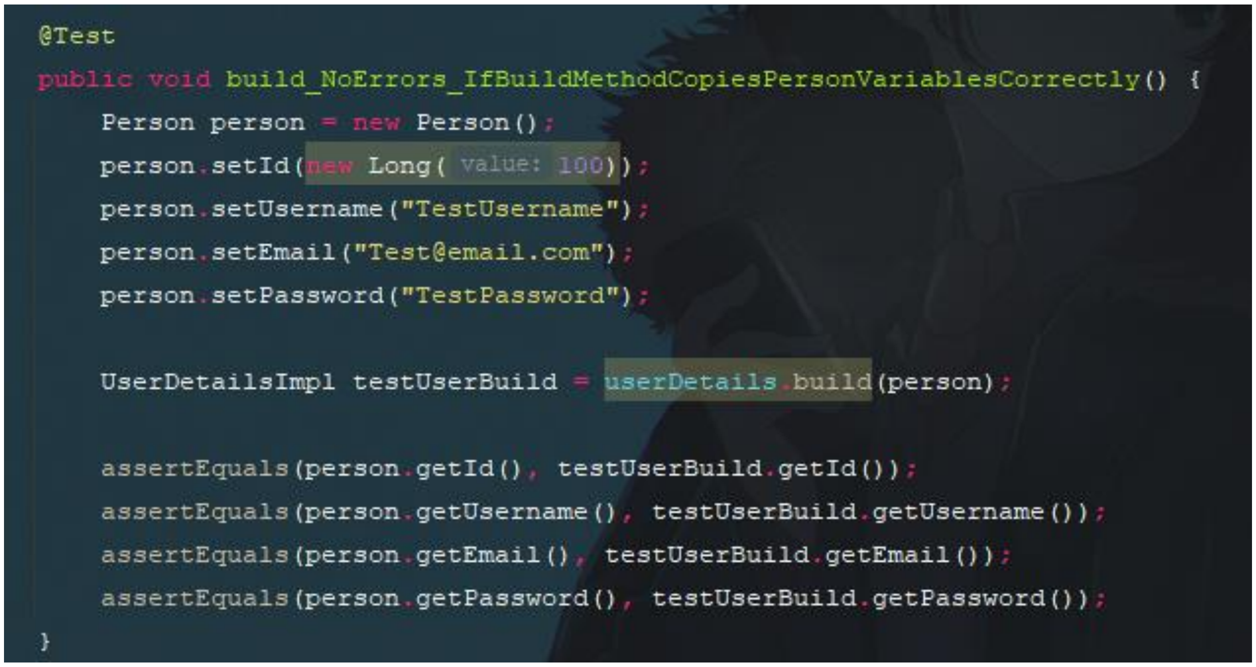
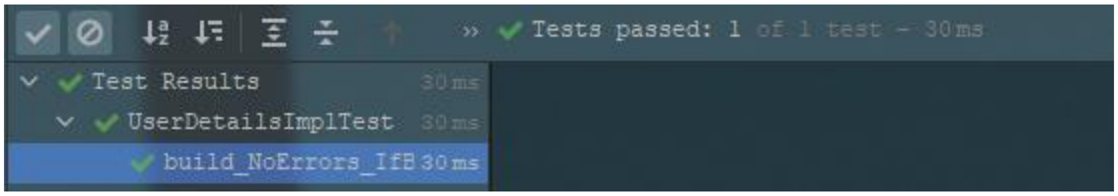
Person Test 01:



```java
@Test
public void build_NoErrors_IfBuildMethodCopiesPersonVariablesCorrectly() {
    Person person = new Person();
    person.setId(new Long( value: 100));
    person.setUsername("TestUsername");
    person.setEmail("Test@email.com");
    person.setPassword("TestPassword");

    UserDetailsImpl testUserBuild = userDetails.build(person);

    assertEquals(person.getId(), testUserBuild.getId());
    assertEquals(person.getUsername(), testUserBuild.getUsername());
    assertEquals(person.getEmail(), testUserBuild.getEmail());
    assertEquals(person.getPassword(), testUserBuild.getPassword());
}
```

Output:

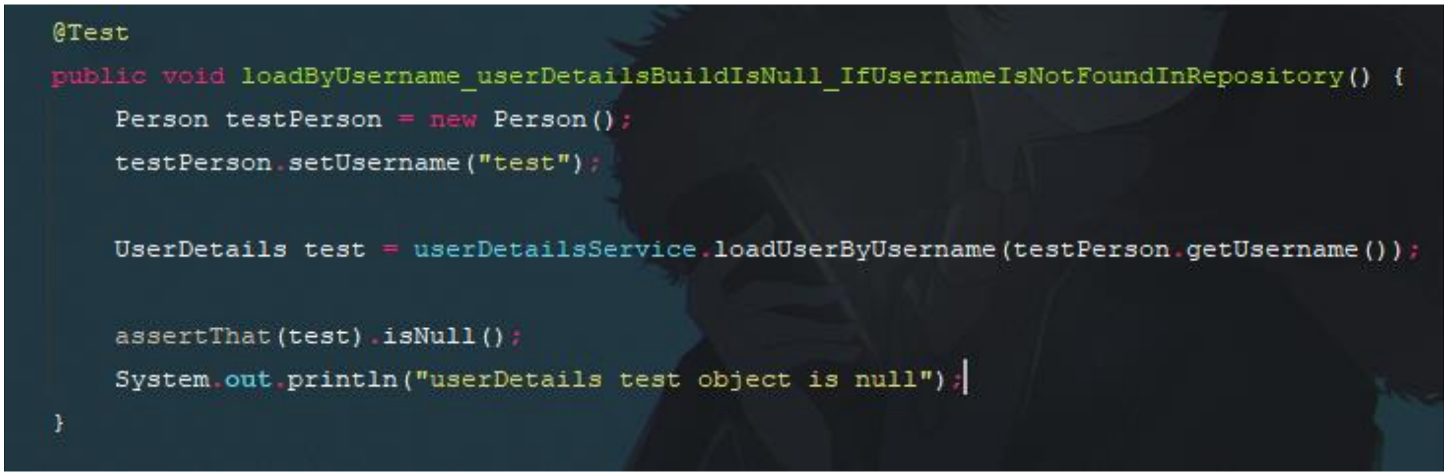


```
» ✓ Tests passed: 1 of 1 test – 30 ms
✓ Test Results                30 ms
  ✓ UserDetailsImplTest       30 ms
    ✓ build_NoErrors_IfB      30 ms
```

Comments:

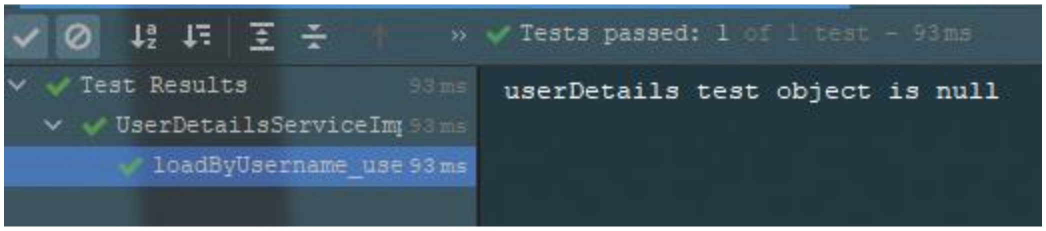
This test asserted that given a person with variables that are not null, if a UserDetailsImpl object was to be instantiated from it using the build() method, then the object should have variables equal to the person. The test passed all four asserts.

Person Test 02:



```java
@Test
public void loadByUsername_userDetailsBuildIsNull_IfUsernameIsNotFoundInRepository() {
    Person testPerson = new Person();
    testPerson.setUsername("test");

    UserDetails test = userDetailsService.loadUserByUsername(testPerson.getUsername());

    assertThat(test).isNull();
    System.out.println("userDetails test object is null");
}
```

Output:

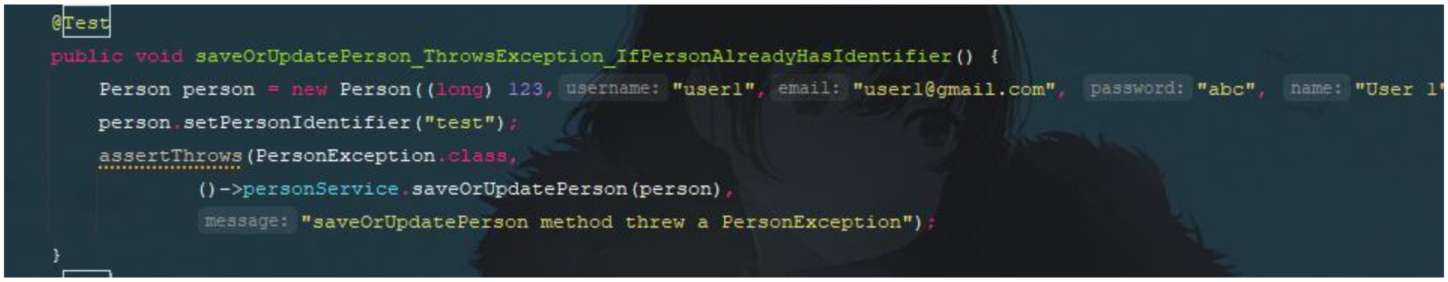


```
Tests passed: 1 of 1 test – 93 ms
Test Results 93 ms
  UserDetailsServiceImp 93 ms
    loadByUsername_use 93 ms

userDetails test object is null
```

Comments:

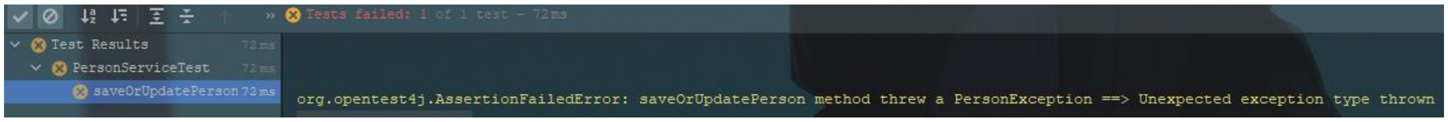
This test asserted that given a person with username “test", if it was not added into the repository, and a UserDetails object were to be instantiated via the loadUserByUsername() method, then the instantiated object would be null. The tests passes and the console prints that the object is null.

Person Test 03:



```
@Test
public void saveOrUpdatePerson_ThrowsException_IfPersonAlreadyHasIdentifier() {
    Person person = new Person((long) 123, username: "user1", email: "user1@gmail.com", password: "abc", name: "User 1
    person.setPersonIdentifier("test");
    assertThrows(PersonException.class,
            ()->personService.saveOrUpdatePerson(person),
            message: "saveOrUpdatePerson method threw a PersonException");
}
```

Output:

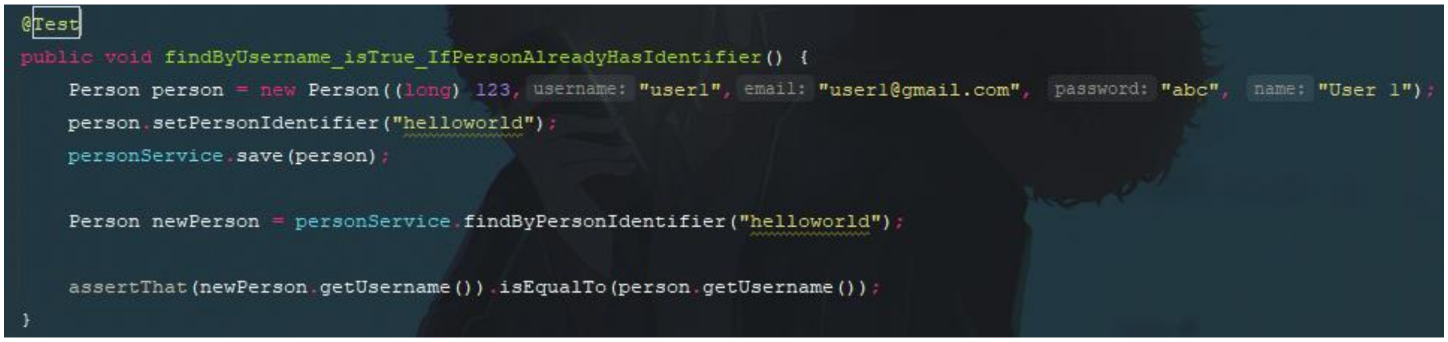


```
Tests failed: 1 of 1 test – 72 ms
Test Results 72 ms
  PersonServiceTest 72 ms
    saveOrUpdatePerson 72 ms

org.opentest4j.AssertionFailedError: saveOrUpdatePerson method threw a PersonException ==> Unexpected exception type thrown
```

Comments:

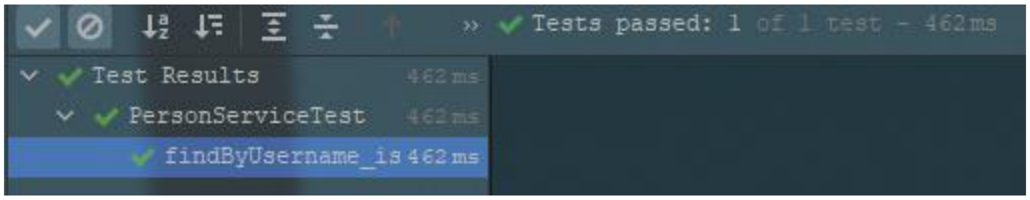
This test asserts that given a person with personIdentifier not null, when it is used as a parameter in the saveOrUpdatePerson() method, then the method would throw an exception. For this test, it fails as the expected exception was a PersonException, and the actual one thrown was a NullPointerException.

Person Test 04:



```
@Test
public void findByUsername_isTrue_IfPersonAlreadyHasIdentifier() {
    Person person = new Person((long) 123, username: "user1", email: "user1@gmail.com", password: "abc", name: "User 1");
    person.setPersonIdentifier("helloworld");
    personService.save(person);

    Person newPerson = personService.findByPersonIdentifier("helloworld");

    assertThat(newPerson.getUsername()).isEqualTo(person.getUsername());
}
```

Output:

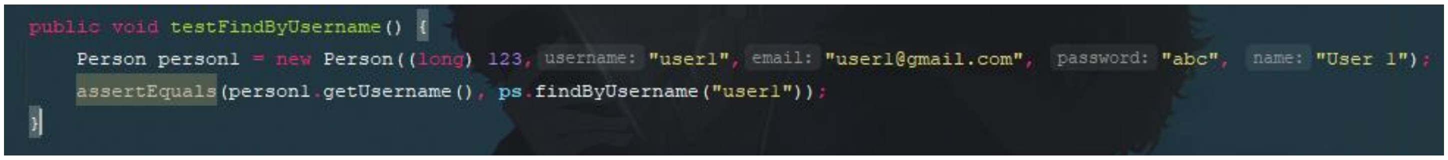


```
Tests passed: 1 of 1 test – 462 ms
Test Results 462 ms
  PersonServiceTest 462 ms
    findByUsername_is 462 ms
```

Comments:

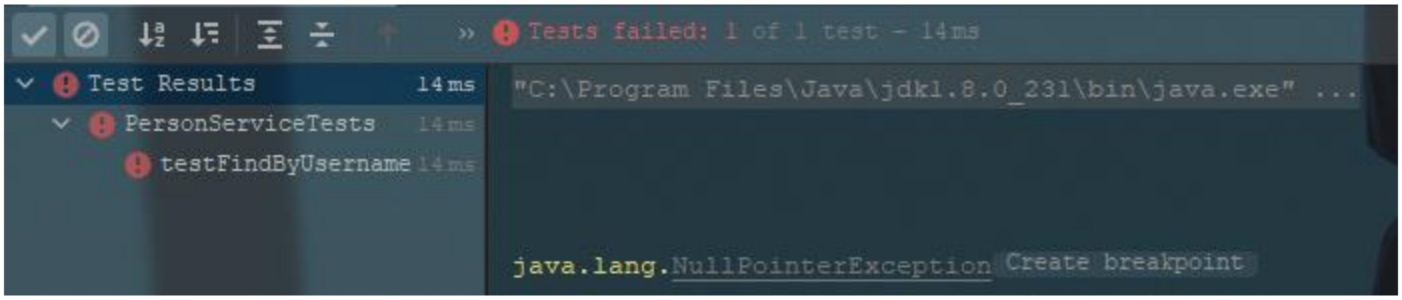
This test asserts that given a person with a person identifier and is saved into the system, when a new person is instantiated by finding the original person via the findByPersonIdentifier() method, then the new person will have an equal username to the original person.

Person Test 05:



```java
public void testFindByUsername() {
    Person person1 = new Person((long) 123, username: "user1", email: "user1@gmail.com", password: "abc", name: "User 1");
    assertEquals(person1.getUsername(), ps.findByUsername("user1"));
}
```

Output:

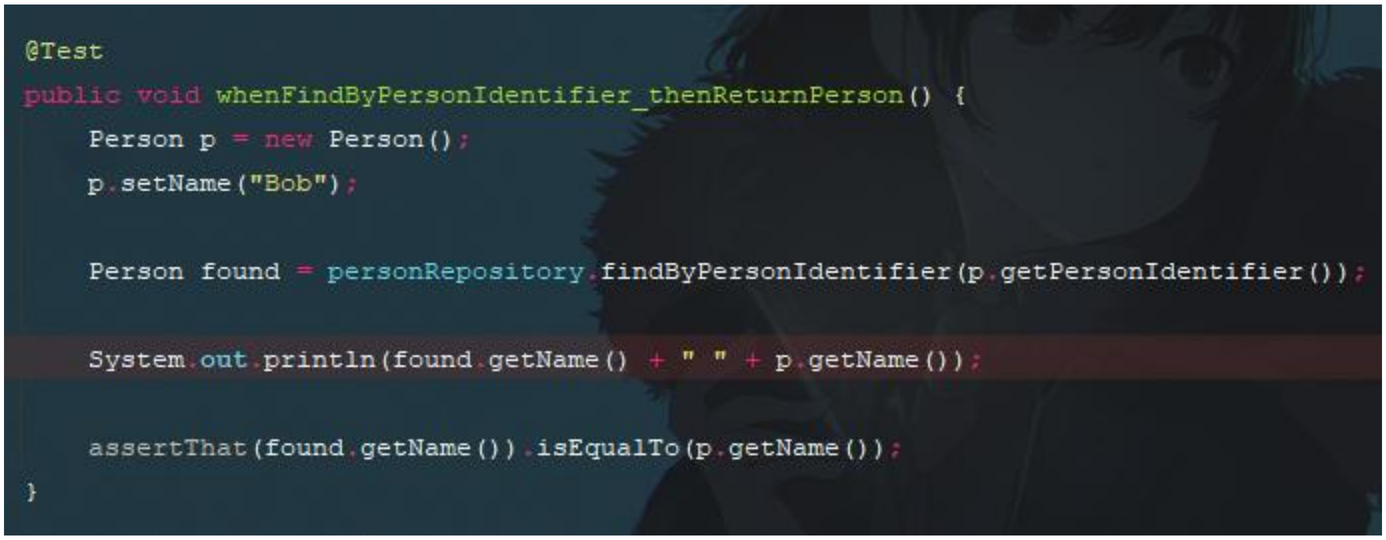


```
Tests failed: 1 of 1 test – 14 ms
Test Results 14 ms
  PersonServiceTests 14 ms
    testFindByUsername 14 ms

"C:\Program Files\Java\jdk1.8.0_231\bin\java.exe" ...

java.lang.NullPointerException
```

Comments:

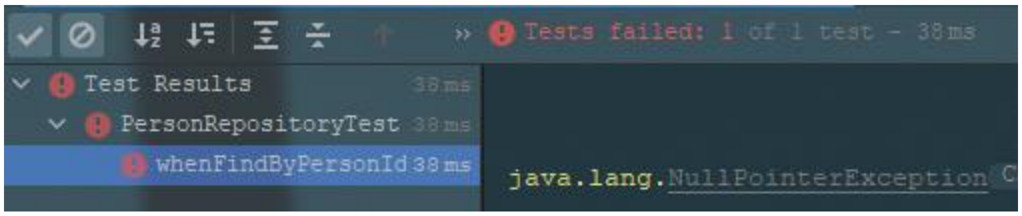
This test asserted that given a person, and the findByUsername() method is called using the person's username, then the person's username should be equal to the output of that method. The method failed due to a NullPointerException.

Person Test 06:



```java
@Test
public void whenFindByPersonIdentifier_thenReturnPerson() {
    Person p = new Person();
    p.setName("Bob");

    Person found = personRepository.findByPersonIdentifier(p.getPersonIdentifier());

    System.out.println(found.getName() + " " + p.getName());

    assertThat(found.getName()).isEqualTo(p.getName());
}
```

Output:



```
Tests failed: 1 of 1 test – 38 ms
Test Results 38 ms
  PersonRepositoryTest 38 ms
    whenFindByPersonId 38 ms
java.lang.NullPointerException
```

Comments:

This test asserted that given a person with name “Bob", when a new person is instantiated via the findByPersonIdentifier() method with the original person’s personal identifier, then the two persons’ name should be “Bob". The test failed due to a NullPointerException.