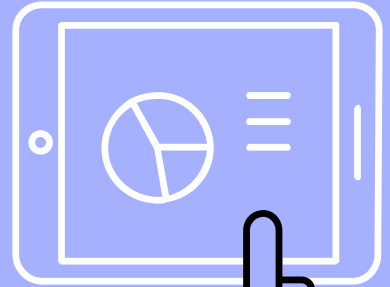
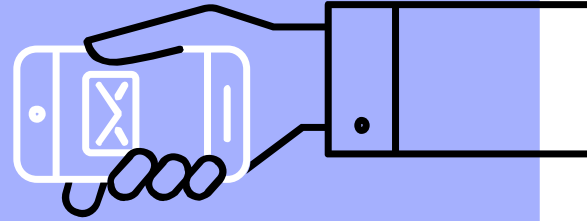
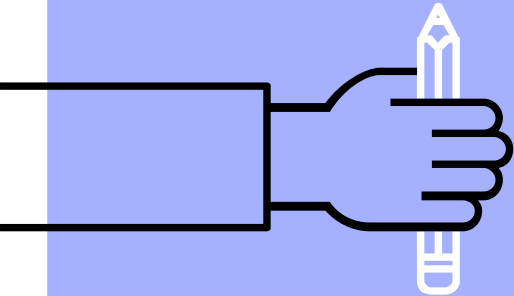
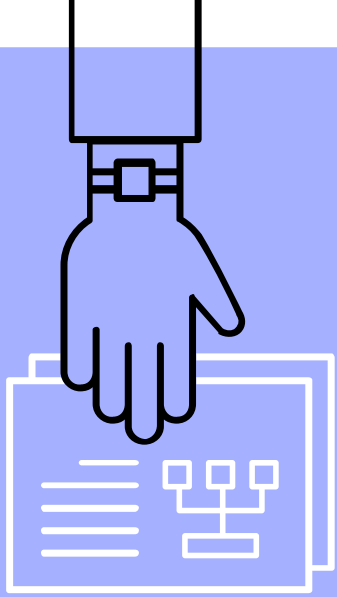


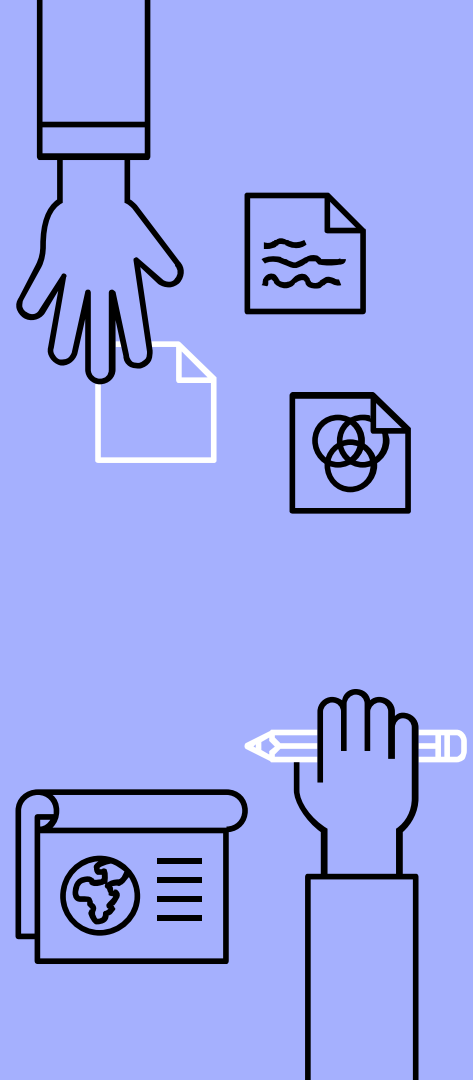
# To Do List Web App Project

By Gie-Anne Fortuna



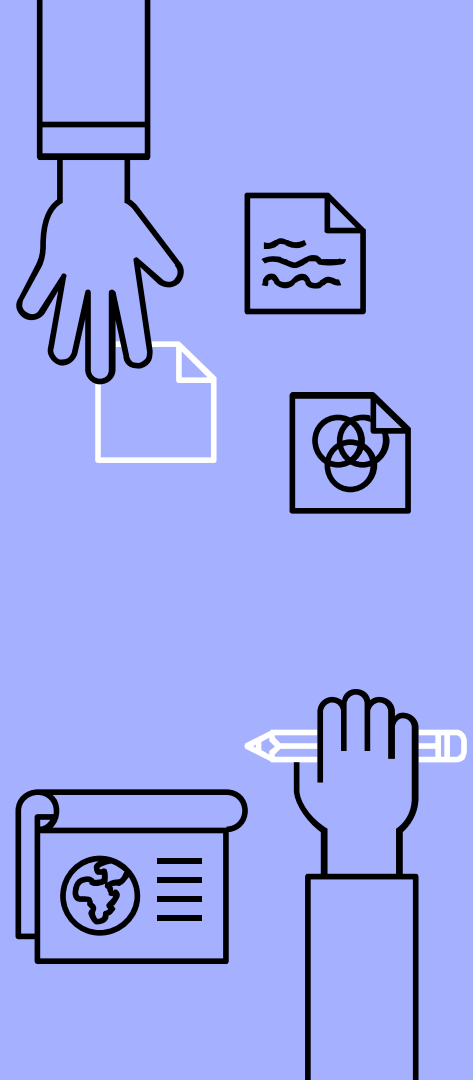
# MUST HAVE

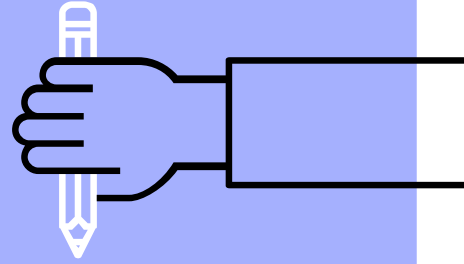
A functional web app that allows the user to perform CRUD functionalities to the two entities-Item and To Do List.



# SHOULD HAVE

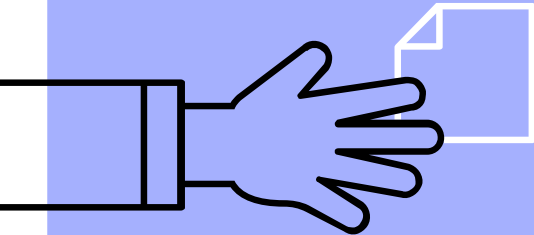
A web app that allows the user to CRUD tasks in the To Do List they belong to.





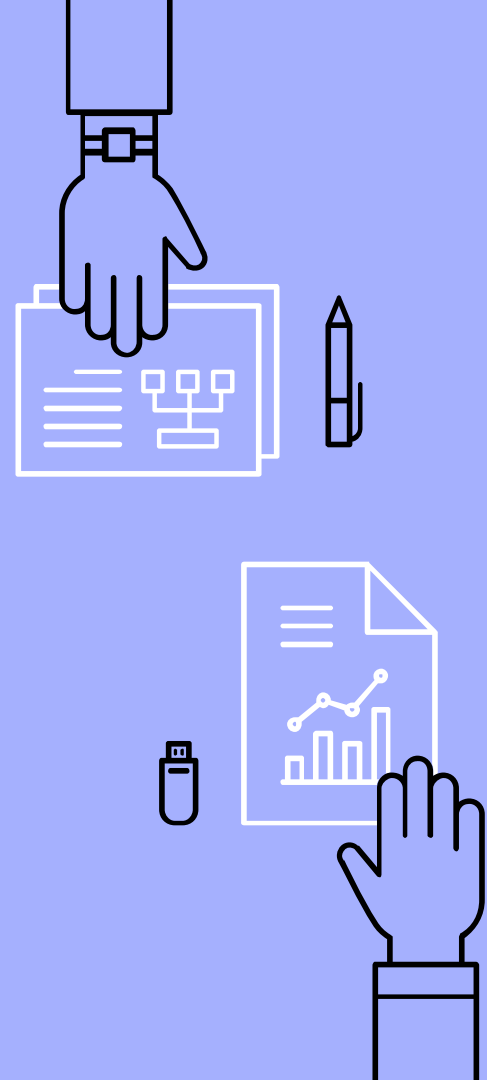
# TECHNOLOGIES

Here are the software used



- ▶ VCS/Code Management: Git/Github
- ▶ Project Management: Jira-Scrum
- ▶ Database Management: MySQL
- ▶ Back-end: Java
- ▶ **API Development: Spring**
- ▶ **Front-end: HTML,CSS,JavaScript**
- ▶ Build Tool: Maven
- ▶ Testing
  - Static Analysis: SonarQube
  - Unit/Integration: JUnit/**Mockito**
  - **User-Acceptance: Selenium**

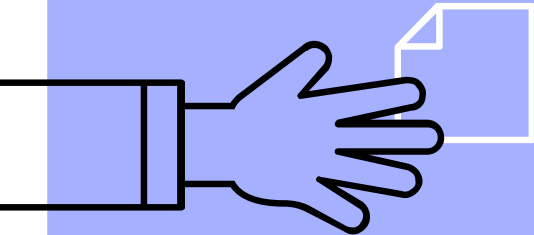
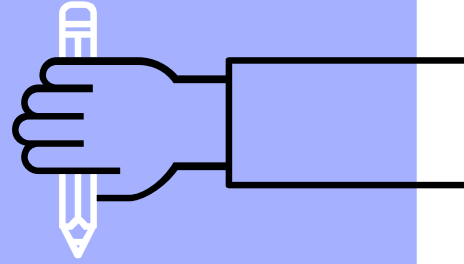
(New for this project)



“

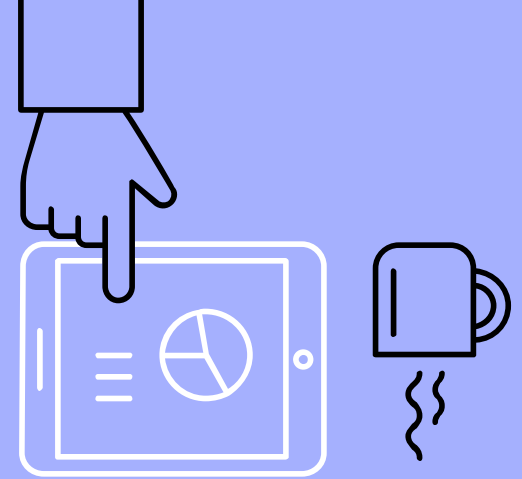
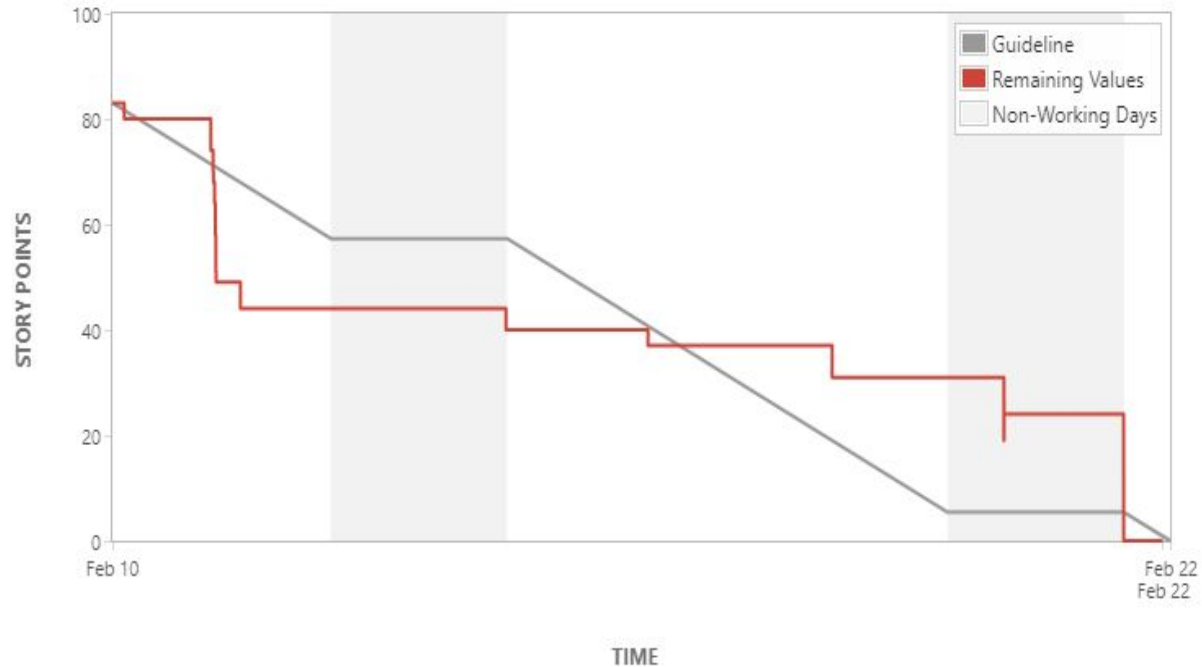
*A goal without a plan is  
just a wish.*

# PLANNING



PLANNING:

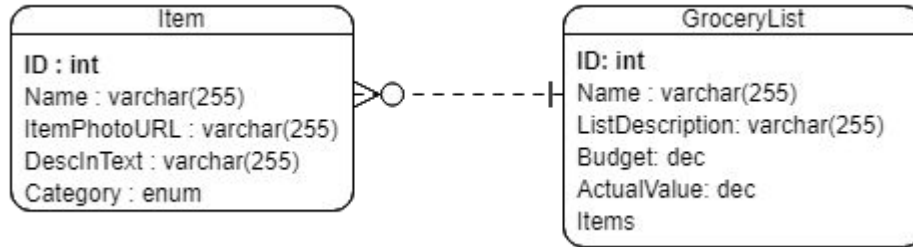
## Jira: Scrum board



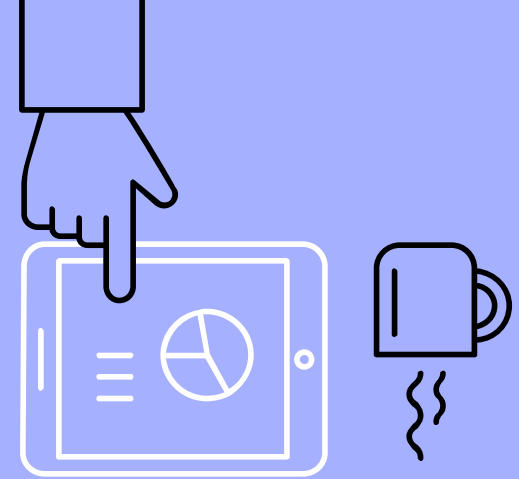


PLANNING:

# Entity Relationship Diagram (start)



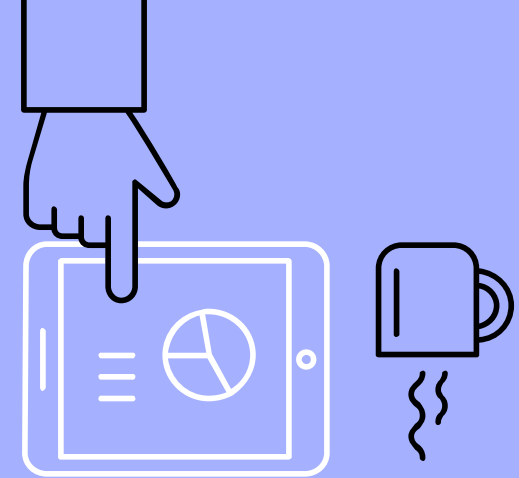
[final](#)



PLANNING:

# RISK ASSESSMENT

Risk	Likelihood	Impact	Risk Level
Incomplete Project	M	H	M
Equipment issues	M	M	L
Broken Frontend	L	H	L
Jar file	L	M	L
Project Spec Misinterpretation	M	H	M
Merge conflicts	M	M	L
Not understanding starter code	M	H	M



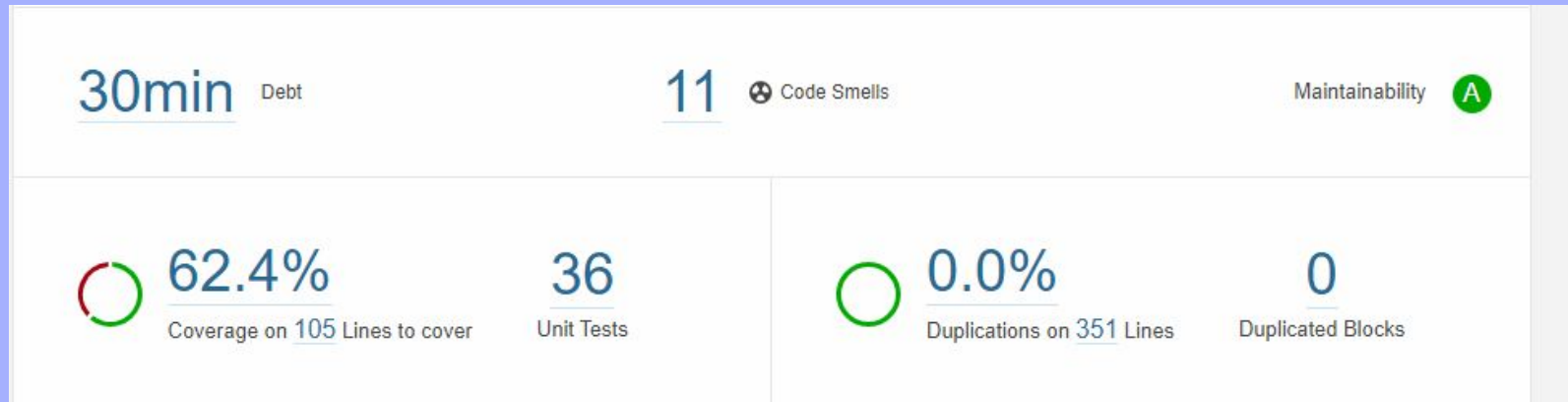
✓ Spring Boot Start

h2 database

demo

# DEMONSTRATION

# SONARQUBE COVERAGE



# STS COVERAGE as JUNIT 5

Element	Coverage	Covered Instructio...	Missed Instructions	Total Instructions
▼ TDL_Project	67.7 %	2,986	1,424	4,410
> src/test/java	68.6 %	2,286	1,045	3,331
▼ src/main/java	64.9 %	700	379	1,079
> com.qa.main.persistence.domain	50.1 %	215	214	429
> com.qa.main.dto	47.8 %	144	157	301
> com.qa.main	37.5 %	3	5	8
> com.qa.main.utils	94.2 %	49	3	52
> com.qa.main.config	100.0 %	7	0	7
> com.qa.main.controller	100.0 %	112	0	112
> com.qa.main.service	100.0 %	170	0	170






Runs: 42/42

Errors: 0






Failures:

- > ItemControllerTest [Runner: JUnit 5] (1.638 s)
- > ItemIntegrationTest [Runner: JUnit 5] (2.016 s)
- > ToDoListControllerTest [Runner: JUnit 5] (0.071 s)
- > ToDoListIntegrationTest [Runner: JUnit 5] (1.276 s)
- > TdlApplicationTests [Runner: JUnit 5] (0.029 s)
- > POJOTest [Runner: JUnit 5] (0.987 s)
- > ToDoListServiceTest [Runner: JUnit 5] (0.926 s)
- > ItemServiceTest [Runner: JUnit 5] (1.908 s)
- > WebAppTest [Runner: JUnit 5] (19.780 s)

# ACCEPTANCE TESTS with SELENIUM

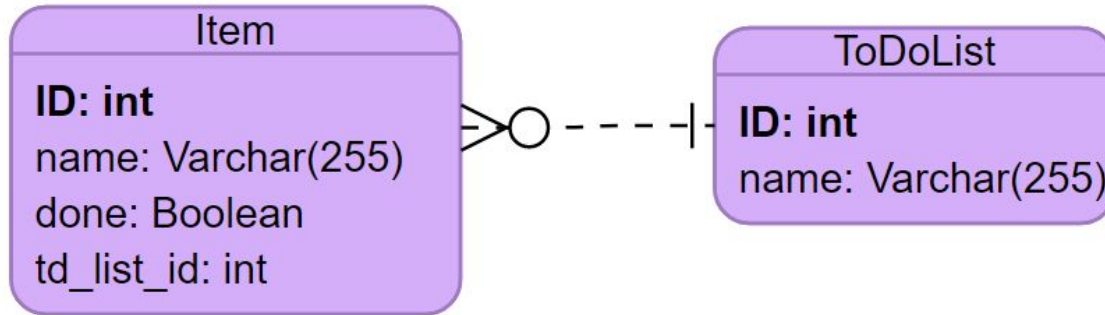
TESTS	
    	
Update list by Id Test	<span>Pass</span>
Create a item test	<span>Pass</span>
Create a list test	<span>Pass</span>
Read list by Id Test	<span>Pass</span>
Delete list Test	<span>Pass</span>
Read all lists test	<span>Pass</span>

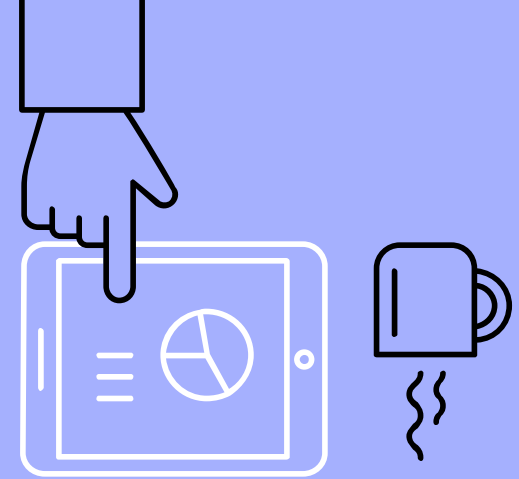
Update list by Id Test		
<span>2021-02-21 20:17:13</span> <span>2021-02-21 20:17:16</span> <span>0h 0m 2s+534ms</span>		
STATUS	TIMESTAMP	DETAILS
	20:17:13	Given - we can access the To Do List webpage
	20:17:14	When we read the first list
	20:17:15	And when we click the edit/pencil button
	20:17:16	Then - I should see this list with new name
	20:17:16	List updated successfully.

RETROSPECTIVE:

# Entity Relationship Diagram (final)



[start](#)



```

classDiagram
    package com.qa.main.controller {
        class ItemController {
            -ItemService
            +create() : ItemDto
            +readAll(): List<ItemDto>
            +readById(): ItemDto
            +update(): ItemDto
            +delete(): ItemDto
            +findItemsInList(): List<ItemDto>
        }
        class ToDoListController {
            -ToDoListService
            +create(): ToDoListDto
            +readAll(): List<ToDoListDto>
            +readById(): ToDoListDto
            +update(): ToDoListDto
            +delete(): ToDoListDto
        }
    }

    package com.qa.main.service {
        class ItemService {
            -repo : ItemRepo
            -mapper : ModelMapper
            +create() : ItemDto
            +readAll(): List<ItemDto>
            +readById(): ItemDto
            +update(): ItemDto
            +delete(): boolean
            +findItemsInList(): List<ItemDto>
            +mapToTDLDto(): ItemDto
        }
        class ToDoListService {
            -repo : ToDoListRepo
            -mapper : ModelMapper
            +create(): ToDoListDto
            +readAll(): List<ToDoListDto>
            +readById(): ToDoListDto
            +update(): ToDoListDto
            +delete(): boolean
            +mapToTDLDto(): ToDoListDto
        }
    }

    package com.qa.main.persistence.repo {
        class ItemRepo {
            +findItemsInList(): List<Item>
        }
        class ToDoListRepo
    }

    package com.qa.main.dto {
        class ItemDto {
            -id: Long
            -name: String
            -tdList: ToDoList
            -id: Long
        }
        class ToDoListDto {
            -id: Long
            -name: String
            -tdList: ToDoList
            -id: Long
        }
    }

    package com.qa.main.persistence.domain {
        class Item {
            -id: Long
            -name: String
            -tdList: ToDoList
        }
        class ToDoList {
            -id: Long
            -name: String
        }
    }

    package config {
        class AppConfig {
            +mapper: ModelMapper
        }
    }

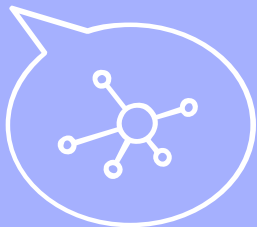
    package utils {
        class SpringBeanUtil {
            +mergeNotNull: void
            -getNullPropName: String[]
        }
    }

    com.qa.main.controller.ItemController ..> com.qa.main.service.ItemService
    com.qa.main.controller.ItemController ..> com.qa.main.persistence.repo.ItemRepo
    com.qa.main.controller.ItemController ..> com.qa.main.persistence.repo.ToDoListRepo
    com.qa.main.controller.ToDoListController ..> com.qa.main.service.ToDoListService
    com.qa.main.controller.ToDoListController ..> com.qa.main.persistence.repo.ToDoListRepo
    com.qa.main.service.ItemService ..> com.qa.main.persistence.repo.ItemRepo
    com.qa.main.service.ItemService ..> com.qa.main.persistence.repo.ToDoListRepo
    com.qa.main.service.ToDoListService ..> com.qa.main.persistence.repo.ToDoListRepo
    com.qa.main.service.ItemService ..> com.qa.main.dto.ItemDto
    com.qa.main.service.ToDoListService ..> com.qa.main.dto.ToDoListDto
    com.qa.main.service.ItemService ..> com.qa.main.persistence.domain.Item
    com.qa.main.service.ToDoListService ..> com.qa.main.persistence.domain.ToDoList
    com.qa.main.dto.ItemDto ..> com.qa.main.persistence.domain.Item
    com.qa.main.dto.ToDoListDto ..> com.qa.main.persistence.domain.ToDoList
    com.qa.main.persistence.domain.Item ..> com.qa.main.persistence.domain.ToDoList
    
```

16



**F  
I  
N  
A  
L**



What went well?



**DELIVERED MVP**



**ADDED EXTRA FEATURES**

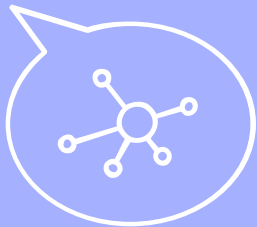


**LEARNED NEW TECHNOLOGIES**

**R  
E  
V  
I  
E  
W**



**F  
I  
N  
A  
L**



What could be done to improve?



**MORE FREQUENT GIT COMMITS**



**BETTER TIME MANAGEMENT**



**MVP FIRST**

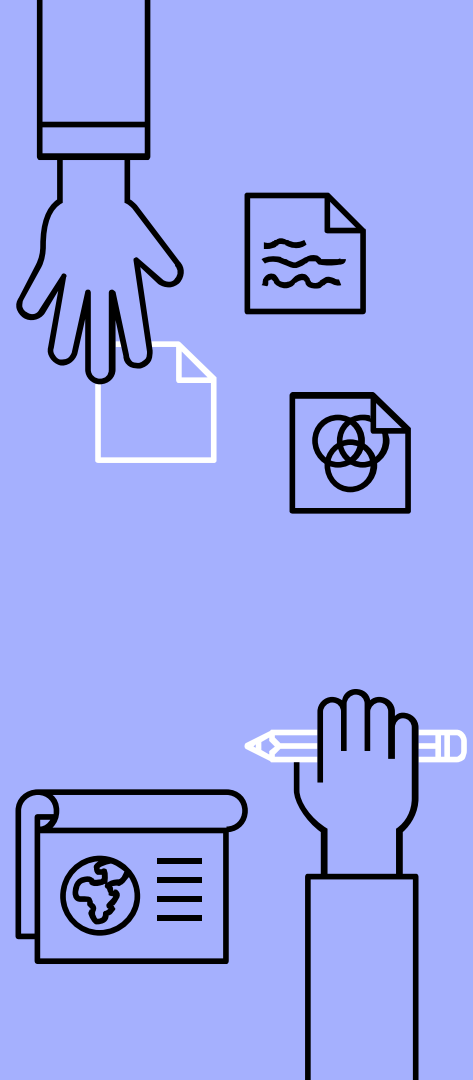
**R  
E  
V  
I  
E  
W**



## COULD HAVE

A web app that has extra features such as searching for a list by name or description.

Mark task as done using checkboxes.

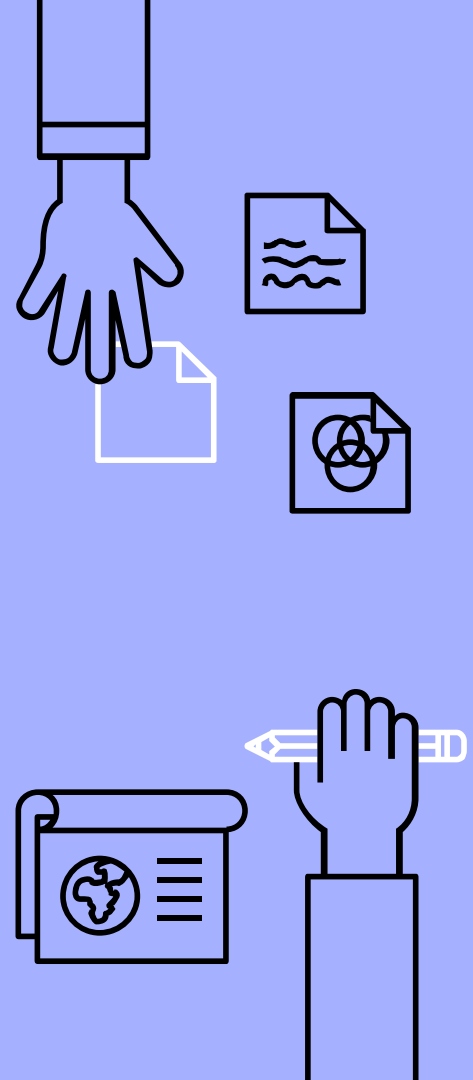


# WOULD HAVE

A web app that asks for user identification in order to access the lists they previously created.

Sessions.

Email notifications.



# THANKS!

Any questions?

