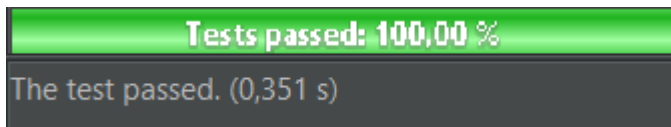


## USER OSZTÁLY:

Ez a Junit teszt a User osztályon végez egy get user by id http kérést, ellenőrizve, hogy egy érvényes ID-vel lekért felhasználó adatai helyesen jelennek-e meg, beleértve az emailt, és hogy a HTTP státuszkód 200 legyen.

Jó teszt: Egy get user by id az adatbázisban létező id-vel.



Hozzá tartozó kódrészlet:

```
@Test
public void testGetUserById_withValidId_returnsUser() {

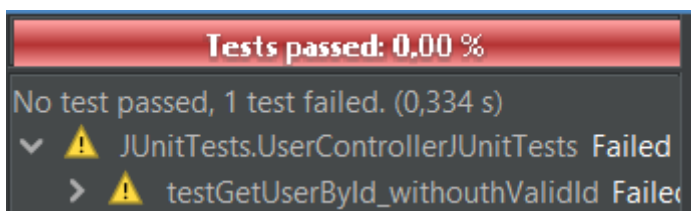
    Response response = client.target(BASE_URI)
        .path("getUserById")
        .queryParams("id", 1)
        .request(MediaType.APPLICATION_JSON)
        .get();

    assertEquals(200, response.getStatus(), "A státusznak 200-nak kell lennie");

    JSONObject responseBody = new JSONObject(response.readEntity(String.class));
    assertTrue(responseBody.has("id"), "A válaszban szerepelnie kell az 'id' mezőnek");
    assertEquals("User1@example.com", responseBody.getString("email"), "Az emailnek meg kell egyeznie");

    response.close();
}
```

Rossz teszt: A get user by id az adatbázisban nem létező id-vel



Hozzá tartozó kódrészlet:

```
@Test
public void testGetUserById_withouthValidId() {

    Response response = client.target(BASE_URI)
        .path("getUserById")
        .queryParams("id", 9999)
        .request(MediaType.APPLICATION_JSON)
        .get();

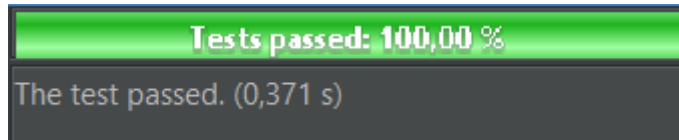
    assertEquals(200, response.getStatus(), "A státusznak 200-nak kell lennie");

    JSONObject responseBody = new JSONObject(response.readEntity(String.class));
    assertTrue(responseBody.has("id"), "A válaszban szerepelnie kell az 'id' mezőnek");
    assertEquals("User9999@example.com", responseBody.getString("email"), "Az emailnek meg kell egyeznie");

    response.close();
}
```

A testRegisterUser\_withValidData\_returnsSuccess teszt a POST /user/registerUser végpontot ellenőrzi, várva a 200-as státuszkódot és "success" választ érvényes adatokra.

Jó teszt mivel az adatok még nem léteztek az adatbázisban:



```
@Test
public void testRegisterUser_withValidData_returnsSuccess() {
    JSONObject requestBody = new JSONObject()
        .put("email", "test.elok@example.com")
        .put("firstName", "Test")
        .put("lastName", "Elok")
        .put("userName", "teastelek")
        .put("picture", "base64image")
        .put("password", "Test123!@#");

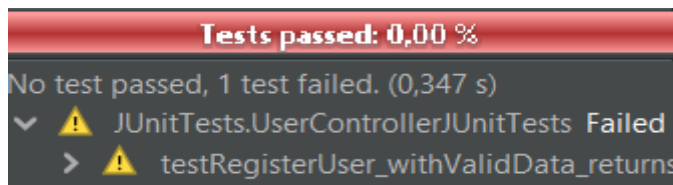
    Response response = client.target(BASE_URI)
        .path("registerUser")
        .request(MediaType.APPLICATION_JSON)
        .post(Entity.json(requestBody.toString()));

    assertEquals(200, response.getStatus(), "A státusznak 200-nak kell lennie");

    JSONObject responseBody = new JSONObject(response.readEntity(String.class));
    assertEquals("success", responseBody.getString("status"), "A státusz 'success' kell legyen");
    assertEquals(200, responseBody.getInt("statusCode"), "A státuszkódnak 200-nak kell lennie");

    response.close();
}
```

Rossz teszt mivel az adatbázisban már létezik ez a felhasználó:



```
@Test
public void testRegisterUser_withValidData_returnsSuccess() {
    JSONObject requestBody = new JSONObject()
        .put("email", "test.elok@example.com")
        .put("firstName", "Test")
        .put("lastName", "Elok")
        .put("userName", "teastelek")
        .put("picture", "base64image")
        .put("password", "Test123!@#");

    Response response = client.target(BASE_URI)
        .path("registerUser")
        .request(MediaType.APPLICATION_JSON)
        .post(Entity.json(requestBody.toString()));

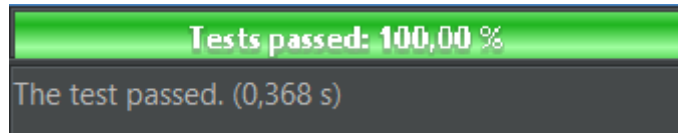
    assertEquals(200, response.getStatus(), "A státusznak 200-nak kell lennie");

    JSONObject responseBody = new JSONObject(response.readEntity(String.class));
    assertEquals("success", responseBody.getString("status"), "A státusz 'success' kell legyen");
    assertEquals(200, responseBody.getInt("statusCode"), "A státuszkódnak 200-nak kell lennie");

    response.close();
}
```

A testRegisterAdmin\_withValidData\_returnsSuccess teszt a POST /user/registerAdmin végpontot ellenőrzi, várva a 200-as státuszkódot és "success" választ érvényes admin adatokra.

Jó teszt mivel az adatok még nem léteztek az adatbázisban:



```
@Test
public void testRegisterAdmin_withValidData_returnsSuccess() {
    JSONObject requestBody = new JSONObject()
        .put("email", "admin@example.com")
        .put("firstName", "Admin")
        .put("lastName", "FOnck")
        .put("userName", "adminfo")
        .put("picture", "base64image")
        .put("password", "Admin123!@#");

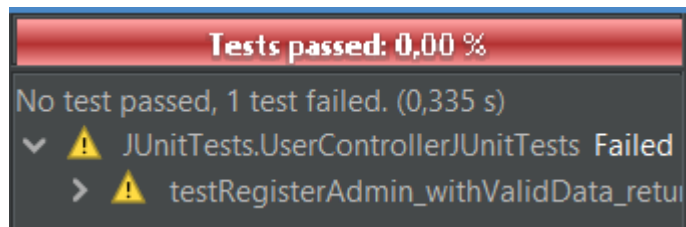
    Response response = client.target(BASE_URI)
        .path("registerAdmin")
        .request(MediaType.APPLICATION_JSON)
        .post(Entity.json(requestBody.toString()));

    assertEquals(200, response.getStatus(), "A státusznak 200-nak kell lennie");

    JSONObject responseBody = new JSONObject(response.readEntity(String.class));
    assertEquals("success", responseBody.getString("status"), "A státusz 'success' kell legyen");
    assertEquals(200, responseBody.getInt("statusCode"), "A státuszkódnak 200-nak kell lennie");

    response.close();
}
```

Rossz teszt mivel az adatbázisban már létezik ez a felhasználó:



```
@Test
public void testRegisterAdmin_withValidData_returnsSuccess() {
    JSONObject requestBody = new JSONObject()
        .put("email", "admin@example.com")
        .put("firstName", "Admin")
        .put("lastName", "FOnck")
        .put("userName", "adminfo")
        .put("picture", "base64image")
        .put("password", "Admin123!@#");

    Response response = client.target(BASE_URI)
        .path("registerAdmin")
        .request(MediaType.APPLICATION_JSON)
        .post(Entity.json(requestBody.toString()));

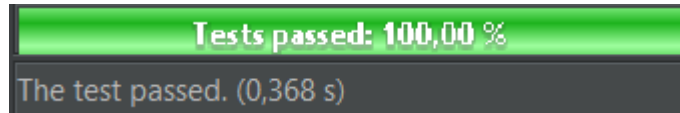
    assertEquals(200, response.getStatus(), "A státusznak 200-nak kell lennie");

    JSONObject responseBody = new JSONObject(response.readEntity(String.class));
    assertEquals("success", responseBody.getString("status"), "A státusz 'success' kell legyen");
    assertEquals(200, responseBody.getInt("statusCode"), "A státuszkódnak 200-nak kell lennie");

    response.close();
}
```

A `testGetAllUsers_withExistingUsers_returnsUserList` teszt a `GET /user/getAllUsers` végpontot ellenőrzi, várva a 200-as státuszkódot és a nem üres felhasználói listát tartalmazó választ.

Jó teszt:



```
@Test
public void testGetAllUsers_withExistingUsers_returnsUserList() {
    Response response = client.target(BASE_URI)
        .path("getAllUsers")
        .request(MediaType.APPLICATION_JSON)
        .get();

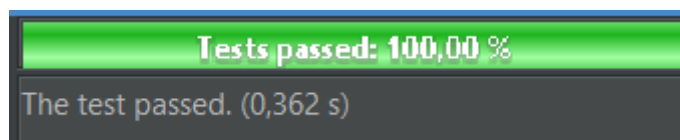
    assertEquals(200, response.getStatus(), "A státusznak 200-nak kell lennie");

    JSONObject responseBody = new JSONObject(response.readEntity(String.class));
    assertEquals(200, responseBody.getInt("statusCode"), "A státuszkódnak 200-nak kell lennie");
    assertTrue(responseBody.has("users"), "A válaszban szerepelnie kell a 'users' tömbnek");
    assertTrue(responseBody.getJSONArray("users").length() > 0, "A felhasználók listája nem üres");

    response.close();
}
```

A `testPasswordChangeByUserId_withValidData_returnsSuccess` teszt a `PUT /user/passwordChangeByUserId` végpontot ellenőrzi, várva a 200-as státuszkódot és "success" választ érvényes jelszóváltoztatási adatokra.

Jó teszt jó adatok esetén:



```
@Test
public void testPasswordChangeByUserId_withValidData_returnsSuccess() {
    // Jelszó módosítása
    JSONObject requestBody = new JSONObject()
        .put("userId", 2)
        .put("oldPassword", "password123")
        .put("newPassword", "New123!@#");

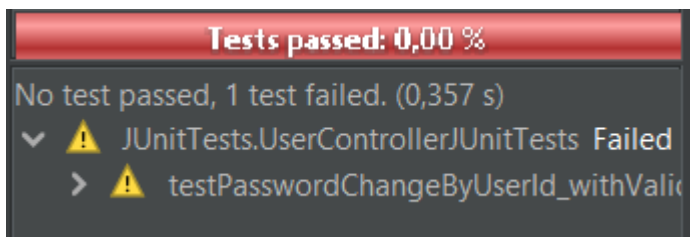
    Response response = client.target(BASE_URI)
        .path("passwordChangeByUserId")
        .request(MediaType.APPLICATION_JSON)
        .put(Entity.json(requestBody.toString()));

    assertEquals(200, response.getStatus(), "A státusznak 200-nak kell lennie");

    JSONObject responseBody = new JSONObject(response.readEntity(String.class));
    assertEquals("success", responseBody.getString("status"), "A státusz 'success' kell legyen");
    assertEquals(200, responseBody.getInt("statusCode"), "A státuszkódnak 200-nak kell lennie");

    response.close();
}
```

Rossz teszt hibás jelszó esetén:



```
@Test
public void testPasswordChangeByUserId_withValidData_returnsSuccess() {

    // Jelszó módosítása
    JSONObject requestBody = new JSONObject()
        .put("userId", 2)
        .put("oldPassword", "Rozserpassword123")
        .put("newPassword", "New123!@#");

    Response response = client.target(BASE_URI)
        .path("passwordChangeByUserId")
        .request(MediaType.APPLICATION_JSON)
        .put(Entity.json(requestBody.toString()));

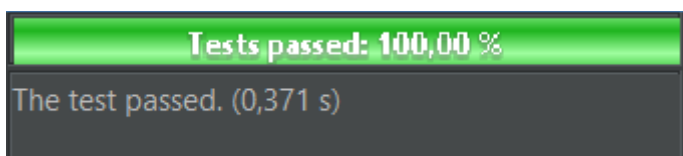
    assertEquals(200, response.getStatus(), "A státusznak 200-nak kell lennie");

    JSONObject responseBody = new JSONObject(response.readEntity(String.class));
    assertEquals("success", responseBody.getString("status"), "A státusz 'success' kell legyen");
    assertEquals(200, responseBody.getInt("statusCode"), "A státuszkódnak 200-nak kell lennie");

    response.close();
}
```

A testUsernameChangeByUserId\_withValidData\_returnsSuccess teszt a PUT /user/usernameChangeByUserId végpontot ellenőrzi, várva a 200-as státuszkódot, "success" választ és az új felhasználónevet érvényes adatokra.

Jó teszt jó id esetén:



```
@Test
public void testUsernameChangeByUserId_withValidData_returnsSuccess() {

    // Felhasználónév módosítása
    JSONObject requestBody = new JSONObject()
        .put("userId", 2)
        .put("newUsername", "ujtest");

    Response response = client.target(BASE_URI)
        .path("usernameChangeByUserId")
        .request(MediaType.APPLICATION_JSON)
        .put(Entity.json(requestBody.toString()));

    assertEquals(200, response.getStatus(), "A státusznak 200-nak kell lennie");

    JSONObject responseBody = new JSONObject(response.readEntity(String.class));
    assertEquals("success", responseBody.getString("status"), "A státusz 'success' kell legyen");
    assertEquals(200, responseBody.getInt("statusCode"), "A státuszkódnak 200-nak kell lennie");
    assertEquals("ujtest", responseBody.getJSONObject("result").getString("username"), "A felhasználónévnek meg kell változnie");

    response.close();
}
```

Rossz teszt rossz id esetén:

**Tests passed: 0,00 %**

No test passed, 1 test failed. (0,361 s)

▼

⚠

JUnitTests.UserControllerJUnitTests Failed

>

⚠

testUsernameChangeByUserId\_withValidData\_returnsSuccess()

```
@Test
public void testUsernameChangeByUserId_withValidData_returnsSuccess() {

    // Felhasználó módosítása
    JSONObject requestBody = new JSONObject()
        .put("userId", 1000)
        .put("newUsername", "ujteszt");

    Response response = client.target(BASE_URI)
        .path("usernameChangeByUserId")
        .request(MediaType.APPLICATION_JSON)
        .put(Entity.json(requestBody.toString()));

    assertEquals(200, response.getStatus(), "A státusznak 200-nak kell lennie");

    JSONObject responseBody = new JSONObject(response.readEntity(String.class));
    assertEquals("success", responseBody.getString("status"), "A státusz 'success' kell legyen");
    assertEquals(200, responseBody.getInt("statusCode"), "A státuszadatok 200-nak kell lennie");
    assertEquals("ujteszt", responseBody.getJSONObject("result").getString("username"), "A felhasználónévnek meg kell változnia");

    response.close();
}
```

A testDeleteUser\_withValidId\_returnsSuccess teszt a DELETE /user/deleteUser végpontot ellenőrzi, várva a 200-as státuszkódot és "success" választ egy érvényes felhasználó ID törlésére.

Jó teszt:

**Tests passed: 100,00 %**

The test passed. (0,371 s)

```
@Test
public void testDeleteUser_withValidId_returnsSuccess() {

    // Felhasználó törlése
    Response response = client.target(BASE_URI)
        .path("deleteUser")
        .queryParam("id", 4)
        .request(MediaType.APPLICATION_JSON)
        .delete();

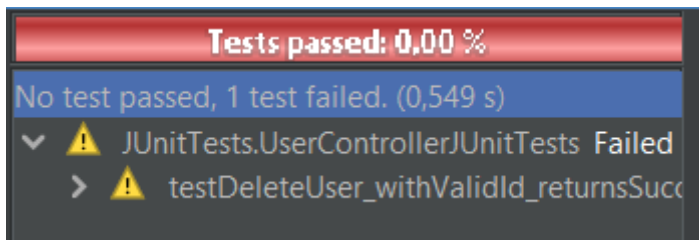
    assertEquals(200, response.getStatus(), "A státusznak 200-nak kell lennie");

    JSONObject responseBody = new JSONObject(response.readEntity(String.class));
    assertEquals("success", responseBody.getString("result"), "A válasz 'success' kell legyen");

    response.close();
}
```



Rossz teszt mivel a user már törölve lett:



```
@Test
public void testDeleteUser_withValidId_returnsSuccess() {

    // Felhasználó törlése
    Response response = client.target(BASE_URI)
        .path("deleteUser")
        .queryParam("id", 4)
        .request(MediaType.APPLICATION_JSON)
        .delete();

    assertEquals(200, response.getStatus(), "A státusznak 200-nak kell lennie");

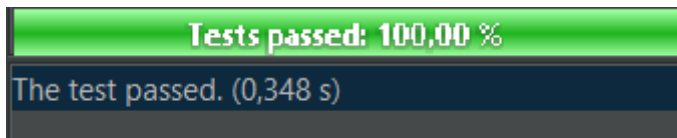
    JSONObject responseBody = new JSONObject(response.readEntity(String.class));
    assertEquals("success", responseBody.getString("result"), "A válasz 'success' kell legyen");

    response.close();
}
```

## SHELFS OSZTÁLY:

A `testAddShelfToStorage_withValidData_returnsSuccess` teszt a `POST /shelves/addShelfToStorage` végpontot ellenőrzi, várva a 201-es státuszkódot, "Shelf successfully added to storage" üzenetet és a helyes polcnevet érvényes adatokra.

Jó teszt:



```
@Test
public void testAddShelfToStorage_withValidData_returnsSuccess() {
    JSONObject requestBody = new JSONObject()
        .put("storageId", 1)
        .put("shelfName", "TestShelf 1")
        .put("locationIn", "Aisle 3");

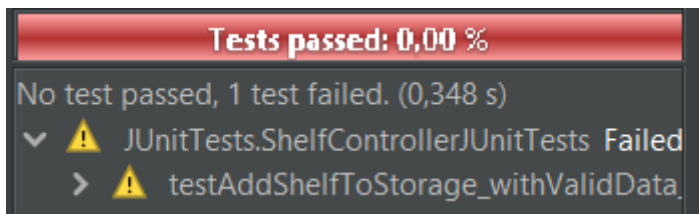
    Response response = client.target(BASE_URI)
        .path("addShelfToStorage")
        .request(MediaType.APPLICATION_JSON)
        .post(Entity.json(requestBody.toString()));

    assertEquals(201, response.getStatus(), "A státusznak 201-nek kell lennie");

    JSONObject responseBody = new JSONObject(response.readEntity(String.class));
    assertEquals(201, responseBody.getInt("statusCode"), "A státuszkódnak 201-nek kell lennie");
    assertEquals("Shelf successfully added to storage", responseBody.getString("message"), "A válaszüzenet helyes kell legyen");
    assertEquals("TestShelf 1", responseBody.getString("shelfName"), "A polc neve helyes kell legyen");

    response.close();
}
```

Rossz teszt rossz id esetén:



```
@Test
public void testAddShelfToStorage_withValidData_returnsSuccess() {
    JSONObject requestBody = new JSONObject()
        .put("storageId", 100)
        .put("shelfName", "TestShelf 1")
        .put("locationIn", "Aisle 3");

    Response response = client.target(BASE_URI)
        .path("addShelfToStorage")
        .request(MediaType.APPLICATION_JSON)
        .post(Entity.json(requestBody.toString()));

    assertEquals(201, response.getStatus(), "A státusznak 201-nek kell lennie");

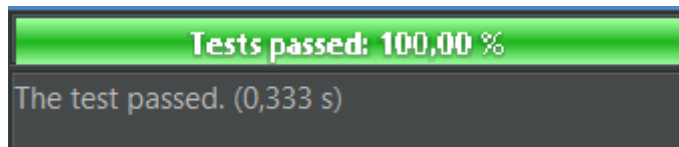
    JSONObject responseBody = new JSONObject(response.readEntity(String.class));
    assertEquals(201, responseBody.getInt("statusCode"), "A státuszkódnak 201-nek kell lennie");
    assertEquals("Shelf successfully added to storage", responseBody.getString("message"), "A válaszüzenet helyes kell legyen");
    assertEquals("TestShelf 1", responseBody.getString("shelfName"), "A polc neve helyes kell legyen");

    response.close();
}
```



A `testGetAllShelves_withExistingShelves_returnsShelfList` teszt a `GET /shelves/getAllShelves` végpontot ellenőrzi, várva a 200-as státuszkódot és a nem üres polclista választ.

Jó teszt:



```
@Test
public void testGetAllShelves_withExistingShelves_returnsShelfList() {
    Response response = client.target(BASE_URI)
        .path("getAllShelves")
        .request(MediaType.APPLICATION_JSON)
        .get();

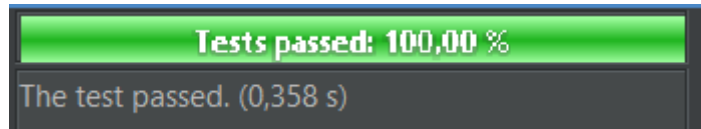
    assertEquals(200, response.getStatus(), "A státusznak 200-nak kell lennie");

    JSONObject responseBody = new JSONObject(response.readEntity(String.class));
    assertEquals(200, responseBody.getInt("statusCode"), "A státuszkódnak 200-nak kell lennie");
    assertTrue(responseBody.has("shelves"), "A válaszban szerepelnie kell a 'shelves' tombnak");
    assertTrue(responseBody.getJSONArray("shelves").length() > 0, "A polcok listája nem üres");

    response.close();
}
```

A `testGetShelfsById_withValidId_returnsShelf` teszt a `GET /shelfs/getShelfsById` végpontot ellenőrzi, várva a 200-as státuszkódot, az ID mezőt és a "Shelf a" nevet egy érvényes polc ID lekérdezésére.

Jó teszt:



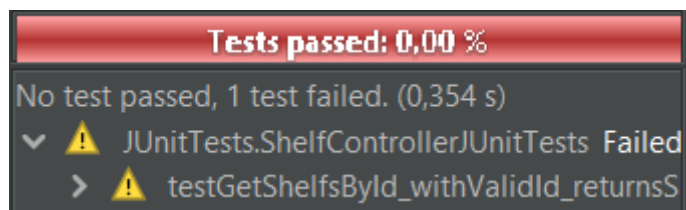
```
@Test
public void testGetShelfsById_withValidId_returnsShelf() {
    Response response = client.target(BASE_URI)
        .path("getShelfsById")
        .queryParams("id", 1)
        .request(MediaType.APPLICATION_JSON)
        .get();

    assertEquals(200, response.getStatus(), "A státusznak 200-nak kell lennie");

    JSONObject responseBody = new JSONObject(response.readEntity(String.class));
    assertTrue(responseBody.has("id"), "A válaszban szerepelnie kell az 'id' mezőnek");
    assertEquals("Shelf a", responseBody.getString("name"), "A polc nevének meg kell egyeznie");

    response.close();
}
```

Rossz teszt mivel nem létező id lett megadva:



```
@Test
public void testGetShelfsById_withValidId_returnsShelf() {
    Response response = client.target(BASE_URI)
        .path("getShelfsById")
        .queryParams("id", 100)
        .request(MediaType.APPLICATION_JSON)
        .get();

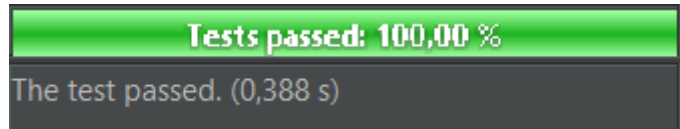
    assertEquals(200, response.getStatus(), "A státusznak 200-nak kell lennie");

    JSONObject responseBody = new JSONObject(response.readEntity(String.class));
    assertTrue(responseBody.has("id"), "A válaszban szerepelnie kell az 'id' mezőnek");
    assertEquals("Shelf a", responseBody.getString("name"), "A polc nevének meg kell egyeznie");

    response.close();
}
```

A `testGetShelvesByStorageId_withValidStorageId_returnsShelves` teszt a `GET /shelves/getShelvesByStorageId` végpontot ellenőrzi, várva a 200-as státuszkódot, nem üres polclistát és a "StorageShelf" nevet egy érvényes tároló ID-re.

Jó teszt:



```
@Test
public void testGetShelvesByStorageId_withValidStorageId_returnsShelves() {
    Response response = client.target(BASE_URI)
        .path("getShelvesByStorageId")
        .queryParam("storageId", 1)
        .request(MediaType.APPLICATION_JSON)
        .get();

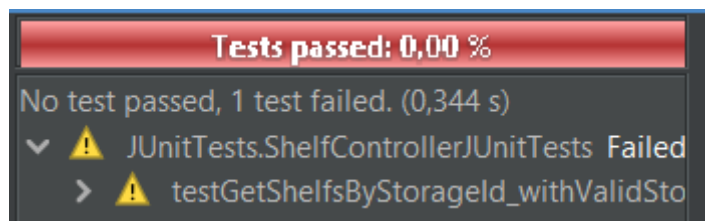
    assertEquals(200, response.getStatus(), "A státusznak 200-nak kell lennie");

    JSONObject responseBody = new JSONObject(response.readEntity(String.class));
    assertEquals(200, responseBody.getInt("statusCode"), "A státuszkódnak 200-nak kell lennie");
    assertTrue(responseBody.has("shelves"), "A válaszban szerepelnie kell a 'shelves' tömbnek");
    assertTrue(responseBody.getJSONArray("shelves").length() > 0, "A polcok listája nem üres");

    JSONArray shelves = responseBody.getJSONArray("shelves");
    JSONObject firstShelf = shelves.getJSONObject(0);
    assertEquals("Shelf B", firstShelf.getString("shelfName"), "A polc nevének meg kell egyeznie");

    response.close();
}
```

Rossz teszt rossz storage id esetén:



```
@Test
public void testGetShelvesByStorageId_withValidStorageId_returnsShelves() {
    Response response = client.target(BASE_URI)
        .path("getShelvesByStorageId")
        .queryParam("storageId", 100)
        .request(MediaType.APPLICATION_JSON)
        .get();

    assertEquals(200, response.getStatus(), "A státusznak 200-nak kell lennie");

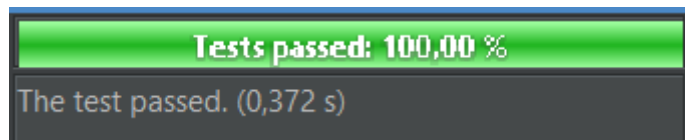
    JSONObject responseBody = new JSONObject(response.readEntity(String.class));
    assertEquals(200, responseBody.getInt("statusCode"), "A státuszkódnak 200-nak kell lennie");
    assertTrue(responseBody.has("shelves"), "A válaszban szerepelnie kell a 'shelves' tömbnek");
    assertTrue(responseBody.getJSONArray("shelves").length() > 0, "A polcok listája nem üres");

    JSONArray shelves = responseBody.getJSONArray("shelves");
    JSONObject firstShelf = shelves.getJSONObject(0);
    assertEquals("Shelf B", firstShelf.getString("shelfName"), "A polc nevének meg kell egyeznie");

    response.close();
}
```

A testDeleteShelfFromStorage\_withValidId\_returnsSuccess teszt a DELETE /shelves/deleteShelfFromStorage végpontot ellenőrzi, várva a 200-as státuszkódot és "success" választ egy érvényes polc ID törlésére.

Jó teszt:



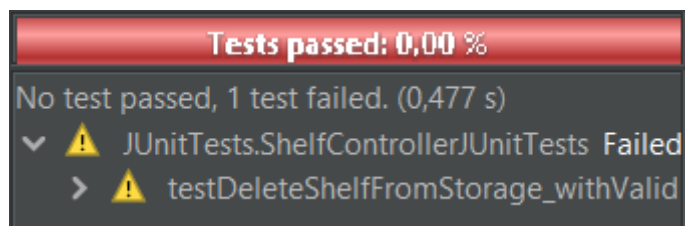
```
@Test
public void testDeleteShelfFromStorage_withValidId_returnsSuccess() {
    Response response = client.target(BASE_URI)
        .path("deleteShelfFromStorage")
        .queryParams("id", 12)
        .request(MediaType.APPLICATION_JSON)
        .delete();

    assertEquals(200, response.getStatus(), "A státusznak 200-nak kell lennie");

    JSONObject responseBody = new JSONObject(response.readEntity(String.class));
    assertEquals("success", responseBody.getString("result"), "A válasz 'success' kell legyen");

    response.close();
}
```

Rossz teszt nem létező ID esetén:



```
@Test
public void testDeleteShelfFromStorage_withValidId_returnsSuccess() {
    Response response = client.target(BASE_URI)
        .path("deleteShelfFromStorage")
        .queryParams("id", 12)
        .request(MediaType.APPLICATION_JSON)
        .delete();

    assertEquals(200, response.getStatus(), "A státusznak 200-nak kell lennie");

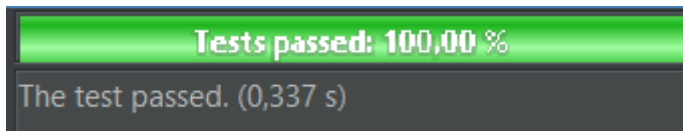
    JSONObject responseBody = new JSONObject(response.readEntity(String.class));
    assertEquals("success", responseBody.getString("result"), "A válasz 'success' kell legyen");

    response.close();
}
```

## STORAGE OSZTÁLY:

testAddStorage\_withValidData\_returnsSuccess: Ellenőrzi, hogy a POST /storage/addStorage végpont 201-es státuszkóddal és sikerüzenettel hozzáad egy új tárolót érvényes adatokkal.

Jó teszt:



```
@Test
public void testAddStorage_withValidData_returnsSuccess() {
    JSONObject requestBody = new JSONObject()
        .put("storageName", "TestStorage")
        .put("location", "Building A");

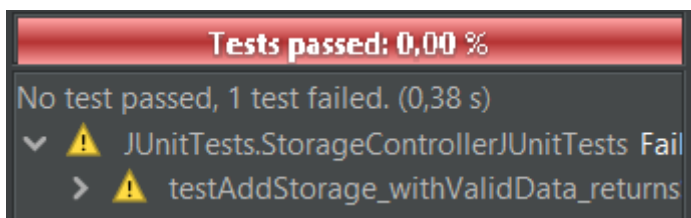
    Response response = client.target(BASE_URI)
        .path("addStorage")
        .request(MediaType.APPLICATION_JSON)
        .post(Entity.json(requestBody.toString()));

    assertEquals(201, response.getStatus(), "A státusznak 201-nek kell lennie");

    JSONObject responseBody = new JSONObject(response.readEntity(String.class));
    assertEquals(200, responseBody.getInt("statusCode"), "A státuszkódnak 200-nak kell lennie");
    assertEquals("Storage successfully added", responseBody.getString("message"), "A válaszüzenet helyes kell legyen");
    assertEquals("TestStorage", responseBody.getString("storageName"), "A tároló neve helyes kell legyen");

    response.close();
}
```

Rossz teszt:



```
@Test
public void testAddStorage_withValidData_returnsSuccess() {
    JSONObject requestBody = new JSONObject()
        .put("storageName", "")
        .put("location", "");

    Response response = client.target(BASE_URI)
        .path("addStorage")
        .request(MediaType.APPLICATION_JSON)
        .post(Entity.json(requestBody.toString()));

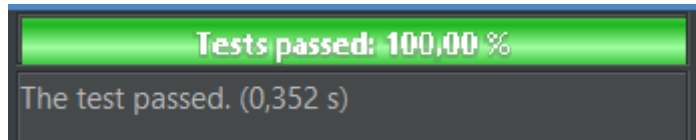
    assertEquals(201, response.getStatus(), "A státusznak 201-nek kell lennie");

    JSONObject responseBody = new JSONObject(response.readEntity(String.class));
    assertEquals(200, responseBody.getInt("statusCode"), "A státuszkódnak 200-nak kell lennie");
    assertEquals("Storage successfully added", responseBody.getString("message"), "A válaszüzenet helyes kell legyen");
    assertEquals("TestStorage", responseBody.getString("storageName"), "A tároló neve helyes kell legyen");

    response.close();
}
```

testGetAllStorages\_withExistingStorages\_returnsStorageList: Ellenőrzi, hogy a GET /storage/getAllStorages végpont 200-as státuszkóddal és nem üres tárolólistával tér vissza meglévő tárolók esetén.

Jó teszt:



```
@Test
public void testGetAllStorages_withExistingStorages_returnsStorageList() {
    Response response = client.target(BASE_URI)
        .path("getAllStorages")
        .request(MediaType.APPLICATION_JSON)
        .get();

    assertEquals(200, response.getStatus(), "A státusznak 200-nak kell lennie");

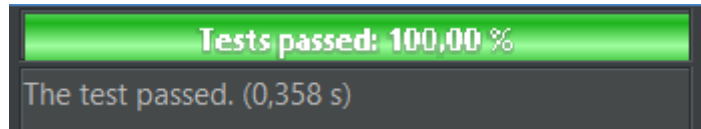
    JSONObject responseBody = new JSONObject(response.readEntity(String.class));
    assertEquals(200, responseBody.getInt("statusCode"), "A státuszkódnak 200-nak kell lennie");
    assertTrue(responseBody.has("Storages"), "A válaszban szerepelnie kell a 'Storages' tömbnek");
    assertTrue(responseBody.getJSONArray("Storages").length() > 0, "A tárolók listája nem üres");

    response.close();
}
```



testDeleteStorageById\_withValidId\_returnsSuccess: Ellenőrzi, hogy a DELETE /storage/deleteStorageById végpont 200-as státuszkóddal és sikerüzenettel töröl egy létező tárolót érvényes ID alapján.

Jó teszt:



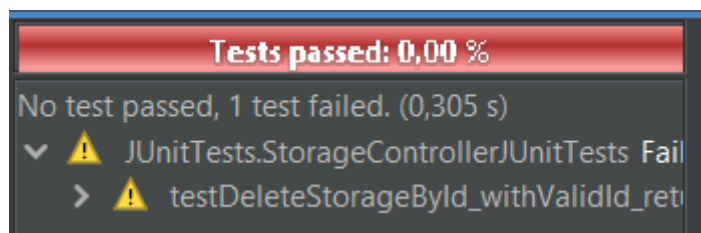
```
#Test
public void testDeleteStorageById_withValidId_returnsSuccess() {
    // Felrötelezünk, hogy létezik egy tároló ID-val !
    Response response = client.target(BASE_URI)
        .path("/deleteStorageById")
        .queryParam("id", 3)
        .request(MediaType.APPLICATION_JSON)
        .delete();

    assertEquals(200, response.getStatus(), "A státusznak 200-nak kell lennie");

    JSONObject responseBody = new JSONObject(response.readEntity(String.class));
    assertEquals(200, responseBody.getInt("statusCode"), "A státuszkódnak 200-nak kell lennie");
    assertEquals("Storage successfully deleted", responseBody.getString("message"), "A válaszüzenet helyes kell legyen");

    response.close();
}
```

Rossz teszt:



```
#Test
public void testDeleteStorageById_withValidId_returnsSuccess() {
    // Felrötelezünk, hogy létezik egy tároló ID-val !
    Response response = client.target(BASE_URI)
        .path("/deleteStorageById")
        .queryParam("id", 9999)
        .request(MediaType.APPLICATION_JSON)
        .delete();

    assertEquals(200, response.getStatus(), "A státusznak 200-nak kell lennie");

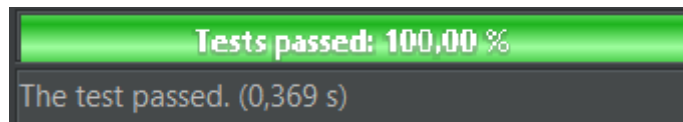
    JSONObject responseBody = new JSONObject(response.readEntity(String.class));
    assertEquals(200, responseBody.getInt("statusCode"), "A státuszkódnak 200-nak kell lennie");
    assertEquals("Storage successfully deleted", responseBody.getString("message"), "A válaszüzenet helyes kell legyen");

    response.close();
}
```

## PALLET OSZTÁLY:

testAddPalletToShelf\_withValidData\_returnsSuccess: Ellenőrzi, hogy a POST /pallet/addPalletToShelf végpont 201-es státuszkóddal és sikerüzenettel hozzáad egy új raklapot egy polchoz érvényes adatokkal.

Jó teszt:



```
@Test
public void testAddPalletToShelf_withValidData_returnsSuccess() {
    JSONObject requestBody = new JSONObject()
        .put("skuCode", "ITEM001")
        .put("shelfId", 1)
        .put("height", 80);

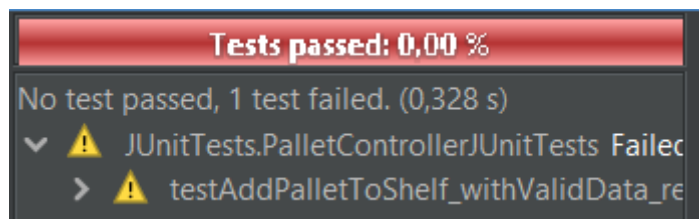
    Response response = client.target(BASE_URI)
        .path("/addPalletToShelf")
        .request(MediaType.APPLICATION_JSON)
        .post(Entity.json(requestBody.toString()));

    assertEquals(201, response.getStatus(), "A státusznak 201-nek kell lennie");

    JSONObject responseBody = new JSONObject(response.readEntity(String.class));
    assertEquals(201, responseBody.getInt("statusCode"), "A státuszkódnak 201-nek kell lennie");
    assertEquals("Pallet successfully added to shelf", responseBody.getString("message"), "A válaszüzenet helyes kell legyen");
    assertEquals("ITEM001", responseBody.getString("skuCode"), "A SKU kód helyes kell legyen");

    response.close();
}
```

Rossz teszt:



```
@Test
public void testAddPalletToShelf_withValidData_returnsSuccess() {
    JSONObject requestBody = new JSONObject()
        .put("skuCode", "FAKESKU")
        .put("shelfId", 1)
        .put("height", 80);

    Response response = client.target(BASE_URI)
        .path("/addPalletToShelf")
        .request(MediaType.APPLICATION_JSON)
        .post(Entity.json(requestBody.toString()));

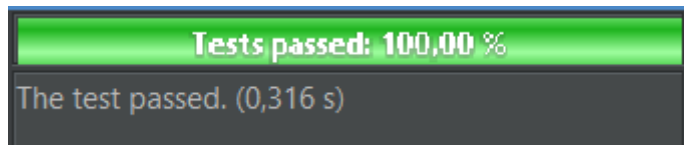
    assertEquals(201, response.getStatus(), "A státusznak 201-nek kell lennie");

    JSONObject responseBody = new JSONObject(response.readEntity(String.class));
    assertEquals(201, responseBody.getInt("statusCode"), "A státuszkódnak 201-nek kell lennie");
    assertEquals("Pallet successfully added to shelf", responseBody.getString("message"), "A válaszüzenet helyes kell legyen");
    assertEquals("FAKESKU", responseBody.getString("skuCode"), "A SKU kód helyes kell legyen");

    response.close();
}
```

testGetPalletsById\_withValidId\_returnsPallet: Ellenőrzi, hogy a GET /pallet/getPalletsById végpont 200-as státuszkóddal és a megfelelő raklap adataival tér vissza egy érvényes ID esetén.

Jó teszt:



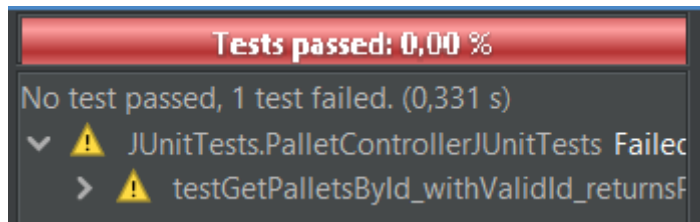
```
@Test
public void testGetPalletsById_withValidId_returnsPallet() {
    Response response = client.target(BASE_URI)
        .path("getPalletsById")
        .queryParam("id", 5)
        .request(MediaType.APPLICATION_JSON)
        .get();

    assertEquals(200, response.getStatus(), "A státusznak 200-nak kell lennie");

    JSONObject responseBody = new JSONObject(response.readEntity(String.class));
    assertTrue(responseBody.has("id"), "A válaszban szerepelnie kell az 'id' mezőnek");
    assertEquals(5, responseBody.getInt("id"), "A raklap ID-jának meg kell egyeznie");

    response.close();
}
```

Rossz teszt:



```
@Test
public void testGetPalletsById_withValidId_returnsPallet() {
    Response response = client.target(BASE_URI)
        .path("getPalletsById")
        .queryParam("id", 1)
        .request(MediaType.APPLICATION_JSON)
        .get();

    assertEquals(200, response.getStatus(), "A státusznak 200-nak kell lennie");

    JSONObject responseBody = new JSONObject(response.readEntity(String.class));
    assertTrue(responseBody.has("id"), "A válaszban szerepelnie kell az 'id' mezőnek");
    assertEquals(1, responseBody.getInt("id"), "A raklap ID-jának meg kell egyeznie");

    response.close();
}
```

testMovePalletBetweenShelves\_withValidData\_returnsSuccess: Ellenőrzi, hogy a POST /pallet/movePalletBetweenShelves végpont 200-as státuszkóddal és sikerüzenettel áthelyez egy raklapot két polc között érvényes adatokkal.

Jó teszt:



```
@Test
public void testMovePalletBetweenShelves_withValidData_returnsSuccess() {
    JSONObject requestBody = new JSONObject()
        .put("palletId", 10)
        .put("fromShelfId", 1)
        .put("toShelfId", 3)
        .put("newId", 40);

    Response response = client.target(BASE_URI)
        .path("movePalletBetweenShelves")
        .request(MediaType.APPLICATION_JSON)
        .post(Entity.json(requestBody.toString()));

    assertEquals(200, response.getStatus(), "A státusznak 200-nak kell lennie");

    JSONObject responseBody = new JSONObject(response.readEntity(String.class));
    assertEquals(200, responseBody.getInt("statusCode"), "A státuszkódnak 200-nak kell lennie");
    assertEquals("Pallet successfully moved between shelves", responseBody.getString("message"), "A válaszüzenet helyes kell legyen");
    assertEquals(10, responseBody.getInt("palletId"), "A raklap ID-jának meg kell egyeznie");

    response.close();
}
```

Rossz teszt:



```
@Test
public void testMovePalletBetweenShelves_withValidData_returnsSuccess() {
    JSONObject requestBody = new JSONObject()
        .put("palletId", 1)
        .put("fromShelfId", 1)
        .put("toShelfId", 100)
        .put("newId", 40);

    Response response = client.target(BASE_URI)
        .path("movePalletBetweenShelves")
        .request(MediaType.APPLICATION_JSON)
        .post(Entity.json(requestBody.toString()));

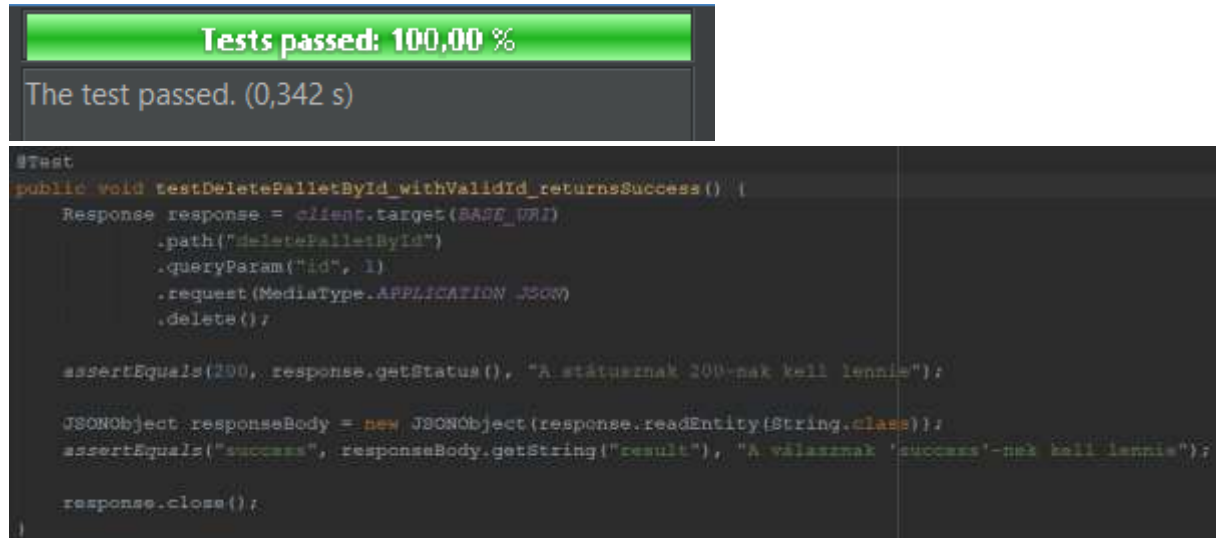
    assertEquals(200, response.getStatus(), "A státusznak 200-nak kell lennie");

    JSONObject responseBody = new JSONObject(response.readEntity(String.class));
    assertEquals(200, responseBody.getInt("statusCode"), "A státuszkódnak 200-nak kell lennie");
    assertEquals("Pallet successfully moved between shelves", responseBody.getString("message"), "A válaszüzenet helyes kell legyen");
    assertEquals(1, responseBody.getInt("palletId"), "A raklap ID-jának meg kell egyeznie");

    response.close();
}
```

testDeletePalletById\_withValidId\_returnsSuccess: Ellenőrzi, hogy a DELETE /pallet/deletePalletById végpont 200-as státuszkóddal és sikerüzenettel töröl egy létező raklapot érvényes ID alapján.

Jó teszt:



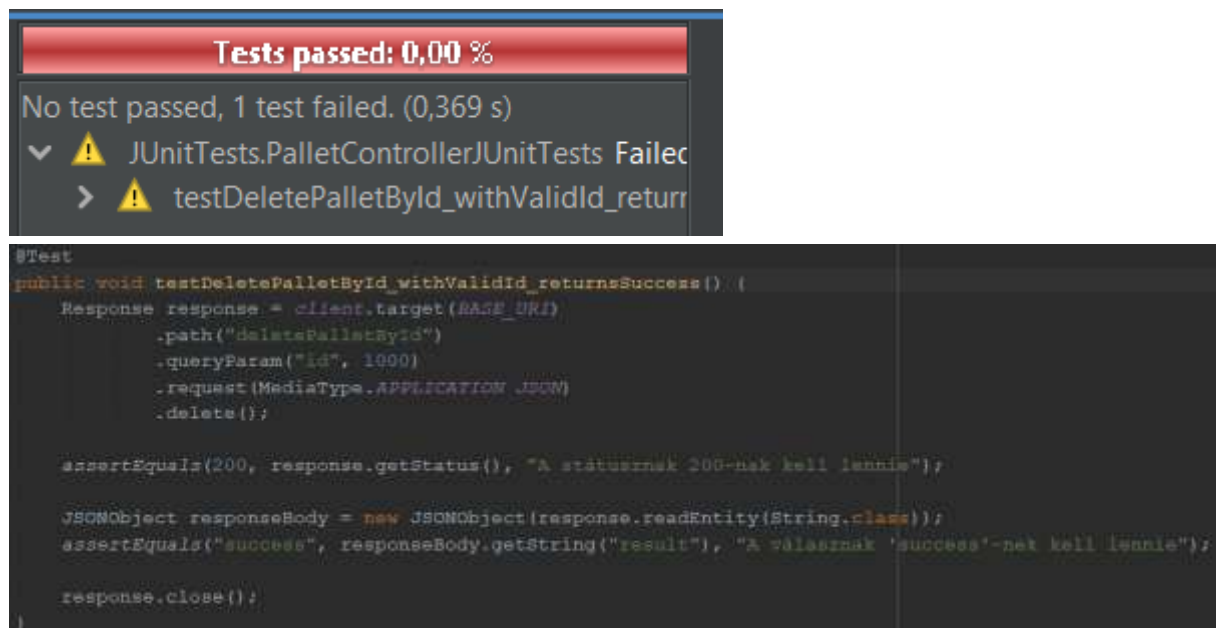
```
#Test
public void testDeletePalletById_withValidId_returnsSuccess() {
    Response response = client.target(BASE_URI)
        .path("deletePalletById")
        .queryParams("id", 1)
        .request(MediaType.APPLICATION_JSON)
        .delete();

    assertEquals(200, response.getStatus(), "A státusznak 200-nak kell lennie");

    JSONObject responseBody = new JSONObject(response.readEntity(String.class));
    assertEquals("success", responseBody.getString("result"), "A válasznak 'success'-nek kell lennie");

    response.close();
}
```

Rossz teszt:



```
#Test
public void testDeletePalletById_withValidId_returnsSuccess() {
    Response response = client.target(BASE_URI)
        .path("deletePalletById")
        .queryParams("id", 1000)
        .request(MediaType.APPLICATION_JSON)
        .delete();

    assertEquals(200, response.getStatus(), "A státusznak 200-nak kell lennie");

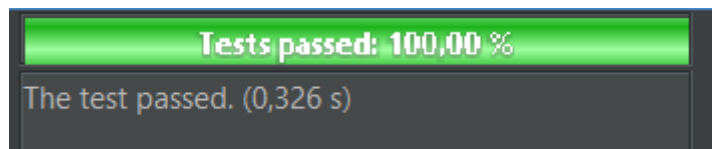
    JSONObject responseBody = new JSONObject(response.readEntity(String.class));
    assertEquals("success", responseBody.getString("result"), "A válasznak 'success'-nek kell lennie");

    response.close();
}
```

## ITEMS OSZTÁLY:

testAddItem\_withValidData\_returnsSuccess: Ellenőrzi, hogy a POST /items/addItem végpont 201-es státuszkóddal és sikerüzenettel hozzáad egy új terméket érvényes adatokkal.

Jó teszt:



```
@Test
public void testAddItem_withValidData_returnsSuccess() {
    JSONObject requestBody = new JSONObject()
        .put("sku", "TESTSKU123")
        .put("type", "wood")
        .put("name", "Test Item")
        .put("amount", 10)
        .put("price", 99.99)
        .put("weight", 5.5)
        .put("size", 1.0)
        .put("description", "Test description");

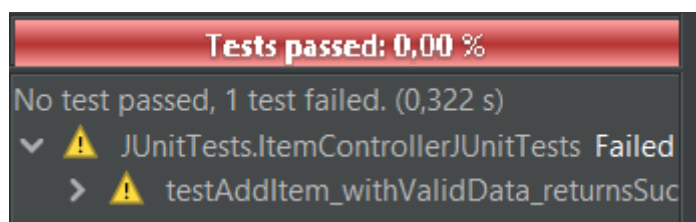
    Response response = client.target(BASE_URI)
        .path("addItem")
        .request(MediaType.APPLICATION_JSON)
        .post(Entity.json(requestBody.toString()));

    assertEquals(201, response.getStatus(), "A státusznak 201-nek kell lennie");

    JSONObject responseBody = new JSONObject(response.readEntity(String.class));
    assertEquals(201, responseBody.getInt("statusCode"), "A státuszkódnak 201-nek kell lennie");
    assertEquals("Item successfully added", responseBody.getString("message"), "A válaszüzenet helyes kell legyen");
    assertEquals("TESTSKU123", responseBody.getString("sku"), "A SKU kód helyes kell legyen");

    response.close();
}
```

Rossz teszt:



```
@Test
public void testAddItem_withValidData_returnsSuccess() {
    JSONObject requestBody = new JSONObject()
        .put("sku", "TESTSKU123")
        .put("type", "BAITTES")
        .put("name", "Test Item")
        .put("amount", 10)
        .put("price", 99.99)
        .put("weight", 5.5)
        .put("size", 1.0)
        .put("description", "Test description");

    Response response = client.target(BASE_URI)
        .path("addItem")
        .request(MediaType.APPLICATION_JSON)
        .post(Entity.json(requestBody.toString()));

    assertEquals(201, response.getStatus(), "A státusznak 201-nek kell lennie");

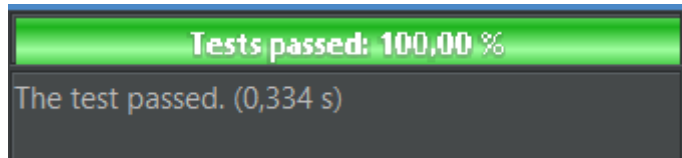
    JSONObject responseBody = new JSONObject(response.readEntity(String.class));
    assertEquals(201, responseBody.getInt("statusCode"), "A státuszkódnak 201-nek kell lennie");
    assertEquals("Item successfully added", responseBody.getString("message"), "A válaszüzenet helyes kell legyen");
    assertEquals("TESTSKU123", responseBody.getString("sku"), "A SKU kód helyes kell legyen");

    response.close();
}
```



testGetItemList\_withExistingItems\_returnsItemList: Ellenőrzi, hogy a GET /items/getItemList végpont 200-as státuszkóddal és nem üres terméklistával tér vissza meglévő termékek esetén.

Jó teszt:



```
@Test
public void testGetItemList_withExistingItems_returnsItemList() {
    Response response = client.target(BASE_URI)
        .path("getItemList")
        .request(MediaType.APPLICATION_JSON)
        .get();

    assertEquals(200, response.getStatus(), "A státusznak 200-nak kell lennie");

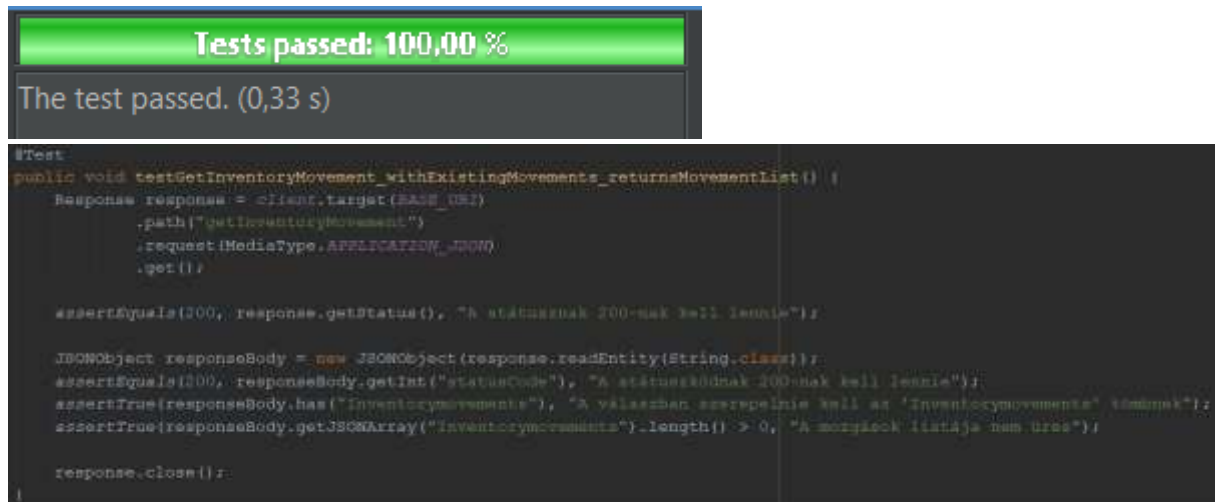
    JSONObject responseBody = new JSONObject(response.readEntity(String.class));
    assertEquals(200, responseBody.getInt("statusCode"), "A státuszkódnak 200-nak kell lennie");
    assertTrue(responseBody.has("items"), "A válaszban szerepelnie kell az 'items' tömbnek");
    assertTrue(responseBody.getJSONArray("items").length() > 0, "A termékek listája nem üres");

    response.close();
}
```

## INVENTORY MOVEMENT OSZTÁLY:

testGetInventoryMovement\_withExistingMovements\_returnsMovementList: Ellenőrzi, hogy a GET /inventorymovement/getInventoryMovement végpont 200-as státuszkóddal és nem üres mozgáslistával tér vissza meglévő mozgások esetén.

Jó teszt:



```
#Test
public void testGetInventoryMovement_withExistingMovements_returnsMovementList() {
    Response response = client.target(BASE_URI)
        .path("getInventoryMovement")
        .request(MediaType.APPLICATION_JSON)
        .get();

    assertEquals(200, response.getStatus(), "A státusznak 200-nak kell lennie");

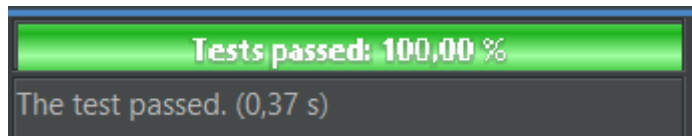
    JSONObject responseBody = new JSONObject(response.readEntity(String.class));
    assertEquals(200, responseBody.getInt("statusCode"), "A státuszkódnak 200-nak kell lennie");
    assertTrue(responseBody.has("inventorymovements"), "A válaszban szerepelnie kell az 'inventorymovements' tagnak");
    assertTrue(responseBody.getJSONArray("inventorymovements").length() > 0, "A mozgások listája nem üres");

    response.close();
}
```

## MOVEMENT REQUEST OSZTÁLY:

testGetMovementRequests\_withExistingRequests\_returnsRequestList: Ellenőrzi, hogy a GET /movementrequests/getMovementRequests végpont 200-as státuszkóddal és nem üres kéréslem listával tér vissza meglévő kérések esetén.

Jó teszt:



```
8Test
public void testGetMovementRequests_withExistingRequests_returnsRequestList() {
    Response response = client.target(BASE_URI)
        .path("getMovementRequests")
        .request(MediaType.APPLICATION_JSON)
        .get();

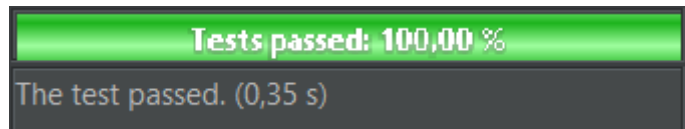
    assertEquals(200, response.getStatus(), "A státusznak 200-nak kell lennie");

    JSONObject responseBody = new JSONObject(response.readEntity(String.class));
    assertEquals(200, responseBody.getInt("statusCode"), "A státuszkódnak 200-nak kell lennie");
    assertTrue(responseBody.has("MovementRequests"), "A válasznak szerepelnie kell a 'MovementRequests' tömbnek");
    assertTrue(responseBody.getJSONArray("MovementRequests").length() > 0, "A kérések listája nem üres");

    response.close();
}
```

testCreateAddMovementRequest\_withValidData\_returnsSuccess: Ellenőrzi, hogy a POST /movementrequests/createAddMovementRequest végpont 200-as státuszkóddal és sikerüzenettel létrehoz egy új hozzáadási kérelmet érvényes adatokkal.

Jó teszt:



```
@Test
public void testCreateAddMovementRequest_withValidData_returnsSuccess() {
    JSONObject requestBody = new JSONObject()
        .put("adminId", 11)
        .put("galleryId", 10)
        .put("roomId", 4)
        .put("timestamp", "2021-04-20 12:00:00");

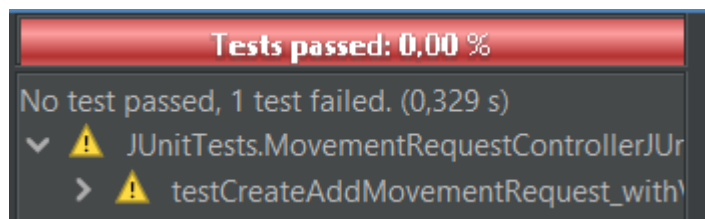
    Response response = client.target(BASE_URI)
        .path("createAddMovementRequest")
        .request(MediaType.APPLICATION_JSON)
        .post(Entity.json(requestBody.toString()));

    assertEquals(200, response.getStatus(), "A státusznak 200-nak kell lennie");

    JSONObject responseBody = new JSONObject(response.readEntity(String.class));
    assertEquals(200, responseBody.getInt("statusCode"), "A státuszkódnak 200-nak kell lennie");
    assertEquals("Add movement request created successfully", responseBody.getString("message"), "A válaszüzenet helyes kell legyen");

    response.close();
}
```

Rossz teszt:



```
@Test
public void testCreateAddMovementRequest_withValidData_returnsSuccess() {
    JSONObject requestBody = new JSONObject()
        .put("adminId", 11001)
        .put("galleryId", 1000)
        .put("roomId", 400)
        .put("timestamp", "2021-04-20 12:00:00");

    Response response = client.target(BASE_URI)
        .path("createAddMovementRequest")
        .request(MediaType.APPLICATION_JSON)
        .post(Entity.json(requestBody.toString()));

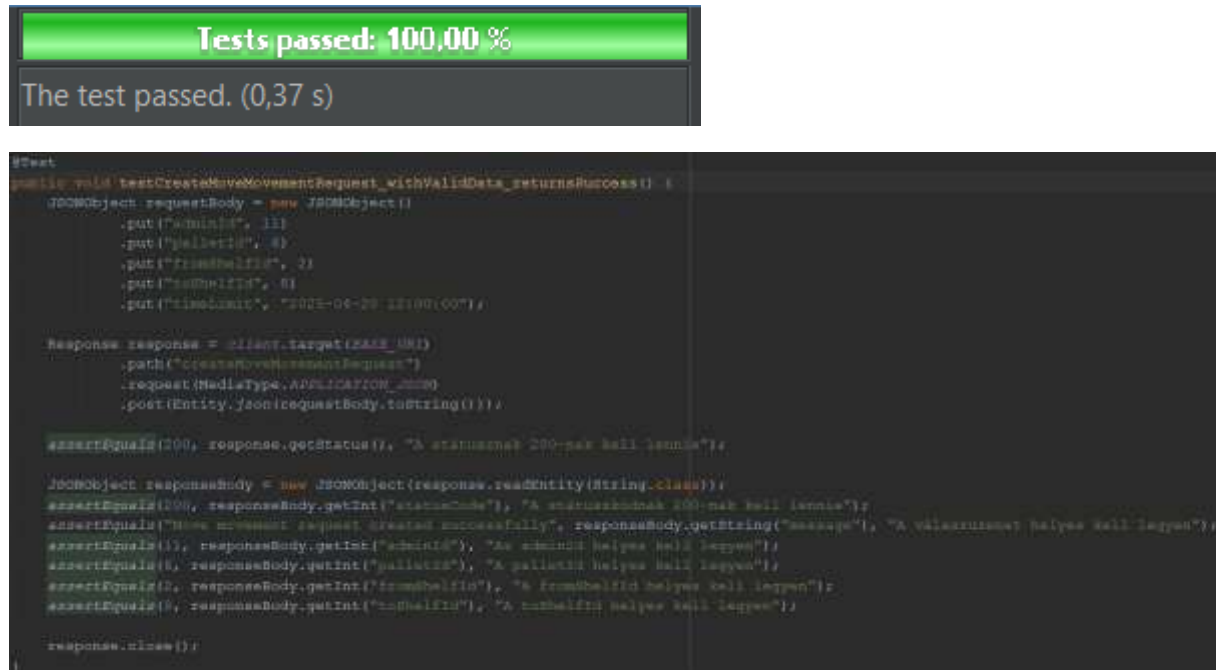
    assertEquals(200, response.getStatus(), "A státusznak 200-nak kell lennie");

    JSONObject responseBody = new JSONObject(response.readEntity(String.class));
    assertEquals(200, responseBody.getInt("statusCode"), "A státuszkódnak 200-nak kell lennie");
    assertEquals("Add movement request created successfully", responseBody.getString("message"), "A válaszüzenet helyes kell legyen");

    response.close();
}
```

testCreateMoveMovementRequest\_withValidData\_returnsSuccess: Ellenőrzi, hogy a POST /movementrequests/createMoveMovementRequest végpont 200-as státuszkóddal és sikerüzenettel létrehoz egy új mozgatási kérelmet érvényes adatokkal, továbbá ellenőrzi a visszaadott adatokat.

Jó teszt:



```
#Test
public void testCreateMoveMovementRequest_withValidData_returnsSuccess() {
    JSONObject requestBody = new JSONObject()
        .put("adminId", 111)
        .put("palletId", 4)
        .put("fromShelfId", 2)
        .put("toShelfId", 8)
        .put("timeLimit", "2023-06-20 12:00:00");

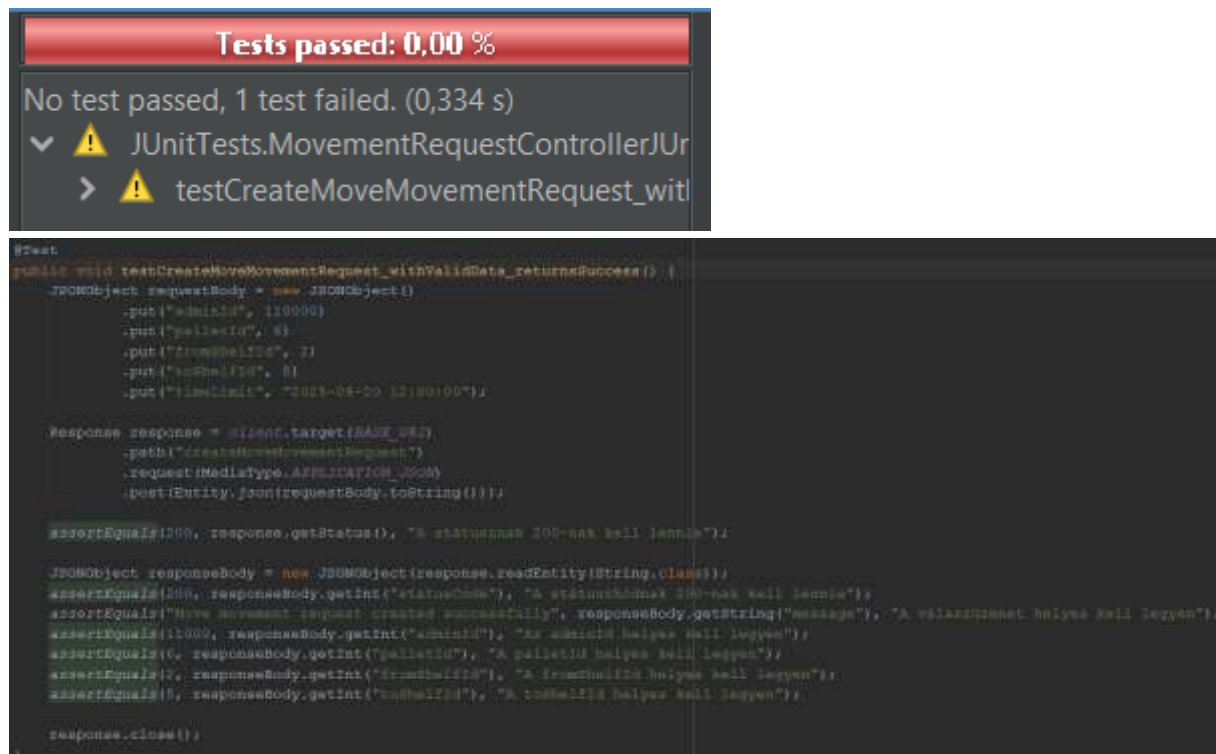
    Response response = client.target(BASE_URI)
        .path("createMoveMovementRequest")
        .request(MediaType.APPLICATION_JSON)
        .post(Entity.json(requestBody.toString()));

    assertEquals(200, response.getStatus(), "A státusznak 200-nak kell lennie");

    JSONObject responseBody = new JSONObject(response.readEntity(String.class));
    assertEquals(200, responseBody.getInt("statusCode"), "A státuszkódnak 200-nak kell lennie");
    assertEquals("Move movement request created successfully", responseBody.getString("message"), "A válaszüzenet helyes kell legyen");
    assertEquals(1, responseBody.getInt("adminId"), "Az adminId helyes kell legyen");
    assertEquals(4, responseBody.getInt("palletId"), "A palletId helyes kell legyen");
    assertEquals(2, responseBody.getInt("fromShelfId"), "A fromShelfId helyes kell legyen");
    assertEquals(8, responseBody.getInt("toShelfId"), "A toShelfId helyes kell legyen");

    response.close();
}
```

Rossz teszt:



```
#Test
public void testCreateMoveMovementRequest_withValidData_returnsSuccess() {
    JSONObject requestBody = new JSONObject()
        .put("adminId", 110000)
        .put("palletId", 4)
        .put("fromShelfId", 2)
        .put("toShelfId", 8)
        .put("timeLimit", "2023-06-20 12:00:00");

    Response response = client.target(BASE_URI)
        .path("createMoveMovementRequest")
        .request(MediaType.APPLICATION_JSON)
        .post(Entity.json(requestBody.toString()));


    assertEquals(200, response.getStatus(), "A státusznak 200-nak kell lennie");

    JSONObject responseBody = new JSONObject(response.readEntity(String.class));
    assertEquals(200, responseBody.getInt("statusCode"), "A státuszkódnak 200-nak kell lennie");
    assertEquals("Move movement request created successfully", responseBody.getString("message"), "A válaszüzenet helyes kell legyen");
    assertEquals(110000, responseBody.getInt("adminId"), "Az adminId helyes kell legyen");
    assertEquals(4, responseBody.getInt("palletId"), "A palletId helyes kell legyen");
    assertEquals(2, responseBody.getInt("fromShelfId"), "A fromShelfId helyes kell legyen");
    assertEquals(8, responseBody.getInt("toShelfId"), "A toShelfId helyes kell legyen");

    response.close();
}
```

testCreateRemoveMovementRequest\_withValidData\_returnsSuccess: Ellenőrzi, hogy a POST /movementrequests/createRemoveMovementRequest végpont 200-as státuszkóddal és sikerüzenettel létrehoz egy új eltávolítási kérelmet érvényes adatokkal.

Jó teszt:



```
First
public void testCreateRemoveMovementRequest_withValidData_returnsSuccess() {
    JSONObject requestBody = new JSONObject()
        .put("adminId", 11)
        .put("palletId", 5)
        .put("fromPalletId", 2)
        .put("timeLimit", "2015-04-20 12:00:00");

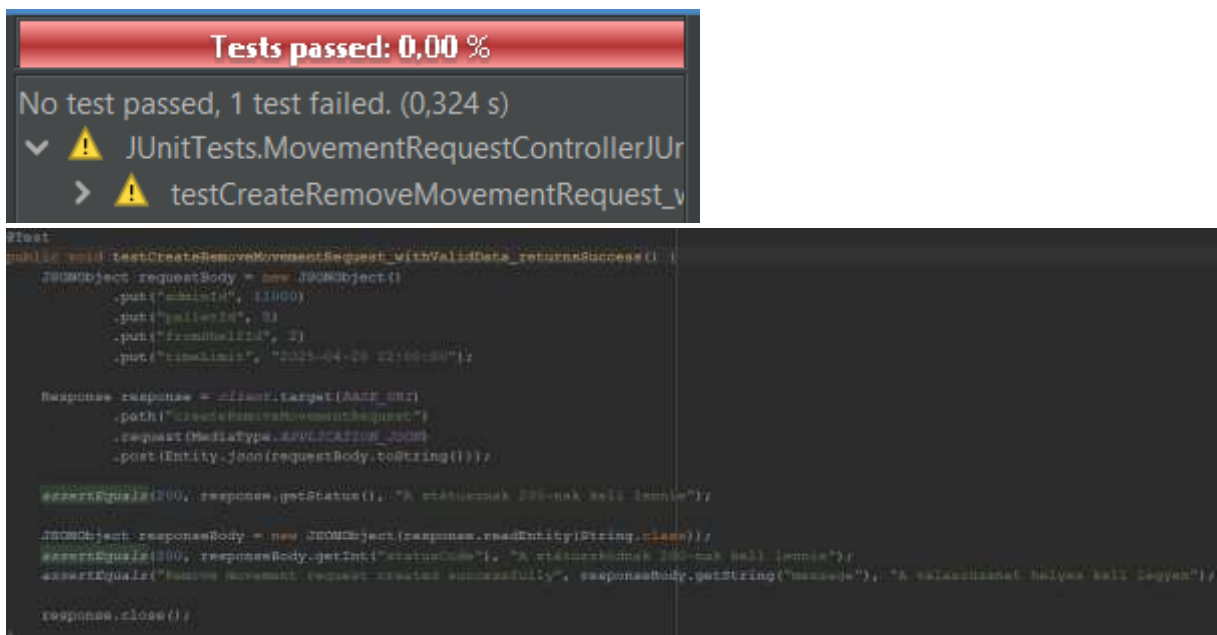
    Response response = client.target(BASE_URI)
        .path("createRemoveMovementRequest")
        .request(MediaType.APPLICATION_JSON)
        .post(Entity.json(requestBody.toString()));

    assertEquals(200, response.getStatus(), "A státusznak 200-nak kell lennie");

    JSONObject responseBody = new JSONObject(response.readEntity(String.class));
    assertEquals(200, responseBody.getInt("statusCode"), "A státuszkódnak 200-nak kell lennie");
    assertEquals("Remove movement request created successfully", responseBody.getString("message"), "A válaszüzenet helyes kell legyen");

    response.close();
}
```

Rossz teszt:



```
First
public void testCreateRemoveMovementRequest_withValidData_returnsSuccess() {
    JSONObject requestBody = new JSONObject()
        .put("adminId", 11000)
        .put("palletId", 5)
        .put("fromPalletId", 2)
        .put("timeLimit", "2015-04-20 12:00:00");

    Response response = client.target(BASE_URI)
        .path("createRemoveMovementRequest")
        .request(MediaType.APPLICATION_JSON)
        .post(Entity.json(requestBody.toString()));

    assertEquals(200, response.getStatus(), "A státusznak 200-nak kell lennie");

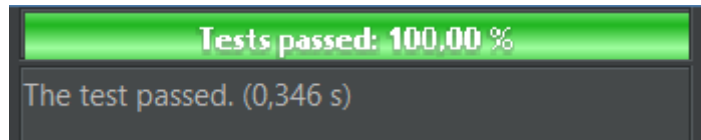
    JSONObject responseBody = new JSONObject(response.readEntity(String.class));
    assertEquals(200, responseBody.getInt("statusCode"), "A státuszkódnak 200-nak kell lennie");
    assertEquals("Remove movement request created successfully", responseBody.getString("message"), "A válaszüzenet helyes kell legyen");

    response.close();
}
```



testCompleteMovementRequest\_withValidData\_returnsSuccess: Ellenőrzi, hogy a POST /movementrequests/completeMovementRequest végpont 200-as státuszkóddal és sikerüzenettel teljesít egy létező kérelmet érvényes adatokkal, továbbá ellenőrzi a visszaadott azonosítókat.

Jó teszt:



```
@Test
public void testCompleteMovementRequest_withValidData_returnsSuccess() {
    JSONObject requestBody = new JSONObject()
        .put("movementRequestId", 6)
        .put("userId", 12);

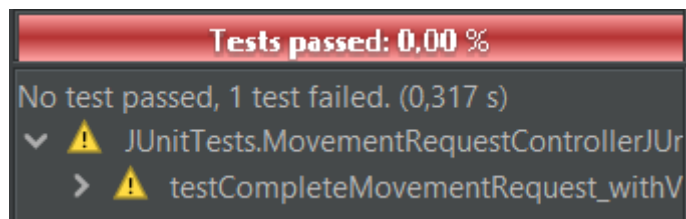
    Response response = client.target(BASE_URI)
        .path("completeMovementRequest")
        .request(MediaType.APPLICATION_JSON)
        .post(Entity.json(requestBody.toString()));

    assertEquals(200, response.getStatus(), "A státusznak 200-nak kell lennie");

    JSONObject responseBody = new JSONObject(response.readEntity(String.class));
    assertEquals(200, responseBody.getInt("statusCode"), "A státuszkódnak 200-nak kell lennie");
    assertTrue(responseBody.getString("message").contains("successfully completed"), "A válaszzenet helyes kell legyen");
    assertEquals(6, responseBody.getInt("movementRequestId"), "A movementRequestId helyes kell legyen");
    assertEquals(12, responseBody.getInt("userId"), "A userId helyes kell legyen");

    response.close();
}
```

Rossz teszt:



```
@Test
public void testCompleteMovementRequest_withValidData_returnsSuccess() {
    JSONObject requestBody = new JSONObject()
        .put("movementRequestId", 6)
        .put("userId", 12000);

    Response response = client.target(BASE_URI)
        .path("completeMovementRequest")
        .request(MediaType.APPLICATION_JSON)
        .post(Entity.json(requestBody.toString()));

    assertEquals(200, response.getStatus(), "A státusznak 200-nak kell lennie");

    JSONObject responseBody = new JSONObject(response.readEntity(String.class));
    assertEquals(200, responseBody.getInt("statusCode"), "A státuszkódnak 200-nak kell lennie");
    assertTrue(responseBody.getString("message").contains("successfully completed"), "A válaszzenet helyes kell legyen");
    assertEquals(6, responseBody.getInt("movementRequestId"), "A movementRequestId helyes kell legyen");
    assertEquals(12000, responseBody.getInt("userId"), "A userId helyes kell legyen");

    response.close();
}
```

## SELENIUM TESZTEK:

A testLogIn teszt sikeres bejelentkezést ellenőriz helyes felhasználónév és jelszó megadásával.

```
@Test
public void testLogIn() throws InterruptedException {
    Thread.sleep(2000);
    WebElement username = driver.findElement(By.className("login-input"));
    username.sendKeys("asd");

    WebElement password = driver.findElement(By.className("password-input"));
    password.sendKeys("asd");

    driver.findElement(By.className("signIn")).click();
    Thread.sleep(2000);
    String actualResult = driver.findElement(By.tagName("p")).getText();
    String expectedResult = "MAIN";
    Assert.assertEquals(actualResult, expectedResult);
}
```

**Tests passed: 100,00 %**

The test passed. (0,346 s)