

Sumar

1. Crearea unui proiect Maven cu Serenity BDD și JUnit în IntelliJ IDEA.....	1
2. Structura proiectului Maven cu Serenity BDD și JUnit	3
3. Alte posibile configurări ale proiectului Maven	4
4. Setarea browser-ului web pentru rularea testelor	5
5. Execuția testelor	6
6. Generarea raportului Serenity BDD pentru testele executate.....	7
Varianta 1	7
Varianta 2	8
7. Vizualizarea raportului Serenity BDD	8
8. Data Driven Testing	9

Lista de Figuri

Figure 1 Crearea unui proiect Maven cu Serenity BDD și JUnit	2
Figure 2 Adăugarea tipului de proiect Serenity BDD cu JUnit în lista de tipuri de proiecte	2
Figure 3. Completarea numelui pachetului root și a numelui proiectului Serenity	3
Figure 4. Vizualizarea și modificarea configurărilor proiectului Maven.....	3
Figure 5. Structura proiectului Maven cu Serenity BDD și JUnit	4
Figure 6. Setarea în variabila <i>Path</i> a folderului care conține driverele pentru browser-ele web folosite la testare	5
Figure 7. Setarea implicită a browser-ului web Mozilla Firefox	5
Figure 8. Crearea unei configurații de rulare a testelor în Serenity BDD	6
Figure 9. Setarea parametrilor pentru comanda <i>verify</i> , cu setarea browser-ului utilizat la execuție	6
Figure 10. Fereastra Maven Projects și comenzile Maven	7
Figure 11. Fereastra Maven Projects și opțiunea Execute Maven Goal.....	7
Figure 12. Fereastra de comenzi Maven pentru generarea raportului de testare	8
Figure 13. Fereastra Terminal cu execuția comenzii de generare a raportului de testare	8
Figure 14 Vizualizarea raportului Serenity BDD	8
Figure 15 Crearea fișierului cu date de test	9
Figure 16 Clasă de test parametrizată, folosind un fișier <i>.csv</i>	9
Figure 17 Raportul Serenity BDD pentru testele din fișierul <i>.csv</i>	10

1. Crearea unui proiect Maven cu Serenity BDD și JUnit în IntelliJ IDEA

1. în meniul **File** ---> **New** ---> **Project**;
2. se selectează din tipul de proiect **Maven**;
3. se alege o versiune un SDK, se recomandă **versiunea 14.0.2** sau o versiune ulterioară;
4. se bifează opțiunea **Create from archetype**;

5. se alege din lista de tipuri de proiecte **serenity-junit-archetype** (vezi Figure 1), versiunea **2.0.81**, apoi **Next**;
6. dacă tipul de proiect nu se găsește în listă, se va adăuga prin click pe butonul **Add Archetype...** și se completează (vezi Figure 2):
 - **GroupId**: **net.serenity-bdd**
 - **ArtifactId**: **serenity-junit-archetype**
 - **Version**: **2.0.81**apoi **OK**;
7. Lista de tipuri de proiecte Maven bazate pe Serenity poate fi accesată de la adresa web: <https://mvnrepository.com/artifact/net.serenity-bdd>.

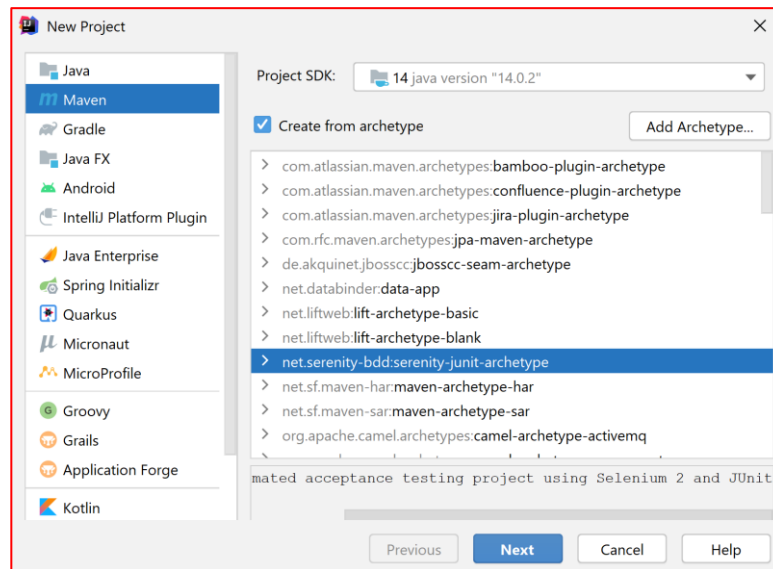


Figure 1 Crearea unui proiect Maven cu Serenity BDD și JUnit

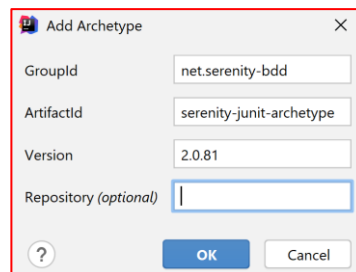


Figure 2 Adăugarea tipului de proiect Serenity BDD cu JUnit în lista de tipuri de proiecte

8. se completează **Name**, i.e., numele folderului asociat proiectului (vezi Figure 3); implicit acesta va fi și numele proiectului, i.e., **ArtifactId**;
9. se completează **GroupId**, i.e. numele pachetului root (vezi Figure 3);
10. se completează **ArtifactId**, i.e., numele proiectului (vezi Figure 3), apoi **Next**;

New Project

Name: AutomationDemo

Location: E:\TDT\TDT2020-2021\Labs\Lab04\AutomationDemo

▼ Artifact Coordinates

GroupId: org.example
The name of the artifact group, usually a company domain

ArtifactId: AutomationDemo
The name of the artifact within the group, usually a project name

Version: 1.0-SNAPSHOT

Previous Next Cancel Help

Figure 3. Completarea numelui pachetului root și a numelui proiectului Serenity

11. se pot vizualiza și modifica configurările proiectului (vezi Figure 4), apoi **Finish**.

New Project

Maven home path: Bundled (Maven 3)
(Version: 3.6.3)

User settings file: C:\Users\Dumitru\.m2\settings.xml

Local repository: C:\Users\Dumitru\.m2\repository

Properties

groupId	org.example
artifactId	AutomationDemo
version	1.0-SNAPSHOT
archetypeGroupId	net.serenity-bdd
archetypeArtifactId	serenity-junit-archetype
archetypeVersion	2.0.81
archetypeRepository	C:/Users/Dumitru/.m2/repository

Previous Finish Cancel Help

Figure 4. Vizualizarea și modificarea configurărilor proiectului Maven

2. Structura proiectului Maven cu Serenity BDD și JUnit

- după creare, proiectul are o structură similară cu a unui proiect Maven obișnuit, care conține doar pachetul **test/java** (vezi Figure 5);
 - pachetul **main/java** nu este generat deoarece aplicația care urmează să fie testată este accesată prin intermediul unei adrese web, așadar nu este necesar să adăugăm cod sursă într-un asemenea pachet;
- pachetul **test/java** conține câteva subpachete:
 - features;**
 - pages;**
 - steps**
 + câteva clase generate care implementează șablonul **Page Object Model** (vezi [Curs 05 – Intro to Automation and Serenity](#) (Prezentare Evozon)), necesar pentru reprezentarea *paginilor* unei aplicații web, a *acțiunilor* desfășurate asupra elementelor interfeței și a *testelor* funcționale implementate.

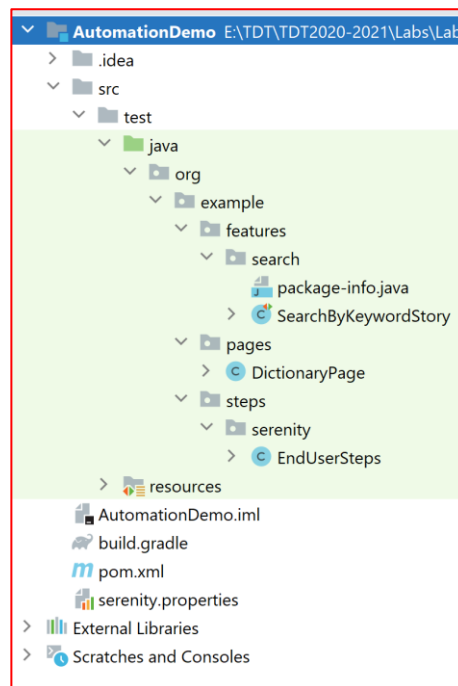


Figure 5. Structura proiectului Maven cu Serenity BDD și JUnit

3. Alte posibile configurări ale proiectului Maven

La crearea proiectului Maven, în funcție de versiunea Java instalată, cât și de alte configurări și setări existente, mai pot fi necesare și următoarele:

1. este posibil ca la descărcarea resurselor asociate proiectului Serenity BDD să fie necesară modificarea protocolului utilizat (din HTTP în HTTPS) în fișierul **pom.xml** la descrierea adresei de acces la repository-ul folosit, e.g.,
`<url>http://jcenter.bintray.com</url>`
2. anumite versiuni de Java SE nu conțin implicit JAXB APIs; acestea pot fi incluse manual în fișierul **pom.xml** ca dependențe în secțiunea `<dependencies>...`
`</dependencies>`

```
<dependency>
  <groupId>javax.xml.bind</groupId>
  <artifactId>jaxb-api</artifactId>
  <version>2.3.0</version>
</dependency>

<dependency>
  <groupId>com.sun.xml.bind</groupId>
  <artifactId>jaxb-impl</artifactId>
  <version>2.3.0</version>
</dependency>

<dependency>
  <groupId>javax.activation</groupId>
  <artifactId>activation</artifactId>
  <version>1.1.1</version>
</dependency>

<dependency>
  <groupId>com.sun.xml.bind</groupId>
  <artifactId>jaxb-core</artifactId>
  <version>2.3.0</version>
</dependency>
```

Așadar, aceste configurări sunt necesare doar dacă la rularea testelor incluse implicit în proiect apar erori. Altfel, alte modificări în fișierul **pom.xml** nu sunt obligatorii.

4. Setarea browser-ului web pentru rularea testelor

1. la crearea proiectului Maven cu Serenity BDD și JUnit, browser-ul web Mozilla Firefox este setat ca browser implicit pentru rularea testelor;
2. driverele pentru browser-ele web folosite la testare:
 - Firefox:
 - adresa web: <https://github.com/mozilla/geckodriver/releases>;
 - Chrome:
 - adresa web: <https://sites.google.com/a/chromium.org/chromedriver/downloads>;
3. driverele se descarcă, se dezarchivează și se salvează într-un folder, e.g., **c:\drivers**, de unde pot fi folosite ulterior de orice proiect de testare;
4. în variabila de mediu **Path** se adaugă calea către folderul care conține driverele pentru browser-ele web, i.e., **c:\drivers** (vezi Figure 6);

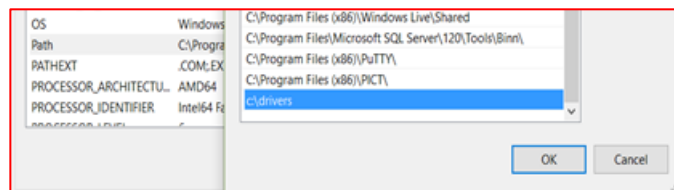


Figure 6. Setarea în variabila *Path* a folderului care conține driverele pentru browser-ele web folosite la testare

În funcție de versiunea de proiect Serenity BDD, e.g., 1.8.4 sau 2.0.81, indicarea browser-ului folosit pentru rularea testelor se face în mod diferit.

Pentru versiunea 1.8.4:

1. fișierul **pom.xml** generat conține specificarea referitoare la browser-ul folosit la rularea testelor, i.e., implicit **firefox** (vezi Figure 7);
2. pentru schimbarea browser-ului se modifică în fișierul **pom.xml** tipul driverului, i.e., **chrome** sau **firefox**;

```
<name>Serenity project with JUnit and WebDriver</name>

<properties>
  <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
  <serenity.version>1.8.3</serenity.version>
  <webdriver.driver>firefox</webdriver.driver>
</properties>
```

Figure 7. Setarea implicită a browser-ului web Mozilla Firefox

Pentru versiunea 2.0.81:

1. fișierul **pom.xml** generat nu conține specificarea referitoare la browser-ul folosit la rularea testelor;
2. ulterior, adăugarea elementului `<webdriver.driver>chrome</webdriver.driver>` nu are efectul dorit;
3. în fereastra de comenzi Maven, pentru comanda **verify** se creează o configurație de rulare nouă, actualizând-o pe cea implicită: click dreapta pe **verify** ---> **Modify [configuration]** (vezi Figure 8);
4. pentru comanda **verify** se utilizează ca parametri **context** și **webdriver.driver** (vezi Figure 9):

```
verify -Dcontext=chrome -Dwebdriver.driver=chrome -f pom.xml
```

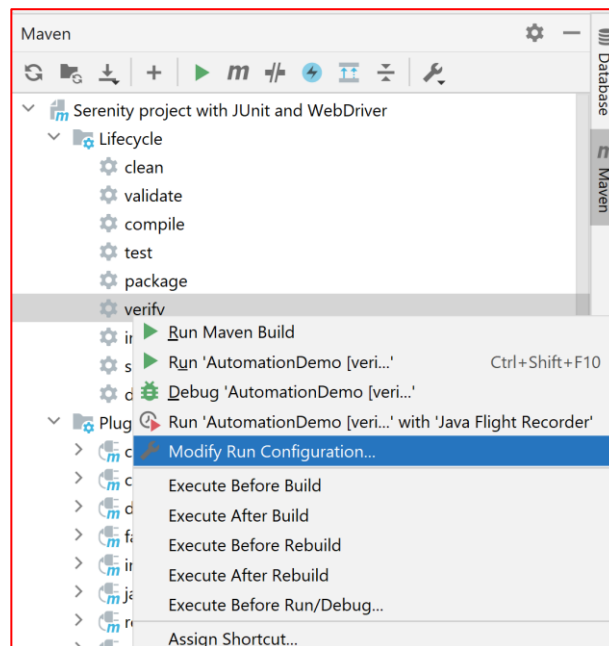


Figure 8. Crearea unei configurații de rulare a testelor în Serenity BDD

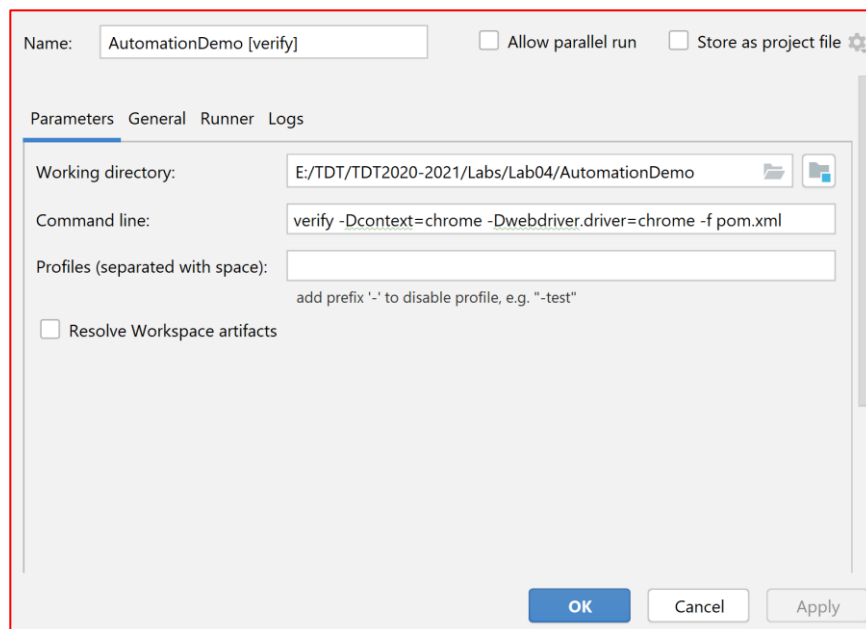


Figure 9. Setarea parametrilor pentru comanda verify, cu setarea browser-ului utilizat la execuție

5. Execuția testelor

Opțiuni de rulare a testelor:

1. click dreapta pe un test sau o clasă de test în **Project Explorer**, e.g., `SearchByKeywordStory` și selectarea opțiunii **SearchByKeywordStory**;
2. în fereastra de comenzi Maven, opțiunea **Run Configurations**, dublu click pe configurația de rulare create, e.g., `AutomationDemo`.

6. Generarea raportului Serenity BDD pentru testele executate

Varianta 1

1. **execuția unui test nu implică și generarea raportului de testare;**
2. din meniul **View** ---> **Tool Windows** ---> **Maven Projects** se deschide fereastra proiectelor gestionate cu Maven (vezi Figure 10);
3. se realizează execuția tuturor testelor folosind comanda **verify** sau execuția individuală a unui test sau clase de teste (vezi Figure 9, Figure 10 din Secțiunea Execuția testelor);
4. din meniul ferestrei proiectelor gestionate cu Maven se alege opțiunea **Execute Maven Goal** (vezi Figure 11);
5. în fereastra de comenzi Maven se completează comanda `mvn serenity:aggregate`, apoi **Execute** (vezi Figure 12);
6. raportul generat va fi salvat în folderul proiectului în `\target\site\serenity`;

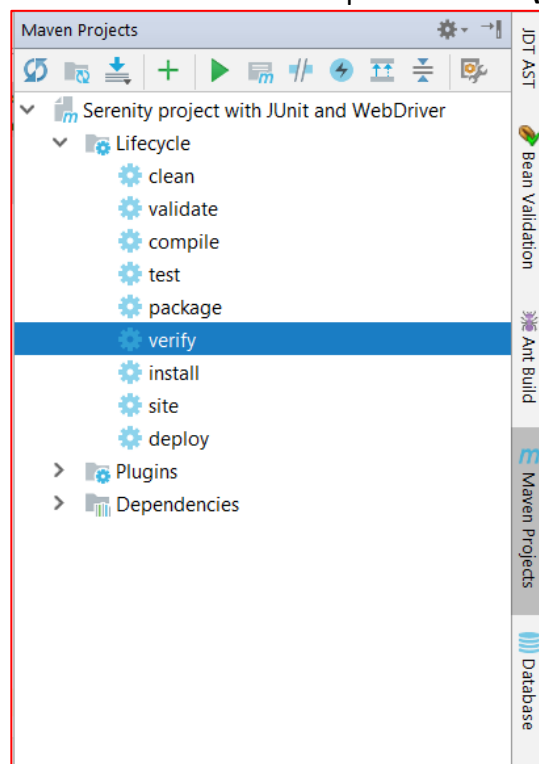


Figure 10. Fereastra Maven Projects și comenzile Maven

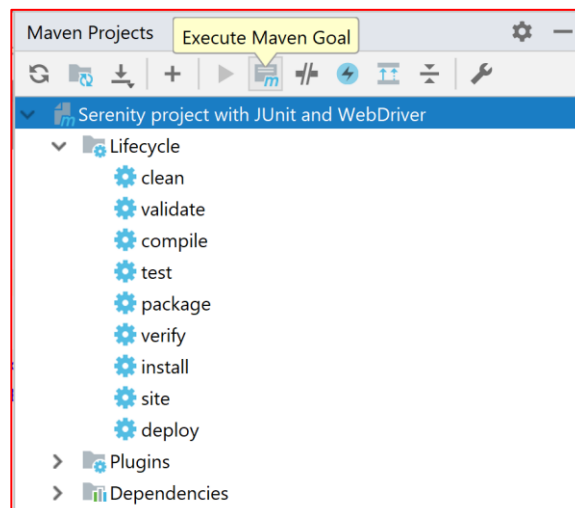


Figure 11. Fereastra Maven Projects și opțiunea Execute Maven Goal

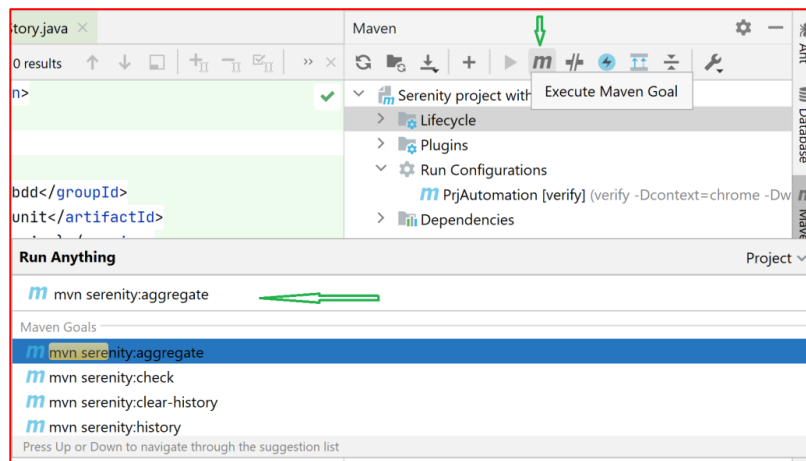


Figure 12. Fereastra de comenzi Maven pentru generarea raportului de testare

Varianta 2

1. **presupune ca Maven sa fie instalat local** și folderul în care Maven este instalat să fie inclus în variabila de mediu **Path**;
2. din meniul **View** ---> **Tool Windows** ---> **Terminal** se deschide fereastra care permite execuția comenzilor din linia de comandă;
3. în această fereastră se execută comanda Maven:
mvn serenity:aggregate <enter> (vezi Figure 13);
4. raportul generat va fi salvat în folderul proiectului în **\target\site\serenity**;

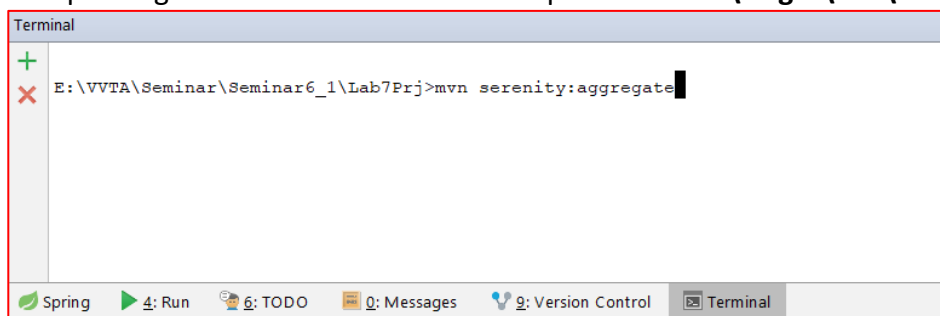


Figure 13. Fereastra Terminal cu execuția comenzii de generare a raportului de testare

7. Vizualizarea raportului Serenity BDD

- din folderul **\target\site\serenity** se încarcă într-un browser web fișierul **index.html** (vezi Figure 14).

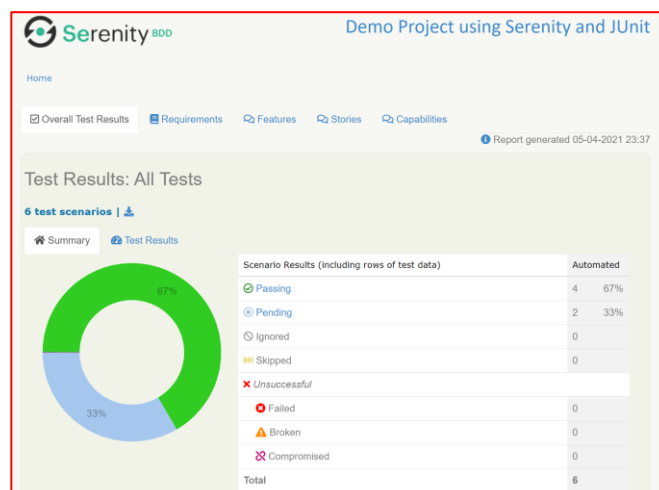
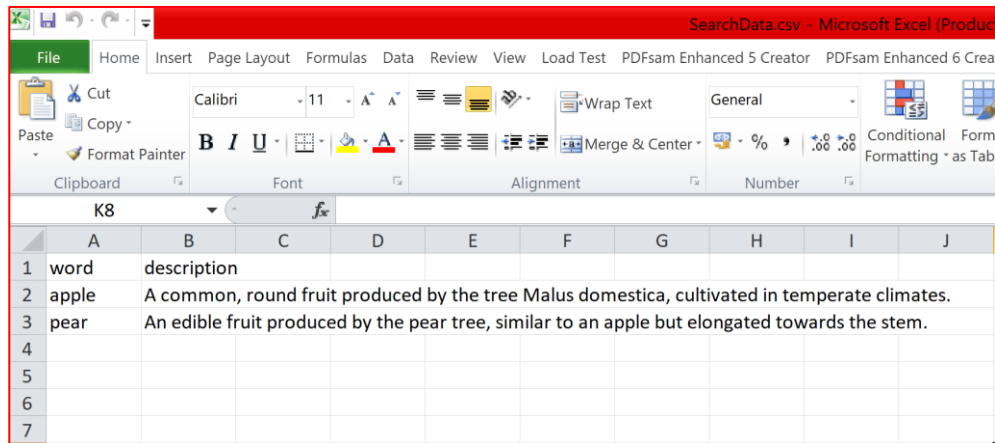


Figure 14 Vizualizarea raportului Serenity BDD

8. Data Driven Testing

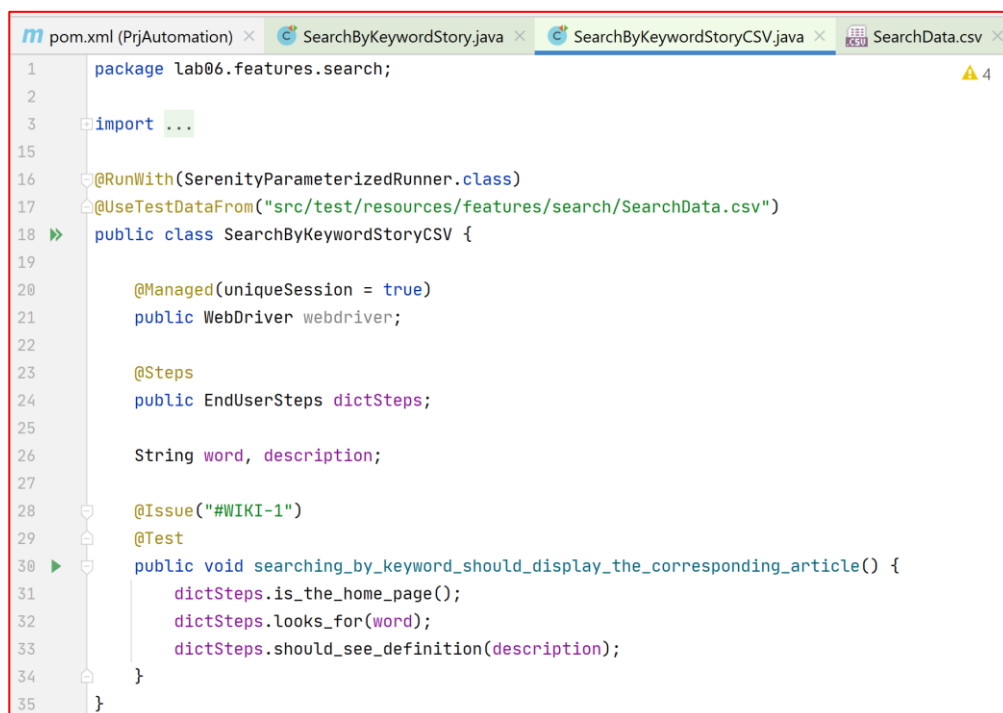
1. se creează fișierul **WikiTestData.csv** cu date de test (vezi Figure 15);
2. prima linie din fișier indică structura tabelului cu date de intrare;
3. următoarele linii conțin date de intrare pentru cazuri de testare individuale;



	A	B	C	D	E	F	G	H	I	J
1	word	description								
2	apple	A common, round fruit produced by the tree Malus domestica, cultivated in temperate climates.								
3	pear	An edible fruit produced by the pear tree, similar to an apple but elongated towards the stem.								
4										
5										
6										
7										

Figure 15 Crearea fișierului cu date de test

4. fișierul **WikiTestData.csv** se salvează în folderul **src/test/resources**;
5. clasa de test va fi executată folosind un *runner parametrizat*;
6. clasa de test reprezintă un șablon de test care va fi executat pentru fiecare înregistrare din fișierul .csv, fișierul fiind dat ca parametru (vezi Figure 16);
7. rularea testelor se realizează ca în Secțiunea 5;
8. generarea raportului Serenity BDD se realizează ca în Secțiunea 6;
9. vizualizarea raportului Serenity BDD indică execuția aceluiași test folosind date de intrare diferite, preluate din fișierul .csv, dat ca parametru (vezi Figure 17).



```

1 package lab06.features.search;
2
3 import ...
4
15
16 @RunWith(SerenityParameterizedRunner.class)
17 @UseTestDataFrom("src/test/resources/features/search/SearchData.csv")
18 public class SearchByKeywordStoryCSV {
19
20     @Managed(uniqueSession = true)
21     public WebDriver webdriver;
22
23     @Steps
24     public EndUserSteps dictSteps;
25
26     String word, description;
27
28     @Issue("#WIKI-1")
29     @Test
30     public void searching_by_keyword_should_display_the_corresponding_article() {
31         dictSteps.is_the_home_page();
32         dictSteps.looks_for(word);
33         dictSteps.should_see_definition(description);
34     }
35 }

```

Figure 16 Clasă de test parametrizată, folosind un fișier .csv

Examples:

#	Word	Description
1	apple	A common, round fruit produced by the tree <i>Malus domestica</i> , cultivated in temperate climates.
2	pear	An edible fruit produced by the pear tree, similar to an apple but elongated towards the stem.













Steps	Screenshots	Outcome	⌚
<div>+</div> Searching by keyword should display the corresponding article	 → 	SUCCESS	22.25s
<div>✓</div> Is the home page		SUCCESS	8.76s
<div>+</div> Looks for: apple	 → 	SUCCESS	11.2s
<div>✓</div> Should see definition: A common, round fruit produced by the tree <i>Malus domestica</i> , cultivated in temperate climates.		SUCCESS	2.3s
<div>+</div> Searching by keyword should display the corresponding article	 → 	SUCCESS	18.22s
<div>✓</div> Is the home page		SUCCESS	14.2s
<div>+</div> Looks for: pear	 → 	SUCCESS	3.59s
<div>✓</div> Should see definition: An edible fruit produced by the pear tree, similar to an apple but elongated towards the stem.		SUCCESS	0.44s
		SUCCESS	40.48s

Figure 17 Raportul Serenity BDD pentru testele din fișierul .csv