

E-commerce store with Spring boot



Outline

- Project structure
- Testing
 - Unit tests
 - Integration tests

Endpoints

Method	Endpoint	description
POST	/api/auth/login	User can login to system
POST	/api/auth/signup	New user can register to system
POST	/api/auth/logout	User can logout from sysetm
GET	/api/user/profile	A logged in user can see its info
GET	/api/user/orders	A logged in user can see the list of his orders
GET	/order/{id}	A logged in user can see details of an order(products, ...)
POST	/api/user/profile/edit	A logged in User can post its new profile information
GET	/api/user/address	A logged in user can get a list of its submitted addresses
POST	/api/user/address/add	A logged in user can submit a new address
PUT	/api/user/address/{id}	A logged in user can edit a previously submitted address
GET	/api/products	Any user can see the list of available products
GET	/api/product/{id}	Any user can see the details of a product

Technologies

- Spring boot
- Spring security
- Postgresql

```
<dependency>  
  <groupId>org.springframework.boot</groupId>  
  <artifactId>spring-boot-starter-web</artifactId>  
</dependency>  
  
<dependency>  
  <groupId>org.springframework.boot</groupId>  
  <artifactId>spring-boot-starter-test</artifactId>  
  <scope>test</scope>  
</dependency>  
  
<dependency>  
  <groupId>org.postgresql</groupId>  
  <artifactId>postgresql</artifactId>  
</dependency>
```

Models

- User(id, firstName, lastName, phoneNumber, email, password)
- Address(id, User, city, streetAddress, postalCode)
- Product(id, name, price, category)
- Order(id, User)

Controllers

- UserAuthController(/**api/auth**/*)
- UserAccountController(/**api/user**/*)
- UserAddressController(/**api/user/address**/*)
- ProductController(/**api/products**/*)

• UserAuthController •

```
@RestController
@RequestMapping("/api/auth")
public class UserAuthController {

    @Autowired
    private AuthenticationManager authenticationManager;

    @Autowired
    private UserService userService;

    @PostMapping("/login")
    public ResponseEntity<String> authenticateUser(
        @Valid @NonNull @RequestBody UserLoginDto loginDto)
    {
        Authentication auth = authenticationManager.authenticate(
            new UsernamePasswordAuthenticationToken(
                loginDto.getEmail(), loginDto.getPassword())
        );

        SecurityContextHolder.getContext().setAuthentication(auth);
        return new ResponseEntity<>("user is now logged in", HttpStatus.OK);
    }

    @PostMapping("/signup")
    public ResponseEntity<String> signupUserAccount(
        @Valid @NonNull @RequestBody UserSignupDto signupDto)
    {
        try {
            userService.save(signupDto);
            return new ResponseEntity<>(HttpStatus.OK);
        } catch (DataIntegrityViolationException e) {
            return new ResponseEntity<>("email already exists", HttpStatus.FORBIDDEN);
        }
    }
}
```


Unit test dependencies

- Junit
- Mockito
- MockMVC
- Spring boot starter test

```
<dependency>
  <groupId>io.rest-assured</groupId>
  <artifactId>spring-mock-mvc</artifactId>
  <scope>test</scope>
</dependency>

<dependency>
  <groupId>junit</groupId>
  <artifactId>junit</artifactId>
  <version>4.13.2</version>
  <scope>test</scope>
</dependency>

<dependency>
  <groupId>org.mockito</groupId>
  <artifactId>mockito-core</artifactId>
  <version>4.4.0</version>
  <scope>test</scope>
</dependency>
```


• Unit test structure •

```

@RunWith(SpringRunner.class)
@WebMvcTest({Controller.class})
public class ControllerTest {
    @Autowired
    private MockMvc mockMvc;

    @MockBean
    Service service;

    @Test
    public void test1() {
    }

    @Test
    @WithMockUser(username="fatemehkarimi@gamil.com", password="admin") {}
}

```

Example 1

```
@Test
public void successUserLogin() throws Exception {
    Mockito.when(userService.loadUserByUsername("fatemehkarimi@gmail.com")).thenReturn(
        new org.springframework.security.core.userdetails.User(
            user1.getEmail(), user1.getPassword(), new HashSet<GrantedAuthority>() )
    );
    Object loginData = new Object() {
        public final String email = "fatemehkarimi@gmail.com";
        public final String password = "admin";
    };

    String json = objectMapper.writeValueAsString(loginData);
    mockMvc.perform(MockMvcRequestBuilders
        .post("/api/auth/login")
        .content(json)
        .contentType(MediaType.APPLICATION_JSON))
        .andExpect(status().isOk())
        .andExpect(content().string("user is now logged in"));
}
```

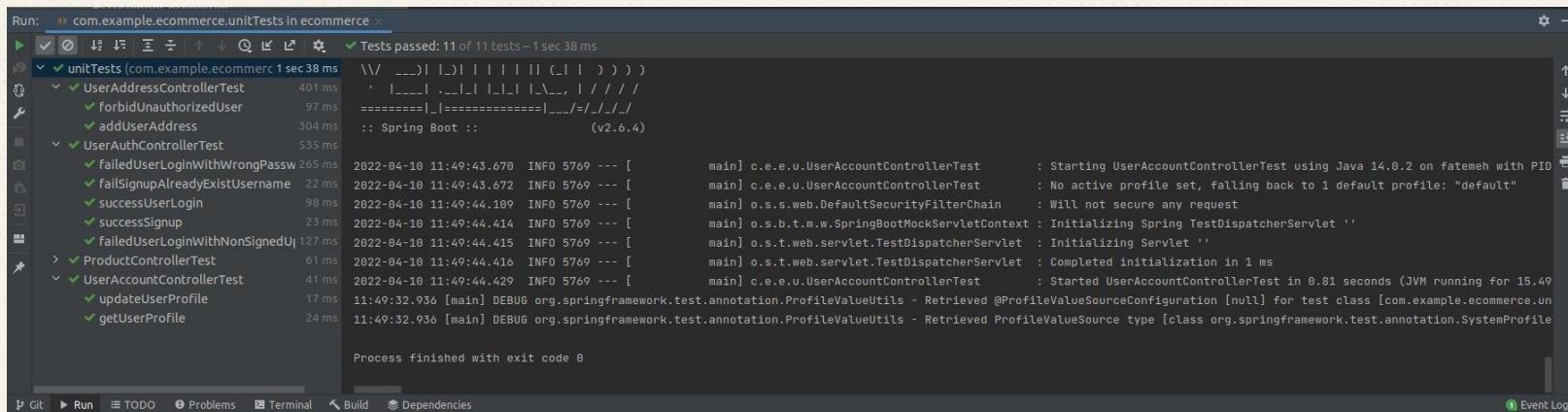
Example 2

```
User user = new User(
    UUID.fromString("aae82711-1546-4118-aaa0-de58f2d3c18b"),
    "fatemeh",
    "karimi",
    "09123456789",
    "fatemehkarimi@gmail.com",
    "$2a$10$EdJnVTWFWG4j9VorOu07W.7fkn2AB/P.Od4TUbFO4x.vqraR80Tq2");

@Test
@WithMockUser(username="fatemehkarimi@gmail.com", password="admin")
public void getUserProfile() throws Exception {
    Mockito.when(userDao.findByEmail(user.getEmail())).thenReturn(user);

    mockMvc.perform(MockMvcRequestBuilders
        .get("/api/user/profile"))
        .andExpect(status().isOk())
        .andExpect(MockMvcResultMatchers.jsonPath("$.email").value(user.getEmail()));
}
```

Running tests



The screenshot shows an IDE window titled "Run: com.example.ecommerce.unitTests in ecommerce". The top status bar indicates "Tests passed: 11 of 11 tests - 1 sec 38 ms". The left sidebar displays a tree view of the test results:

- unitTests (com.example.ecommerce) 1 sec 38 ms
 - UserAddressControllerTest 401 ms
 - ✓ forbidUnauthorizedUser 97 ms
 - ✓ addUserAddress 304 ms
 - UserAuthControllerTest 535 ms
 - ✓ failedUserLoginWithWrongPassword 265 ms
 - ✓ failSignUpAlreadyExistUsername 22 ms
 - ✓ successUserLogin 98 ms
 - ✓ successSignUp 23 ms
 - ✓ failedUserLoginWithNonSignedUp 127 ms
 - > ProductControllerTest 61 ms
 - UserAccountControllerTest 41 ms
 - ✓ updateUserProfile 17 ms
 - ✓ getUserProfile 24 ms

The right pane shows the output of the tests, including log messages and stack traces. The output is as follows:

```

2022-04-10 11:49:43.670 INFO 5769 --- [main] c.e.e.u.UserAccountControllerTest : Starting UserAccountControllerTest using Java 14.0.2 on fatemeh with PID
2022-04-10 11:49:43.672 INFO 5769 --- [main] c.e.e.u.UserAccountControllerTest : No active profile set, falling back to 1 default profile: "default"
2022-04-10 11:49:44.109 INFO 5769 --- [main] o.s.s.web.DefaultSecurityFilterChain : Will not secure any request
2022-04-10 11:49:44.414 INFO 5769 --- [main] o.s.b.t.m.w.SpringBootMockServletContext : Initializing Spring TestDispatcherServlet ''
2022-04-10 11:49:44.415 INFO 5769 --- [main] o.s.t.web.servlet.TestDispatcherServlet : Initializing Servlet ''
2022-04-10 11:49:44.416 INFO 5769 --- [main] o.s.t.web.servlet.TestDispatcherServlet : Completed initialization in 1 ms
2022-04-10 11:49:44.429 INFO 5769 --- [main] c.e.e.u.UserAccountControllerTest : Started UserAccountControllerTest in 0.81 seconds (JVM running for 15.49
11:49:32.936 [main] DEBUG org.springframework.test.annotation.ProfileValueUtils - Retrieved @ProfileValueSourceConfiguration [null] for test class [com.example.ecommerce.un
11:49:32.936 [main] DEBUG org.springframework.test.annotation.ProfileValueUtils - Retrieved ProfileValueSource type [class org.springframework.test.annotation.SystemProfile

Process finished with exit code 0

```

Integration tests dependencies

- Junit
- Rest-assured
- Cucumber
- Cucumber reporting

```
<dependency>
  <groupId>io.cucumber</groupId>
  <artifactId>cucumber-junit</artifactId>
  <version>7.2.3</version>
  <scope>test</scope>
</dependency>

<dependency>
  <groupId>io.rest-assured</groupId>
  <artifactId>rest-assured</artifactId>
  <scope>test</scope>
</dependency>

<dependency>
  <groupId>io.rest-assured</groupId>
  <artifactId>spring-mock-mvc</artifactId>
  <scope>test</scope>
</dependency>

<dependency>
  <groupId>junit</groupId>
  <artifactId>junit</artifactId>
  <version>4.13.2</version>
  <scope>test</scope>
</dependency>
```

Configuration

```
@RunWith(Cucumber.class)
@cucumberContextConfiguration
@cucumberOptions(
    plugin={"pretty", "json:target/cucumber-report.json", "html:target/cucumber-report"},
    glue={"src/test/java/com/example/ecommerce/Cucumber/glue"},
    features="src/test/resources")
@SpringBootTest(webEnvironment=SpringBootTest.WebEnvironment.RANDOM_PORT)
public class CucumberIntegrationTest {

}
```


Gherkin test Scenarios

Scenario Outline: User can edit his address

Given i am a logged in user with email = "<email>" and password = "<password>"

When i send a request to URL "/api/user/address/<id>" to update the city of address with id = "<id>" to "<city>"

Then i can see that the city name of that address has been updated at URL "/api/user/address"

Examples:

email	password	id	city	
fatemehkarimi.1998@yahoo.com	admin	10	Shiraz	

Given

```
@Before
public void setUp() {
    RestAssured.baseURI = BASE_URI;
    RestAssured.port = port;
}

@Given("i am a logged in user with email = {string} and password = {string}")
public void i_am_a_logged_in_user_with_email_and_password(
    String user_email, String user_password) throws Exception
{
    this.email = user_email;
    this.password = user_password;

    Object loginData = new Object() {
        public String email = "fatemehkarimi.1998@yahoo.com";
        public String password = "admin";
    };

    String json = objectMapper.writeValueAsString(loginData);
    this.cookie = given().contentType(ContentType.JSON)
        .body(json).post("/api/auth/login")
        .headers().getValue("Set-Cookie");
    this.cookie = this.cookie.split(";")[0];
}
```

When

```
@When("i send a request to URL {string} to update the city of address with id = {string} to {string}")
public void i_send_a_request_to_url_to_update_the_city_of_address_with_id_to(
    String endpoint, String addressId, String updated_cityName) throws Exception
{
    this.updated_cityName = updated_cityName;
    this.addressId = addressId;

    Object object = new Object() {
        public String city = updated_cityName;
    };

    String json = objectMapper.writeValueAsString(object);
    given().header("Cookie", this.cookie)
        .contentType(ContentType.JSON)
        .body(json)
        .put(endpoint)
        .then().statusCode(200);
}
```

Then

```
@Then("i can see that the city name of that address has been updated at URL {string}")
public void i_can_see_that_the_city_name_of_that_address_has_been_updated_at_url(String endpoint) throws Exception {
    List<Map<String, Object>> result =
        given().header("Cookie", this.cookie)
            .get(endpoint).then()
            .extract().body().as(new TypeRef<List<Map<String, Object>>>() {});

    for(Map<String, Object> address : result) {
        if(address.get("id").equals(this.addressId)) {
            assertThat(address.get("city")).isEqualTo(this.updated_cityName);
            break;
        }
    }
}
```

Results

Run: Feature: Product

✓ Done: Scenarios 4 of 4 (23 sec 276 ms)

✓ Test Results 3 sec 133 ms

```
2022-04-10 11:52:21.608 INFO 6138 --- [main] .b.t.c.SpringBootTestContextBootstrapper : Found @SpringBootConfiguration com.example.ecommerce.EcommerceApplication
2022-04-10 11:52:21.610 INFO 6138 --- [main] .b.t.c.SpringBootTestContextBootstrapper : Loaded default TestExecutionListener class names from location [META-INF
2022-04-10 11:52:21.610 INFO 6138 --- [main] .b.t.c.SpringBootTestContextBootstrapper : Using TestExecutionListeners: [org.springframework.test.context.web.Serv

4 Scenarios (4 passed)
8 Steps (8 passed)
0m19.373s

View your Cucumber Report at:
https://reports.cucumber.io/reports/32086b20-c856-43a5-8a87-4bd3a515b836

This report was published in collection "ecommerce"

2022-04-10 11:52:24.100 INFO 6138 --- [ionShutdownHook] j.LocalContainerEntityManagerFactoryBean : Closing JPA EntityManagerFactory for persistence unit 'default'
2022-04-10 11:52:24.130 INFO 6138 --- [ionShutdownHook] com.zaxxer.hikari.HikariDataSource : HikariPool-1 - Shutdown initiated...
2022-04-10 11:52:24.163 INFO 6138 --- [ionShutdownHook] com.zaxxer.hikari.HikariDataSource : HikariPool-1 - Shutdown completed.

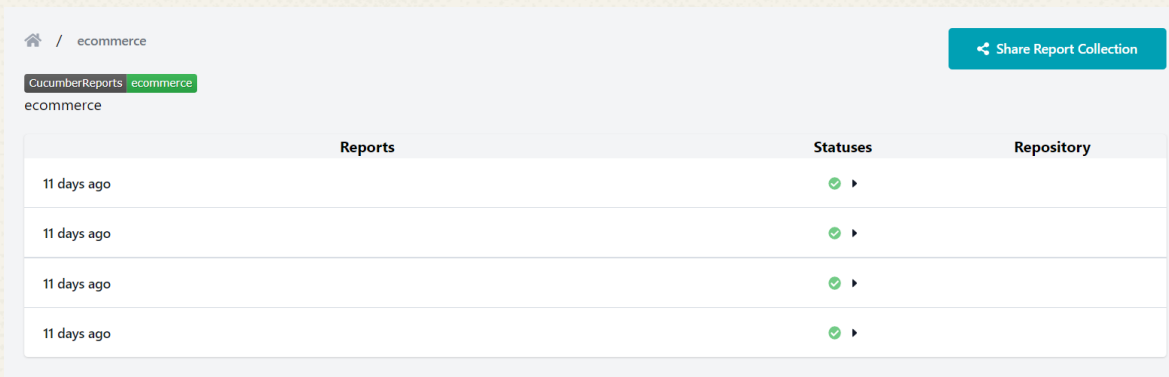
Process finished with exit code 0
```

Tests passed: 4 (a minute ago)

84:77 LF UTF-8 2 spaces master

Reports

- You can create a collection on cucumber website and keep your [reports](#).







The screenshot displays the Cucumber Reports web interface. At the top, there is a breadcrumb navigation showing a home icon followed by '/ ecommerce'. To the right of this is a teal button labeled 'Share Report Collection'. Below the breadcrumb, there is a header bar with 'CucumberReports' in a dark box and 'ecommerce' in a green box. The main content area features a table with three columns: 'Reports', 'Statuses', and 'Repository'. The table contains four rows, each representing a report generated '11 days ago'. Each row shows a green checkmark in the 'Statuses' column and a right-pointing arrow. The 'Repository' column is currently empty.


Reports	Statuses	Repository
11 days ago	✓ ▶	
11 days ago	✓ ▶	
11 days ago	✓ ▶	
11 days ago	✓ ▶	


Reports

14 PASSED

100% passed 14 executed	11 days ago Last Run	10.15 seconds Duration
 Linux	 OpenJDK 64-Bit Server VM 14.0.2+12-Ubuntu-120.04	 cucumber-jvm 7.2.3

>  file:///home/fatemeh/University/advanced%20software%20engineering/ecommerce/src/test/resources/Product.feature

>  file:///home/fatemeh/University/advanced%20software%20engineering/ecommerce/src/test/resources/UserAccount.feature

>  file:///home/fatemeh/University/advanced%20software%20engineering/ecommerce/src/test/resources/UserLogin.feature



✓  file:///home/fatemeh/University/advanced%20software%20engineering/ecommerce/src/test/resources/Product.feature

Report Edit

Feature: any website visitor can see products list and product details
Scenario: User can view product list

✓  When i send a request to URL `"/api/products"` to fetch products

✓  Then i must get a list of products

Scenario Outline: User can see the details of a product

✓  When i send a request to URL `"/api/product/<id>"` to see the details of product `"<id>"`

✓  Then i get an object with id equals to `"<id>"`

Examples:

✓ 	id
✓ 	3
✓ 	5
✓ 	15

>  file:///home/fatemeh/University/advanced%20software%20engineering/ecommerce/src/test/resources/UserAccount.feature

>  file:///home/fatemeh/University/advanced%20software%20engineering/ecommerce/src/test/resources/UserLogin.feature