

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: 15%](#)
- [Requirement definitions \(User Stories\): 15%](#)
- [Correct status / Linking / Branching \(Kanban, Git\): 15%](#)
- [Implementation: 15%](#)
- [Testing: 15%](#)
- [Pipeline \(Continuous Integration and Maven\): 15%](#)
- [Artefacts \(Continuous Delivery\): 10%](#)

Documentation of the process

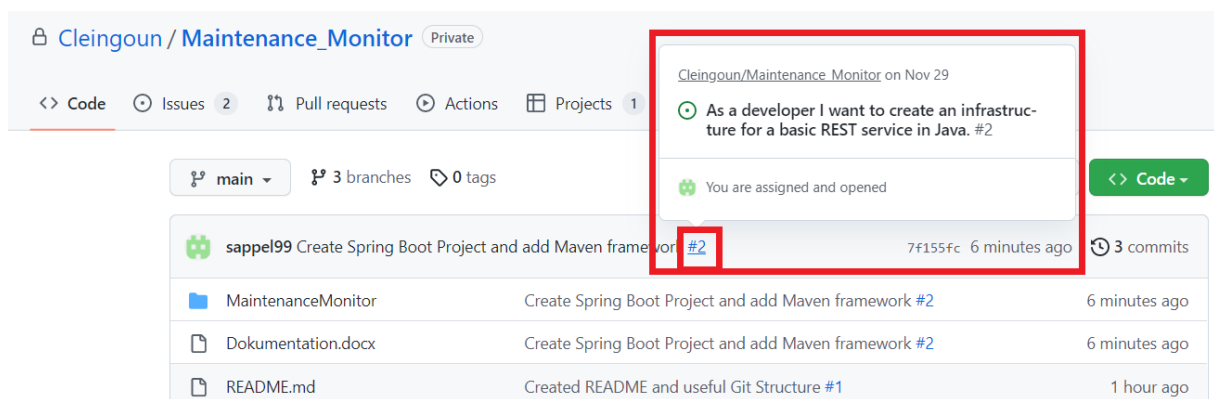
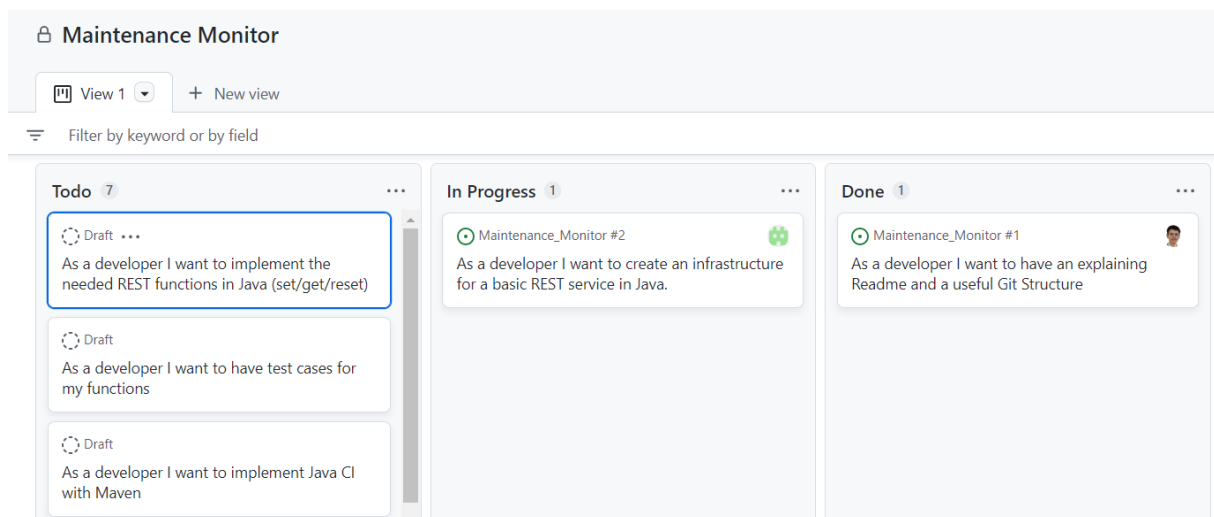
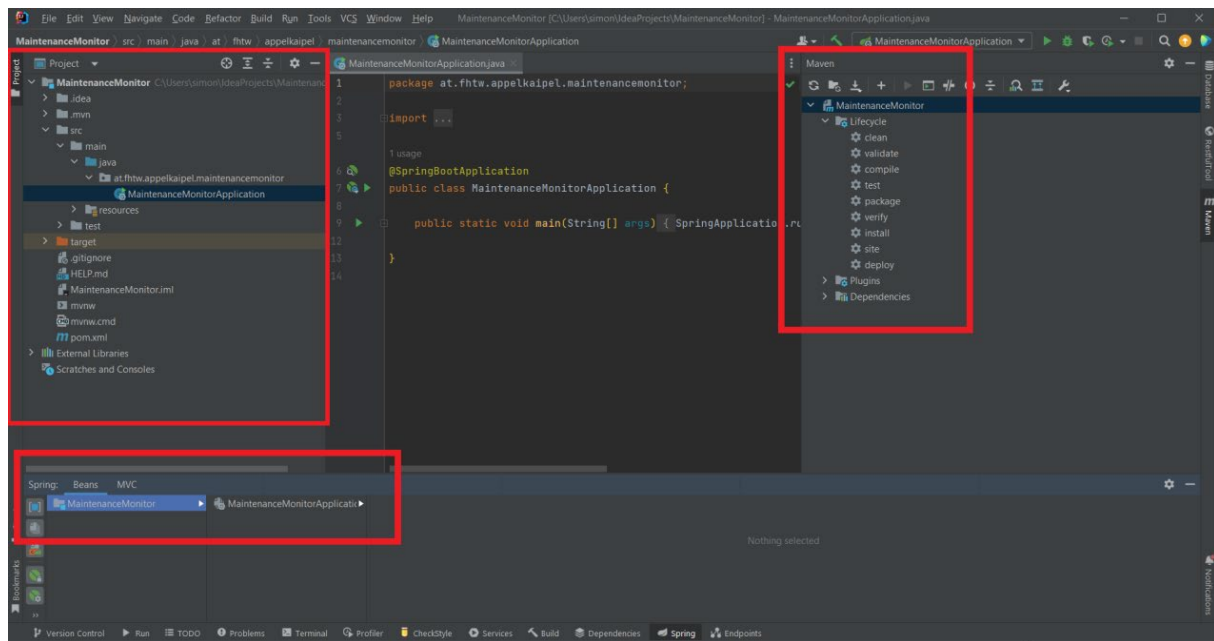
1. Git Repository erstellen, Branches erstellen, erste User Stories definieren

The screenshot displays a GitHub repository named 'Maintenance_Monitor' by user 'Cleingoun'. The repository is private and has 1 watch, 0 forks, and 0 stars. The main branch is 'main', and there are 3 branches in total. A branch manager dropdown is open, showing 'main' as the current branch and 'development' as the default branch. The repository description states: 'We are a small hydro-power electricity supplier near Vienna. Our customers expect electricity around the clock with a service level agreement of 99.95%. This means that the maximum outage of 21 minutes and 54 seconds (monthly in sum) is tolerated. It is easy to derive that service times are very important to us. Huge monitors were installed that should show'.

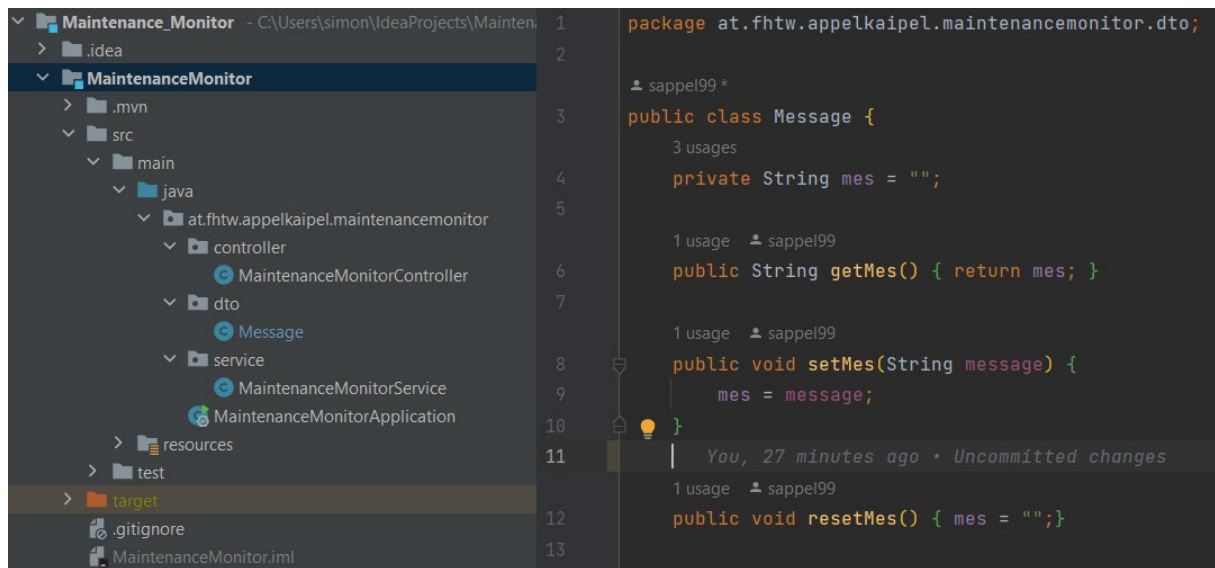
The Kanban board shows three columns: 'Todo' (8 items), 'In Progress' (1 item), and 'Done' (0 items). The 'Todo' column contains two items:

- Maintenance_Monitor #2**: As a developer I want to create an infrastructure for a basic REST service in Java. (Status: Draft)
- Maintenance_Monitor #1**: As a developer I want to have an explaining Readme and a useful Git Structure.

2. Spring Boot Projekt erstellen, Maven einrichten



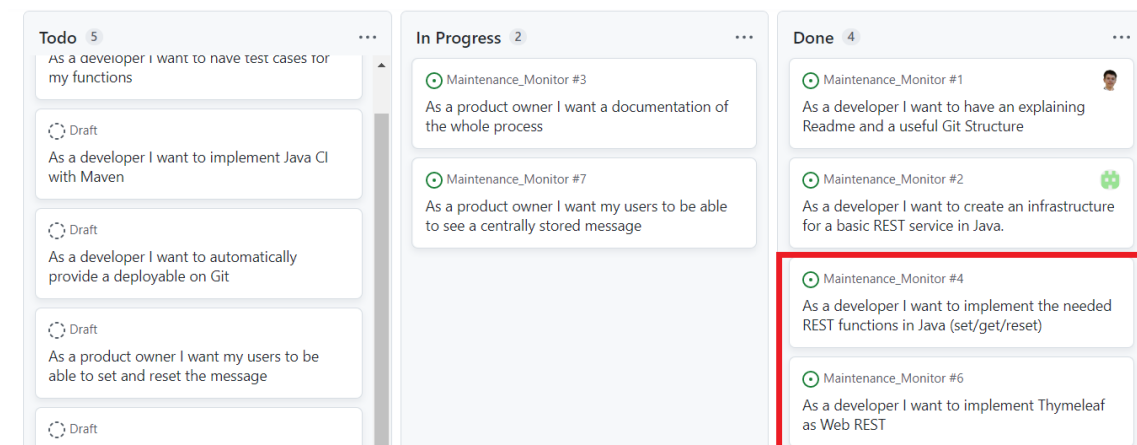
3. Create Controller, Message Class und Service
4. Create Setter, Getter, Reset REST functionality



5. Add Thymeleaf Module Template (/home)

```
@GetMapping("/home")
public String home() {
    return "home";
}
```

```
<!DOCTYPE HTML>
<html xmlns:th="http://www.thymeleaf.org">
<head>
    <title>Home</title>
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
</head>
<body><p th:text="'Ich bin eine Thymeleaf-View!'" />
</body>
</html>
```



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("message", message);

    return "mes";
}
```

localhost:8080/message

Stored Message

The screenshot shows a Kanban board with three columns: 'Todo' (5 items), 'In Progress' (2 items), and 'Done' (5 items). The 'Done' column contains a card for 'Maintenance_Monitor #7' with the text 'As a product owner I want my users to be able to see a centrally stored message'. This card is highlighted with a red box. Other cards in the 'Done' column include 'Maintenance_Monitor #6' and 'Maintenance_Monitor #9'. The 'In Progress' column contains cards for 'Maintenance_Monitor #3' and 'Maintenance_Monitor #9'. The 'Todo' column contains cards for 'Draft' and 'Draft'.

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

Todo 4	In Progress 2	Done 6
<div><div>Draft</div><div>As a user I want an automatically refresh every 5 seconds</div></div>	<div><div>Maintenance_Monitor #3</div><div>As a product owner I want a documentation of the whole process</div></div>	<div><div>Maintenance_Monitor #9</div><div>As a product owner I want my users to be able to set and reset the message</div></div>
<div><div>Draft</div><div>As a developer I want to have test cases for my functions</div></div>	<div><div>Maintenance_Monitor #10 ...</div><div>As a user I want a green monitor in case everything looks fine and a red monitor in case of problems</div></div>	<div><div>Maintenance_Monitor #7</div><div>As a product owner I want my users to be able to see a centrally stored message</div></div>

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

```

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

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

    return "mes";
}

```

localhost:8080/message

Stored Message

localhost:8080/message/error

localhost:8080/message

error

localhost:8080/message/reset

localhost:8080/message

Todo 4	In Progress 1	Done 7
<div><div>Draft</div><div>As a user I want an automatically refresh every 5 seconds</div></div> <div><div>Draft</div><div>As a developer I want to have test cases for my functions</div></div>	<div><div>Maintenance_Monitor #3</div><div>As a product owner I want a documentation of the whole process</div></div>	<div><div>Maintenance_Monitor #10</div><div>As a user I want a green monitor in case everything looks fine and a red monitor in case of problems</div></div> <div><div>Maintenance_Monitor #9</div><div>As a product owner I want my users to be able to set and reset the message</div></div>

9. Add 5 second refresh

Maintenance Monitor

View 1 + New view

Filter by keyword or by field

Todo 3

- Draft
As a developer I want to have test cases for my functions
- Draft
As a developer I want to implement Java CI with Maven

In Progress 2

- Maintenance_Monitor #3
As a product owner I want a documentation of the whole process
- Maintenance_Monitor #12
As a user I want an automatically refresh every 5 seconds

Done 7

- Maintenance_Monitor #10
As a user I want a green monitor in case everything looks fine and a red monitor in case of problems
- Maintenance_Monitor #9
As a product owner I want my users to be able to set and reset the message

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

Todo 3

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

In Progress 1

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

Done 8

- Maintenance_Monitor #12 ...
As a user I want an automatically refresh every 5 seconds

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

```
@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());
}
```

MaintenanceMonitorServiceTests (at.fht 361 ms)	
✓ testSettingMessage()	354 ms
✓ testReset()	2 ms
✓ testGettingMessage()	3 ms
✓ contextLoads()	2 ms

Todo 2	In Progress 1	Done 9
<div>Draft</div> <div>As a developer I want to implement Java CI with Maven</div>	<div>Maintenance_Monitor #3</div> <div>As a product owner I want a documentation of the whole process</div>	<div>Maintenance_Monitor #13</div> <div>As a developer I want to have test cases for my service functions</div>
<div>Draft</div> <div>As a developer I want to automatically provide a deployable on Git</div>		<div>Maintenance_Monitor #12</div> <div>As a user I want an automatically refresh every 5 seconds</div>

11. Add Java CI with Maven

Todo 1

Draft

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

In Progress 2

Maintenance_Monitor #3

As a product owner I want a documentation of the whole process

Maintenance_Monitor #15

As a developer I want to implement Java CI with Maven

Done 9

Maintenance_Monitor #13

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

Maintenance_Monitor #12

As a user I want an automatically refresh every 5 seconds

Java with Maven

By GitHub Actions

Build and test a Java project with Apache Maven.

Configure

Java

3 workflow runs				Event ▾	Status ▾	Branch ▾	Actor ▾
✓	Fix maven.yml (pom path) #15	main	2 minutes ago	1m 29s	...		
Java CI with Maven #11: Commit 8b6f7da pushed by sappel99							
✗	Fix maven.yml #15	main	10 minutes ago	32s	...		
Java CI with Maven #10: Commit 9180ca7 pushed by sappel99							
✗	Create maven.yml #15	main	19 minutes ago	21s	...		
Java CI with Maven #9: Commit 58e79b0 pushed by sappel99							

Todo 1

Draft

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

In Progress 1

Maintenance_Monitor #3

As a product owner I want a documentation of the whole process

Done 10



Maintenance_Monitor #15

As a developer I want to implement Java CI with Maven

12. Automatically upload deployable

```
- 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.
```

5 workflow runs				Event ▼	Status ▼	Branch ▼	Actor ▼
✓	Fix maven.yml (deployable path) #17	main	2 minutes ago	33s	...		
Java CI with Maven #13: Commit db8f671 pushed by sappel99							
✓	Update maven.yml (Add deployable) #17	main	8 minutes ago	32s	...		
Java CI with Maven #12: Commit 7cf2b63 pushed by sappel99							
✓	Fix maven.yml (pom path) #15	main	18 minutes ago	1m 29s	...		
Java CI with Maven #11: Commit 8b6f7da pushed by sappel99							
✗	Fix maven.yml #15	main	26 minutes ago	32s	...		
Java CI with Maven #10: Commit 9180ca7 pushed by sappel99							
✗	Create maven.yml #15	main	35 minutes ago	21s	...		
Java CI with Maven #9: Commit 58e79b0 pushed by sappel99							

Artifacts		
Produced during runtime		
Name	Size	
 deployable	19.1 MB	

Todo 0	In Progress 1	Done 11
	<div>Maintenance_Monitor #3 As a product owner I want a documentation of the whole process</div>	<div>Maintenance_Monitor #17 As a developer I want to automatically provide a deployable on Git</div> <div>Maintenance_Monitor #15 As a developer I want to implement Java CI with Maven</div>

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).

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

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

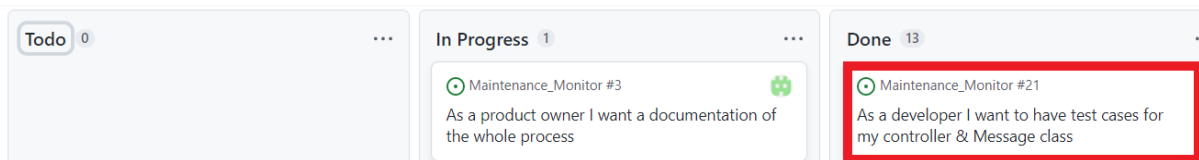
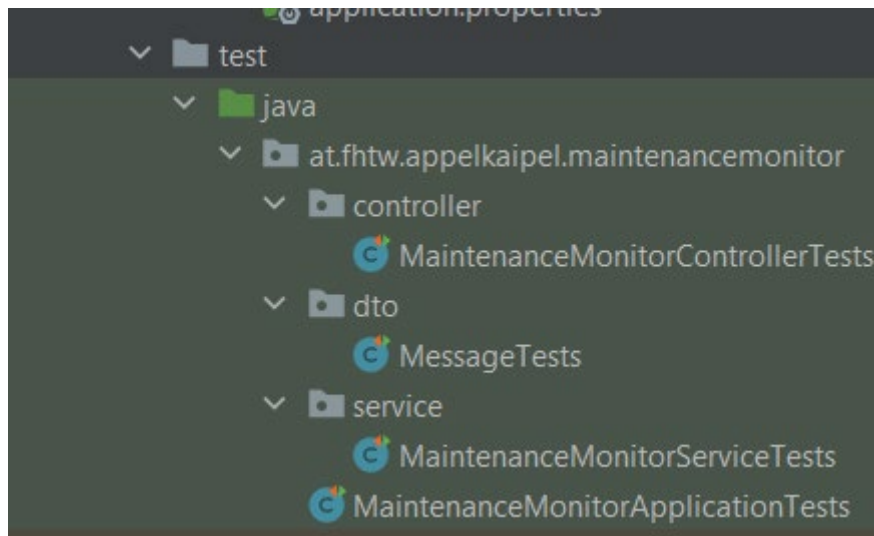
← → ↻ ⓘ localhost:8080/message/test

You just set the message to: "test"

Return to localhost:8080/message to see it

Todo 0	In Progress 1	Done 12
	<div>Maintenance_Monitor #3</div> <div>As a product owner I want a documentation of the whole process</div>	<div>Maintenance_Monitor #18</div> <div>As a user I dont want to see a web error after setting a message</div>

14. Create Unit Tests for Controller & Message Klasse



Requirement definitions (User Stories)

Maintenance Monitor				
<div> <div>View 1</div> <div>+ New view</div> </div>				
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 pi #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

As a user I want a green monitor in case

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)

The screenshot displays a GitHub repository interface for 'Cleingoun / Maintenance_Monitor'. The top navigation bar includes links for Code, Issues (14), Pull requests, Actions, and Projects. A yellow banner indicates 'features had recent pushes 15 minutes ago'. Below this, a 'Switch branches/tags' dropdown menu is open, showing the 'main' branch as the current selection, with other branches like 'development' and 'features' listed. To the right, a list of recent commits is visible, including updates to documentation and implementation of REST functions.

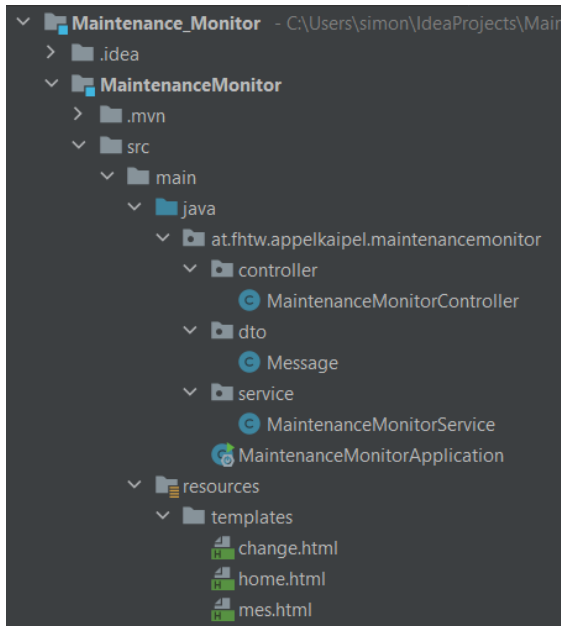
Commits:

- Update Dokumentation (set,get,reset;Thymeleaf init) #3 (sappel99 committed 2 days ago)
- Update Dokumentation (Message storing and red/green) #3 (sappel99 committed 2 days ago)
- Implement Green and Red Monitor (/message/error) #10 (Cleingoun authored and sappel99 committed 2 days ago)
- Implement Set and Reset (/message/(message)) #9 (Cleingoun authored and sappel99 committed 2 days ago)
- Display a centrally stored Message (/message) #7 (Cleingoun authored and sappel99 committed 2 days ago)
- Update Dokumentation (set,get,reset;Thymeleaf init) #3 (sappel99 committed 2 days ago)
- Merge remote-tracking branch 'origin/features' into features (sappel99 committed 2 days ago)
- Update Dokumentation (Message storing and red/green) #3 (sappel99 committed 2 days ago)
- Implement Green and Red Monitor (/message/error) #10 (Cleingoun authored and sappel99 committed 2 days ago)
- Implement Set and Reset (/message/(message)) #9 (Cleingoun authored and sappel99 committed 2 days ago)
- Display a centrally stored Message (/message) #7 (Cleingoun authored and sappel99 committed 2 days ago)

Kanban Board:

- 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: As a user I want a green monitor in case...
- 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

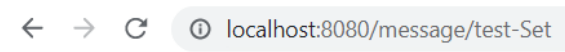
Implementation



Aufruf von localhost:8080/message = **Show Stored Message**



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



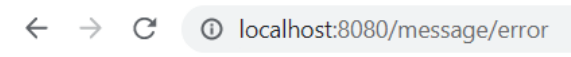
You just set the message to: "test-Set"

Return to localhost:8080/message to see it

Aufruf von localhost:8080/message = **Show Set Message**



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



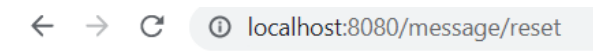
You just set the message to: "error"

Return to localhost:8080/message to see it

Aufruf von localhost:8080/message = **Show Error**



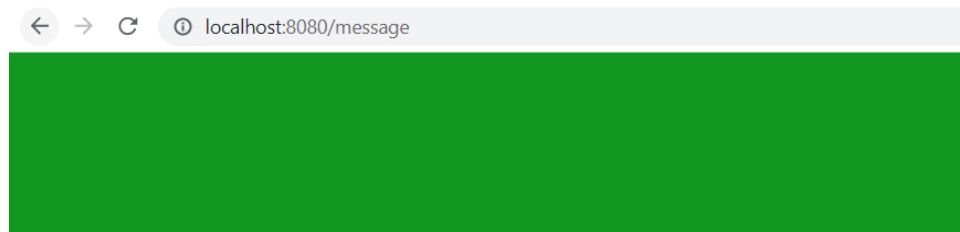
Aufruf von localhost:8080/message/reset = **Reset**



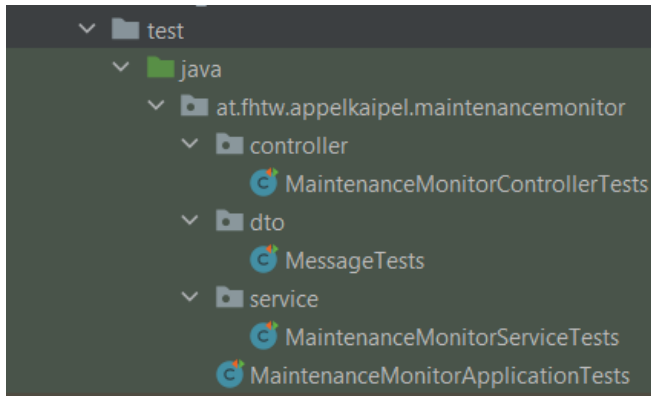
You just set the message to: ""

Return to localhost:8080/message to see it

Aufruf von localhost:8080/message = **Show Reset**



Testing



```
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)

Maven

Profiles

MaintenanceMonitor

Lifecycle

clean

validate

compile

test

package

verify

install

site

deploy

Plugins

Dependencies

```
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://maven.apache.org/POM/4.0.0" xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 https://maven.apache.org/maven-v4_0_0.xsd">
  <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>
</project>
```

Java with Maven

By GitHub Actions

Build and test a Java project with Apache Maven.

Configure

Java

✓ Fix maven.yml (pom path) #15 #11

Summary

Jobs

✓ build

Run details

Usage

Workflow file

Triggered via push 3 hours ago

Status

sappel99 pushed 8b6f7da main

Success

maven.yml

on: push

✓ build 46s

Artefacts (Continuous Delivery)

```
- 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.
```

← Java CI with Maven
Merge pull request #20 from Cleingoun/development #15

- Summary
- Jobs
 - build
- Run details
 - Usage
 - Workflow file

Triggered via push 2 hours ago

sappel99 pushed -> 88b91b4 main

Status

Success

Total duration

43s

Billable time

1m

Artifacts

1

maven.yml

on: push

build

31s

Artifacts

Produced during runtime

Name	Size
deployable	19.1 MB