Testing Report

GTA-VI - A Nedap Theft Dashboard

Antoine Moghaddar	(S1880241)
Michael Raczkiewicz	(S2524732)
Duru Koçak	(S2486431)
Valeria Veravita	(S2456036)
Sadat Ahmad	(S2610736)

Module 4: Data & Information
University of Twente
2020-2021

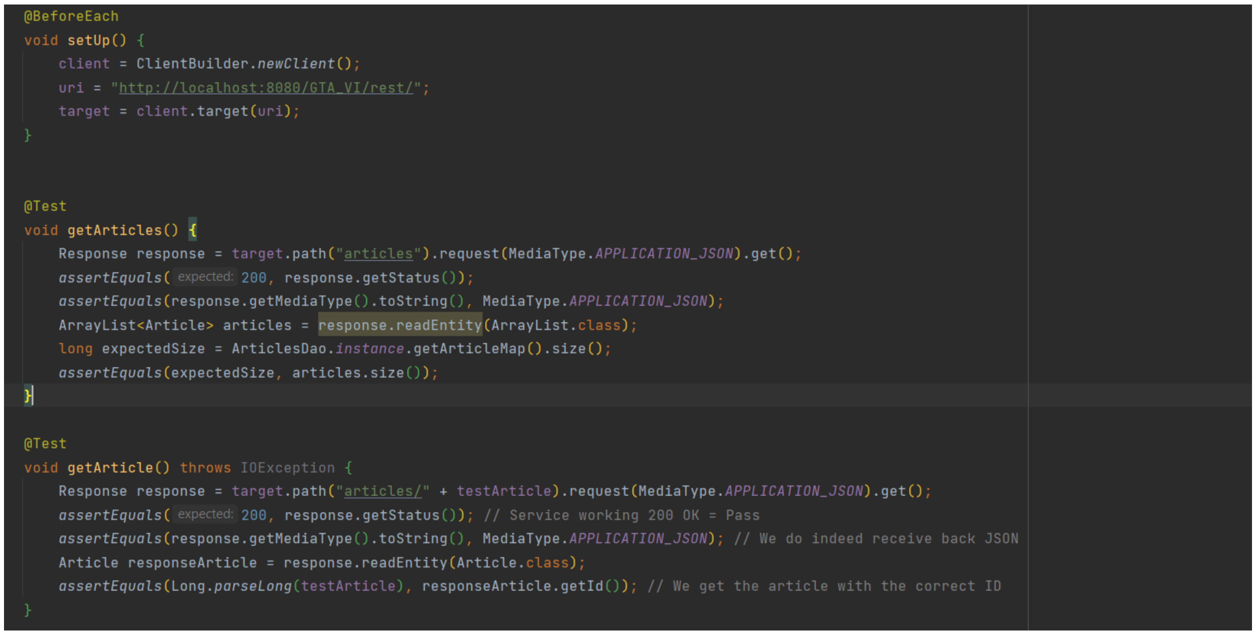
Unit test cases

Unit tests cover all Resource classes and all methods accessible in those classes. This ensures that every piece of the back end architecture works correctly. Some methods are not tested. This is on purpose, as those methods are rather small and serve a purpose of helper methods.

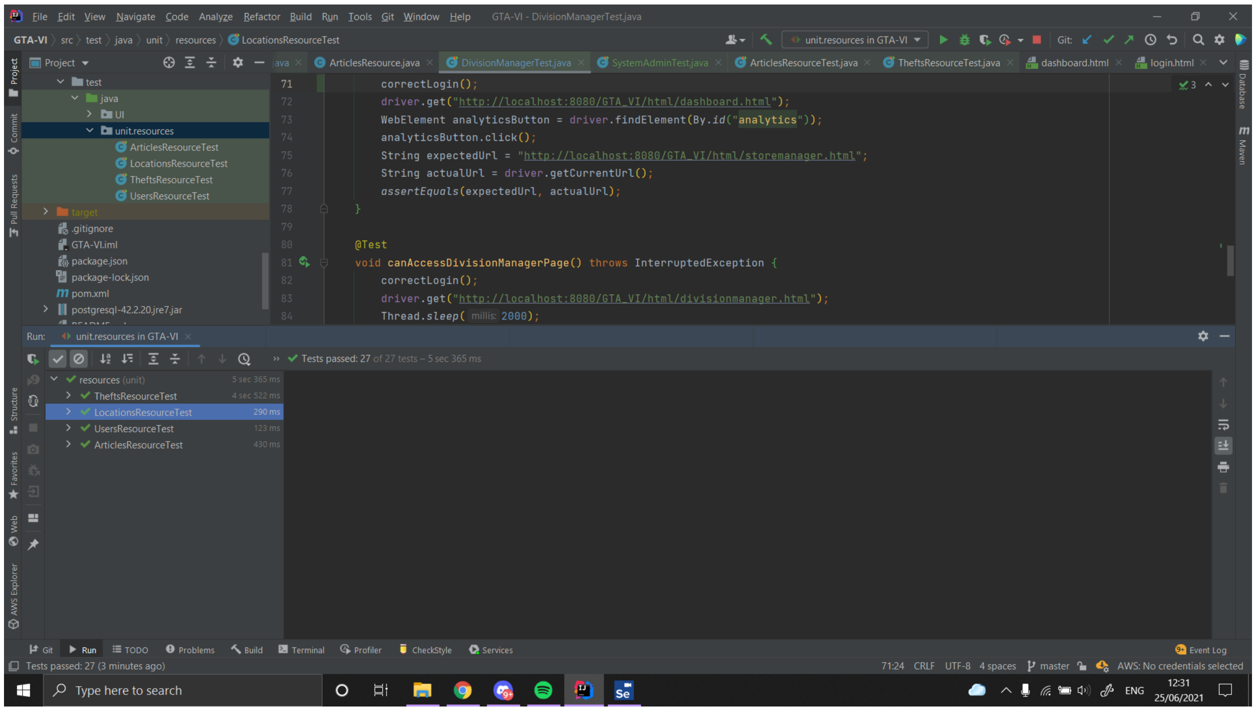
Typical test case consists of following elements sub test cases:

1. Check if the HTTP response code is appropriate. Typically that would be 200 as most of the HTTP requests return data in either text or JSON format. Sometimes the code is 204 which of course is accounted for.
2. Check if the appropriate data format was received. In the majority of cases this will be JSON.
3. Check if returned JSON can be parsed to XMLRootElement type object which ensures that this functionality of the system is working.
4. The final test verifies if the correct article was fetched from the database.

Given that we have about 30 different test cases we can pretty reliably cover the entirety of the back end architecture.



```java
@BeforeEach
void setUp() {
    client = ClientBuilder.newClient();
    uri = "http://localhost:8080/GTA_VI/rest/";
    target = client.target(uri);
}


@Test
void getArticles() {
    Response response = target.path("articles").request(MediaType.APPLICATION_JSON).get();
    assertEquals( expected: 200, response.getStatus());
    assertEquals(response.getMediaType().toString(), MediaType.APPLICATION_JSON);
    ArrayList<Article> articles = response.readEntity(ArrayList.class);
    long expectedSize = ArticlesDao.instance.getArticleMap().size();
    assertEquals(expectedSize, articles.size());
}

@Test
void getArticle() throws IOException {
    Response response = target.path("articles/" + testArticle).request(MediaType.APPLICATION_JSON).get();
    assertEquals( expected: 200, response.getStatus()); // Service working 200 OK = Pass
    assertEquals(response.getMediaType().toString(), MediaType.APPLICATION_JSON); // We do indeed receive back JSON
    Article responseArticle = response.readEntity(Article.class);
    assertEquals(Long.parseLong(testArticle), responseArticle.getId()); // We get the article with the correct ID
}
```



```java
        correctLogin();
        driver.get("http://localhost:8080/GTA_VI/html/dashboard.html");
        WebElement analyticsButton = driver.findElement(By.id("analytics"));
        analyticsButton.click();
        String expectedUrl = "http://localhost:8080/GTA_VI/html/storemanager.html";
        String actualUrl = driver.getCurrentUrl();
        assertEquals(expectedUrl, actualUrl);
    }

    @Test
    void canAccessDivisionManagerPage() throws InterruptedException {
        correctLogin();
        driver.get("http://localhost:8080/GTA_VI/html/divisionmanager.html");
        Thread.sleep( millis: 2000);
```

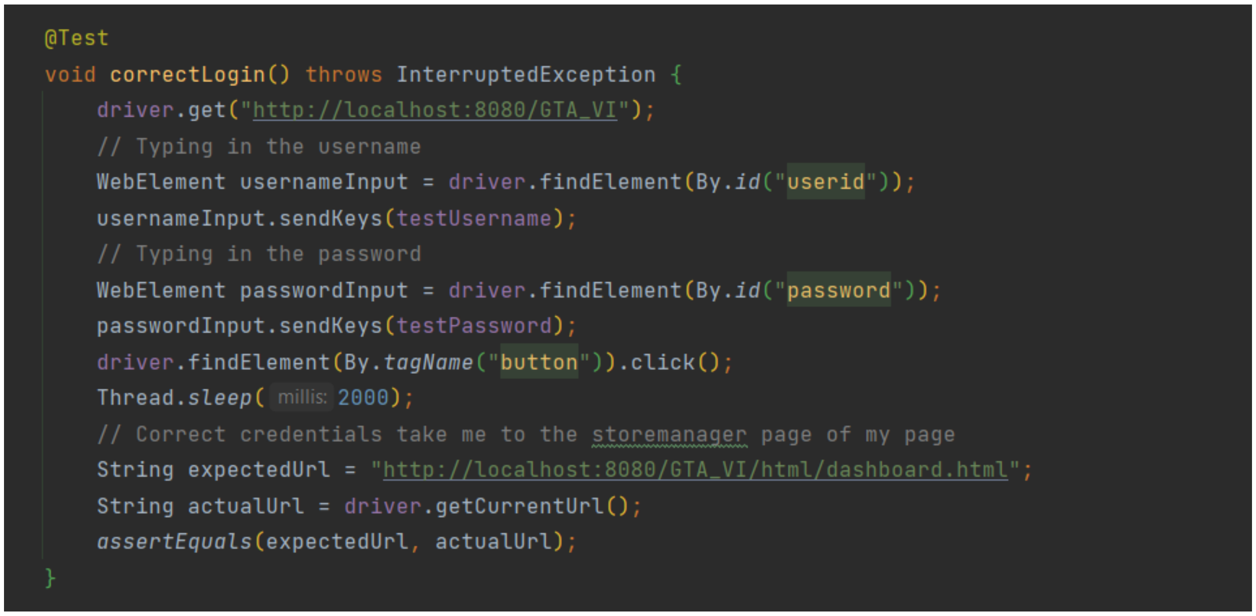
```
✓ Tests passed: 27 of 27 tests – 5 sec 365 ms

✓ resources (unit)              5 sec 365 ms
  ✓ TheftsResourceTest          4 sec 522 ms
  ✓ LocationsResourceTest             290 ms
  ✓ UsersResourceTest                 123 ms
  ✓ ArticlesResourceTest              430 ms
```

Tests passed: 27 (3 minutes ago)

File Edit View Navigate Code Analyze Refactor Build Run Tools Git Window Help
GTA-VI - DivisionManagerTest.java
GTA-VI › src › test › java › unit › resources › LocationsResourceTest
unit.resources in GTA-VI
ArticlesResource.java
DivisionManagerTest.java
SystemAdminTest.java
ArticlesResourceTest.java
TheftsResourceTest.java
dashboard.html
login.html
Project
test
java
UI
unit.resources
ArticlesResourceTest
LocationsResourceTest
TheftsResourceTest
UsersResourceTest
target
.gitignore
GTA-VI.iml
package.json
package-lock.json
pom.xml
postgresql-42.2.20.jre7.jar
Run: unit.resources in GTA-VI
Git Run TODO Problems Build Terminal Profiler CheckStyle Services Event Log
71:24 CRLF UTF-8 4 spaces master AWS: No credentials selected
Type here to search
ENG 12:31 25/06/2021

Selenium test cases

Selenium test cases allow for more elaborate testing. Selenium allowed us to emulate different scenarios one could find oneself in. We have prepared a lot of test cases that are aimed at the token system. In fact, this is the main way we test the tokens. Selenium has proven to be a rewarding tool to test our system in a comprehensive way. Most of the Selenium test cases are structured more or less in the same manner viz. a browser driver (we use chromdriver) is used to fetch our web application on the localhost. An element on the page is then found using the intrinsic methods of the driver and are passed a value or interacted with (‘clicked’). And for the test result, the driver URL is asserted against an expected value.



```java
@Test
void correctLogin() throws InterruptedException {
    driver.get("http://localhost:8080/GTA_VI");
    // Typing in the username
    WebElement usernameInput = driver.findElement(By.id("userid"));
    usernameInput.sendKeys(testUsername);
    // Typing in the password
    WebElement passwordInput = driver.findElement(By.id("password"));
    passwordInput.sendKeys(testPassword);
    driver.findElement(By.tagName("button")).click();
    Thread.sleep( millis: 2000);
    // Correct credentials take me to the storemanager page of my page
    String expectedUrl = "http://localhost:8080/GTA_VI/html/dashboard.html";
    String actualUrl = driver.getCurrentUrl();
    assertEquals(expectedUrl, actualUrl);
}
```