



دانشگاه صنعتی شریف  
دانشکده‌ی مهندسی کامپیوتر

عنوان:

## آزمایش شماره‌ی ۲ - آشنایی با BDD

اعضای تیم

محمد طه جهانی نژاد - ۹۸۱۰۱۳۶۳

ایمان علیپور - ۹۸۱۰۲۰۲۴

نام درس

آزمایشگاه مهندسی نرم افزار

Course Name

**Software Engineering Lab**

نیمسال دوم ۱۴۰۲-۱۴۰۱

نام استاد درس

دکتر سید حسن میریان حسین آبادی

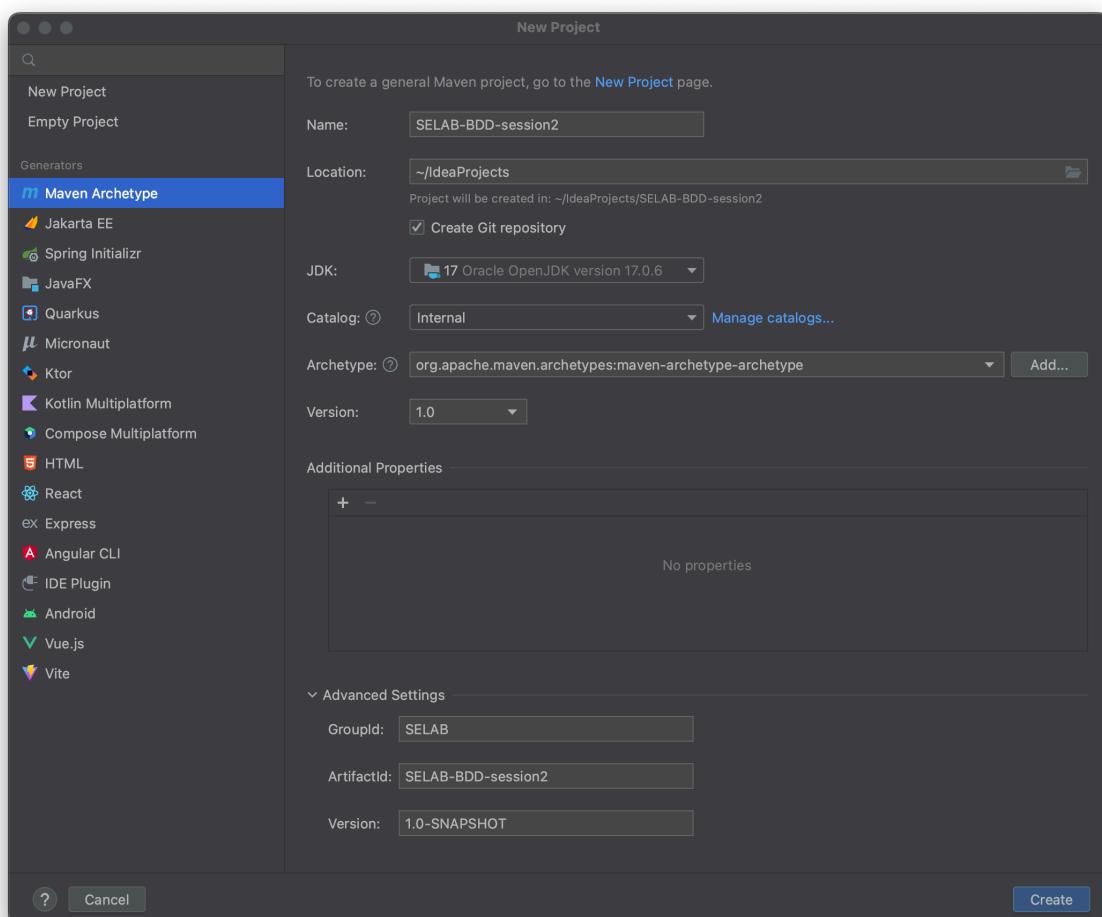
# ۱ مقدمه

در این آزمایش قصد داریم تا با تبدیل نیازمندی‌ها به موارد آزمون با استفاده از روش ایجاد مبتنی بر رفتار (BDD) آشنا شویم و به صورت عملی از آن استفاده کنیم.

کدهای این پروژه در مخزن [گیت‌هاب موجود در این لینک](#) موجود می‌باشند.

## ۲ انجام مراحل موجود در فایل Example.pdf

ابتدا یک پروژه جدید را همانند مثالی که در فایل Example.pdf است ایجاد می‌کنیم. شکل ۱، ۲ در ابتدا پروژه را با JDK 17.0.6 ایجاد کردیم اما به علت مشکلاتی که با cucumber برخورد کردیم از ۹ JDK استفاده کردیم.



شکل ۱

```

<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/maven-v4_0_0.xsd">
  <modelVersion>4.0.0</modelVersion>
  <groupId>SELAB</groupId>
  <artifactId>SELAB-BDD-session2</artifactId>
  <version>1.0-SNAPSHOT</version>
  <name>Archetype - SELAB-BDD-session2</name>
  <url>http://maven.apache.org</url>
</project>

```

Run: [org.apache.maven.plugins:maven-archetype-plugin:RELEASE] [5 min, 16 sec, 514 ms]

```

[INFO] Parameter: groupId, Value: SELAB
[INFO] Parameter: artifactId, Value: SELAB-BDD-session2
[INFO] Parameter: packageName, Value: SELAB
[INFO] Parameter: version, Value: 1.0-SNAPSHOT
[INFO] project created from Old (1.x) Archetype in dir: /private/var/folders/jl/_y3czlc14gbggq71kqclws0000gn/T/archetypeTempo/SELAB-BDD-ses
[INFO] ...
[INFO] BUILD SUCCESS
[INFO] ...
[INFO] Total time: 05:14 min
[INFO] Finished at: 2023-03-19T21:18:53+03:30
[INFO] ...

```

Process finished with exit code 0

شکل ۲

در ادامه های مورد نیاز را به فایل pom.xml اضافه کردیم. شکل ۳

```

<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/maven-v4_0_0.xsd">
  <modelVersion>4.0.0</modelVersion>
  <groupId>SELAB</groupId>
  <artifactId>SELAB-BDD-session2</artifactId>
  <version>1.0-SNAPSHOT</version>
  <name>Archetype - SELAB-BDD-session2</name>
  <url>http://maven.apache.org</url>
  <dependencies>
    <dependency>
      <groupId>info.cukes</groupId>
      <artifactId>cucumber-junit</artifactId>
      <version>1.1.5</version> </dependency>
    <dependency>
      <groupId>junit</groupId>
      <artifactId>junit</artifactId>
      <version>4.12</version>
      <scope>test</scope>
    </dependency>
    <dependency>
      <groupId>info.cukes</groupId>
      <artifactId>cucumber-java</artifactId>
      <version>1.2.5</version>
      <scope>test</scope>
    </dependency>
  </dependencies>
</project>

```

Run: [org.apache.maven.plugins:maven-archetype-plugin:RELEASE] [5 min, 16 sec, 514 ms]

```

[INFO] Parameter: groupId, Value: SELAB
[INFO] Parameter: artifactId, Value: SELAB-BDD-session2
[INFO] Parameter: packageName, Value: SELAB
[INFO] Parameter: version, Value: 1.0-SNAPSHOT
[INFO] project created from Old (1.x) Archetype in dir: /private/var/folders/jl/_y3czlc14gbggq71kqclws0000gn/T/archetypeTempo/SELAB-BDD-ses
[INFO] ...
[INFO] BUILD SUCCESS
[INFO] ...
[INFO] Total time: 05:14 min
[INFO] Finished at: 2023-03-19T21:18:55+03:30
[INFO] ...

```

Process finished with exit code 0

شکل ۳

سپس تست‌ها را اجرا کردیم تا یک دور build انجام شود و همه چیز آماده شود و موارد لازم

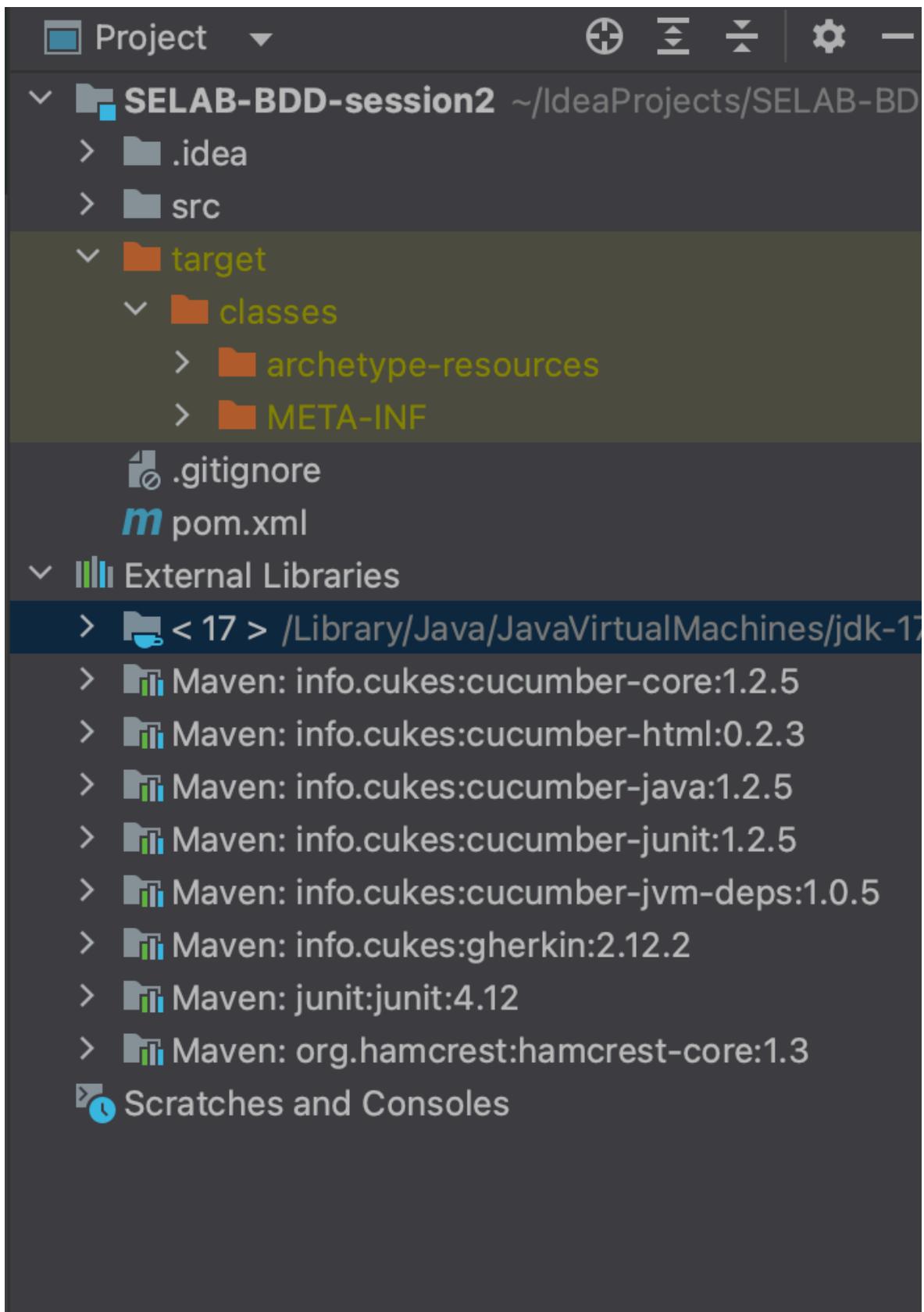
## دانلود شوند. شکل ۴

```
<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">
  <modelVersion>4.0.0</modelVersion>
  <groupId>SELAB</groupId>
  <artifactId>SELAB-BDD-session2</artifactId>
  <version>1.0-SNAPSHOT</version>
  <name>Archetype - SELAB-BDD-session2</name>
  <url>http://maven.apache.org</url>
  <dependencies>
    <dependency>
      <groupId>info.cukes</groupId>
      <artifactId>cucumber-junit</artifactId>
      <version>3.2.4</version>
      <scope>test</scope>
    </dependency>
    <dependency>
      <groupId>junit</groupId>
      <artifactId>junit</artifactId>
      <version>4.12</version>
      <scope>test</scope>
    </dependency>
    <dependency>
      <groupId>info.cukes</groupId>
      <artifactId>cucumber-java</artifactId>
      <version>1.2.5</version>
      <scope>test</scope>
    </dependency>
  </dependencies>
</project>
```

Run: SELAB-BDD-session2 [test] ...  
SELAB-BDD-session2 [test]: At 3/19/23, 9:25 PM with 1 warning  
SELAB-SELAB-BDD-session2jar:1.0-SNAPSHOT 1 warning  
resources 1 warning  
Using platform encoding (UTF-8 actually) to copy filtered resources, i.e. build  
[INFO] --- maven-compiler-plugin:3.1:testCompile (default-testCompile) @ SELAB-BDD-session2 ...  
[INFO] No sources to compile  
[INFO] --- maven-surefire-plugin:2.12.4:test (default-test) @ SELAB-BDD-session2 ...  
[INFO] No tests to run.  
[INFO] BUILD SUCCESS  
[INFO] -----  
[INFO] Total time: 33.347 s  
[INFO] Finished at: 2023-03-19T21:25:57+03:30  
[INFO] -----  
Process finished with exit code 0

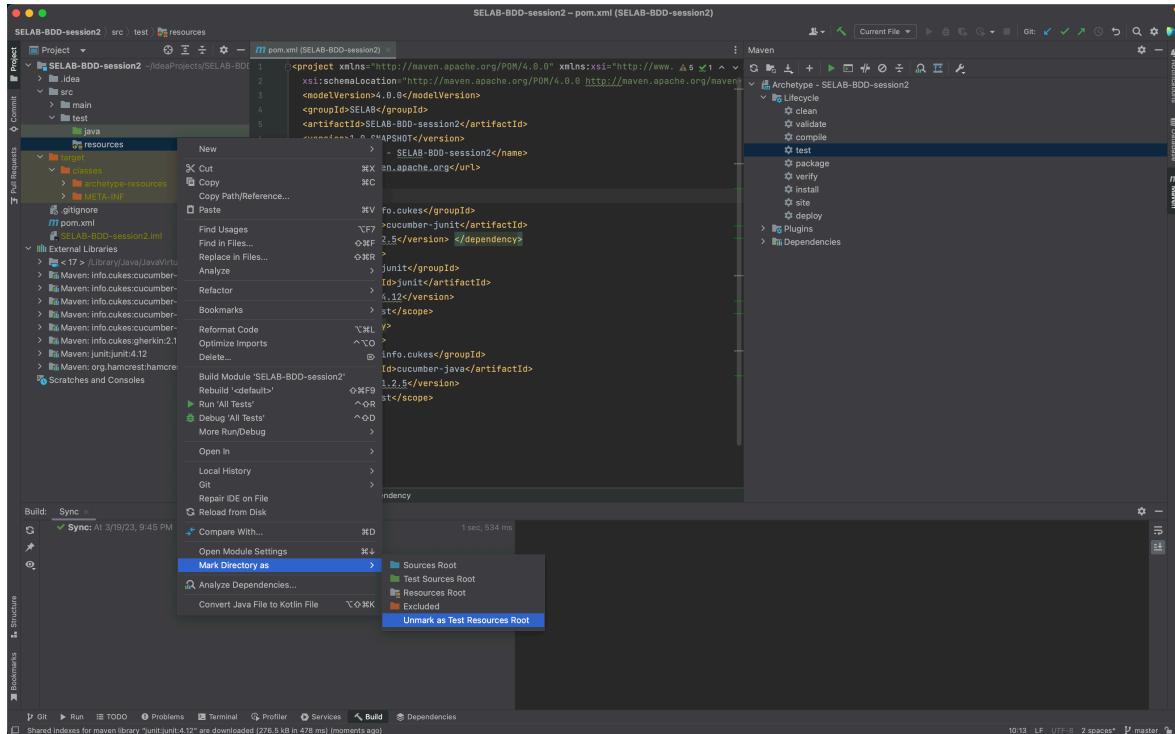
## شکل ۴

ملزومات پروژه در این بخش انجام شده‌اند و ما آماده‌ی اجرای مرحله‌ی بعد هستیم. شکل ۵

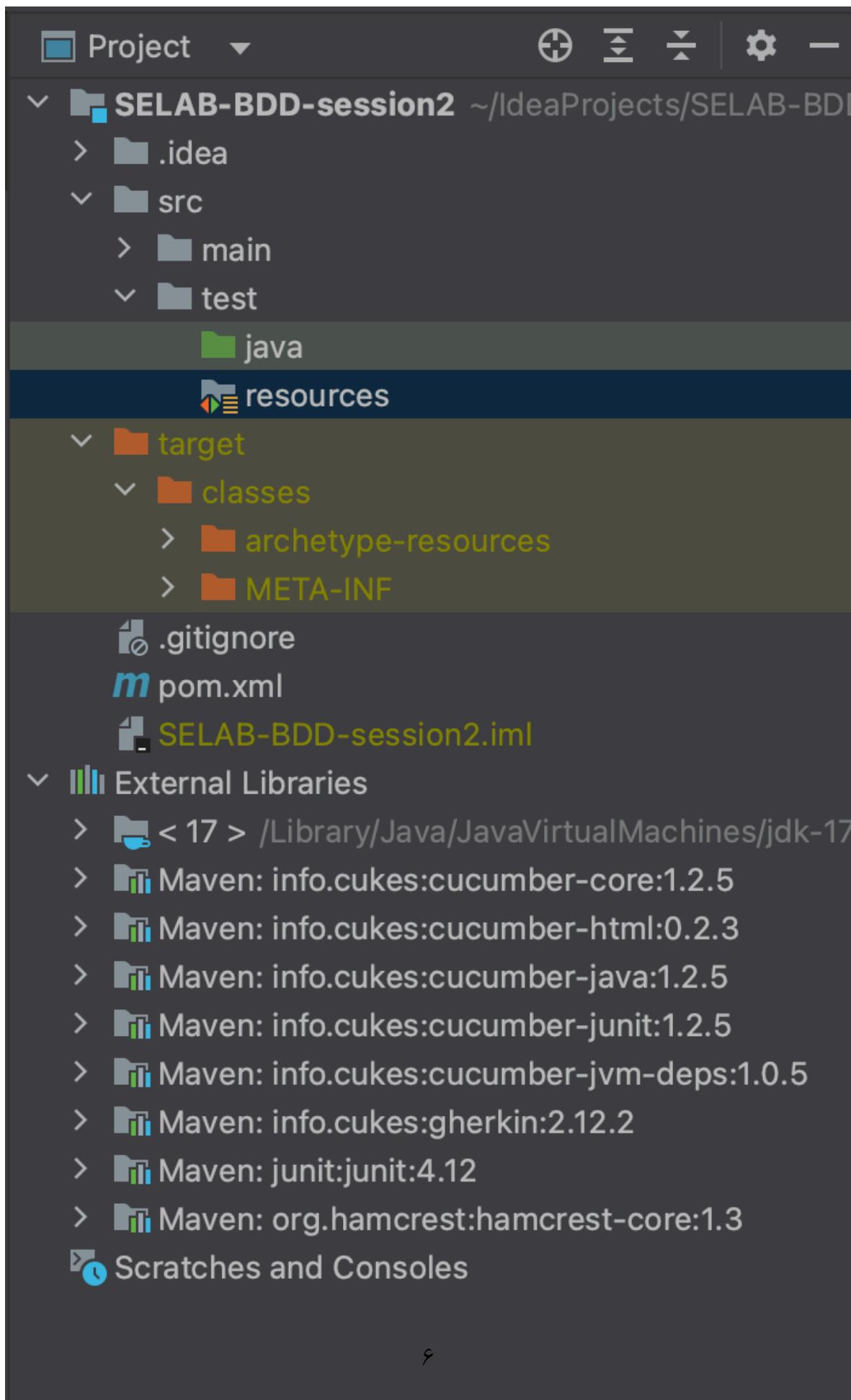


شكل ٥

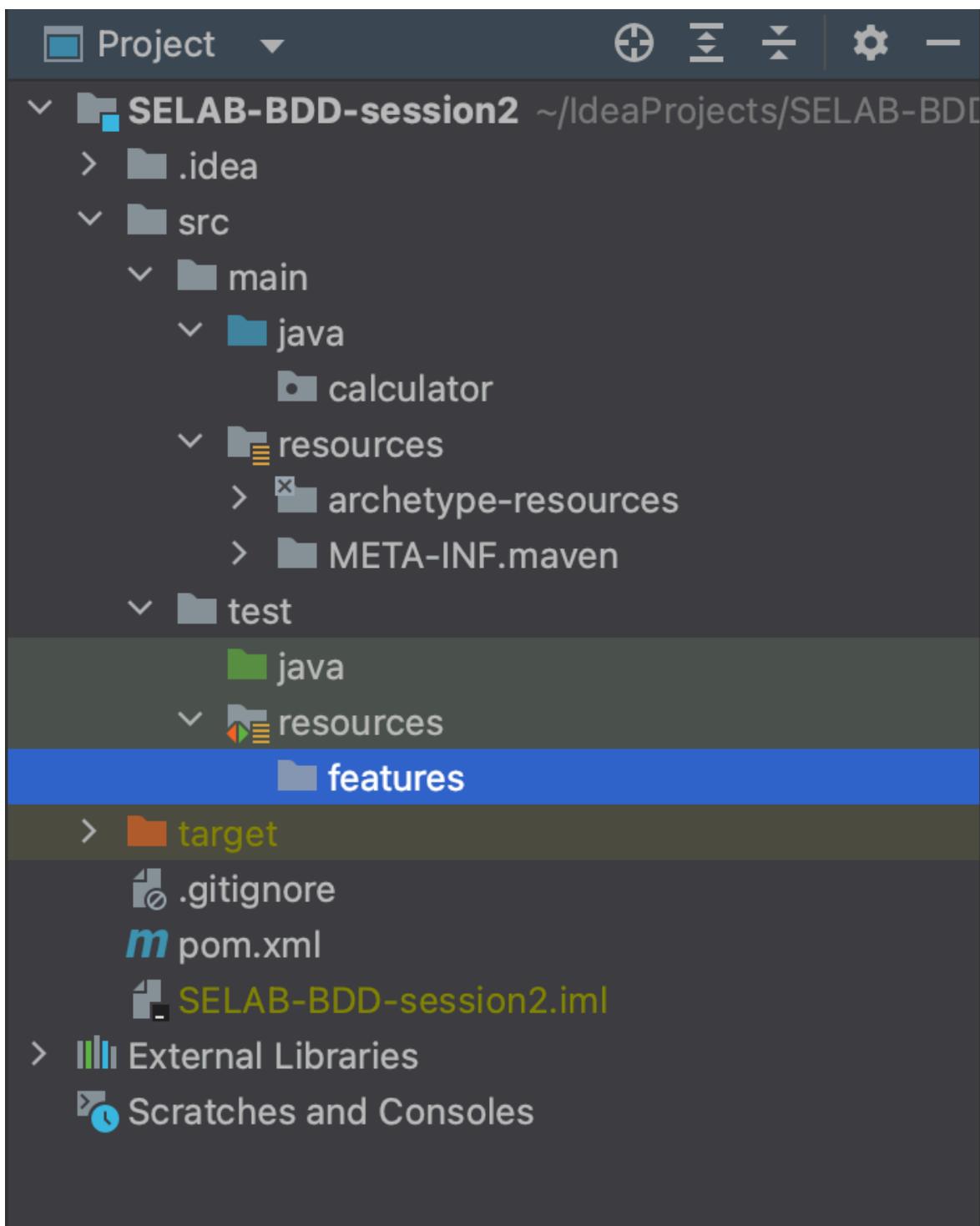
## ایجاد پکیج جدید با نام resources. شکل ۶، ۷



شکل ۶

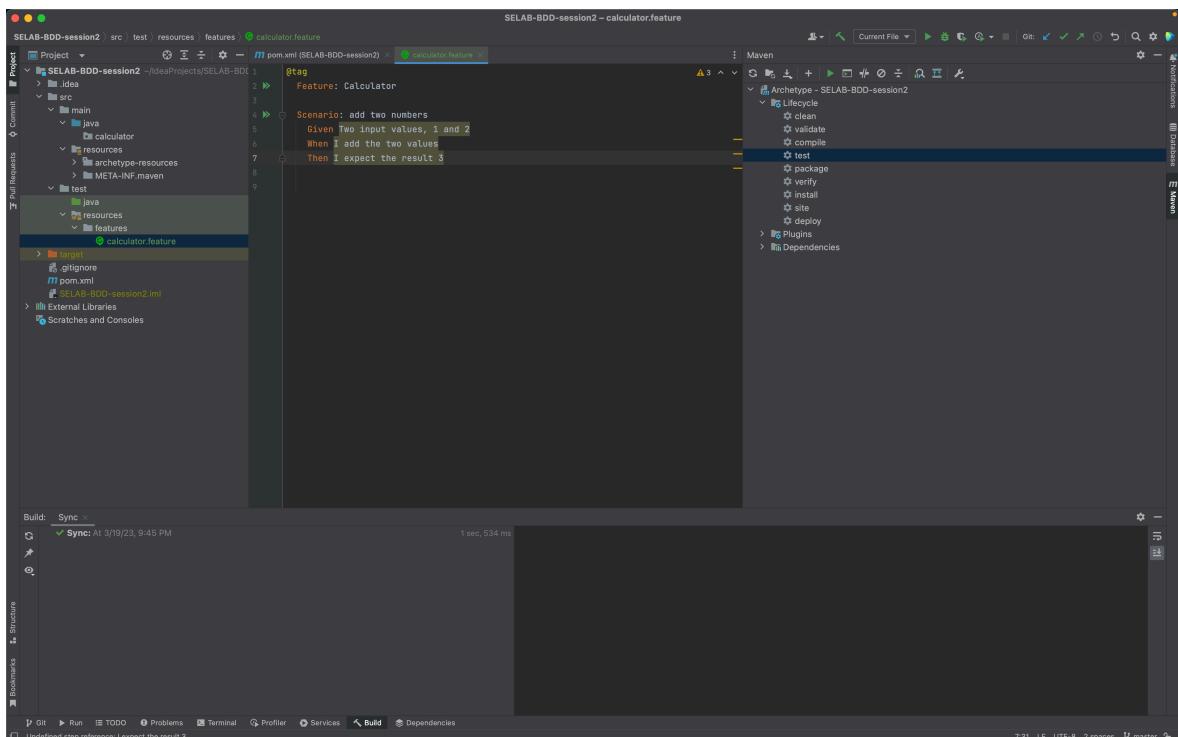


ایجاد پکیج جدید برای features. شکل ۸

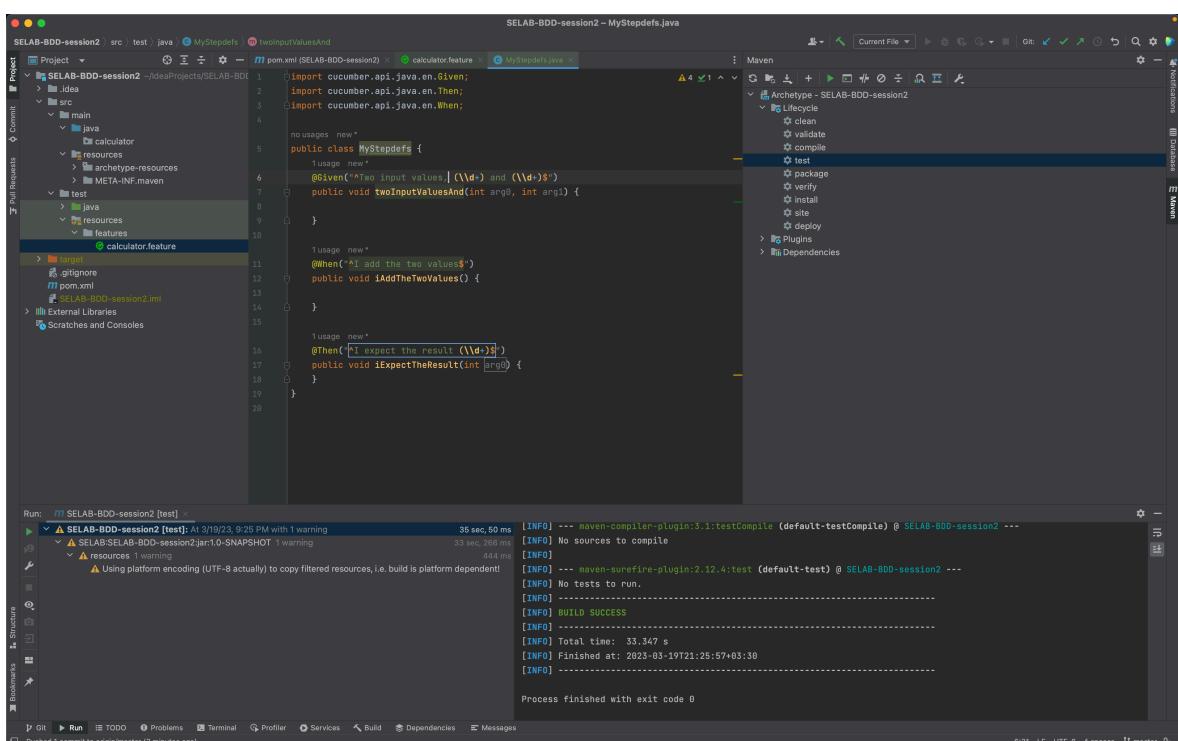


شکل ۸

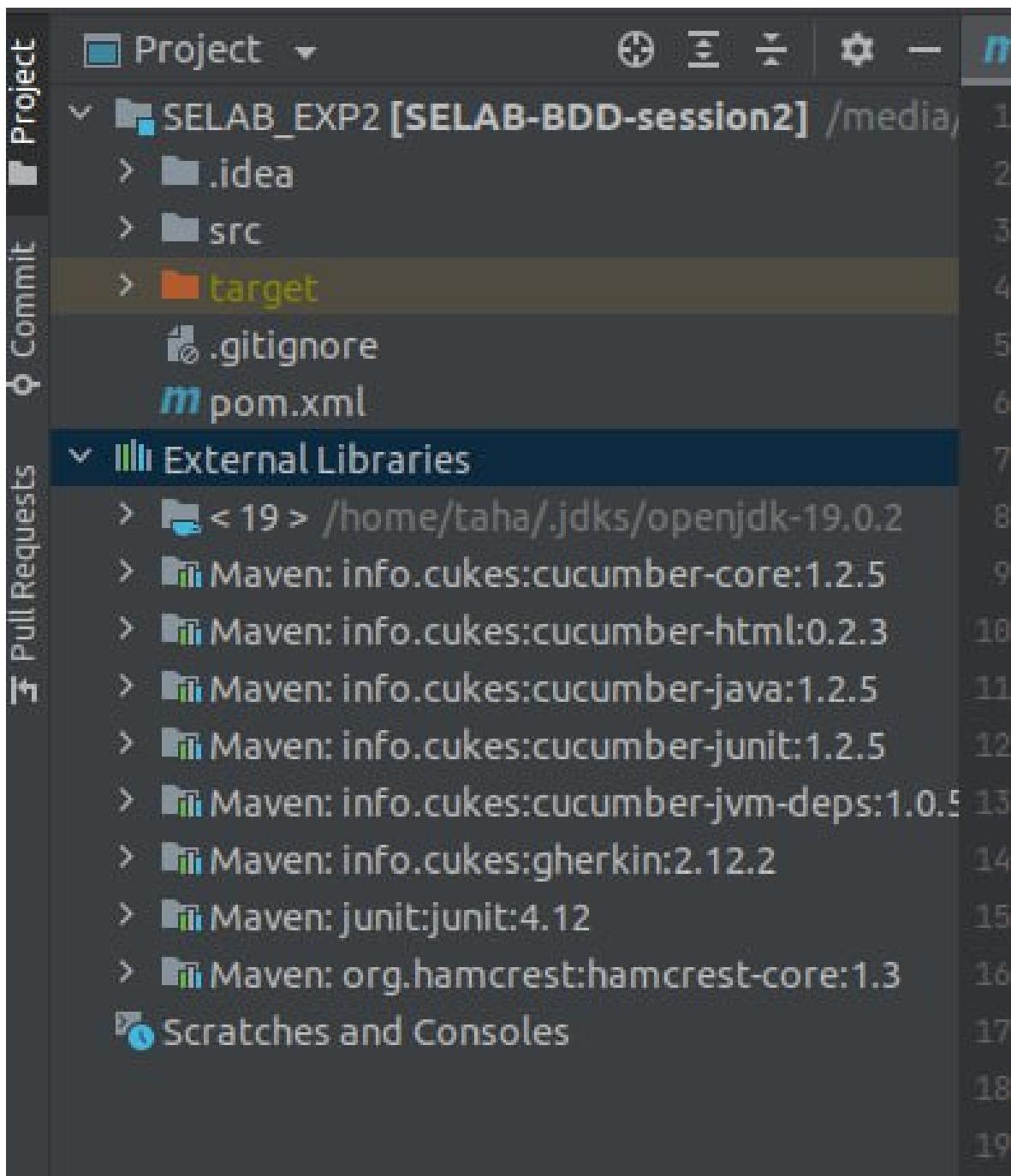
ایجاد فایل جدید با عنوان calculator.feature. شکل ۹ همچنین اضافه کردن توصیفات گفته شده به این فایل با زبان Gherkin. و سپس ایجاد متدهای بدون بدنه در فایل MyStepdefs و تکمیل آنها و رفع مشکل و در واقع اصلاح ورژن Maven در فایل pom.xml. شکل ۱۰، ۱۱، ۱۲



شكل ٩



شكل ١٠



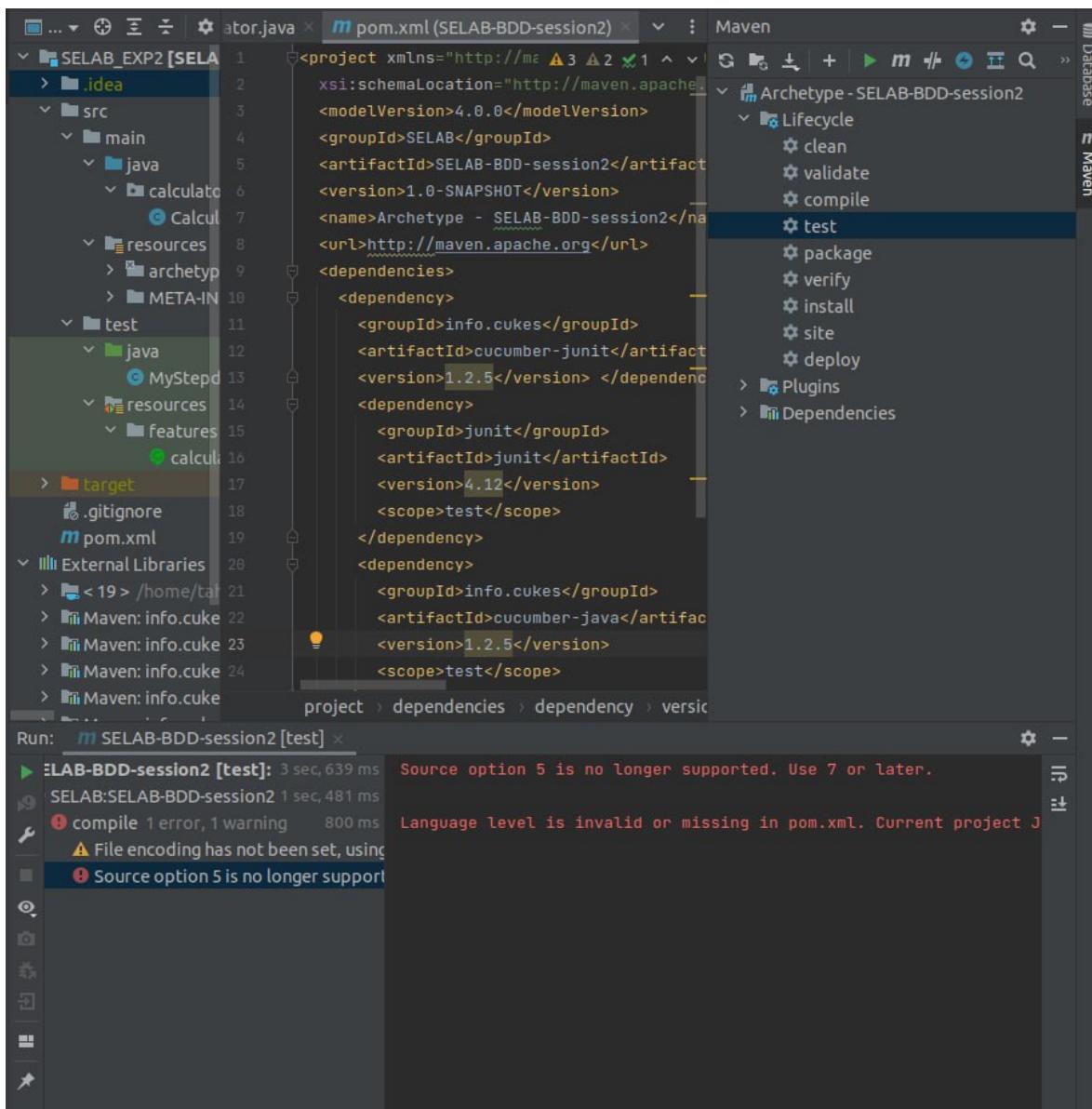
شكل ١١

The screenshot shows an IDE interface with a code editor and a terminal window. The code editor displays a Java file named `MyStepdefs.java`. The file contains Cucumber step definitions and some setup code. A code completion tooltip is open at line 10, highlighting the symbol 'Calculator'. The tooltip includes options like 'Create type parameter 'Calculator'' and 'More actions...'. The terminal window below shows a command prompt on a Linux system.

```
1 import cucumber.api.java.Before;
2 import cucumber.api.java.en.Given;
3 import cucumber.api.java.en.Then;
4 import cucumber.api.java.en.When;
5 import org.junit.Assert;
6
7 public class MyStepdefs {
8     private Calculator calculator;
9     private int
10    private int
11    private int
12    @Before
13    public void before() {
14        calculator = new Calculator();
15    }
16
17    @Given("^Two input values, (\d+) and (\d+)$")
18    public void twoInputValuesAnd(int arg0, int arg1) {
19        value1 = arg0;
20        value2 = arg1;
21    }
22
23    @When("I add the two values$")
24    public void iAddTheTwoValues() {
25
```

Terminal: Local × + ▾ taha@taha-ZenBook-UX461FN-UX461FN:/media/extra/Documents/Term 8/Software Lab/SELAB\_EXP2\$

١٢ شكل



شكل ١٣

The screenshot shows the IntelliJ IDEA interface with the following components:

- Project View:** Shows the project structure under "SELAB\_EXP2 [SELA]". It includes a .idea folder, a src directory with main and test sub-directories, and a target directory.
- pom.xml Editor:** Displays the Maven configuration file. The "maven.compiler.target" setting is highlighted at line 32, value 1.8.
- Maven Tool Window:** Located on the right, it shows the "Lifecycle" section with "test" selected. Other options include clean, validate, compile, package, verify, install, site, and deploy.
- Run Output:** Shows the command-line output for the "SELAB-BDD-session2 [test]" run. The output indicates a successful build with a total time of 1.460 seconds and a finish time of 2023-03-19T22:09:12+03:30.

شکل ۱۴

همانطور که می‌بینید در نهایت تست نوشته شده با موفقیت پاس می‌شود. شکل ۱۵

The screenshot shows the IntelliJ IDEA interface with the project structure for SELAB-BDD-session2. The pom.xml file is open, displaying the Maven configuration. The 'calculator' dependency section is selected. The 'Run' tool window at the bottom shows the execution of 'calculator.feature' resulted in 1 scenario and 3 steps passed. The code editor shows the feature file content.

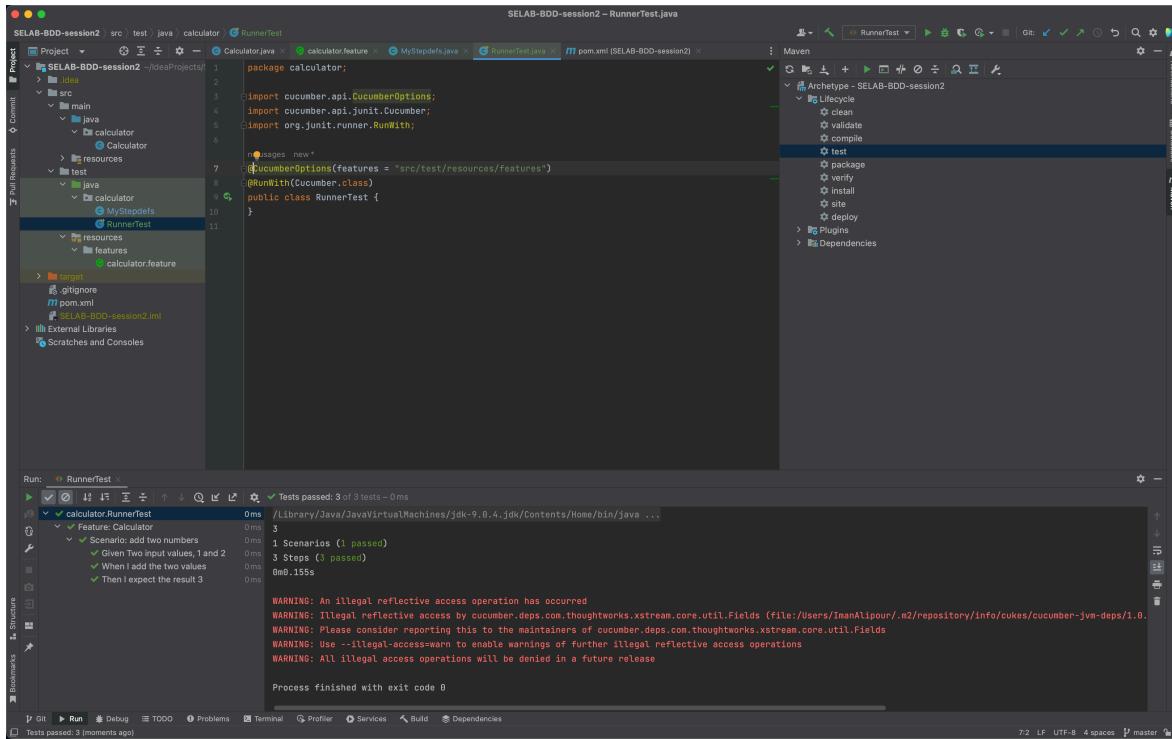
شکل ۱۵

حال تست را با JUnit انجام می‌دهیم و به مشکلی که در شکل ۱۶ مشاهده می‌کنید بروخورد می‌کنیم.

The screenshot shows the IntelliJ IDEA interface with the project structure for SELAB-BDD-session2. The RunnerTest.java file is open, showing the code for running Cucumber tests using JUnit. The 'Run' tool window at the bottom shows the execution of 'calculator.RunnerTest' resulted in 0 features found. The code editor shows the RunnerTest.java file content.

شکل ۱۶

برای رفع این مشکل مسیر فایل features را در RunnerTest می‌دهیم و دوباره تست را ران می‌کنیم. شکل ۱۷



شکل ۱۷

تست را تغییر می‌دهیم تا فیل شدن را نیز مشاهده کنیم. شکل ۱۸

```

SELAB-BDD-session2 - calculator.feature
Feature: Calculator
  Scenario: add two numbers
    Given Two input values, 1 and 2
    When I add the two values
    Then I expect the result 3000

159ms
Testing started at 11:16 PM...
WARNING: An illegal reflective access operation has occurred
WARNING: Illegal reflective access by cucumber.deps.com.thoughtworks.xstream.core.util.Fields (file:/Users/ImanAlipour/m2/repository/info/cukes/cucumber-jvm-deps/1.0.2/cucumber-deps-1.0.2.jar)
WARNING: Please consider reporting this to the maintainers of cucumber.deps.com.thoughtworks.xstream.core.util.Fields
WARNING: Use -illegal-access=warn to enable warnings of further illegal reflective access operations
WARNING: All illegal access operations will be denied in a future release
3
java.lang.AssertionError: Create breakpoint: expected:<3000> but was:<> <1 internal lines>
at org.junit.Assert.failNotEquals(Assert.java:835) <2 internal lines>
at calculator.MyStepdefs.IExpectTheResult(MyStepdefs.java:32)
at *.Then I expect the result 3000(</Users/ImanAlipour/IdeaProjects/SELAB-BDD-session2/src/test/resources/features/calculator.feature:7>)

Failed scenarios:
</Users/ImanAlipour/IdeaProjects/SELAB-BDD-session2/src/test/resources/features/calculator.feature:3> # Scenario: add two numbers

1 Scenarios (1 failed)
3 Steps (1 failed, 2 passed)
0m0.161s

java.lang.AssertionError: Create breakpoint: expected:<3000> but was:<> <1 internal lines>
at org.junit.Assert.failNotEquals(Assert.java:835) <2 internal lines>
at calculator.MyStepdefs.IExpectTheResult(MyStepdefs.java:32)
at *.Then I expect the result 3000(</Users/ImanAlipour/IdeaProjects/SELAB-BDD-session2/src/test/resources/features/calculator.feature:7>)

Process finished with exit code 1

```

شکل ۱۸

حال از استفاده می‌کنیم تا چندین تست را اجرا کنیم. شکل ۱۹ در یکی از مشاهده می‌کنیم. **tests**

```

SELAB-BDD-session2 - calculator.feature
Feature: Calculator
  Scenario: add two numbers
    Given Two input values, 1 and 2
    When I add the two values
    Then I expect the result 3

  Scenario Outline: add two numbers:
    Given Two input values, <first> and <second>
    When I add the two values
    Then I expect the result <result>

  Examples:
    | first | second | result |
    | 1     | 12      | 13   |
    | -1    | 6       | 5    |
    | 2     | 2       | 4    |

159ms
Stopped. Tests passed: 9, ignored: 6 of 18 tests - 0 ms
313
Test ignored.

Test ignored.

Test ignored.

4
4 Scenarios (1 undefined, 3 passed)
12 Steps (2 skipped, 1 undefined, 9 passed)
0m0.143s

You can implement missing steps with the snippets below:

```

شکل ۱۹

## مسئله‌ی ۱ موارد تستی که به مشکل بخوردند کدامند؟ چرا این مشکل پیش آمده است؟

علت این مشکل این است که در یکی از تست‌ها ما از عدد منفی استفاده کردی‌ایم در حلی که در توابع *step* اعداد منفی را لحاظ نکرده بودیم، در واقع مشکل اصلی رجکسی یا *Regular Expression* یی است که برای این بخش استفاده کردیم، در رجکس استفاده شده علائم  $+$  و  $-$  که می‌توانند پیش از عدد بیایند لحاظ نشده‌اند، برای رفع این مسئله رجکس‌ها را به نحوی تغییر می‌دهیم که این علائم را بتوانند بشناسند، سپس تست‌ها را مجدداً ران می‌کنیم و همانطور که مشاهده می‌کنید

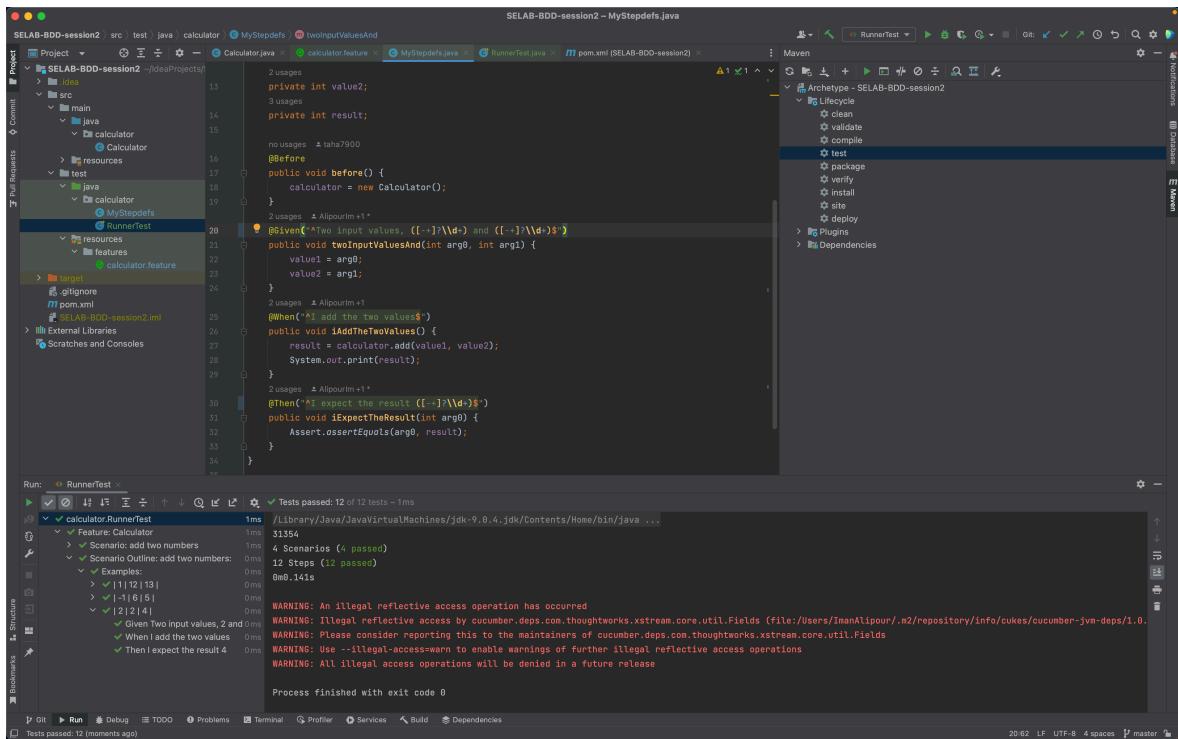
همگی پاس می‌شوند. شکل ۲۰ و ۲۱

```
2 usages  Alipourlm+1 *
@Given("^Two input values, ([+-]?\d+) and ([+-]?\d+)$")
public void twoInputValuesAnd(int arg0, int arg1) {
    value1 = arg0;
    value2 = arg1;
}

2 usages  Alipourlm+1
@When("I add the two values$")
public void iAddTheTwoValues() {
    result = calculator.add(value1, value2);
    System.out.print(result);
}

2 usages  Alipourlm+1 *
@Then("I expect the result ([+-]?\d+)$")
public void iExpectTheResult(int arg0) {
    Assert.assertEquals(arg0, result);
}
```

شکل ۲۰



۲۱ شکل

### ۳ انجام مراحل پیشین برای مثال گفته شده در شرح آزمایش

برای انجام این کار ابتدا به ماشین حسابی که پیشتر ایجاد کردیم تابع زیر را اضافه می‌کنیم. شکل ۲۲ این تابع با گرفتن یک *operator* و یک ورودی بر اساس عملگر داده شده اعمال گفته شده را انجام می‌دهد. همچنین بررسی می‌کند تا عدد داده شده اگر عملگر *MD* نظر مار *sqr* است عدد منفی نباشد و اگر عملگر مدنظر ما *rvs* است، ورودی داده شده عدد صفر نباشد، همچنین اگر عملگر ساپورت نمی‌شد خطایی را گزارش می‌کند.

```
1 package calculator;
2
3     3 usages  ↳ Alipourlm +1
4     public class Calculator {
5         1 usage  ↳ taha7900
6             ↳
7                 ↳
8 @     ↳
9         ↳
10            ↳
11            ↳
12            ↳
13            ↳
14            ↳
15            ↳
16            ↳
17            ↳
18            ↳
19            ↳
20            ↳
21            ↳
22        }
23
```

شکل ۲۲

در ادامه گام‌ها را مانند مثالی که پیشتر انجام دادیم تعریف می‌کنیم. شکل ۲۳ به علت ماهیت کد یک گام نیز برای خطاهای درنظر گرفته‌ایم.

```

3 usages  ▲ Alipourlm +1 *
@Then("^I expect the result ([+-]?\d+)$")
public void iExpectTheResult(double arg0) {
    Assert.assertEquals(arg0, result, delta: 0.000001);
}

2 usages new *
@Given("^Input value ([+-]?\d+) and opt \"([^\"]*)\"$")
public void inputValueAndOpt(int arg0, String arg1) throws Throwable {
    value1 = arg0;
    opt = arg1;
}

2 usages new *
@When("^I run func$")
public void iRunFunc() {
    try{
        result = calculator.func(value1, opt);
        System.out.print(result);
    } catch (Exception e){
        error = e;
    }
}

1 usage new *
@Then("^I expect to get error$")
public void iExpectToGetError() {
    Assert.assertNotNull(error);
}
}

```

شکل ۲۳

دو سناریو می‌نویسیم تا تست‌های اولیه را انجام بدھیم. شکل ۲۴

```

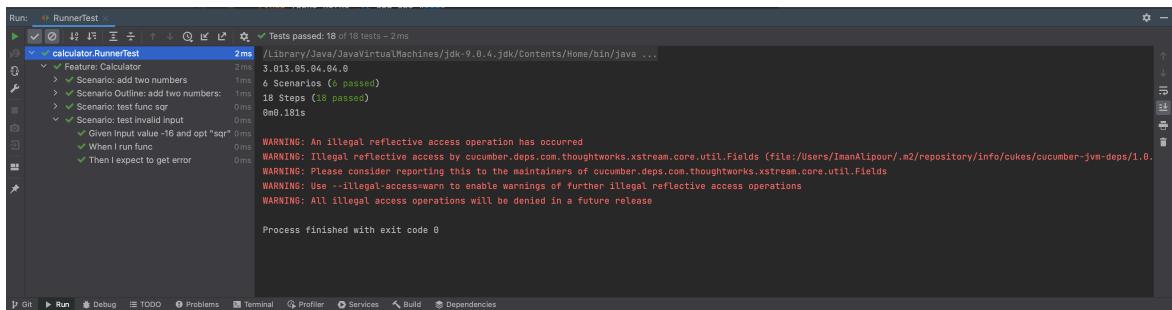
Scenario: test func sqr
  Given Input value 16 and opt "sqr"
  When I run func
  Then I expect the result 4

Scenario: test invalid input
  Given Input value -16 and opt "sqr"
  When I run func
  Then I expect to get error

```

شکل ۲۴

همانطور که مشاهده می‌کنید این سناریوها با موفقیت پاس می‌شوند. شکل ۲۵



شکل ۲۵

حال که از صحبت کلی محیط آگاه شدیم تست‌ها را کاملتر می‌کنیم و سناریوهای فیل شدن را نیز به ان اضافه می‌کنیم. همچنین یک *Scenario Outline* نیز ایجاد می‌کنیم تا تست‌های غیر حالت خاص و بیشتری را به صورت اتوماتیک انجام دهیم، در این تست‌ها حالات مختلف مانند عدد منفی و ... در نظر گرفته شده‌اند. همچنین برای حالات خاص که باید خطأ داده شود سناریوهای خاص منظوره نوشته شده‌اند. شکل ۲۶

```

20 ➤ Scenario: test func sqr
21   Given Input value 16 and opt "sqr"
22   When I run func
23   Then I expect the result 4
24
25 ➤ Scenario: test invalid input sqr
26   Given Input value -16 and opt "sqr"
27   When I run func
28   Then I expect to get error
29
30 ➤ Scenario: test func rvs
31   Given Input value 2 and opt "rvs"
32   When I run func
33   Then I expect the result 0.5
34
35
36 ➤ Scenario: test invalid input rvs
37   Given Input value 0 and opt "rvs"
38   When I run func
39   Then I expect to get error
40
41 ➤ Scenario Outline: test func
42   Given Input value <Input> and opt "<opt>"
43   When I run func
44   Then I expect the result <result>
45
46   Examples:
47     | Input | opt | result |
48     | 4    | rvs | 0.25  |
49     | 4    | sqr | 2      |
50     | 1    | sqr | 1      |
51     | 1    | rvs | 1      |
52     | 9    | sqr | 3      |
53     | 6    | rvs | 0.16666 |
54     | -5   | rvs | -0.2   |
55

```

شكل ٢٦

حال تست‌ها را اجرا می‌کنیم و همان‌طور که مشاهده می‌کنید همگی با موفقیت پاس می‌شوند.

## ٢٧ شکل

The screenshot shows the IntelliJ IDEA interface with the 'Run' tool window open. The 'calculator.RunnerTest' is selected, showing a green checkmark indicating 'Tests passed: 45 of 45 tests - 0ms'. The test results are listed under the feature 'calculator' and its scenarios. The output pane displays the command used ('/Library/Java/JavaVirtualMachines/jdk-9.0.4.jdk/Contents/Home/bin/java ...'), the number of scenarios and steps passed (15 Scenarios (15 passed), 45 Steps (45 passed)), and the time taken (0m0.16s). It also includes several warning messages related to illegal reflective access and cucumber dependencies. The bottom of the window shows the terminal output with the message 'Process finished with exit code 0'.

شکل ٢٧