **Spring Boot**
Makes developing Java projects much easier

➔ **Selenium**
The front-end can be tested using Java in the back-end!

Version Control System

Pick a User Story or Task on Jira

For every Task and User Story I create a new branch in my local repo.

Write some working code

Harder than it sounds.

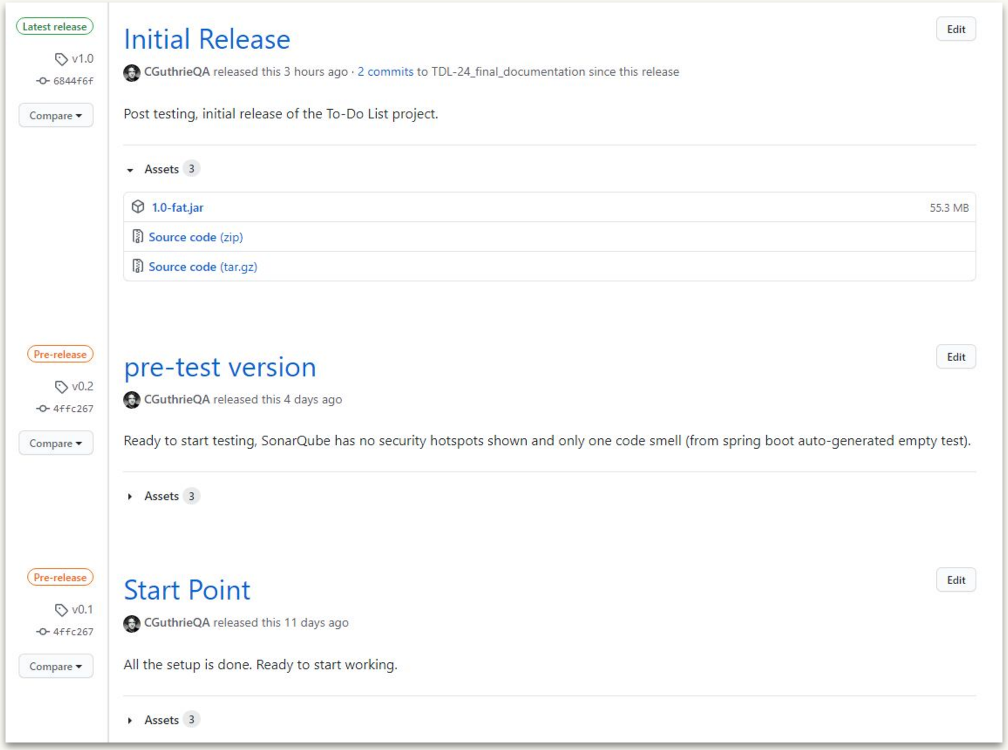
Commit and push to feature branch

I merge them into my dev branch afterwards.

More Version Control!

Releases

On GitHub



Latest release
v1.0
6844f6f
Compare

Initial Release

Edit

CGuthrieQA released this 3 hours ago · 2 commits to TDL-24_final_documentation since this release

Post testing, initial release of the To-Do List project.

Assets 3

1.0-fat.jar	55.3 MB
Source code (zip)	
Source code (tar.gz)	

Pre-release
v0.2
4ffc267
Compare

pre-test version

Edit

CGuthrieQA released this 4 days ago

Ready to start testing, SonarQube has no security hotspots shown and only one code smell (from spring boot auto-generated empty test).

Assets 3

Pre-release
v0.1
4ffc267
Compare

Start Point

Edit

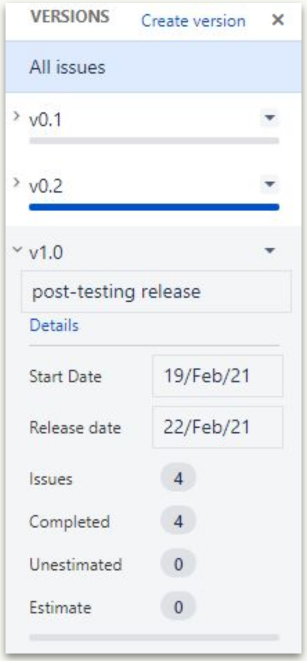
CGuthrieQA released this 11 days ago

All the setup is done. Ready to start working.

Assets 3

Jira was also used to track versions

Issues could be assigned directly to the version and tracked

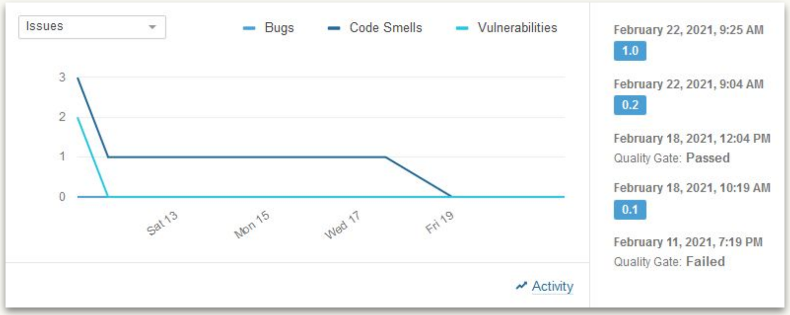


post-testing release

Start Date	19/Feb/21
Release date	22/Feb/21
Issues	4
Completed	4
Unestimated	0
Estimate	0

SonarQube was used as well

To build useful graphs automatically



February 22, 2021, 9:25 AM
1.0

February 22, 2021, 9:04 AM
0.2

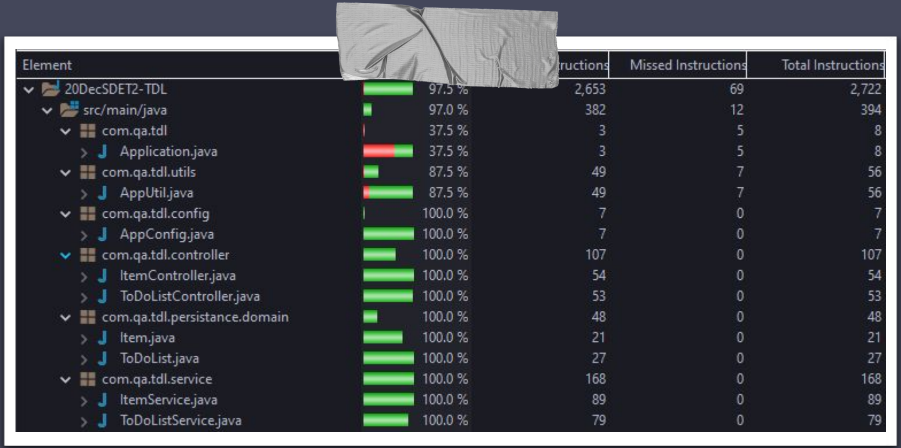
February 18, 2021, 12:04 PM
Quality Gate: Passed

February 18, 2021, 10:19 AM
0.1

February 11, 2021, 7:19 PM
Quality Gate: Failed

How I handled testing to reach **97% coverage** in src/main/java

- ➔ **Write**
  Create code fo a test.
- ➔ **Run**
  To check if it passes and works as intended.
- ➔ **Refactor**
  Remove duplicate code, simplify it.



Element		ructions	Missed Instructions	Total Instructions
20DecSDET2-TDL	97.5 %	2,653	69	2,722
src/main/java	97.0 %	382	12	394
com.qa.tdl	37.5 %	3	5	8
Application.java	37.5 %	3	5	8
com.qa.tdl.utils	87.5 %	49	7	56
AppUtil.java	87.5 %	49	7	56
com.qa.tdl.config	100.0 %	7	0	7
AppConfig.java	100.0 %	7	0	7
com.qa.tdl.controller	100.0 %	107	0	107
ItemController.java	100.0 %	54	0	54
ToDoListController.java	100.0 %	53	0	53
com.qa.tdl.persistance.domain	100.0 %	48	0	48
Item.java	100.0 %	21	0	21
ToDoList.java	100.0 %	27	0	27
com.qa.tdl.service	100.0 %	168	0	168
ItemService.java	100.0 %	89	0	89
ToDoListService.java	100.0 %	79	0	79

CRUD

I made sure to test all of the CRUD functionality of my classes.

That way I knew that my User Stories could be completed without error.

// C reate

// R ead

// U pdate

// D elete

Unit Testing

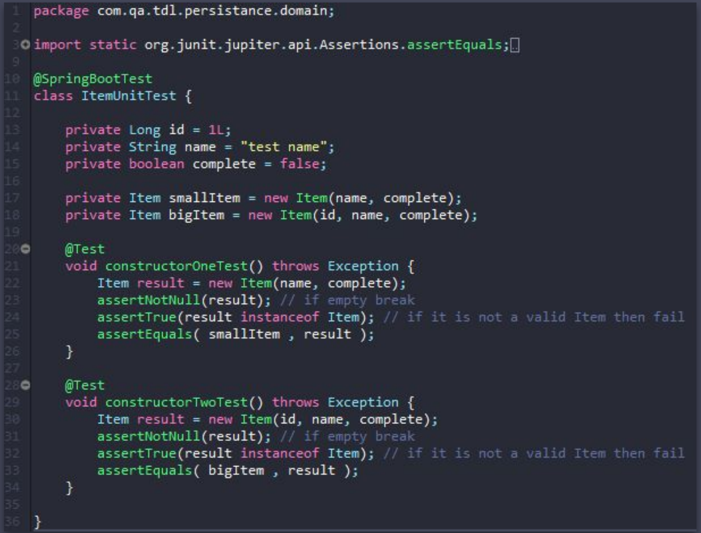
Was used to test the classes.

Covered as many lines of code as possible.

Made sure to write robust assertions.

// missed testing a private constructor

// missed testing the main method

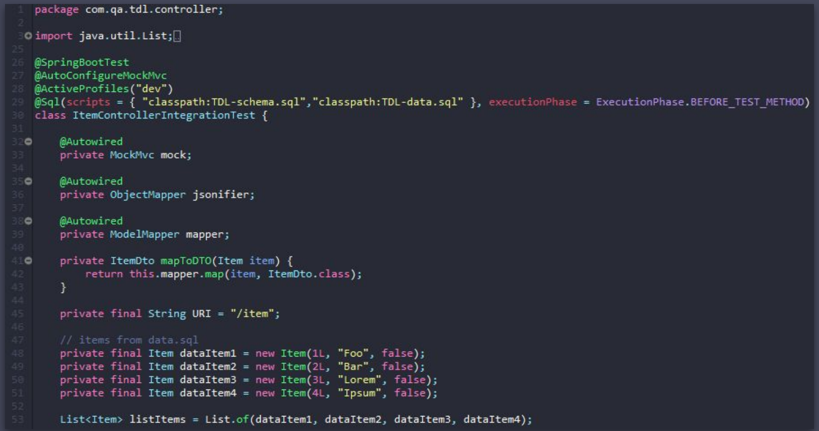


```java
package com.qa.tdl.persistance.domain;

import static org.junit.jupiter.api.Assertions.assertEquals;

@SpringBootTest
class ItemUnitTest {

    private Long id = 1L;
    private String name = "test name";
    private boolean complete = false;

    private Item smallItem = new Item(name, complete);
    private Item bigItem = new Item(id, name, complete);

    @Test
    void constructorOneTest() throws Exception {
        Item result = new Item(name, complete);
        assertNotNull(result); // if empty break
        assertTrue(result instanceof Item); // if it is not a valid Item then fail
        assertEquals( smallItem , result );
    }

    @Test
    void constructorTwoTest() throws Exception {
        Item result = new Item(id, name, complete);
        assertNotNull(result); // if empty break
        assertTrue(result instanceof Item); // if it is not a valid Item then fail
        assertEquals( bigItem , result );
    }

}
```

Integration Testing

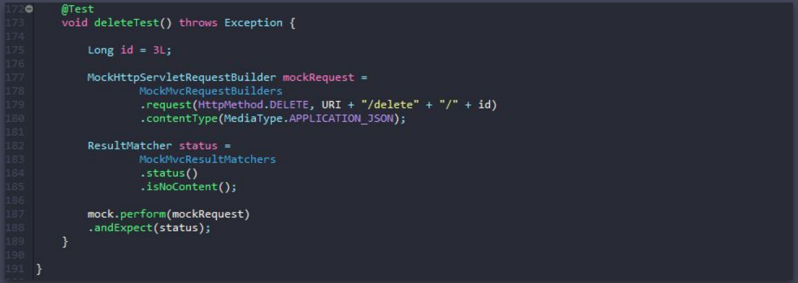
Was used to test link between the controller classes and the service classes.

Used SQL dummy data to assert against.



```java
package com.qa.tdl.controller;

import java.util.List;

@SpringBootTest
@AutoConfigureMockMvc
@ActiveProfiles("dev")
@Sql(scripts = { "classpath:TDL-schema.sql","classpath:TDL-data.sql" }, executionPhase = ExecutionPhase.BEFORE_TEST_METHOD)
class ItemControllerIntegrationTest {

    @Autowired
    private MockMvc mock;

    @Autowired
    private ObjectMapper jsonifier;

    @Autowired
    private ModelMapper mapper;

    private ItemDto mapToDTO(Item item) {
        return this.mapper.map(item, ItemDto.class);
    }

    private final String URI = "/item";

    // items from data.sql
    private final Item dataItem1 = new Item(1L, "Foo", false);
    private final Item dataItem2 = new Item(2L, "Bar", false);
    private final Item dataItem3 = new Item(3L, "Lorem", false);
    private final Item dataItem4 = new Item(4L, "Ipsum", false);

    List<Item> listItems = List.of(dataItem1, dataItem2, dataItem3, dataItem4);
```

...

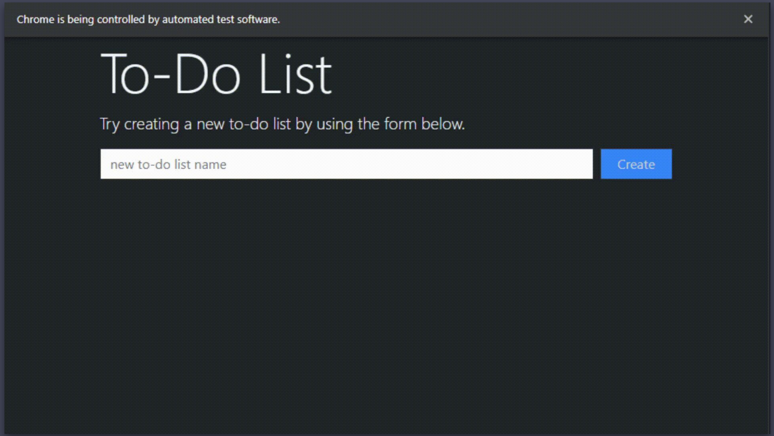


```java
    @Test
    void deleteTest() throws Exception {

        Long id = 3L;

        MockHttpServletRequestBuilder mockRequest =
                MockMvcRequestBuilders
                .request(HttpMethod.DELETE, URI + "/delete" + "/" + id)
                .contentType(MediaType.APPLICATION_JSON);

        ResultMatcher status =
                MockMvcResultMatchers
                .status()
                .isNoContent();

        mock.perform(mockRequest)
        .andExpect(status);
    }

}
```

Acceptance Testing

Used to test user stories on the front-end.

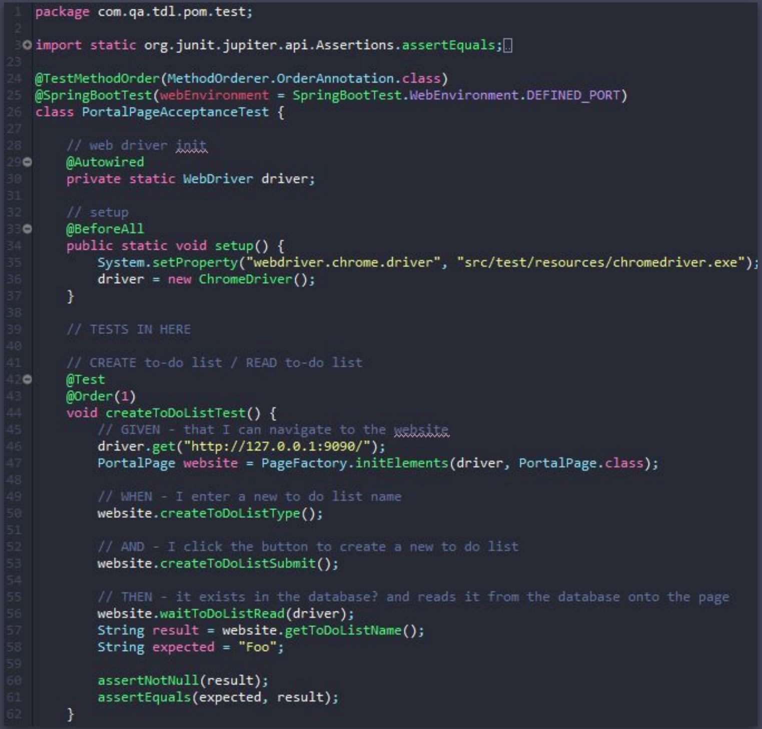
Implemented ordered testing.



Chrome is being controlled by automated test software.

To-Do List

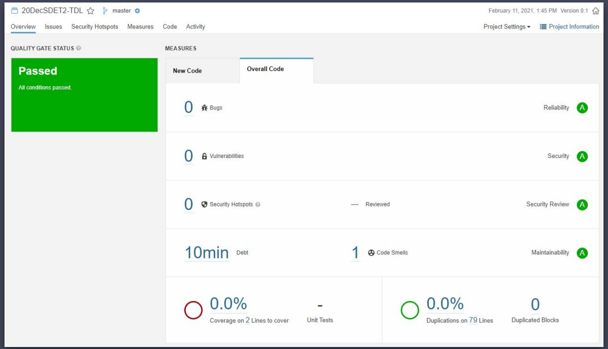
Try creating a new to-do list by using the form below.



```
package com.qa.tdl.pom.test;

import static org.junit.jupiter.api.Assertions.assertEquals;

@TestMethodOrder(MethodOrderer.OrderAnnotation.class)
@SpringBootTest(webEnvironment = SpringBootTest.WebEnvironment.DEFINED_PORT)
class PortalPageAcceptanceTest {

    // web driver init
    @Autowired
    private static WebDriver driver;

    // setup
    @BeforeAll
    public static void setup() {
        System.setProperty("webdriver.chrome.driver", "src/test/resources/chromedriver.exe");
        driver = new ChromeDriver();
    }

    // TESTS IN HERE

    // CREATE to-do list / READ to-do list
    @Test
    @Order(1)
    void createToDoListTest() {
        // GIVEN - that I can navigate to the website
        driver.get("http://127.0.0.1:9090/");
        PortalPage website = PageFactory.initElements(driver, PortalPage.class);

        // WHEN - I enter a new to do list name
        website.createToDoListType();

        // AND - I click the button to create a new to do list
        website.createToDoListSubmit();

        // THEN - it exists in the database? and reads it from the database onto the page
        website.waitToDoListRead(driver);
        String result = website.getToDoListName();
        String expected = "Foo";

        assertNotNull(result);
        assertEquals(expected, result);
    }
```

SonarQube

initial



February 11, 2021, 1:45 PM Version 0.1

QUALITY GATE STATUS

Passed
All conditions passed.

MEASURES

New Code | Overall Code

0 Bugs — Reliability A

0 Vulnerabilities — Security A

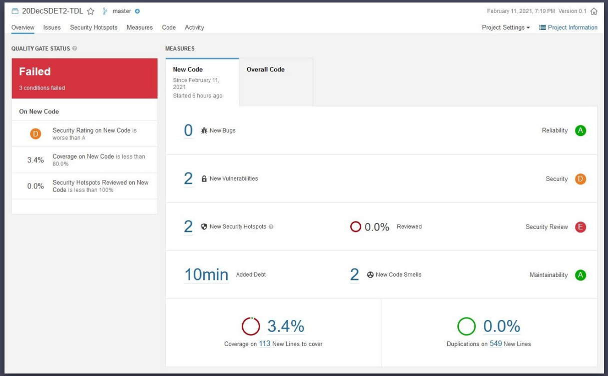
0 Security Hotspots — Reviewed — Security Review A

10min Debt — 1 Code Smells — Maintainability A

0.0% Coverage on 2 Lines to cover — - Unit Tests

0.0% Duplications on 79 Lines — 0 Duplicated Blocks

v0.1



February 11, 2021, 7:19 PM Version 0.1

QUALITY GATE STATUS

Failed
3 conditions failed

On New Code

D Security Rating on New Code is worse than A

3.4% Coverage on New Code is less than 80.0%

0.0% Security Hotspots Reviewed on New Code is less than 100%

MEASURES

New Code (Since February 11, 2021, Started 6 hours ago) | Overall Code

0 New Bugs — Reliability A

2 New Vulnerabilities — Security D

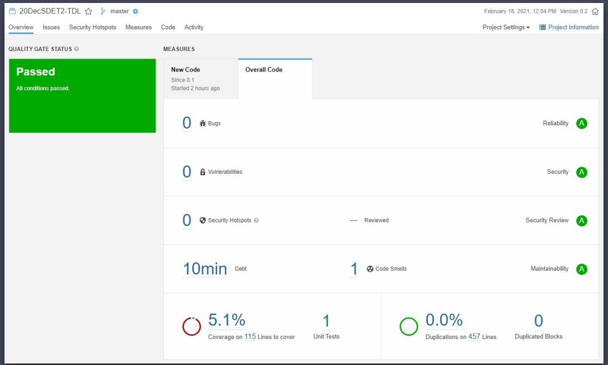
2 New Security Hotspots — 0.0% Reviewed — Security Review E

10min Added Debt — 2 New Code Smells — Maintainability A

3.4% Coverage on 113 New Lines to cover

0.0% Duplications on 549 New Lines

v0.1
fix



February 18, 2021, 12:04 PM Version 0.2

QUALITY GATE STATUS

Passed
All conditions passed.

MEASURES

New Code (Since 0.1, Started 2 hours ago) | Overall Code

0 Bugs — Reliability A

0 Vulnerabilities — Security A

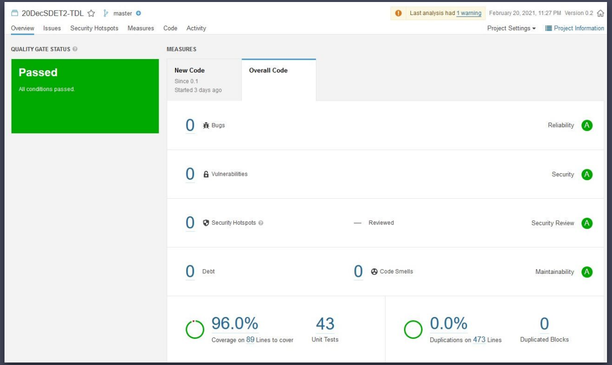
0 Security Hotspots — Reviewed — Security Review A

10min Debt — 1 Code Smells — Maintainability A

5.1% Coverage on 115 Lines to cover — 1 Unit Tests

0.0% Duplications on 457 Lines — 0 Duplicated Blocks

v0.2



Last analysis had 1 warning February 20, 2021, 11:27 PM Version 0.2

QUALITY GATE STATUS

Passed
All conditions passed.

MEASURES

New Code (Since 0.1, Started 3 days ago) | Overall Code

0 Bugs — Reliability A

0 Vulnerabilities — Security A

0 Security Hotspots — Reviewed — Security Review A

0 Debt — 0 Code Smells — Maintainability A

96.0% Coverage on 89 Lines to cover — 43 Unit Tests

0.0% Duplications on 473 Lines — 0 Duplicated Blocks

Demonstration

Time for a live demo of the project

// I hope you like it!

Using a Jira board to plan the sprints proved it is a powerful tool for both large solo projects and group work.

For this project one sprint goal was missed! // documentation

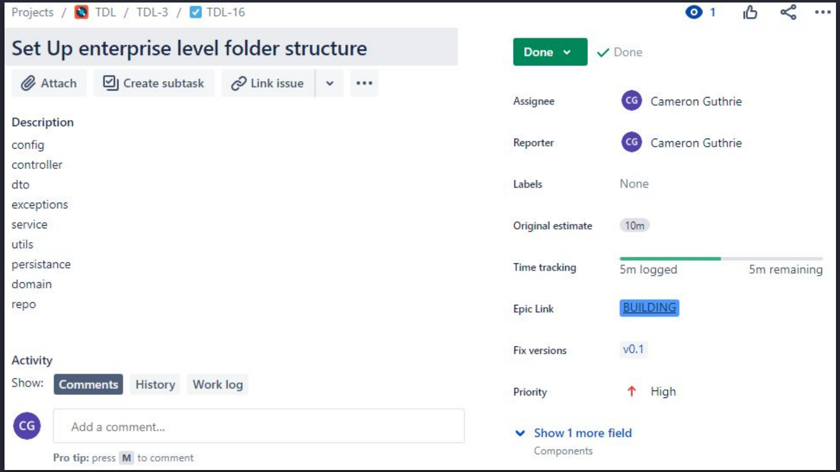
TDL Sprints

By the end of the two sprints, the MVP from the project specification needed to be complete.

- ➔ **Tasks**
  Anything that isn't a user story.
- ➔ **User Stories**
  Specific things that should be doable by someone using the product.

Tasks

- → What the task was
- → Any steps I needed to take
- → A time estimate
- → An epic to keep them organised
- → A version for version control


Projects / TDL / TDL-3 / TDL-16

Set Up enterprise level folder structure

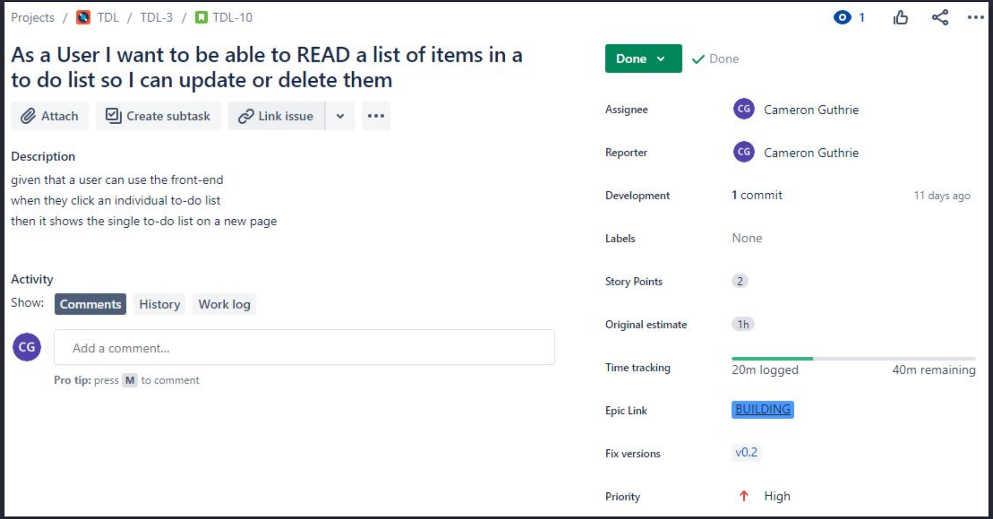
Description
config
controller
dto
exceptions
service
utils
persistance
domain
repo

Activity
Show: Comments History Work log
Add a comment...
Pro tip: press M to comment

Assignee	Cameron Guthrie
Reporter	Cameron Guthrie
Labels	None
Original estimate	10m
Time tracking	5m logged 5m remaining
Epic Link	BUILDING
Fix versions	v0.1
Priority	High

User Stories

- → What the User would want to do
- → Acceptance criteria
- → Story point
- → A time estimate
- → An epic to keep them organised
- → A version for version control



As a User I want to be able to READ a list of items in a to do list so I can update or delete them

Description

given that a user can use the front-end
when they click an individual to-do list
then it shows the single to-do list on a new page

Activity

Show: Comments History Work log

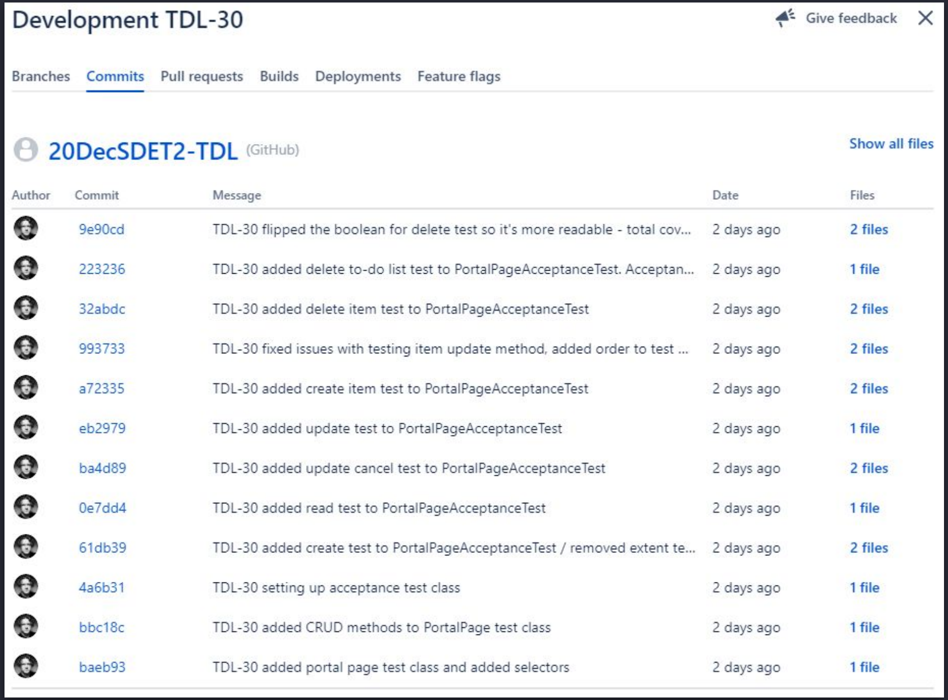
Add a comment...

Pro tip: press M to comment

Assignee	Cameron Guthrie
Reporter	Cameron Guthrie
Development	1 commit 11 days ago
Labels	None
Story Points	2
Original estimate	1h
Time tracking	20m logged 40m remaining
Epic Link	BUILDING
Fix versions	v0.2
Priority	High

GitHub Integration

- → Track branches
- → Track commits



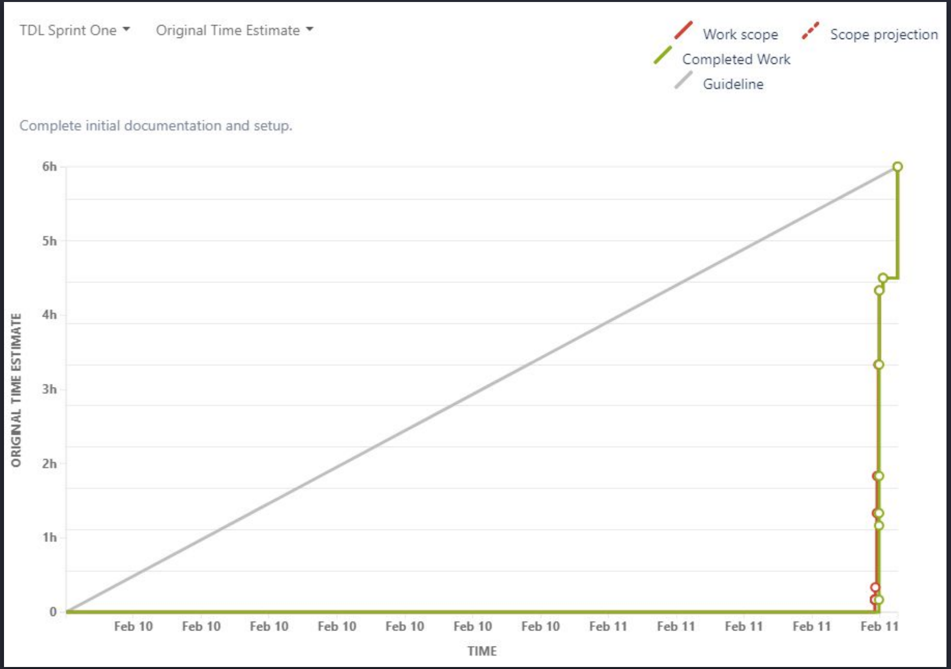
Development TDL-30

20DecSDET2-TDL (GitHub)

Author	Commit	Message	Date	Files
	9e90cd	TDL-30 flipped the boolean for delete test so it's more readable - total cov...	2 days ago	2 files
	223236	TDL-30 added delete to-do list test to PortalPageAcceptanceTest. Acceptan...	2 days ago	1 file
	32abdc	TDL-30 added delete item test to PortalPageAcceptanceTest	2 days ago	2 files
	993733	TDL-30 fixed issues with testing item update method, added order to test ...	2 days ago	2 files
	a72335	TDL-30 added create item test to PortalPageAcceptanceTest	2 days ago	2 files
	eb2979	TDL-30 added update test to PortalPageAcceptanceTest	2 days ago	1 file
	ba4d89	TDL-30 added update cancel test to PortalPageAcceptanceTest	2 days ago	2 files
	0e7dd4	TDL-30 added read test to PortalPageAcceptanceTest	2 days ago	1 file
	61db39	TDL-30 added create test to PortalPageAcceptanceTest / removed extent te...	2 days ago	2 files
	4a6b31	TDL-30 setting up acceptance test class	2 days ago	1 file
	bbc18c	TDL-30 added CRUD methods to PortalPage test class	2 days ago	1 file
	baeb93	TDL-30 added portal page test class and added selectors	2 days ago	1 file

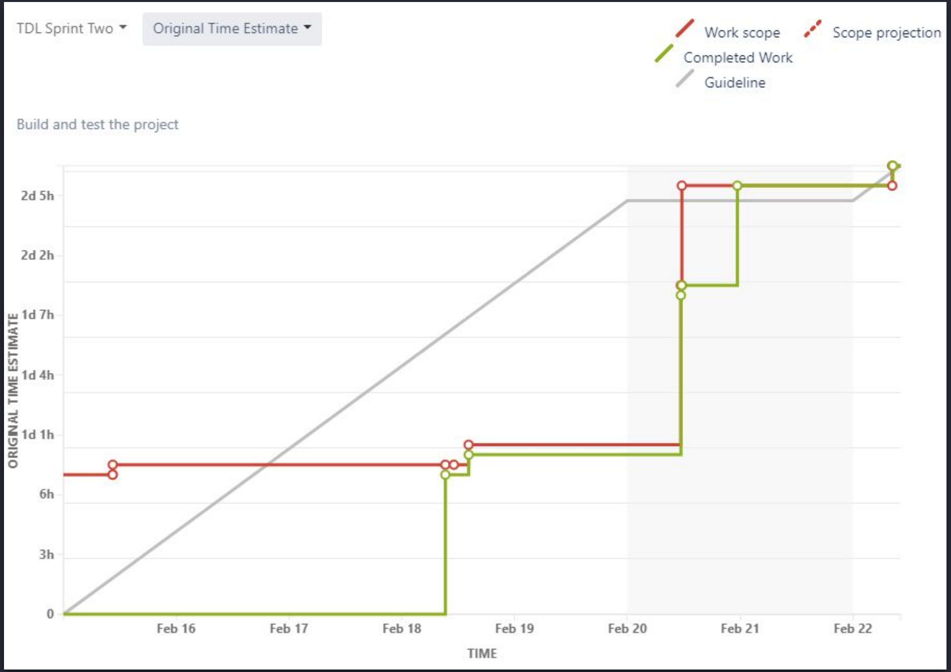
Sprint One

Initial documentation and admin tasks.



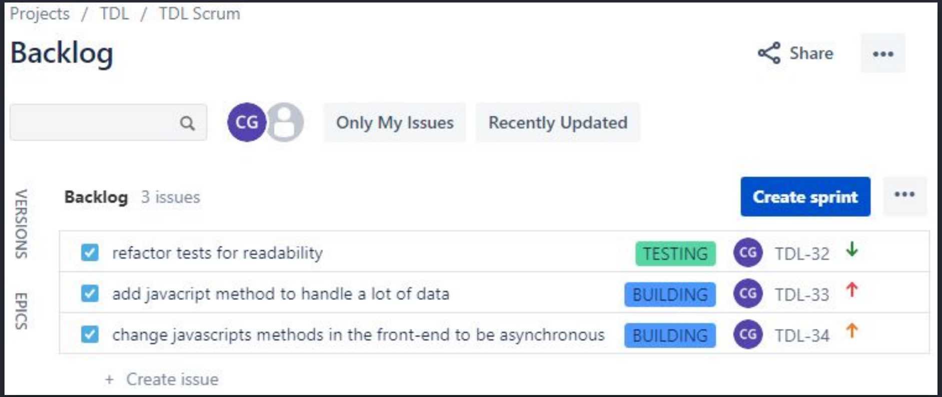
Sprint Two

Building and testing the project.



What went well?
What could be improved?

→ backlog items


Projects / TDL / TDL Scrum

Backlog

Backlog 3 issues

refactor tests for readability	TESTING	CG	TDL-32
add javacript method to handle a lot of data	BUILDING	CG	TDL-33
change javascripts methods in the front-end to be asynchronous	BUILDING	CG	TDL-34

Time Management

Building

Back-End

Maven project with Spring Boot and H2 database

Front-End

Single page in html, css, and javascript

Testing

Back-End

Unit and Integration testing to cover CRUD

Front-End

Acceptance testing to cover User Stories

Documentation

Diagrams and Risk

Needed to be completed both before and alongside the project

Jira Board

Writing out all tasks and user stories and connecting to GitHub

Presentation

Having worked on the IMS project, this was managed better

Conclusion

An exciting and slightly stressful journey to meet the minimum viable product.

➔ **Thank you for listening**
I am aware the presentation was quite verbose

➔ **Do you have any questions?**
Feel free to ask!