Maintenance Monitor

A team of max 3 members should implement a REST-based server in Java (use Spring Boot). The service should be able to manage a centrally stored message and hereby capable to:

- reset the message
- set it to a specific message
- deliver the message to the clients using REST.

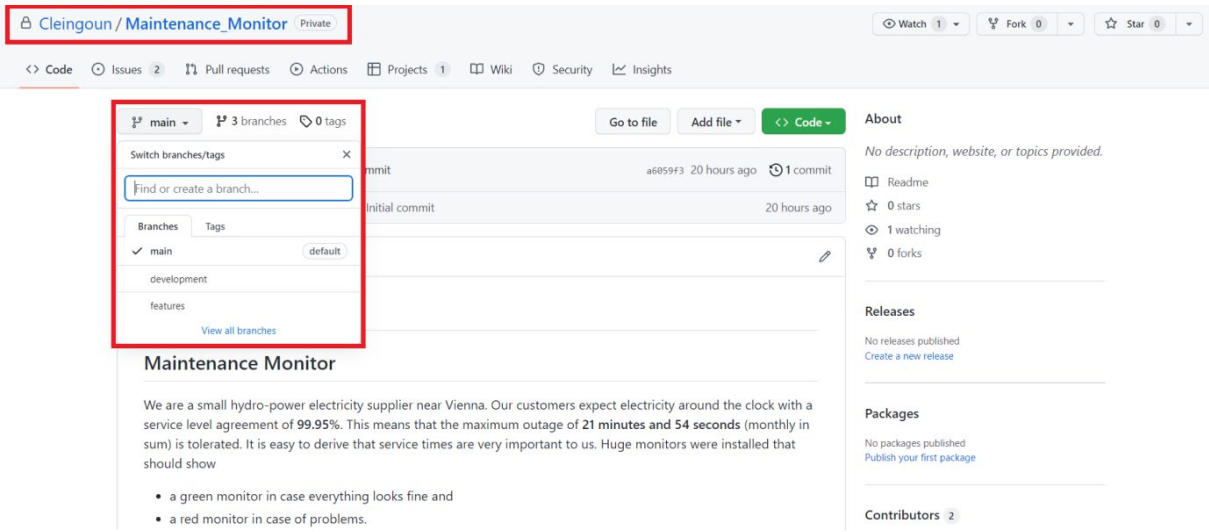
Create a web frontend which is capable to query the message every 5 seconds.

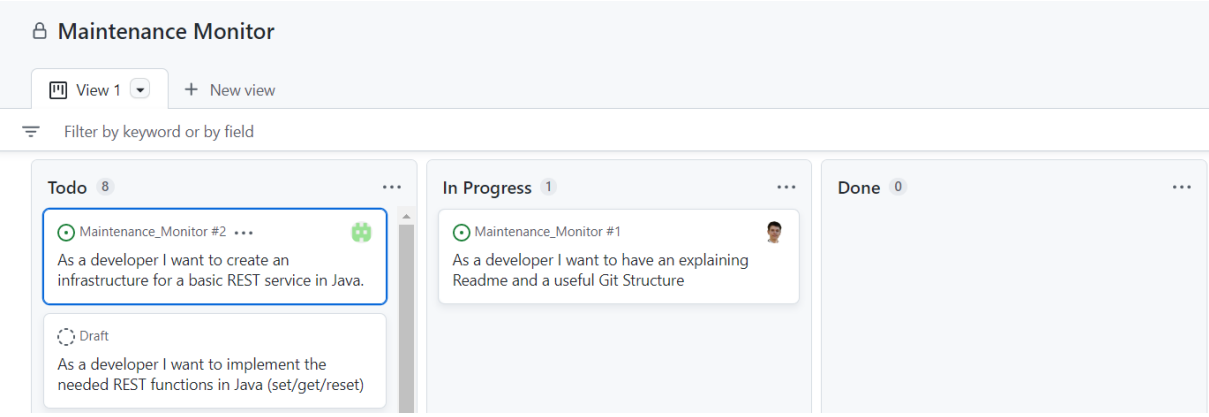
https://github.com/Cleingoun/Maintenance_Monitor

Inhaltsverzeichnis

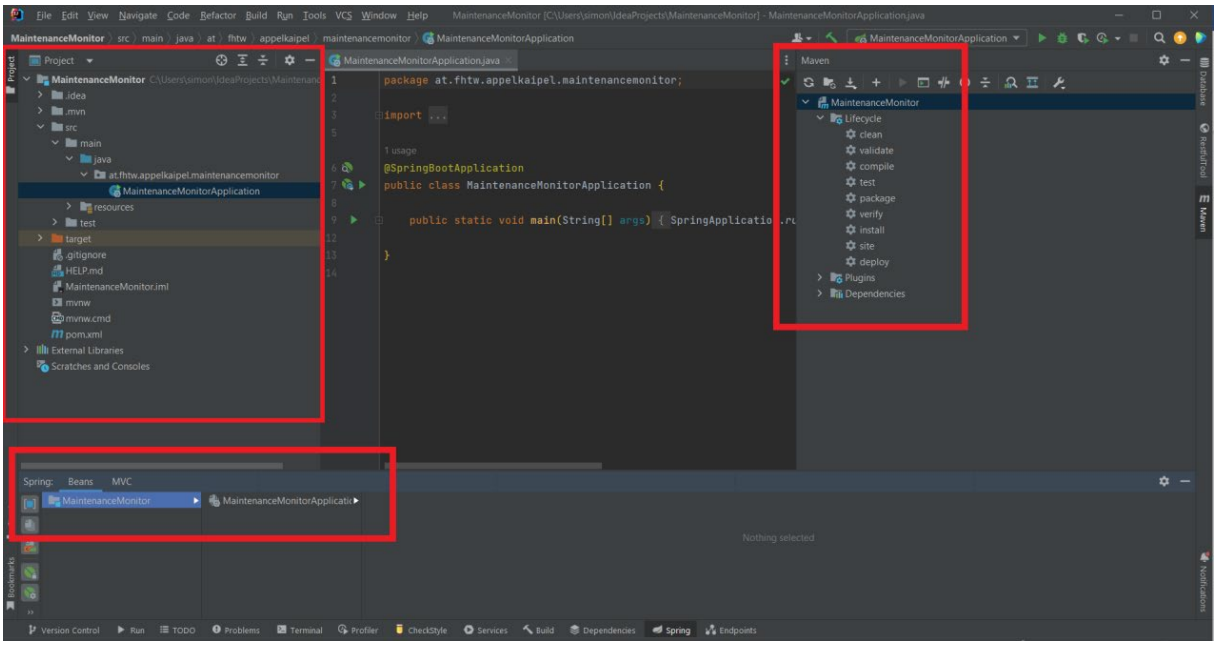
Documentation of the process

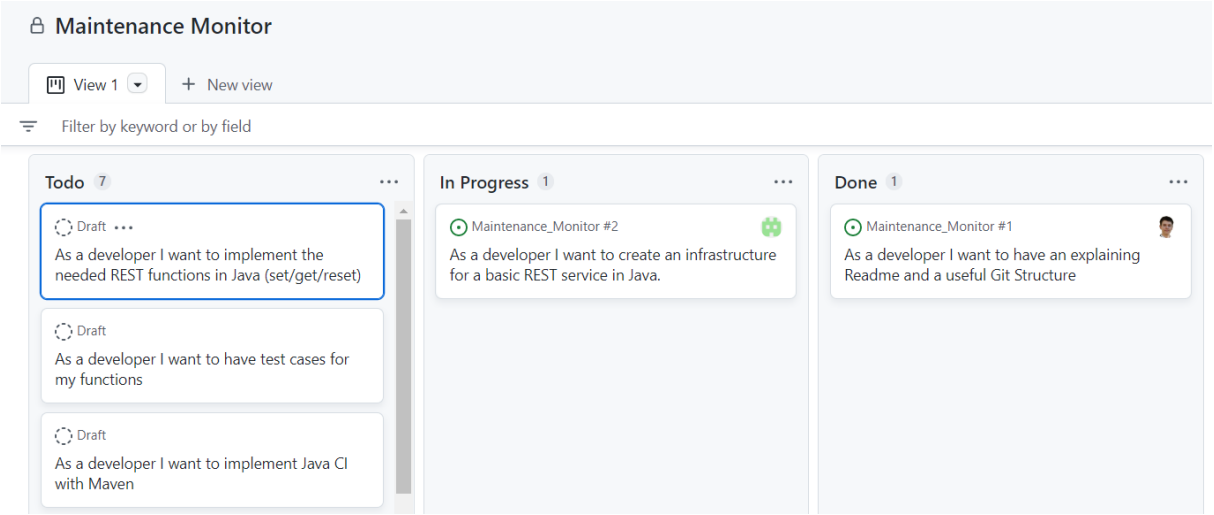
1. Git Repository erstellen, Branches erstellen, erste User Stories definieren

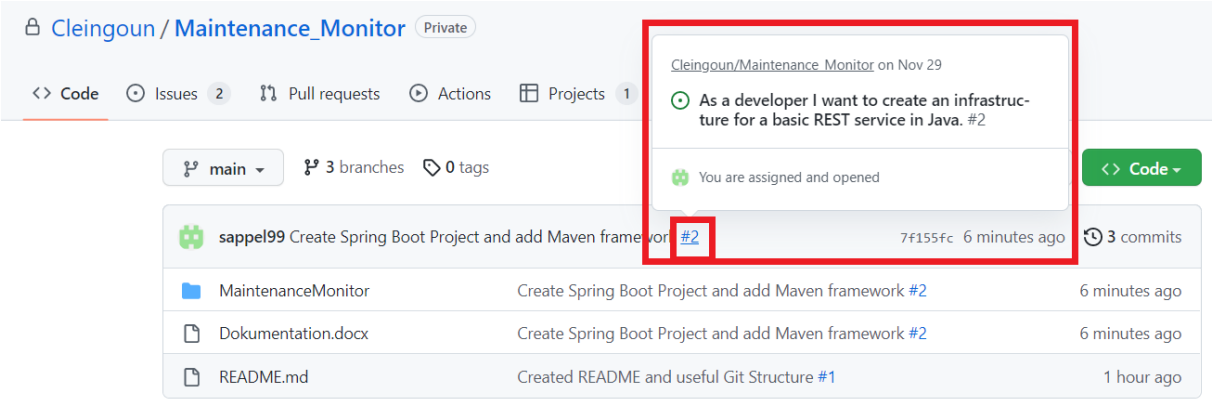




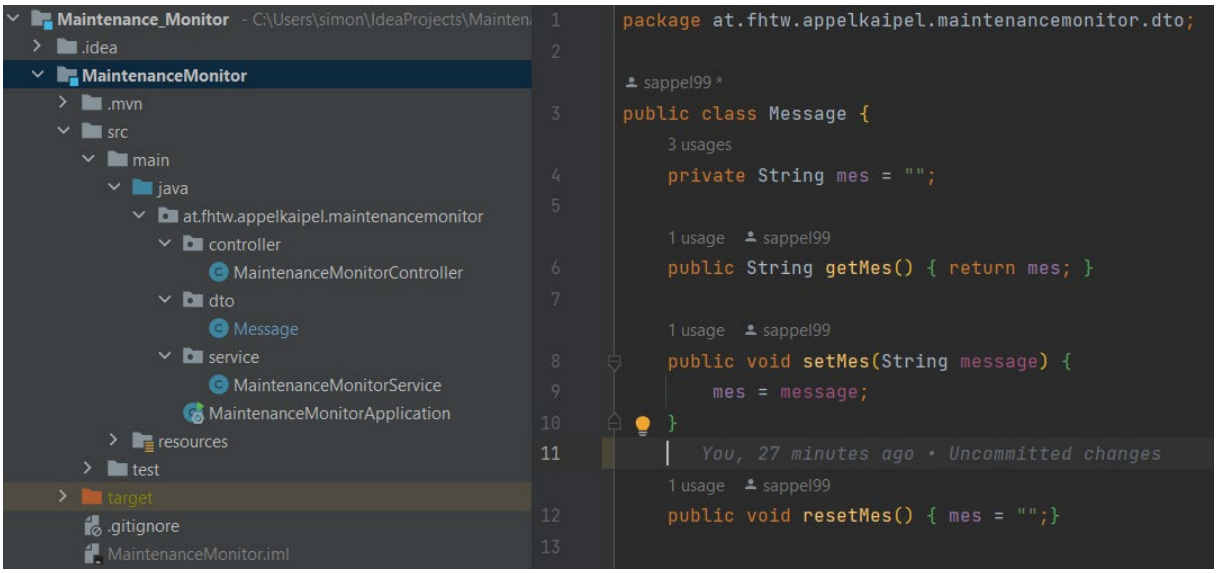
2. Spring Boot Projekt erstellen, Maven einrichten



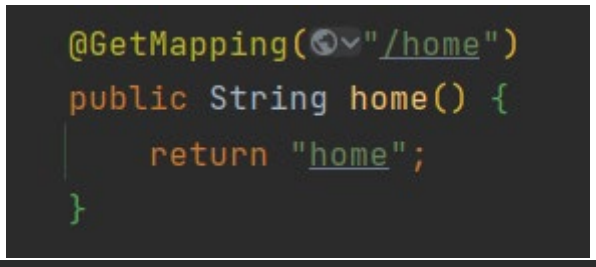


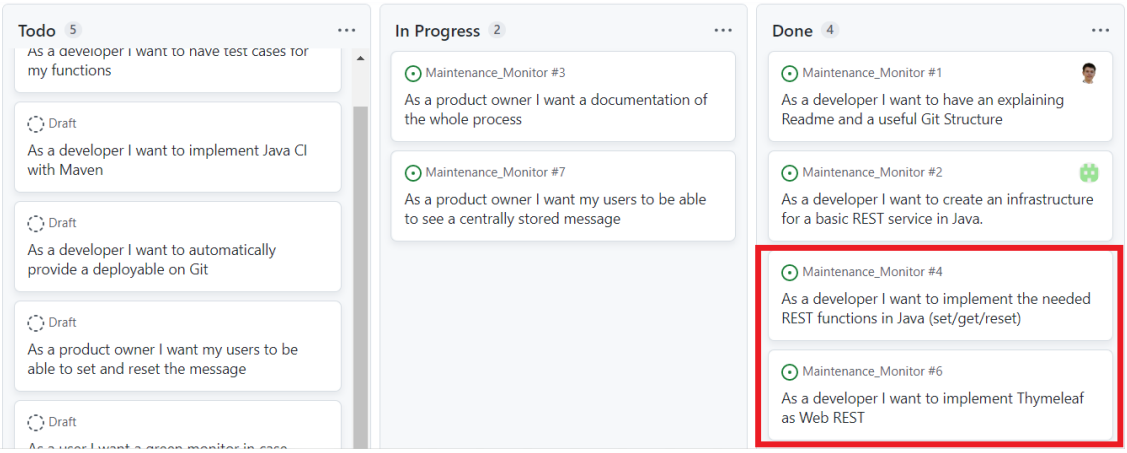


3. Create Controller, Message Class und Service
4. Create Setter, Getter, Reset REST functionality



5. Add Thymeleaf Module Template (/home)





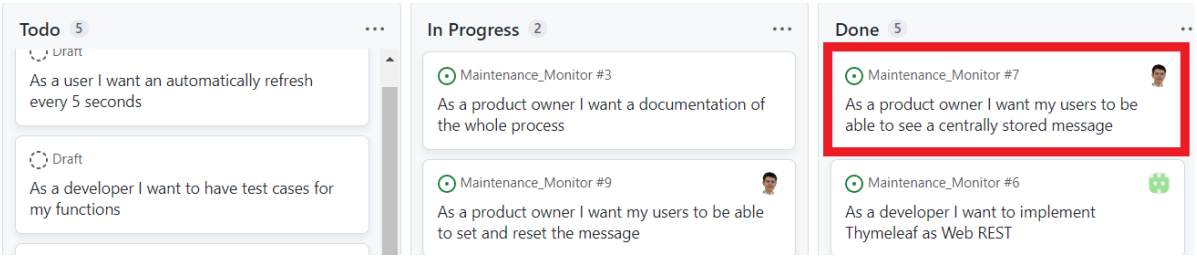
6. Display centrally stored message (/message)

```
3 usages
private String mes = "Stored Message";
```

```
Kaipel Sebastian *
@GetMapping("/message")
String getMes(Model model) {
    String message = monitorService.getMessage();
    model.addAttribute( attributeName: "message", message);

    return "mes";
}
```

localhost:8080/message

Stored Message



7. Set and Reset Message (/message/set)

```
Kaipel Sebastian
@GetMapping("/message/{message}")
String setMes(@PathVariable String message) {
    if (!message.equals("reset")) {
        monitorService.setMessage(message);
    } else {
        monitorService.resetMessage();
    }
    return monitorService.getMessage();
}
```

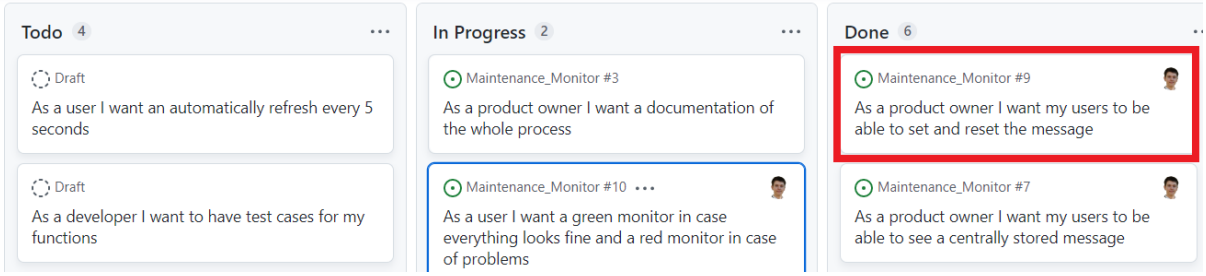
localhost:8080/message

Stored Message

localhost:8080/message/set-msg

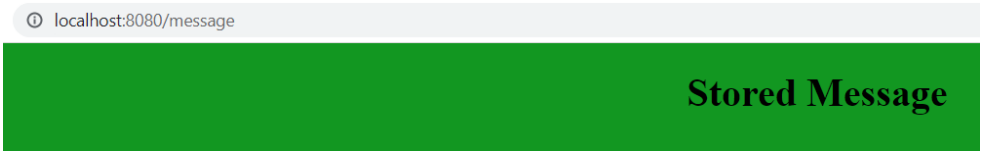
localhost:8080/message

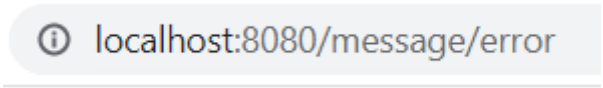
set-msg

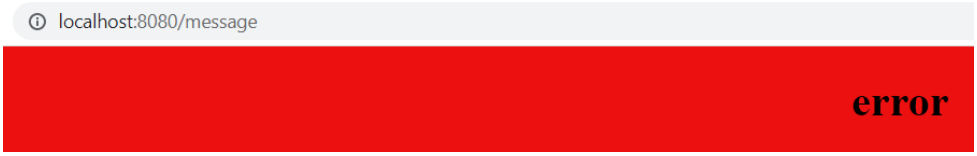


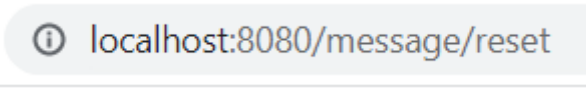
8. Green and Red Monitor (/message/error)

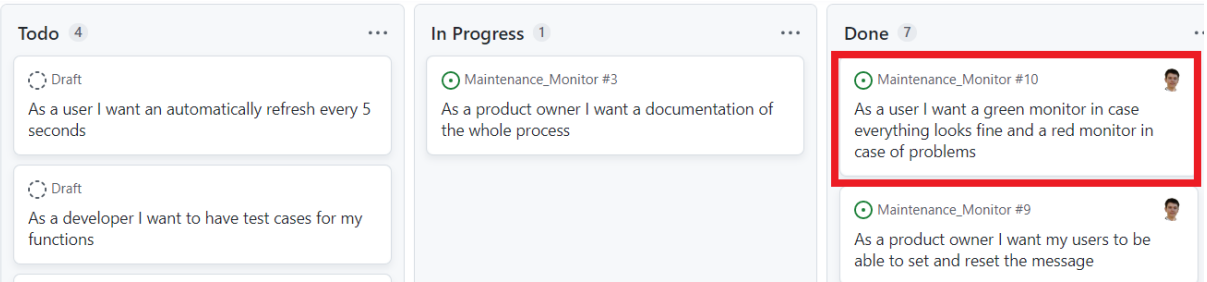
```
Kaipel Sebastian +1
@GetMapping("/message")
String getMes(Model model) {
    String message = monitorService.getMessage();
    model.addAttribute( attributeName: "message", message);

    if (message.equals("error")) {
        model.addAttribute( attributeName: "color",  attributeValue: "#ec1110");
    } else {
        model.addAttribute( attributeName: "color",  attributeValue: "#129721");
    }

    return "mes";
}
```



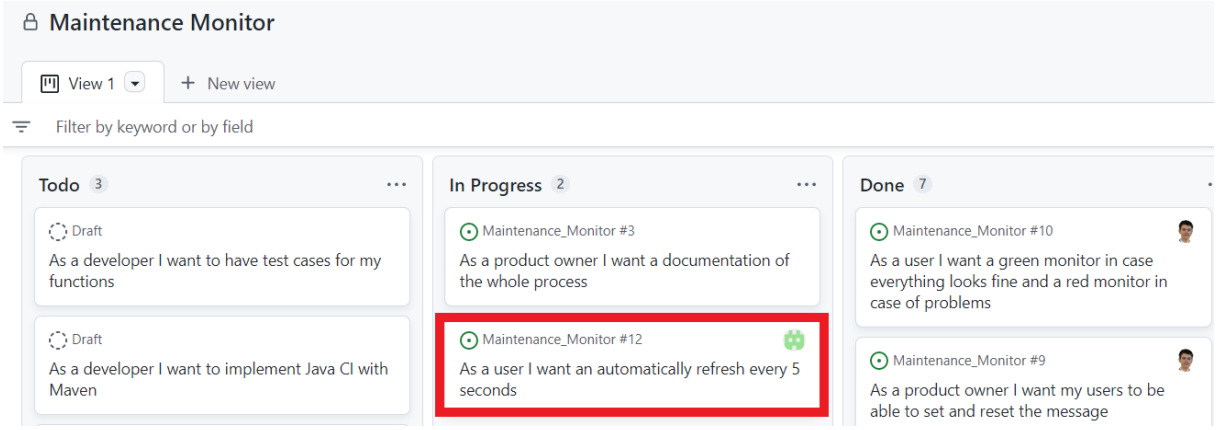




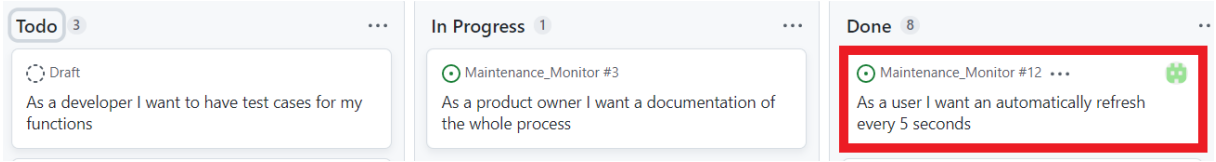




9. Add 5 second refresh

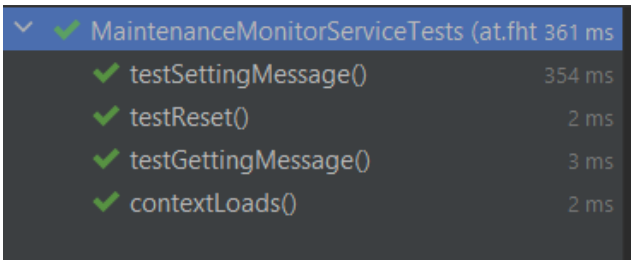


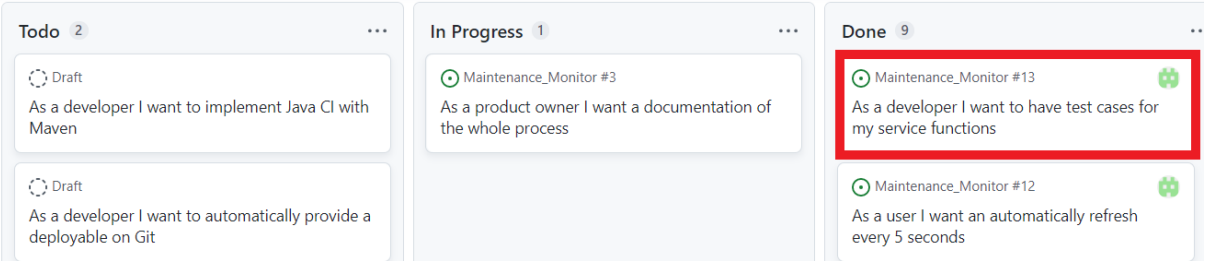

```
<meta http-equiv="refresh" content="5" />
```



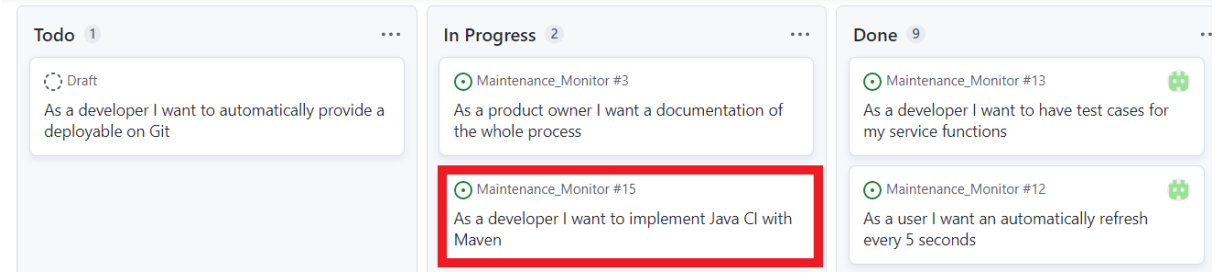
10. Test cases Service Klasse (get,set,reset)

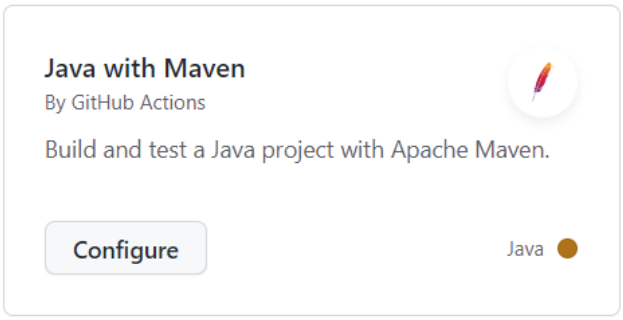
```java
@Test
void contextLoads() {
}

@Test
void testGettingMessage() {
    assertEquals( expected: "Stored Message", monitorService.getMessage());
}

@Test
void testSettingMessage() {
    monitorService.setMessage("Test");
    assertEquals( expected: "Test", monitorService.getMessage());
}

@Test
void testReset() {
    monitorService.setMessage("Test");
    monitorService.resetMessage();
    assertEquals( expected: "", monitorService.getMessage());
}
```

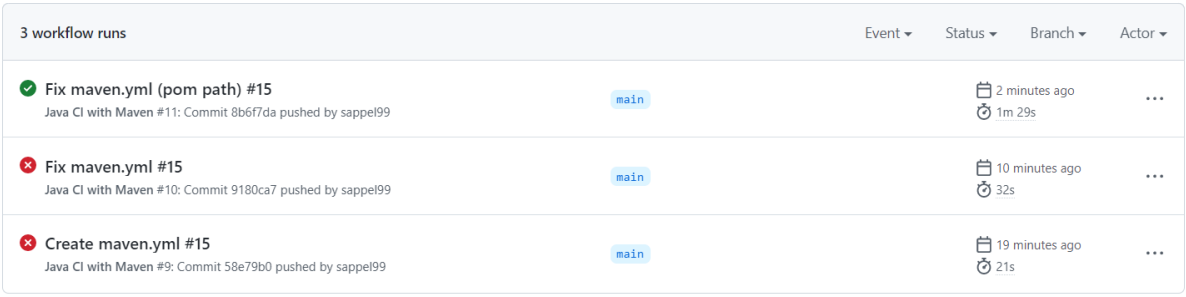


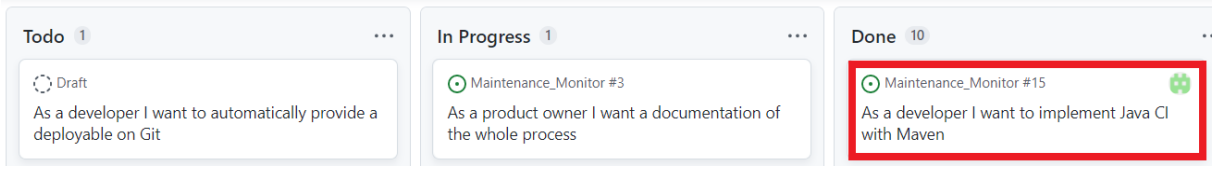


11. Add Java CI with Maven



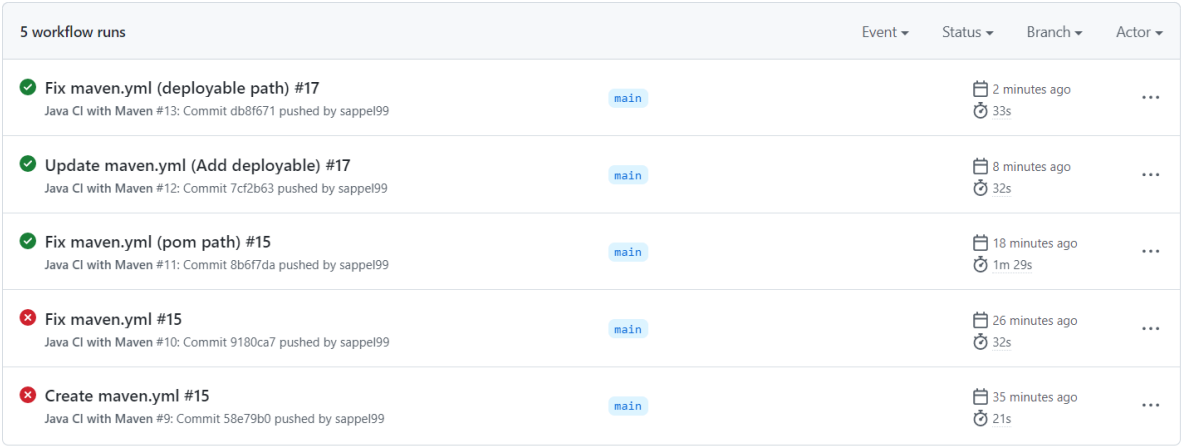


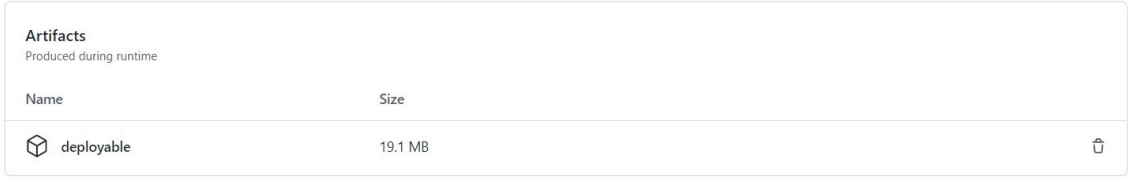


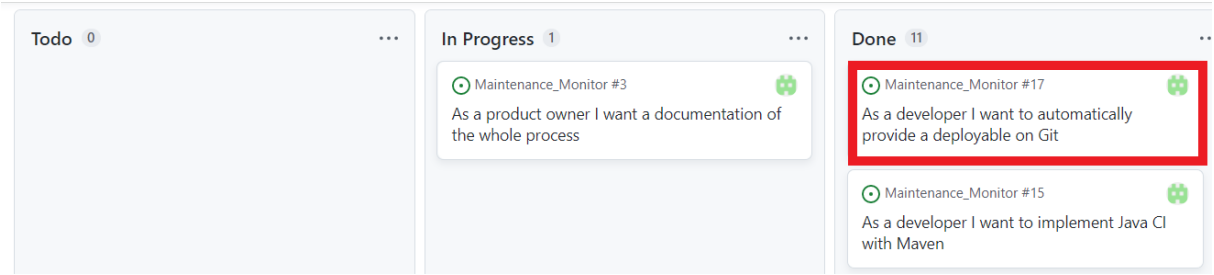


12. Automatically upload deployable

```yaml
- name: Build with Maven
  run: mvn -B package --file MaintenanceMonitor/pom.xml
- name: Upload a Build Artifact
  uses: actions/upload-artifact@v3
  with:
    # Artifact name
    name: deployable
    # A file, directory or wildcard pattern that describes what to upload
    path: MaintenanceMonitor/target/*.jar
    # The desired behavior if no files are found using the provided path.
```







13. Return change.html after setting a message (to not show error

localhost:8080/message/test

Whitelabel Error Page

This application has no explicit mapping for /error, so you are seeing this as a fallback.

Thu Dec 01 18:28:29 CET 2022
There was an unexpected error (type=Internal Server Error, status=500).

change.html | mes.html | MaintenanceMonitorController.java | home.html

```html
<!DOCTYPE HTML>
<html xmlns:th="http://www.thymeleaf.org">
<head>
  <title>Home</title>
  <meta http-equiv="Content-Type" content="text/html; charset
</head>
<body>
<p th:text="${setText}" />
<p th:text="'Return to localhost:8080/message to see it'" />
</body>
</html>
```

```
model.addAttribute( attributeName: "setText", attributeValue: "You just set
return "change";   You, Moments ago • Uncommitted changes
```

localhost:8080/message/test

You just set the message to: "test"

Return to localhost:8080/message to see it

Todo 0

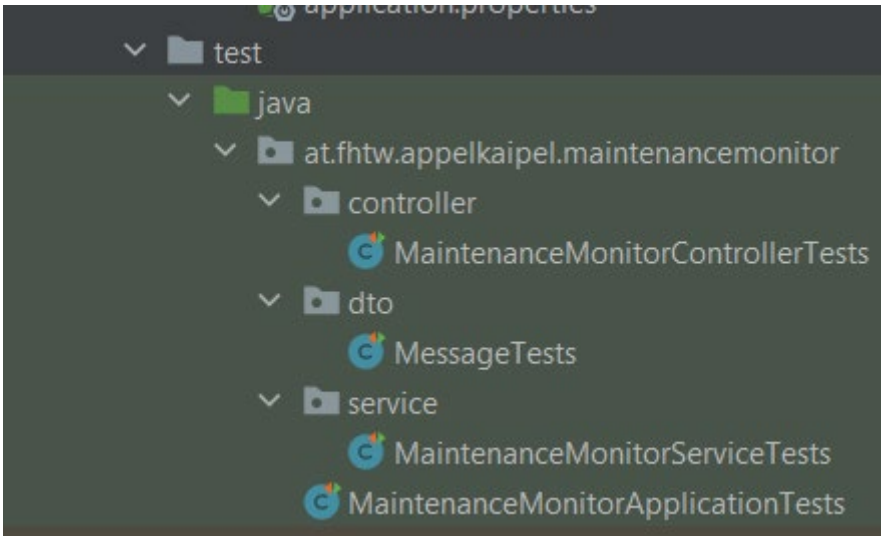
In Progress 1

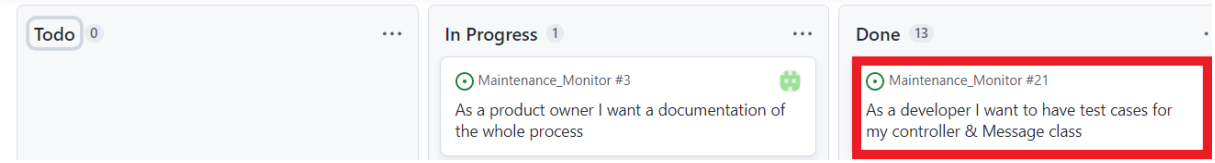
Maintenance_Monitor #3
As a product owner I want a documentation of the whole process

Done 12

Maintenance_Monitor #18
As a user I dont want to see a web error after setting a message

14. Create Unit Tests for Controller & Message Klasse





Requirement definitions (User Stories)

Maintenance Monitor

View 1 | New view

	Title	Assignees	Status
1	As a developer I want to have test cases for my controller & Message class #21		Done
2	As a user I dont want to see a web error after setting a message #18	sappel99	Done
3	As a developer I want to automatically provide a deployable on Git #17	sappel99	Done
4	As a developer I want to implement Java CI with Maven #15	sappel99	Done
5	As a developer I want to have test cases for my service functions #13	sappel99	Done
6	As a user I want an automatically refresh every 5 seconds #12	sappel99	Done
7	As a user I want a green monitor in case everything looks fine and a red monitor in case of pı #10	Cleingoun	Done
8	As a product owner I want my users to be able to set and reset the message #9	Cleingoun	Done
9	As a product owner I want my users to be able to see a centrally stored message #7	Cleingoun	Done
10	As a developer I want to implement Thymeleaf as Web REST #6	sappel99	Done
11	As a developer I want to implement the needed REST functions in Java (set/get/reset) #4	sappel99	Done
12	As a developer I want to create an infrastructure for a basic REST service in Java. #2	sappel99	Done
13	As a developer I want to have an explaining Readme and a useful Git Structure #1	Cleingoun	Done
14	As a product owner I want a documentation of the whole process #3	sappel99	In Progress

Todo 5

As a developer I want to have test cases for my functions

Draft
As a developer I want to implement Java CI with Maven

Draft
As a developer I want to automatically provide a deployable on Git

Draft
As a product owner I want my users to be able to set and reset the message

Draft

In Progress 2

Maintenance_Monitor #3
As a product owner I want a documentation of the whole process

Maintenance_Monitor #7
As a product owner I want my users to be able to see a centrally stored message

Done 4

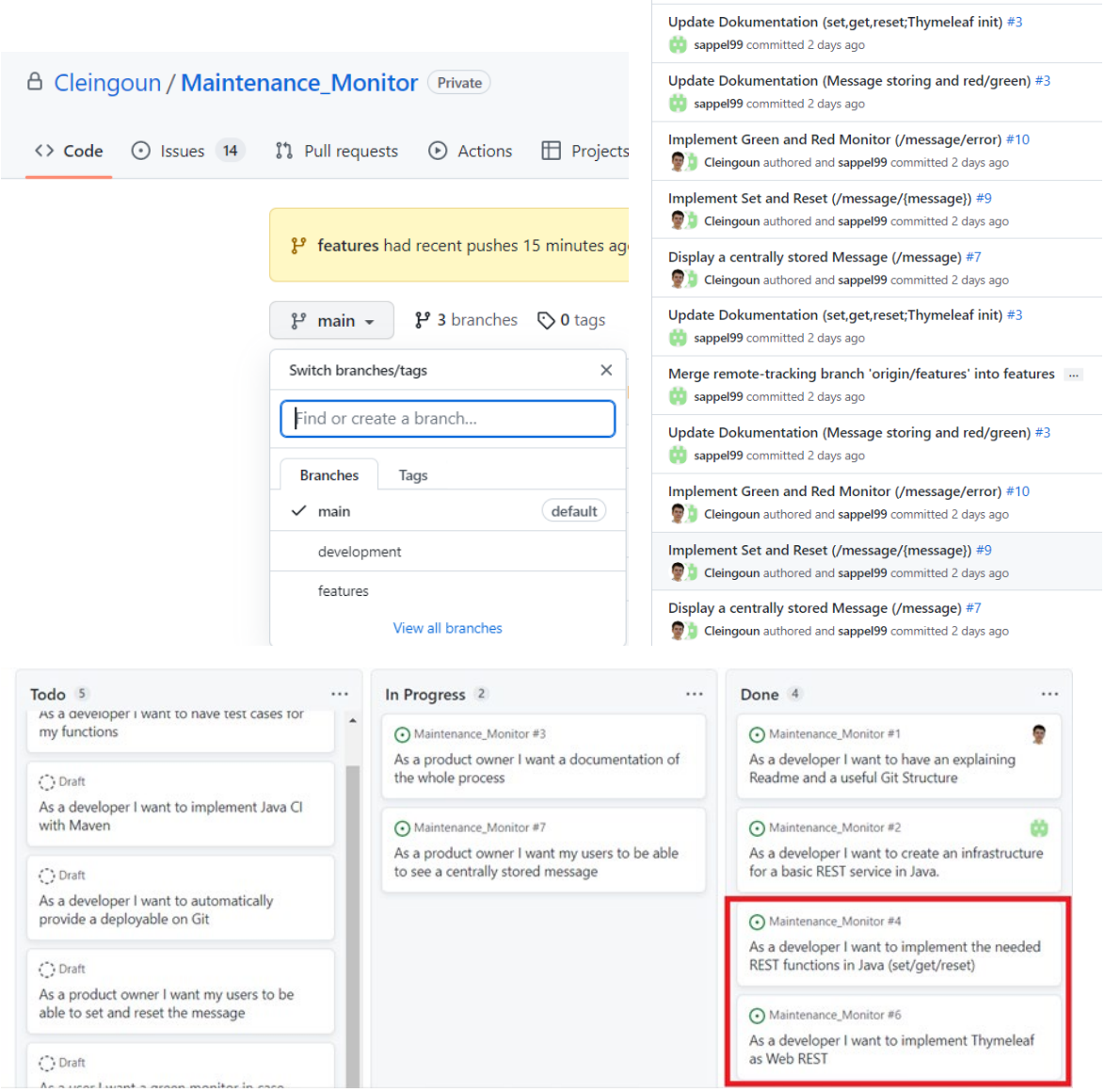
Maintenance_Monitor #1
As a developer I want to have an explaining Readme and a useful Git Structure

Maintenance_Monitor #2
As a developer I want to create an infrastructure for a basic REST service in Java.

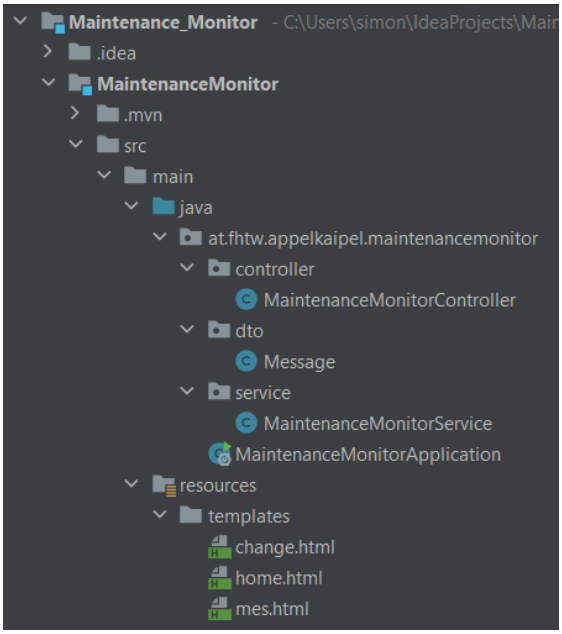
Maintenance_Monitor #4
As a developer I want to implement the needed REST functions in Java (set/get/reset)

Maintenance_Monitor #6
As a developer I want to implement Thymeleaf as Web REST

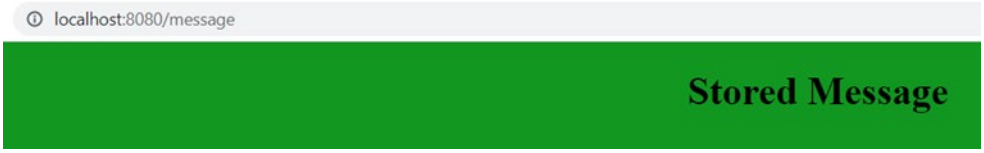
Correct status / Linking / Branching (Kanban, Git)



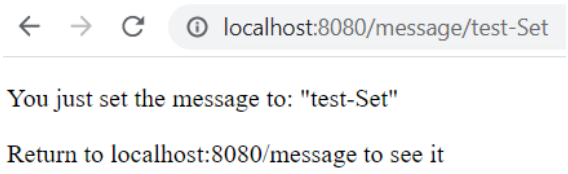
Implementation



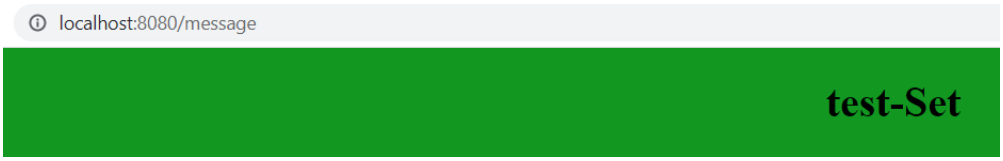
Aufruf von localhost:8080/message = Show Stored Message



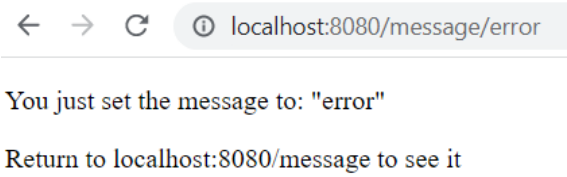
Aufruf von localhost:8080/message/test-Set = Set Message



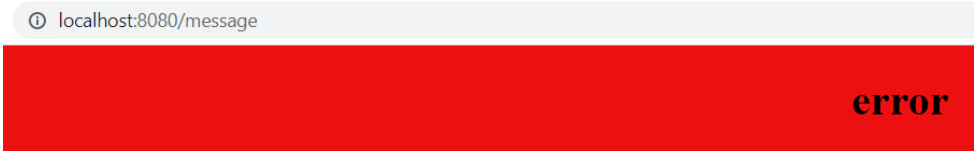
Aufruf von localhost:8080/message = Show Set Message



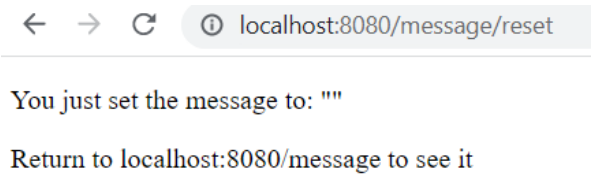
Aufruf von localhost:8080/message/error = Set Error



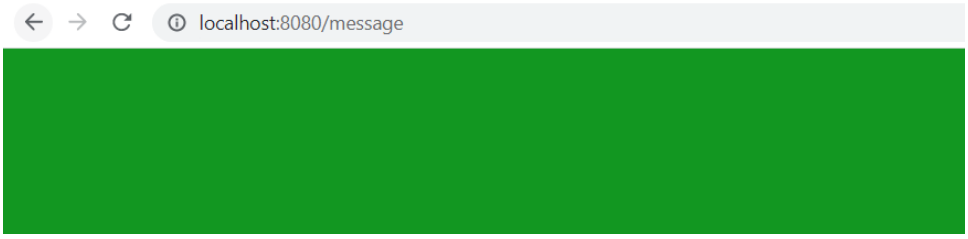
Aufruf von localhost:8080/message = Show Error



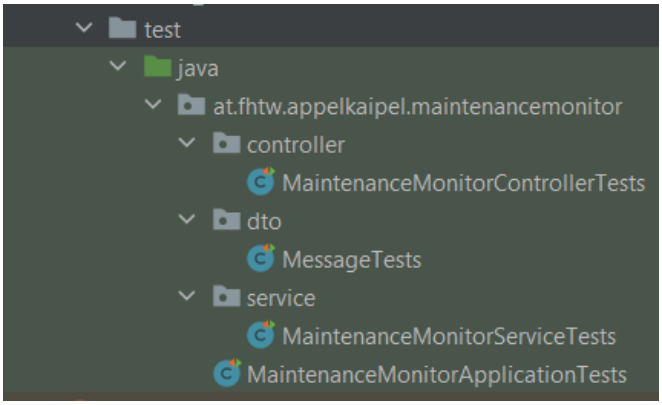
Aufruf von localhost:8080/message/reset = Reset



Aufruf von localhost:8080/message = Show Reset



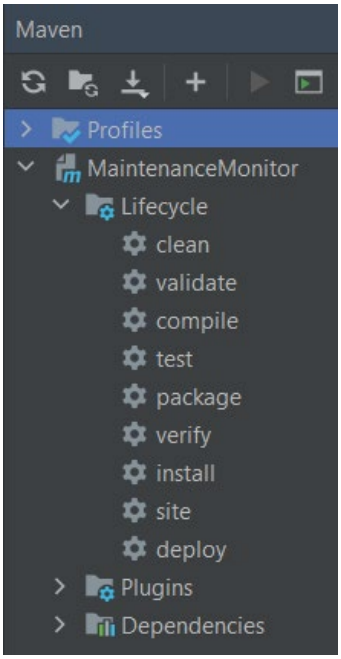
Testing



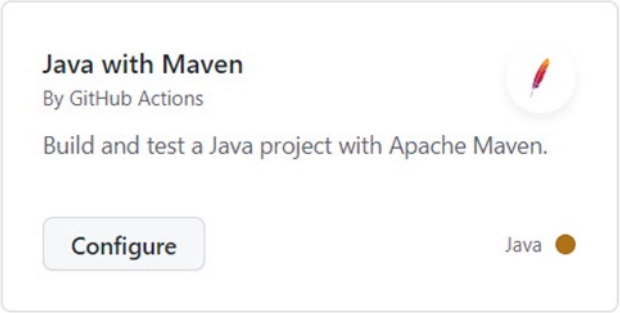

```java
public class MaintenanceMonitorServiceTests {
    6 usages
    private final MaintenanceMonitorService monitorService = new MaintenanceMonitorService();

    sappel99
    @Test
    void contextLoads() {
    }

    sappel99
    @Test
    void testGettingMessage() { assertEquals( expected: "Stored Message", monitorService.getMessage()); }

    sappel99
    @Test
    void testSettingMessage() {
        monitorService.setMessage("Test");
        assertEquals( expected: "Test", monitorService.getMessage());
    }

    sappel99
    @Test
    void testReset() {
        monitorService.setMessage("Test");
        monitorService.resetMessage();
        assertEquals( expected: "", monitorService.getMessage());
    }
}
```

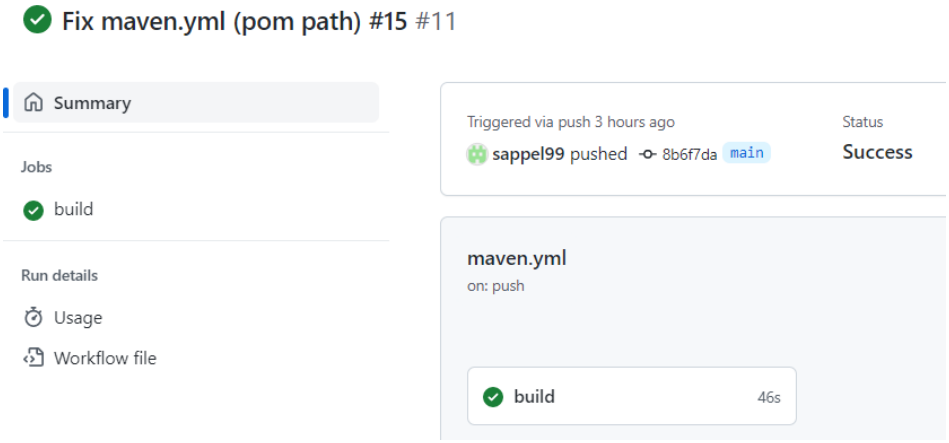
```
[INFO] Results:
[INFO]
[INFO] Tests run: 11, Failures: 0, Errors: 0, Skipped: 0
[INFO]
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time:  10.023 s
[INFO] Finished at: 2022-12-01T20:30:25+01:00
[INFO] ------------------------------------------------------------------------
```

Pipeline (Continuous Integration and Maven)




```
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="ht
    xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 https
    <modelVersion>4.0.0</modelVersion>
    <parent>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-parent</artifactId>
        <version>3.0.0</version>
        <relativePath/> <!-- lookup parent from repository -->
    </parent>
    <groupId>at.fhtw.appelkaipel</groupId>
    <artifactId>MaintenanceMonitor</artifactId>
    <version>0.0.1-SNAPSHOT</version>
    <name>MaintenanceMonitor</name>
    <description>MaintenanceMonitor</description>
    <properties>
        <java.version>17</java.version>
    </properties>
```





Artefacts (Continuous Delivery)

```yaml
- name: Upload a Build Artifact
  uses: actions/upload-artifact@v3
  with:
    # Artifact name
    name: deployable
    # A file, directory or wildcard pattern that describes what to upload
    path: MaintenanceMonitor/target/*.jar
    # The desired behavior if no files are found using the provided path.
```