

# To-Do List Web Application

SDET week 8

# Design Specification

**To create an OOP-based web application, with utilisation of supporting tools, methodologies, and technologies, that encapsulates all fundamental and practical modules covered during training.**

- **Simple webpage using bootstrap, Create tasks organised into lists.**
- **Spring boot application handles from API to Database**
- **Keep it simple, focus on getting the testing right.**

# Planning - Design

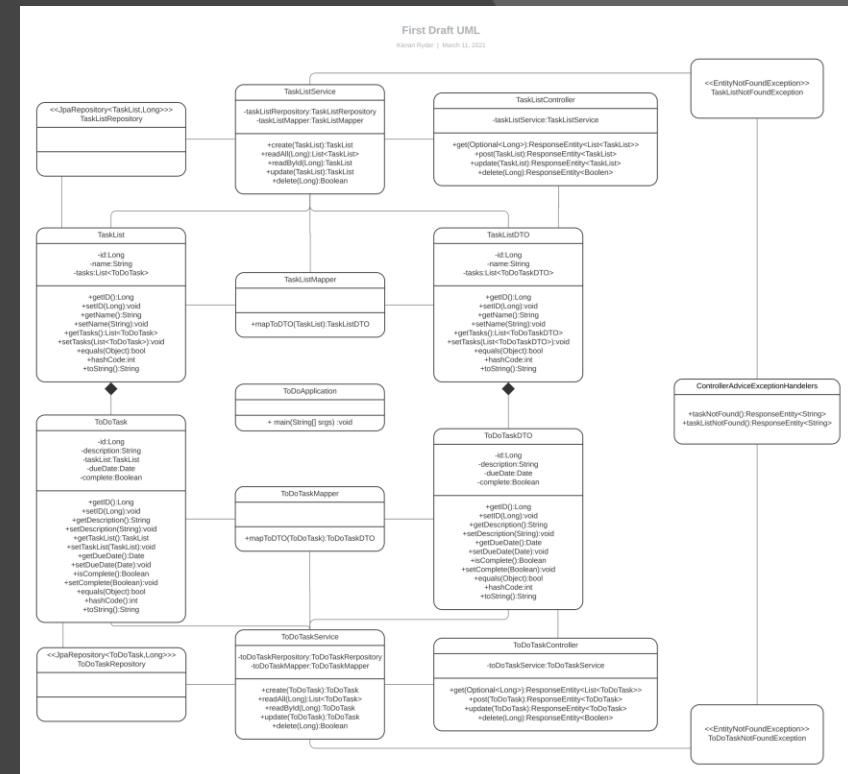
## Wireframe webpage

MY TASKS

LIST NAME	DELETE
TASK 1	✓
TASK 2 DUE	✓
TASK 3	✓
DELETE	EDIT
<del>TASK 4 DUE</del>	✓

NEW TASK

## UML



# Planning – Sprint Tools

- Build task backlog on jira
- Setup git integration with Jira

The screenshot displays the Jira interface for a project named 'To-Do list web-app'. The main view is the 'Backlog' section, which lists 11 issues. The issues are organized by epic, with 'MVP' and 'Extensions' visible in the left sidebar. The 'MVP' epic contains 10 issues, and the 'Extensions' epic contains 1 issue. The issues are listed in a table with columns for issue key, description, priority, and status. The 'Backend setup' issue (TDLWA-10) is highlighted in blue. The right sidebar shows the details for the selected issue, including its title 'Backend setup', a description field, and a list of child issues.

Projects / To-Do list web-app

## Backlog

Search: KR | Epic: | Type: | Create sprint

72 0 0

▼ Backlog (11 issues)

Issue Key	Description	Priority	Status
TDLWA-1	As a User I want my TODO lists presented cleanly so that I can see what I have to do	MVP	6 KR
TDLWA-2	As a User I want to be able to create a new TODO list so that I can keep my tasks separated and orga...	MVP	4 KR
TDLWA-3	As a User I want to be able to mark tasks as complete so that I can tell what still needs to be done	MVP	2 KR
TDLWA-4	As a User I want to add tasks to my TODO lists so that I can full them with tasks	MVP	8 KR
TDLWA-5	As a User I want to optionally assign due dates to my tasks so that I can see when I need to do thing...	MVP	12 KR
TDLWA-6	As a User I want to edit my tasks so that mistakes can be corrected	MVP	4 KR
TDLWA-7	As a User I want to remove tasks so that tasks that are no longer relevant can be removed	MVP	2 KR
TDLWA-8	As a User I want to remove TODO lists so that I can get rid of completed lists.	MVP	2 KR
TDLWA-9	As a User I want to edit my lists titles so that I can correct typos.	MVP	2 KR
✓ TDLWA-10	Backend setup	MVP	15 KR
✓ TDLWA-11	Frontend Setup	MVP	15 KR

+ Create issue

▼ Epic

Issues without epic

> MVP

> Extensions

+ Create Epic

MVP / TDLWA-10

## Backend setup

To Do

Description

Add a description...

Child issues

0% Done

Issue Key	Description	Priority	Status
TDLWA-16	Create Spring project	KR	TO DO
TDLWA-17	create ToDoTask per...	KR	TO DO
TDLWA-18	Create ToDoTaskDA...	KR	TO DO
TDLWA-19	create ToDoTask Ma...	KR	TO DO
TDLWA-20	Create unit tests for ...	KR	TO DO

# Technologies Used

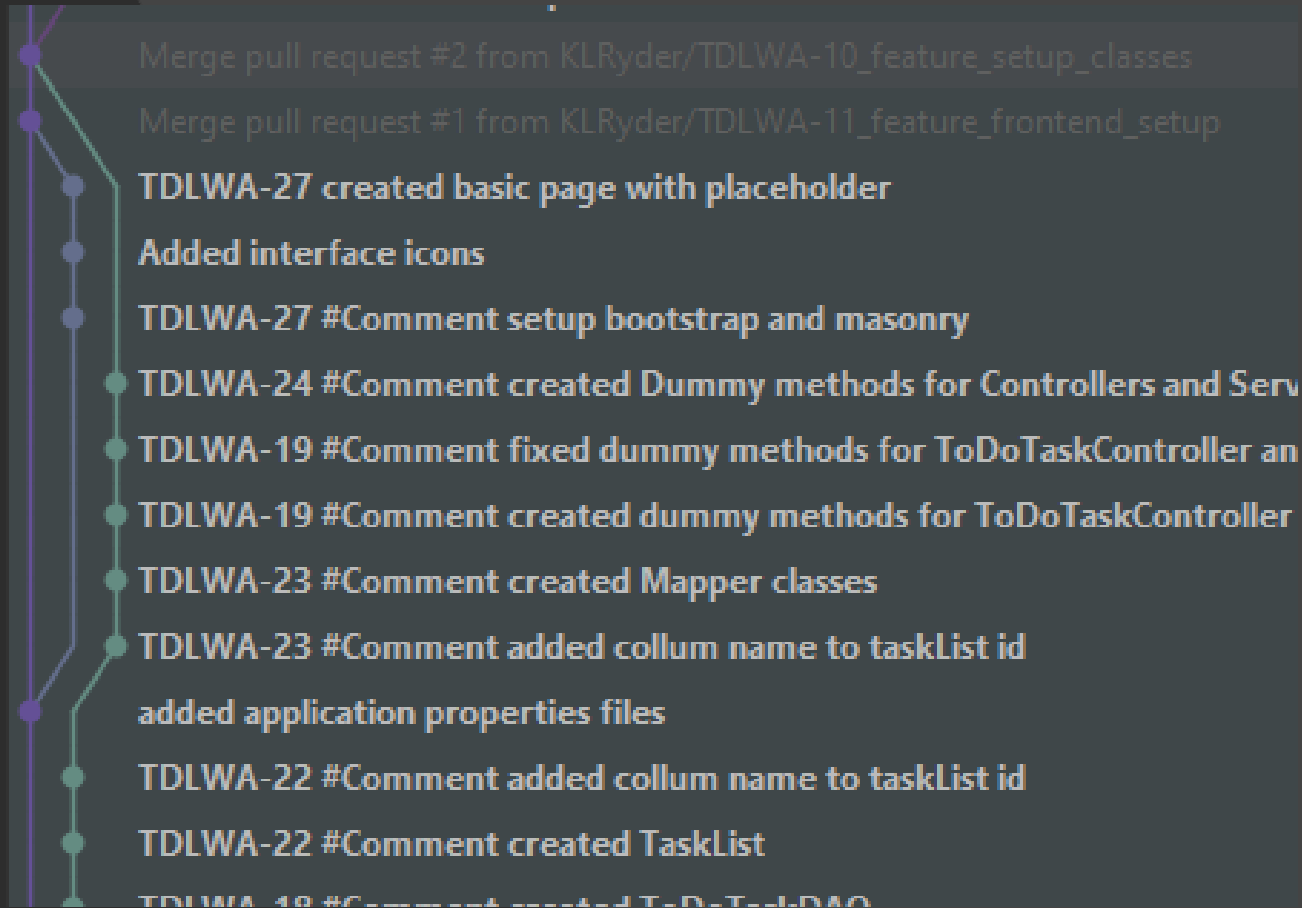
## **Old/Proven**

- Git
- Java
- JUnit5
- JavaScript
- Mockito
- Maven
- MySQL

## **New/Less-experienced**




















- Spring
- HTML/CSS
- Selenium
- SonarQube

# Version control



- All code stored on github:
- [https://github.com/KLRyder/TDLWA\\_project2](https://github.com/KLRyder/TDLWA_project2)
- Using a standard main-dev-feature branching model.
- Feature branches named after corresponding Jira story

# Testing

todo_lists				
<b>todo_lists</b>				
Element	Missed Instructions	Cov.	Missed	
 <a href="#">com.qa.todo_lists.data.dto</a>		81%		
 <a href="#">com.qa.todo_lists.exceptions</a>		33%		
 <a href="#">com.qa.todo_lists</a>		15%		
 <a href="#">com.qa.todo_lists.data.model</a>		97%		
 <a href="#">com.qa.todo_lists.service</a>		100%		
 <a href="#">com.qa.todo_lists.controller</a>		100%		
 <a href="#">com.qa.todo_lists.mappers</a>		100%		
Total	100 of 1,046	90%	0 of 74	
Created				

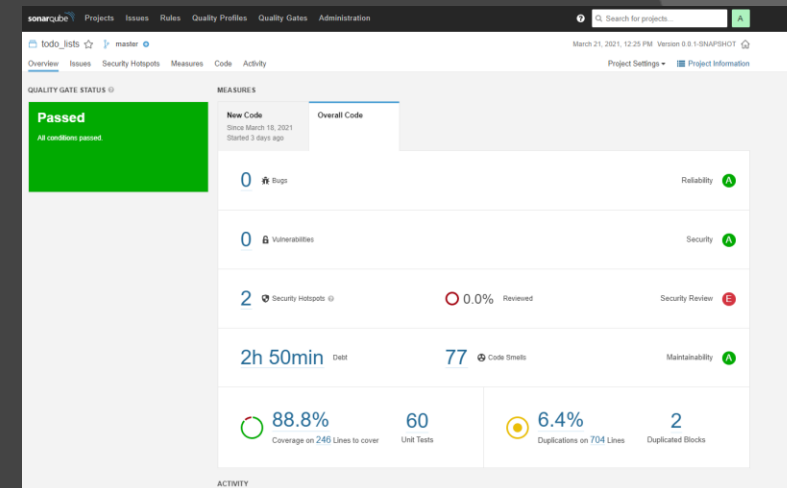
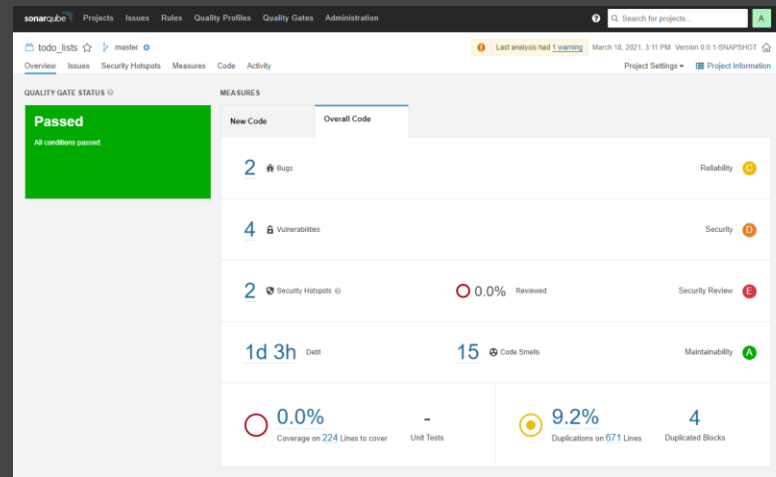
- Coverage: 90% in src/java according to JaCoCo/EclEmma
- Unit tests for all classes with JUnit5
- Integration tests for Controllers and Service classes with JUnit5
- Acceptance testing Webpage with Selenium

# Static Analysis

## Using SonarQube

Examples of fixes and comments on them in directory "statica" on github.

Most code smells are just the use of "public" in JUnit 5



### Resolution

Unresolved	0	False Positive	0
Fixed	89	Removed	0
Won't Fix	4		



# Quick Demonstration

# Self-Evaluation

# Sprint Review

## **Positives**

- Full MVP completion

## **Negatives**

- Not enough time to make website look good

# Sprint Retrospective

## Positives

- Managed time well – Low stress
- Frontend went far better than expected
- Minor – UML was nearly perfect on first pass. Was also satisfyingly symmetrical.
- Project did not need to change much from conception to completion

## Negatives

- Still awful at actually estimating time taken on tasks
- Didn't keep up with documenting workflow very well

# Closing Remarks

- Overall, reasonably happy with outcome considering initial reservations
- Spring is a powerful tool, but the loss of control inherent to inversion of control is not something that I am a fan of.
- Masonry is a fantastic tool, but it did highlight how much nice building this kind of app would be in jQuery.
- Future improvements: make the website not look boring as sin.

Questions?