

# To-do List (TDL) Web Application



By Celina Basa

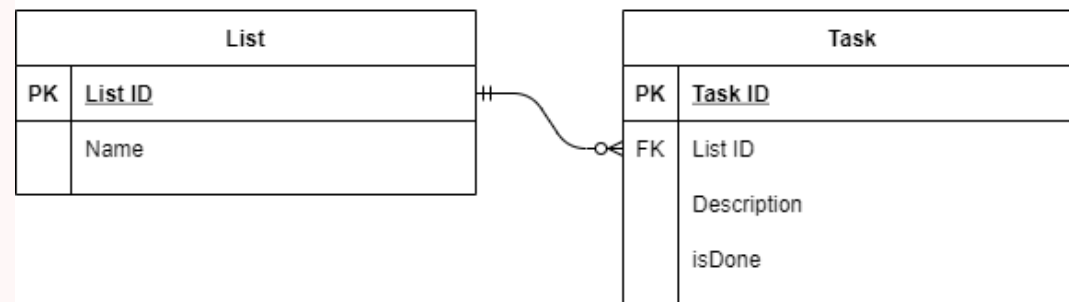
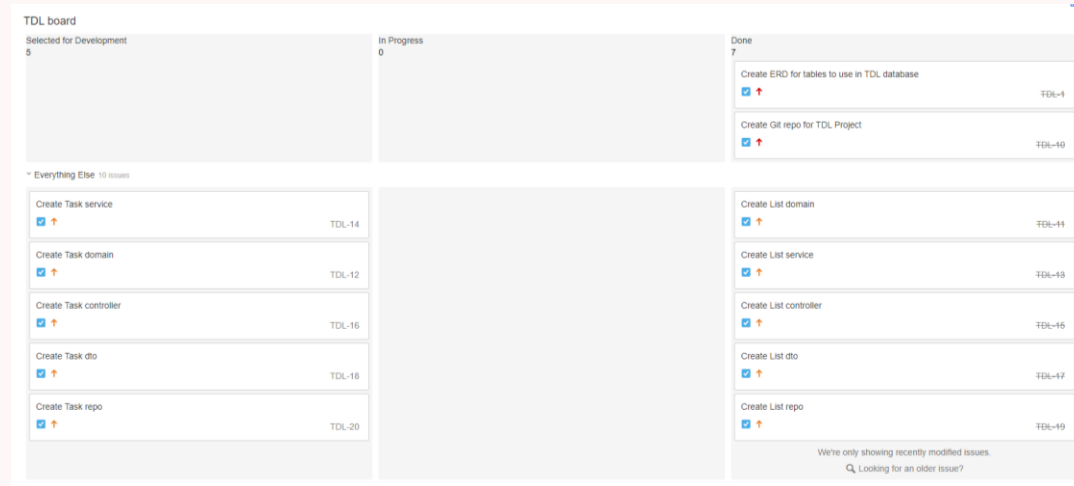
# Introduction

Who am I?

- QA Trainee Consultant

How did I approach the specification?

- Considered concept of minimum viable product
- Started with Kanban board and ERD



# Sprint Plan

Meet the minimum specification requirements:

- At least two entities
- Functional front-end which connects to back-end
- >80% test coverage
- Unit and integration testing
- User-acceptance testing
- Static analysis

# Consultant Journey

## Technologies learnt for this project:

- Spring Boot

- JavaScript

- HTML

- CSS

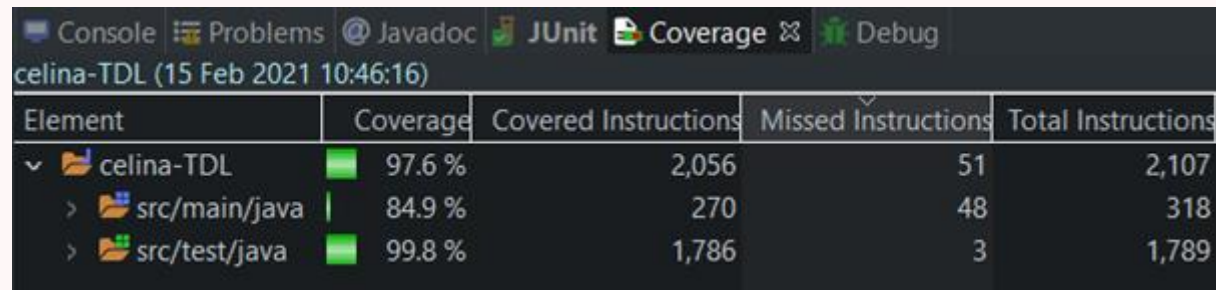
- Selenium

- SonarQube



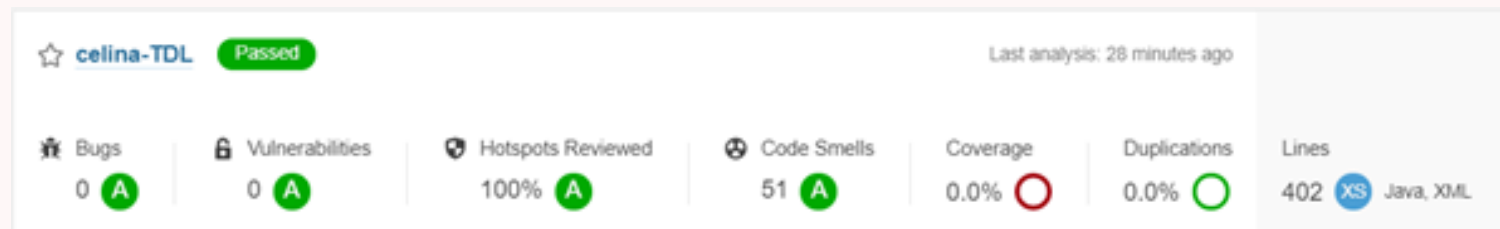
# Testing

- Coverage and static analysis with SonarQube



The screenshot shows an IDE window with tabs for Console, Problems, Javadoc, JUnit, Coverage, and Debug. The Coverage tab is active, displaying a table for 'celina-TDL (15 Feb 2021 10:46:16)'. The table has columns for Element, Coverage, Covered Instructions, Missed Instructions, and Total Instructions. The data is as follows:

Element	Coverage	Covered Instructions	Missed Instructions	Total Instructions
celina-TDL	97.6 %	2,056	51	2,107
src/main/java	84.9 %	270	48	318
src/test/java	99.8 %	1,786	3	1,789



The screenshot shows the SonarQube analysis results for 'celina-TDL'. The status is 'Passed' and the last analysis was 28 minutes ago. The results are summarized in a table:

Bugs	Vulnerabilities	Hotspots Reviewed	Code Smells	Coverage	Duplications	Lines
0 A	0 A	100% A	51 A	0.0% <span style="color: red;">○</span>	0.0% <span style="color: green;">○</span>	402 XS Java, XML

# Demonstration

2 / 1 5 / 2 0 2 1

### What was completed:

- Test coverage of 80% reached
- Fat .jar able to run from the command-line
- Supporting documentation
- Basic CRUD functionality for lists and tasks

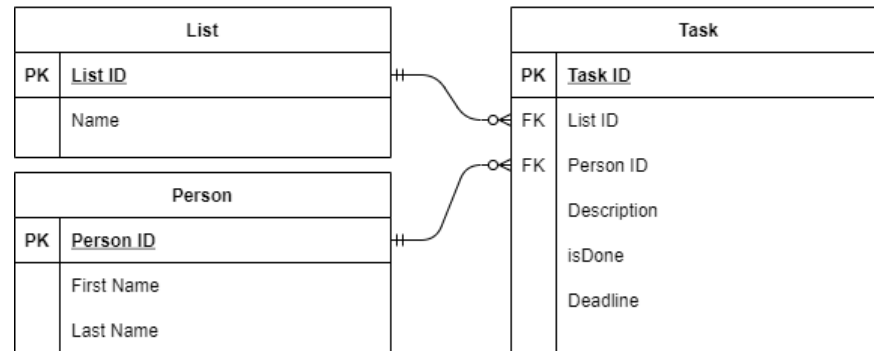
### What got left behind:

- Generating .war file instead of .jar
- Applying more involved styling to front-end

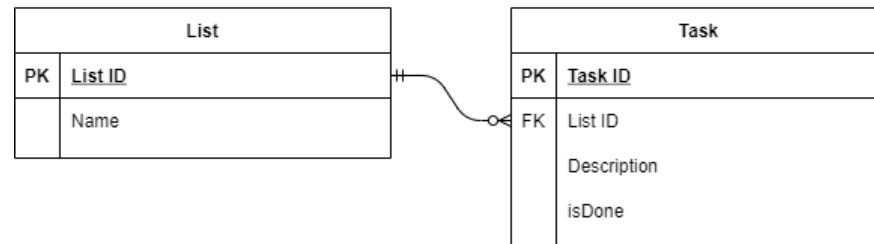
# Sprint Review

# Sprint Retrospective

Intended final ERD for TDL:



Actual final ERD of TDL:





# Sprint Retrospective

```
▼ 2 ■■■■■ src/main/java/com/qa/demo/rest/TaskController.java
↑... @@ -5,6 +5,7 @@
5 5 import org.springframework.beans.factory.annotation.Autowired;
6 6 import org.springframework.http.HttpStatus;
7 7 import org.springframework.http.ResponseEntity;
8 + import org.springframework.web.bind.annotation.CrossOrigin;
8 9 import org.springframework.web.bind.annotation.DeleteMapping;
9 10 import org.springframework.web.bind.annotation.GetMapping;
10 11 import org.springframework.web.bind.annotation.PathVariable;
⋮ @@ -18,6 +19,7 @@
18 19 import com.qa.demo.persistence.dtos.TaskDTO;
19 20 import com.qa.demo.services.TaskService;
20 21
22 + @CrossOrigin
21 23 @RestController
22 24 @RequestMapping("/Task")
23 25 public class TaskController {
24 26
25 27     private TaskService service;
26 28
27 29     @Autowired
28 30     public TaskController(TaskService service) {
29 31         super();
30 32         this.service = service;
31 33     }
}
```

```
▼ 2 ■■■■■ src/main/java/com/qa/demo/rest/ListController.java
↑... @@ -5,6 +5,7 @@
5 5 import org.springframework.beans.factory.annotation.Autowired;
6 6 import org.springframework.http.HttpStatus;
7 7 import org.springframework.http.ResponseEntity;
8 + import org.springframework.web.bind.annotation.CrossOrigin;
8 9 import org.springframework.web.bind.annotation.DeleteMapping;
9 10 import org.springframework.web.bind.annotation.GetMapping;
10 11 import org.springframework.web.bind.annotation.PathVariable;
⋮ @@ -18,6 +19,7 @@
18 19 import com.qa.demo.persistence.dtos.ListDTO;
19 20 import com.qa.demo.services.ListService;
20 21
22 + @CrossOrigin
21 23 @RestController
22 24 @RequestMapping("/List")
23 25 public class ListController {
24 26
25 27     private ListService service;
26 28
27 29     @Autowired
28 30     public ListController(ListService service) {
29 31         super();
30 32         this.service = service;
31 33     }
}
```

# Conclusion/Q&A



Reflections



Future steps