accenturetechnology

Behavior Driven Development (BDD)
Day 1 Labs

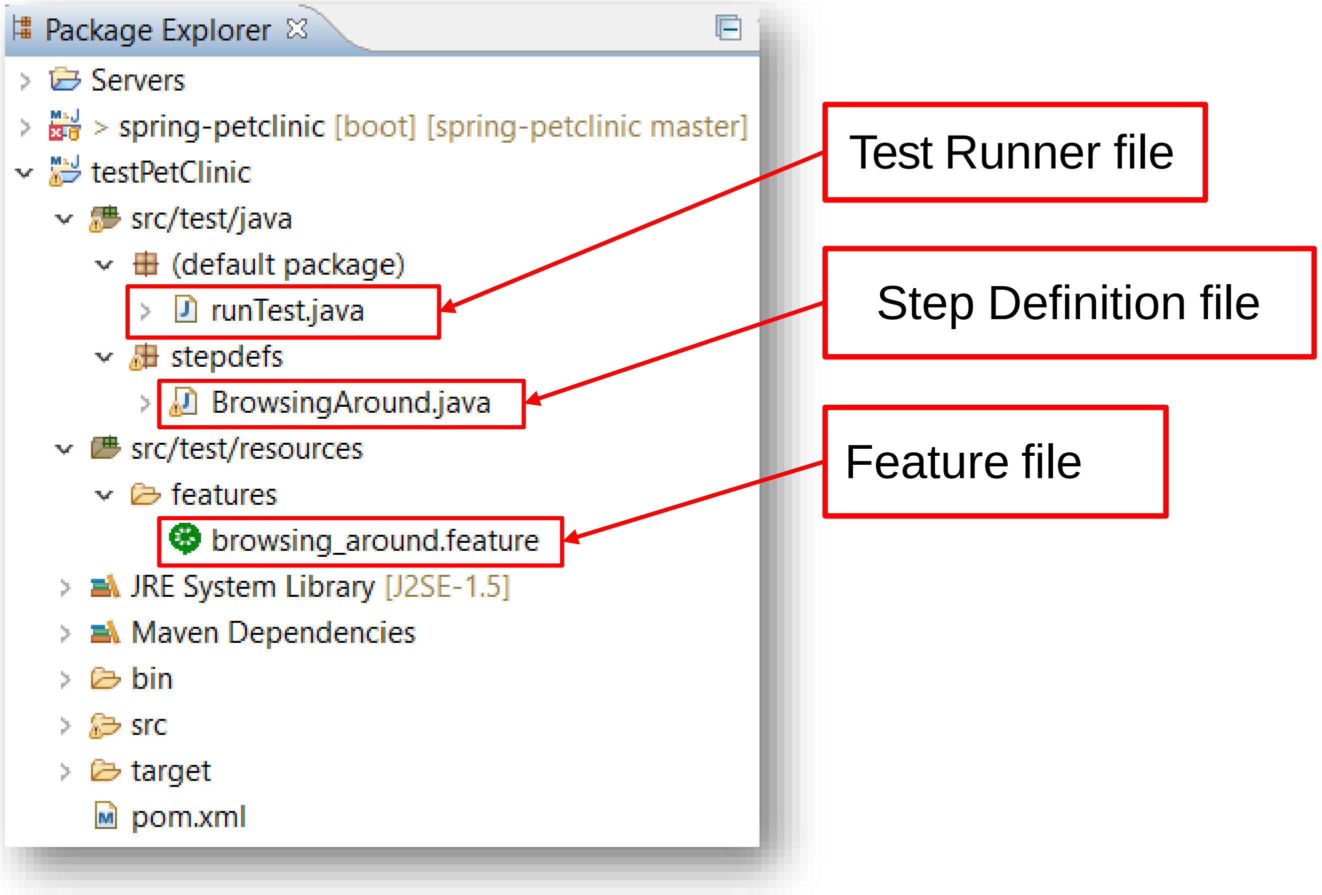
Micro Lab #1 – Explore Feature and Step Definition File

Micro Lab #1 – Exploring Feature and Step Definition Files

Objective: To understand the structure of Feature and Step Definition files

In this quick lab you will:

- Observe the structure of a feature file
- Observe the structure of a step definition file
- Observe the connection between the two
- Observe how both files are coupled to test the Spring Pet Clinic website



Package Explorer

- Servers
- > spring-petclinic [boot] [spring-petclinic master]
- testPetClinic
  - src/test/java
    - (default package)
      - runTest.java
    - stepdefs
      - BrowsingAround.java
  - src/test/resources
    - features
      - browsing_around.feature
  - JRE System Library [J2SE-1.5]
  - Maven Dependencies
  - bin
  - src
  - target
  - pom.xml

Test Runner file

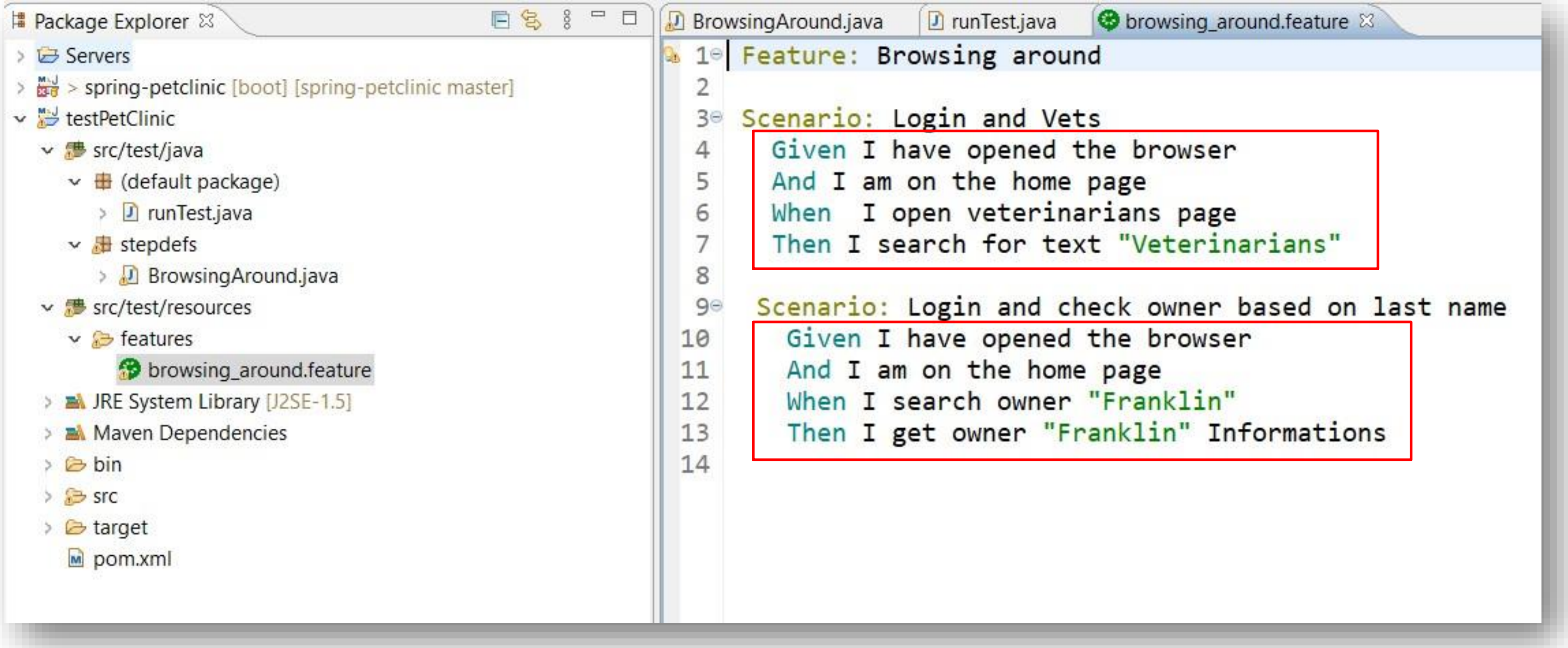
Step Definition file

Feature file

Micro Lab #1 – Exploring Feature and Step Definition Files

Step 1 – Open feature file and observe the Gherkin structure

1. Open file: **browsing_around.feature** and observe the contents of the file
2. Notice the Gherkin structure (Given, When, And, Then)



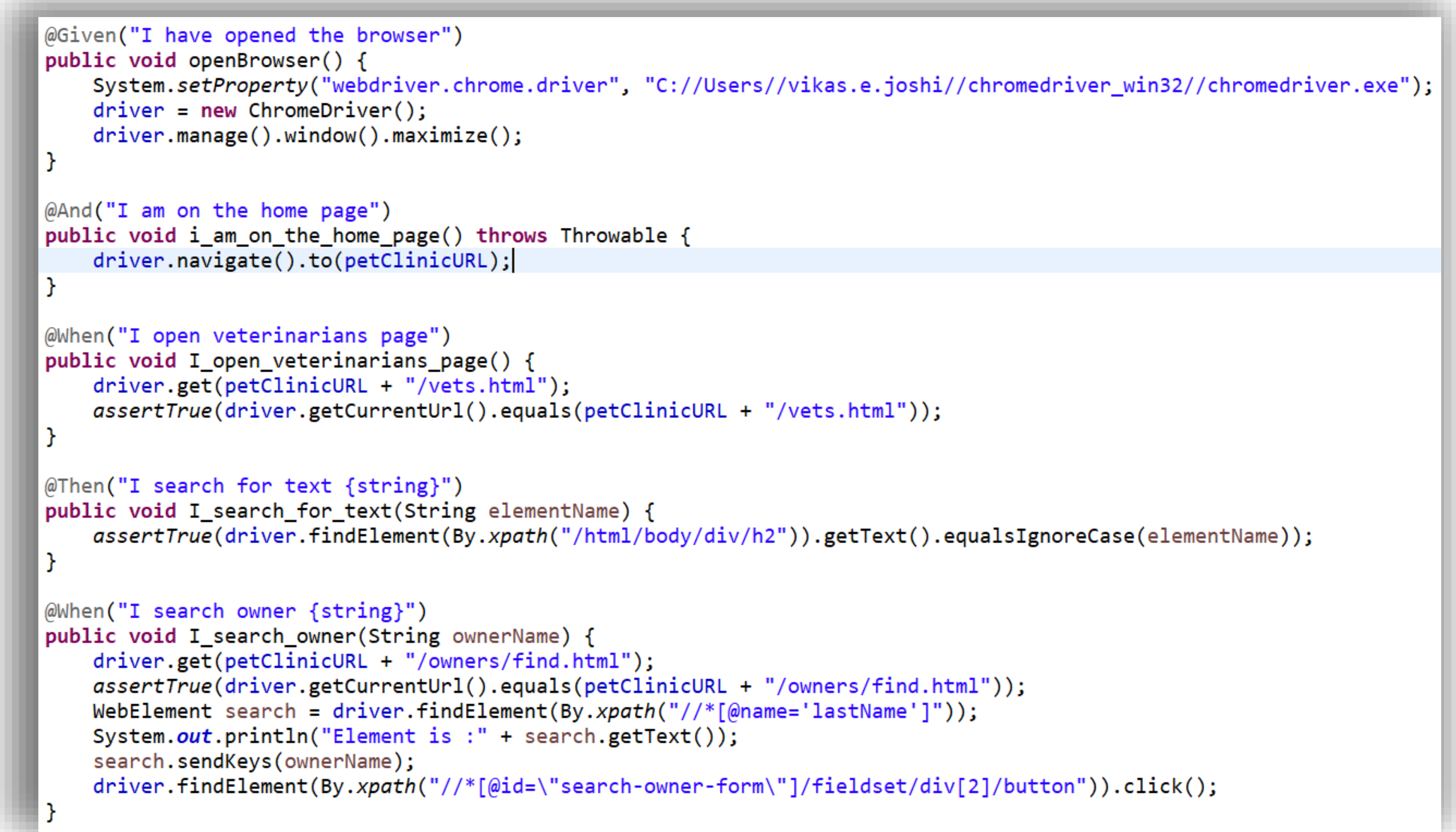
```
Feature: Browsing around

Scenario: Login and Vets
  Given I have opened the browser
  And I am on the home page
  When  I open veterinarians page
  Then I search for text "Veterinarians"

 Scenario: Login and check owner based on last name
  Given I have opened the browser
  And I am on the home page
  When I search owner "Franklin"
  Then I get owner "Franklin" Informations
```

Micro Lab #1 – Exploring Feature and Step Definition Files

Step 2 – Open the step definition file and observe the relation

1. Open file: **BrowsingAround.java** and observe the contents
2. Toggle from the feature back to the step definition file and see if you can catch how they're related

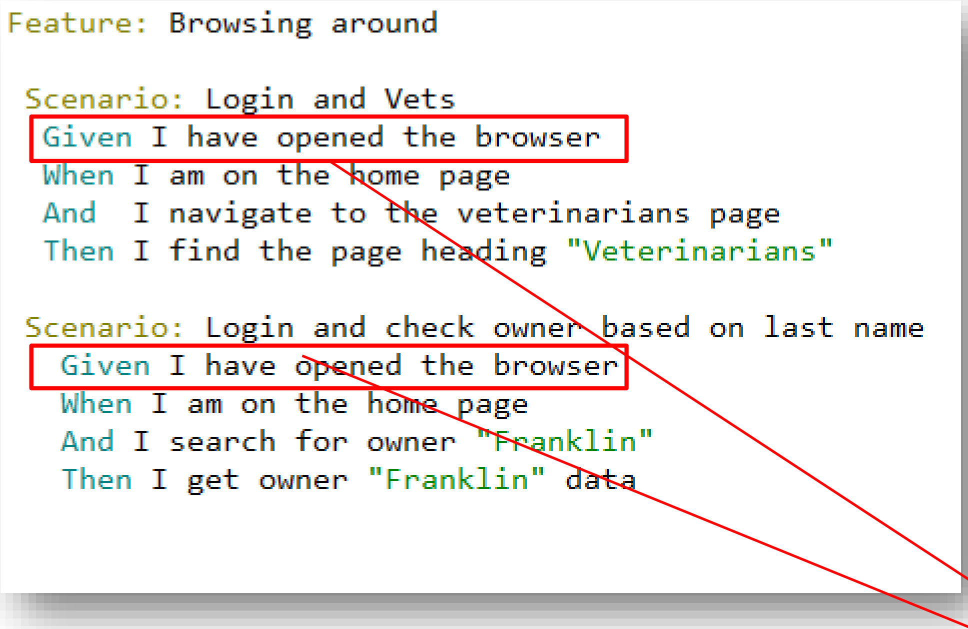
   Hint: look at the annotations "@"



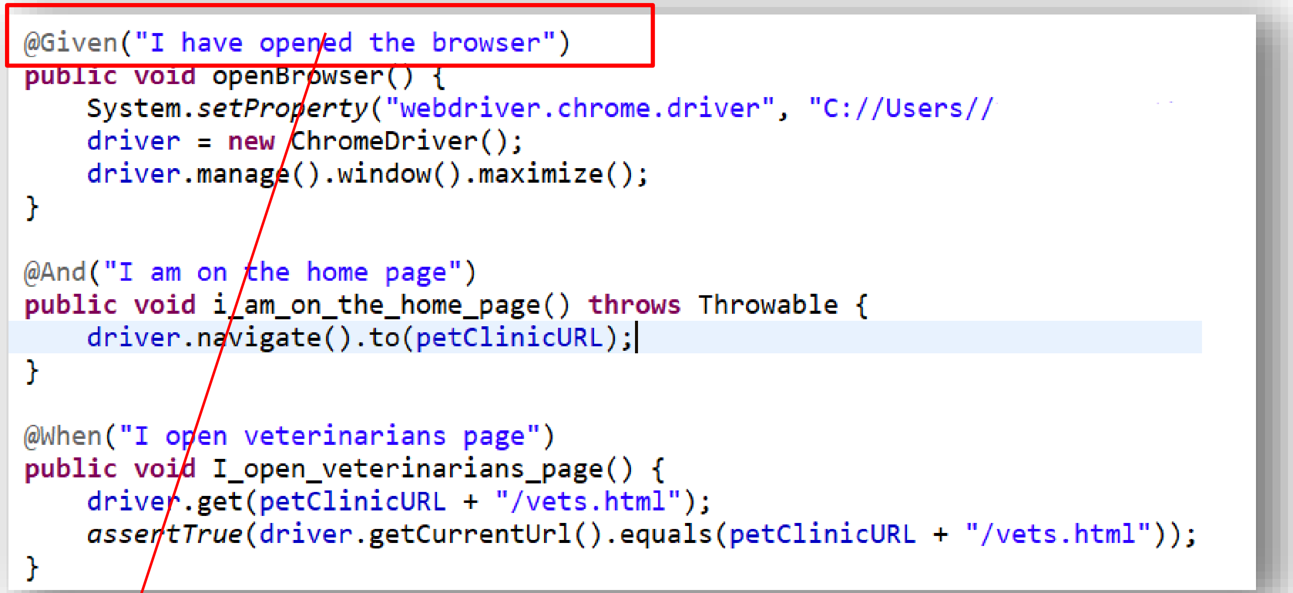
```
@Given("I have opened the browser")
public void openBrowser() {
    System.setProperty("webdriver.chrome.driver", "C://Users//vikas.e.joshi//chromedriver_win32//chromedriver.exe");
    driver = new ChromeDriver();
    driver.manage().window().maximize();
}

@And("I am on the home page")
public void i_am_on_the_home_page() throws Throwable {
    driver.navigate().to(petClinicURL);
}

@When("I open veterinarians page")
public void I_open_veterinarians_page() {
    driver.get(petClinicURL + "/vets.html");
    assertTrue(driver.getCurrentUrl().equals(petClinicURL + "/vets.html"));
}

@Then("I search for text {string}")
public void I_search_for_text(String elementName) {
    assertTrue(driver.findElement(By.xpath("/html/body/div/h2")).getText().equalsIgnoreCase(elementName));
}

@When("I search owner {string}")
public void I_search_owner(String ownerName) {
    driver.get(petClinicURL + "/owners/find.html");
    assertTrue(driver.getCurrentUrl().equals(petClinicURL + "/owners/find.html"));
    WebElement search = driver.findElement(By.xpath("//*[@name='lastName']"));
    System.out.println("Element is :" + search.getText());
    search.sendKeys(ownerName);
    driver.findElement(By.xpath("//*[@id=\"search-owner-form\"]/fieldset/div[2]/button")).click();
}
```

Micro Lab #1 – Exploring Feature and Step Definition Files

Step 3 – Mapping of Step Definitions with Feature



```
Feature: Browsing around

 Scenario: Login and Vets
  Given I have opened the browser
  When I am on the home page
  And  I navigate to the veterinarians page
  Then I find the page heading "Veterinarians"

 Scenario: Login and check owner based on last name
   Given I have opened the browser
   When I am on the home page
   And I search for owner "Franklin"
   Then I get owner "Franklin" data
```



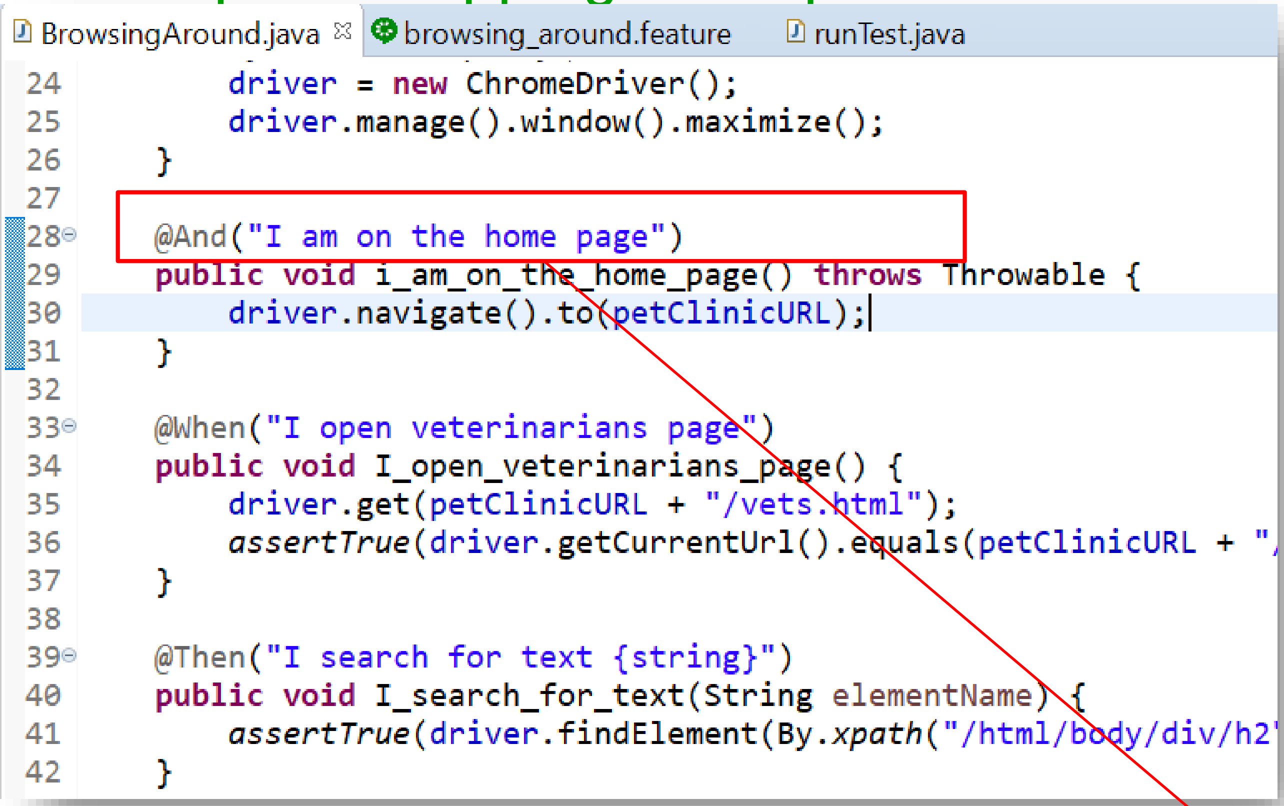
```
@Given("I have opened the browser")
public void openBrowser() {
    System.setProperty("webdriver.chrome.driver", "C://Users//
    driver = new ChromeDriver();
    driver.manage().window().maximize();
}

@And("I am on the home page")
public void i_am_on_the_home_page() throws Throwable {
    driver.navigate().to(petClinicURL);
}

@When("I open veterinarians page")
public void I_open_veterinarians_page() {
    driver.get(petClinicURL + "/vets.html");
    assertTrue(driver.getCurrentUrl().equals(petClinicURL + "/vets.html"));
}
```

Notice the annotation and method name matches the Gherkin "Given" keyword for multiple scenarios

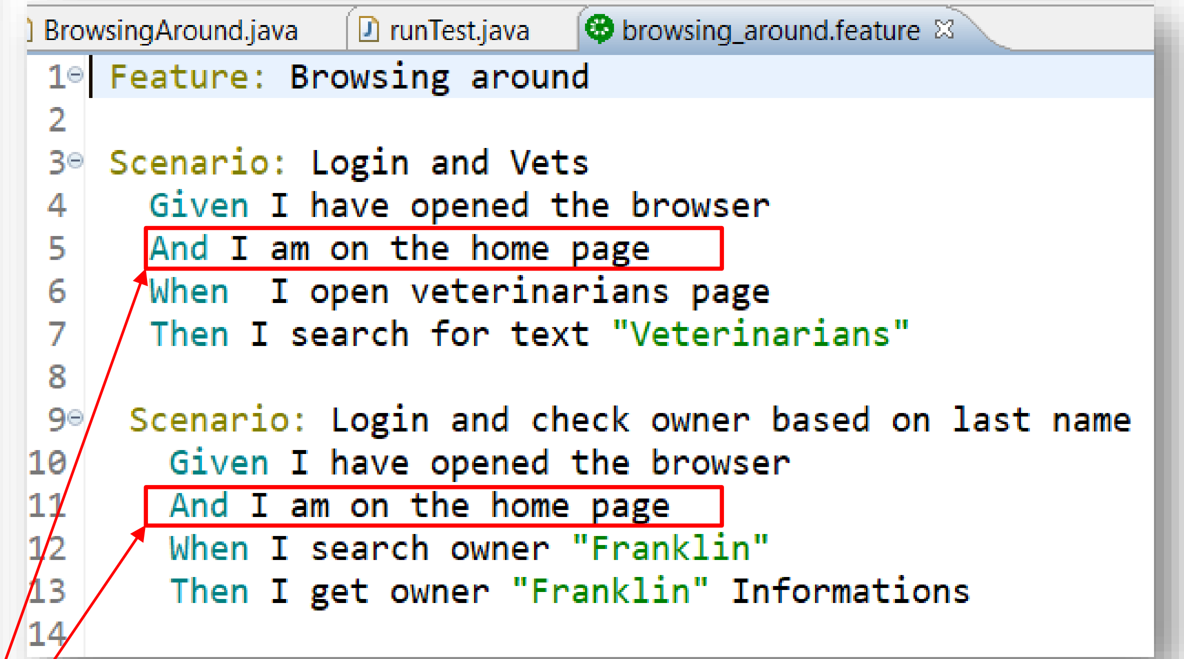
- This method is called anytime "I have opened the browser" is parsed from the feature file

Micro Lab #1 – Exploring Feature and Step Definition Files

Step 3 – Mapping of Step Definitions with Feature



```
        driver = new ChromeDriver();
        driver.manage().window().maximize();
    }

    @And("I am on the home page")
    public void i_am_on_the_home_page() throws Throwable {
        driver.navigate().to(petClinicURL);
    }

    @When("I open veterinarians page")
    public void I_open_veterinarians_page() {
        driver.get(petClinicURL + "/vets.html");
        assertTrue(driver.getCurrentUrl().equals(petClinicURL + "/
    }

    @Then("I search for text {string}")
    public void I_search_for_text(String elementName) {
        assertTrue(driver.findElement(By.xpath("/html/body/div/h2"
    }
```



```
Feature: Browsing around

Scenario: Login and Vets
  Given I have opened the browser
  And I am on the home page
  When  I open veterinarians page
  Then I search for text "Veterinarians"

 Scenario: Login and check owner based on last name
   Given I have opened the browser
   And I am on the home page
   When I search owner "Franklin"
   Then I get owner "Franklin" Informations
```

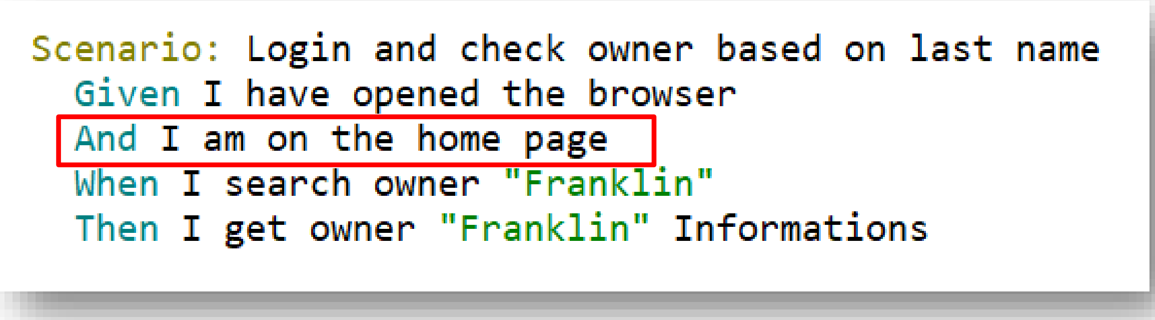
Notice the method title is the same text as in the feature file.

- This method is called anytime “I am on the home page” occurs in the feature file.

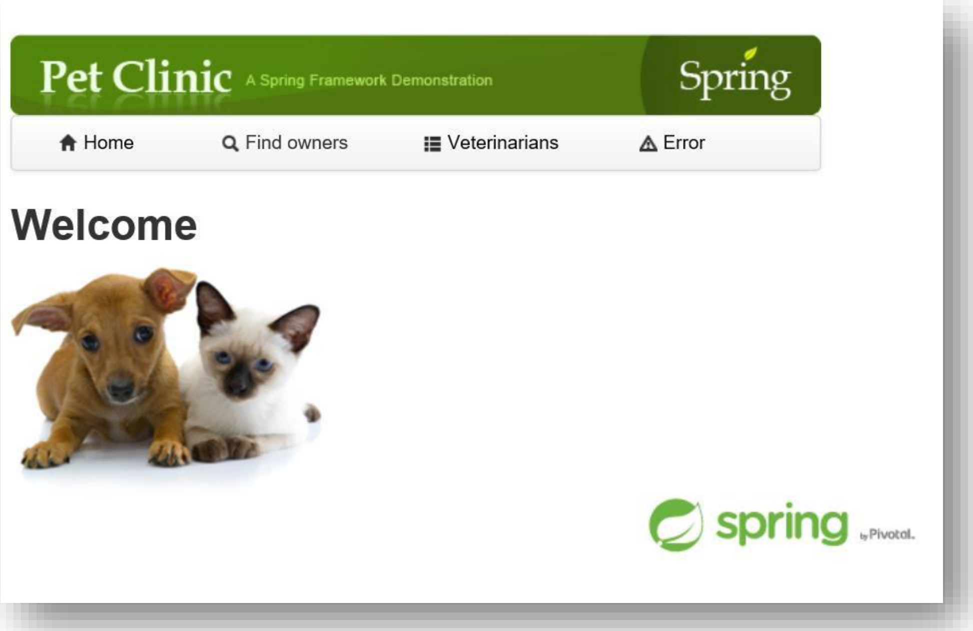
Micro Lab #1 – Exploring Feature and Step Definition Files

Step 4 – How does it integrate to test the Pet Clinic website?

1. Using the scenario: “Login and check owner based on last name”
2. Observe where the “**And**” statement is executed



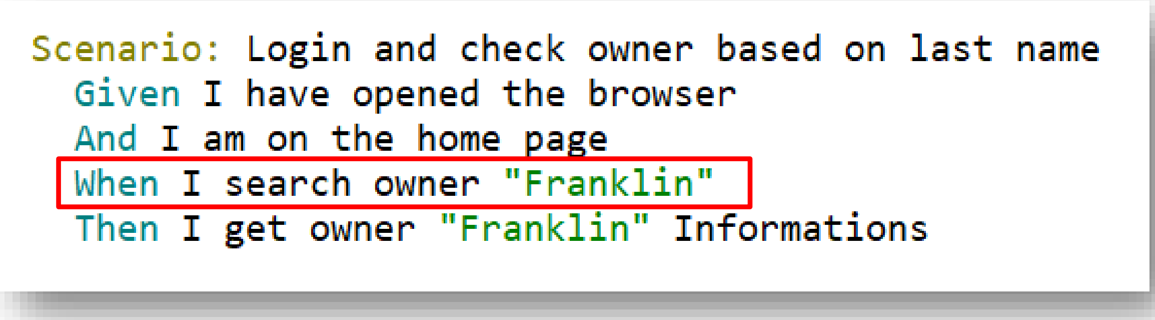
```
Scenario: Login and check owner based on last name
  Given I have opened the browser
  And I am on the home page
  When I search owner "Franklin"
  Then I get owner "Franklin" Informations
```



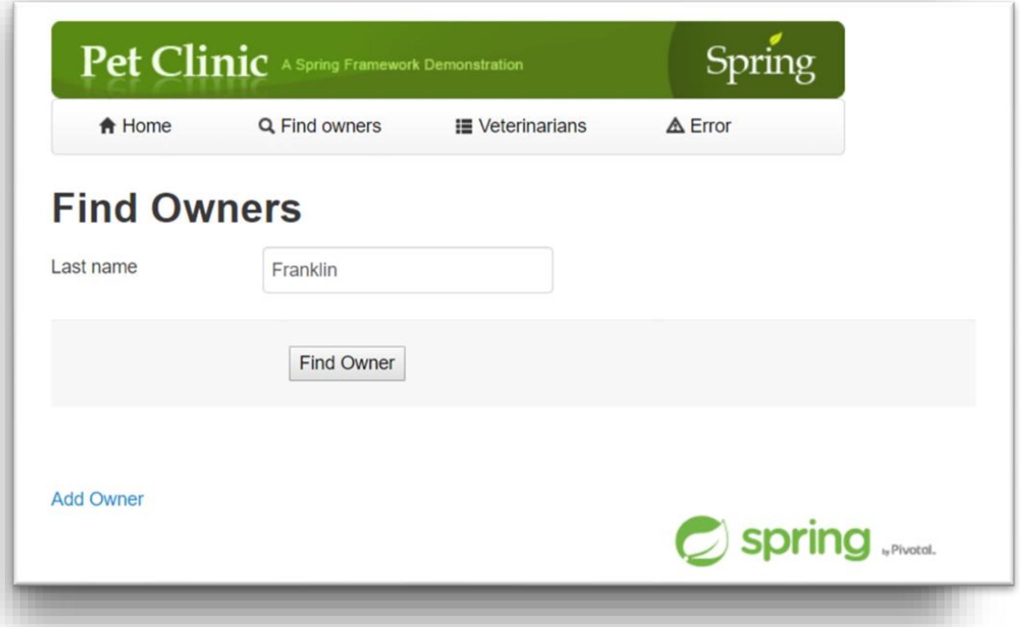
Micro Lab #1 – Exploring Feature and Step Definition Files

Step 4 – How does it integrate to test the Pet Clinic website?

1. Using the scenario: “Login and check owner based on last name”
2. Observe where the “**When**” statement is executed



```
Scenario: Login and check owner based on last name
  Given I have opened the browser
  And I am on the home page
  When I search owner "Franklin"
  Then I get owner "Franklin" Informations
```



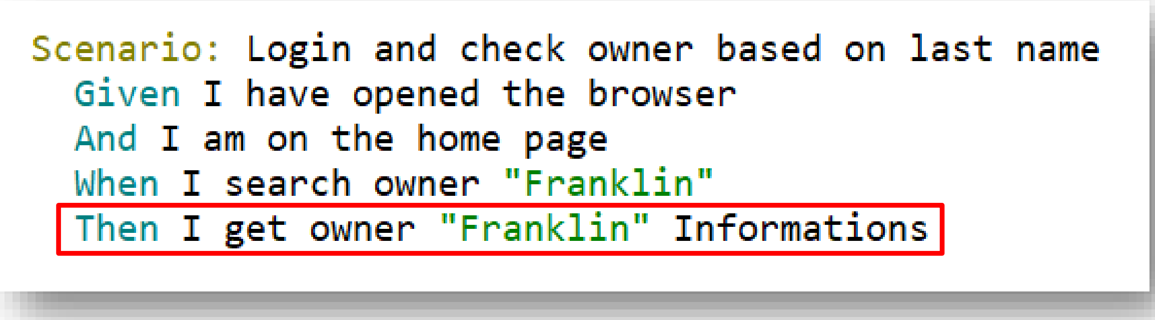
Find Owners

Last name: Franklin

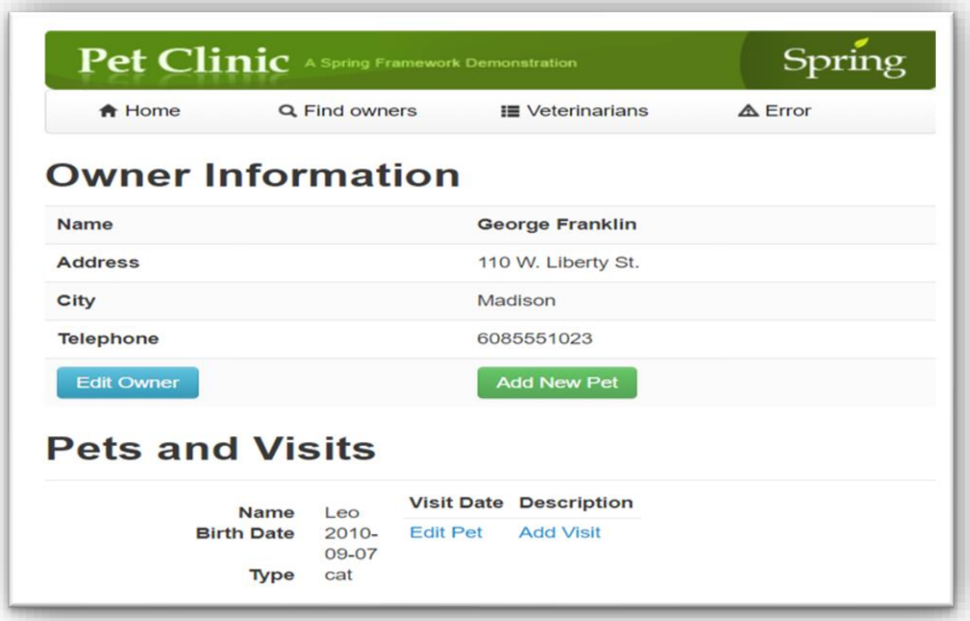
Micro Lab #1 – Exploring Feature and Step Definition Files

Step 4 – How does it integrate to test the Pet Clinic website?

1. Using the scenario: “Login and check owner based on last name”
2. Observe where the “**Then**” statement is executed



```
Scenario: Login and check owner based on last name
  Given I have opened the browser
  And I am on the home page
  When I search owner "Franklin"
  Then I get owner "Franklin" Informations
```



Owner Information

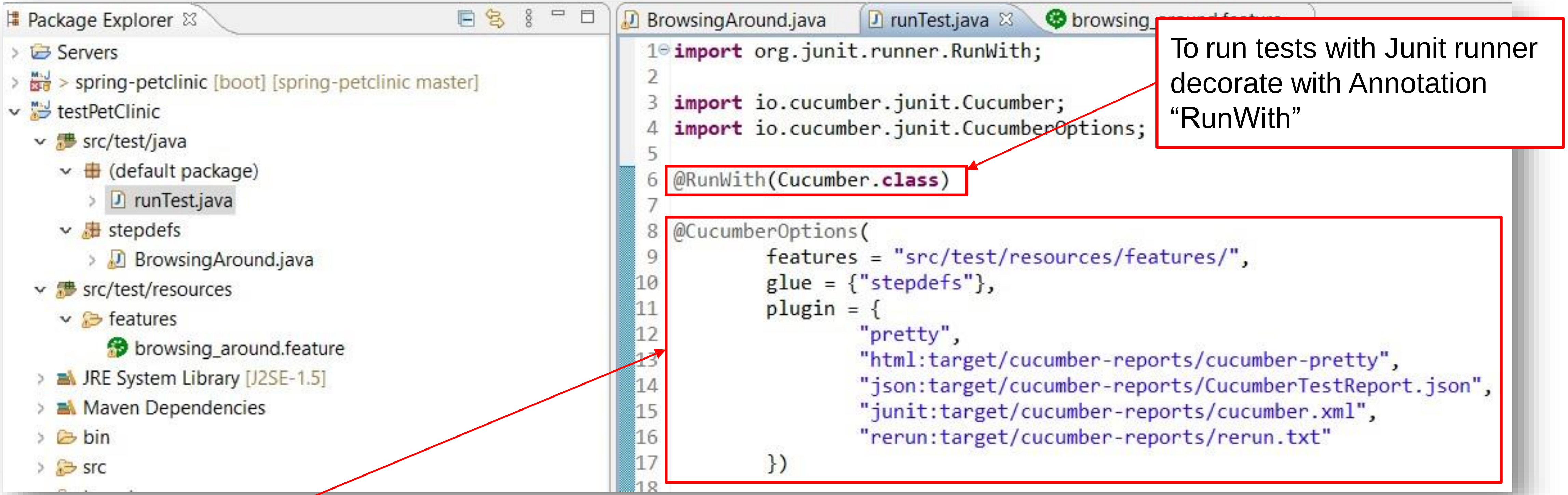
Name	George Franklin
Address	110 W. Liberty St.
City	Madison
Telephone	6085551023

Pets and Visits

		Visit Date	Description
Name	Leo		
Birth Date	2010-09-07		
Type	cat		

Micro Lab #1 – Exploring Feature and Step Definition Files

Step 5 – Observe the Test Runner file



```
import org.junit.runner.RunWith;

import io.cucumber.junit.Cucumber;
import io.cucumber.junit.CucumberOptions;

@RunWith(Cucumber.class)

@CucumberOptions(
        features = "src/test/resources/features/",
        glue = {"stepdefs"},
        plugin = {
                "pretty",
                "html:target/cucumber-reports/cucumber-pretty",
                "json:target/cucumber-reports/CucumberTestReport.json",
                "junit:target/cucumber-reports/cucumber.xml",
                "rerun:target/cucumber-reports/rerun.txt"
        })
```

To run tests with Junit runner decorate with Annotation “RunWith”

Using Annotation “CucumberOptions”:

i. Path to feature files can be defined using “features”

ii. Feature files can be glued (mapped) with Step Definitions using “glue”.

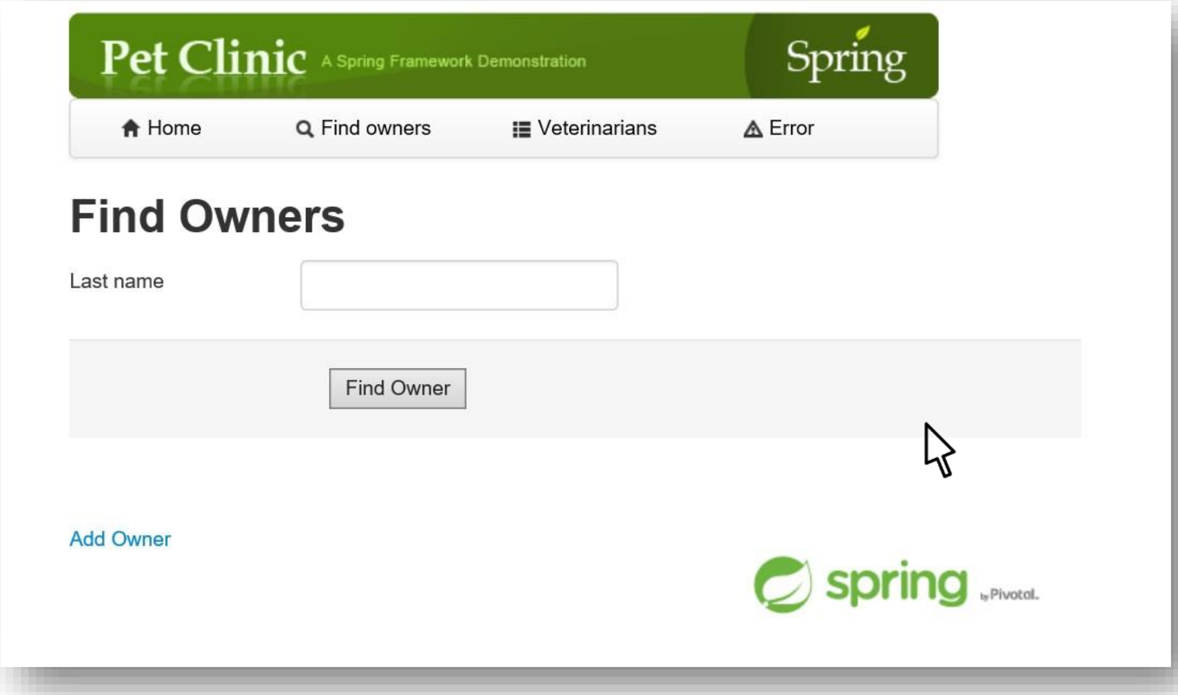
iii. Reports can be generated in different formats. E.g., Json, html or xml using “plugin”.

Micro Lab #2 – Run the Cucumber Test

Micro Lab #2 – Run the Cucumber Test

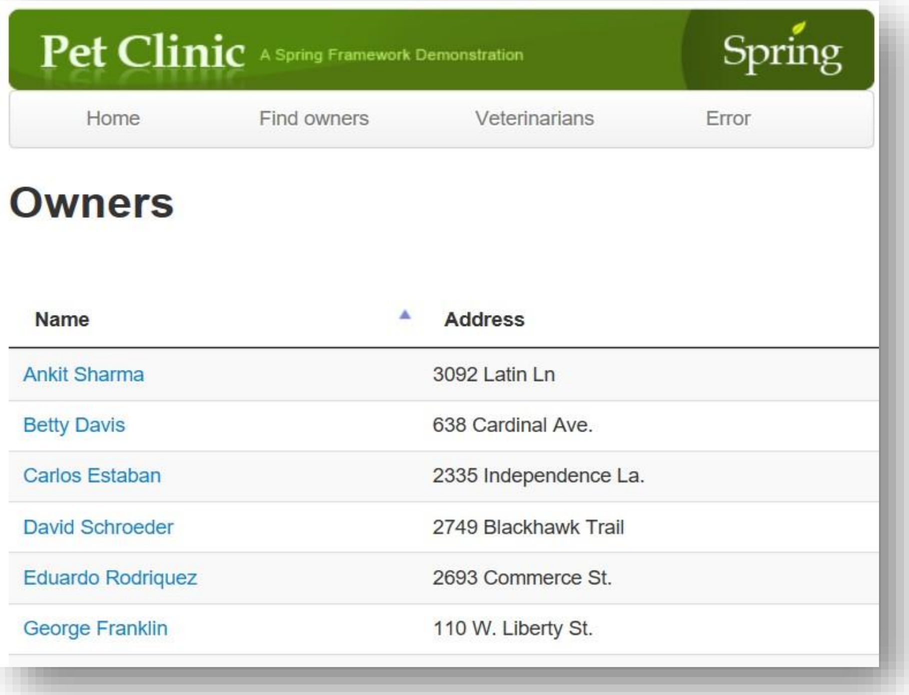
Step 1 – Explore the “Browsing Around” feature functionality

1. View all existing pet owners by submitting the “Find Owners” form with an empty string
2. This will present a list of all pet owners stored in the database
3. Choose any existing last name and replace it within your feature file (see next slide)



Find Owners

Last name



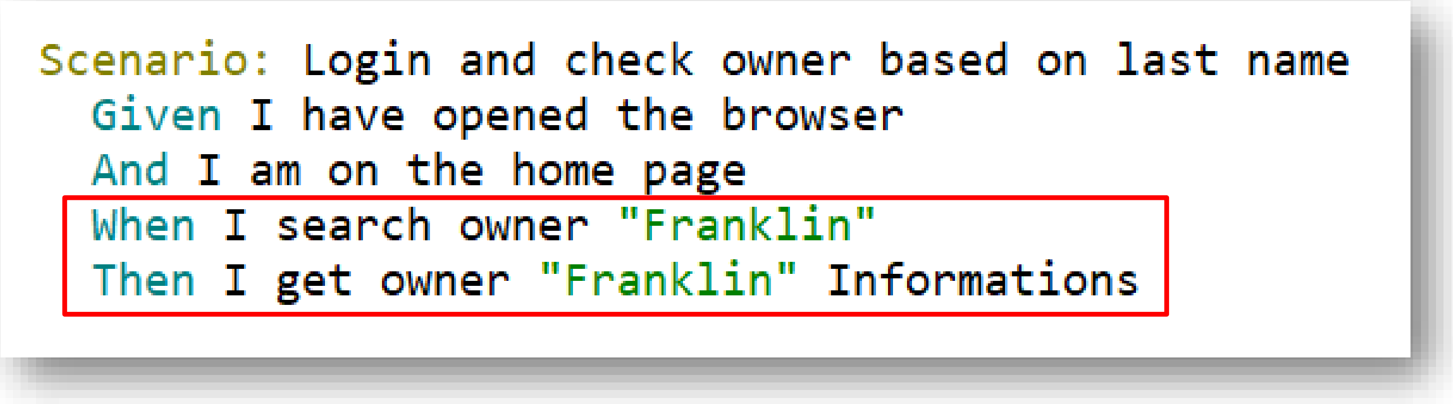
Owners

Name	Address
Ankit Sharma	3092 Latin Ln
Betty Davis	638 Cardinal Ave.
Carlos Estaban	2335 Independence La.
David Schroeder	2749 Blackhawk Trail
Eduardo Rodriquez	2693 Commerce St.
George Franklin	110 W. Liberty St.

Micro Lab #2 – Run the Cucumber Test

Step 1 cont. – Explore the “Browsing Around” feature functionality

1. Our screenshots will keep the pet owners last name “Franklin”, but feel free to select a different one



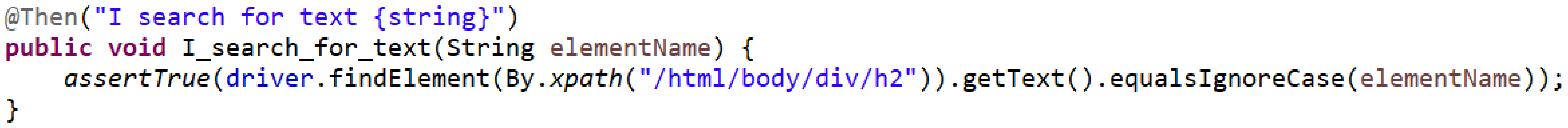
```
Scenario: Login and check owner based on last name
  Given I have opened the browser
  And I am on the home page
  When I search owner "Franklin"
  Then I get owner "Franklin" Informations
```

2. Within your Eclipse workspace, right-click the testPetClinic project, in the menu, select “Run As” > “Maven Test”
3. After a slight pause (10 secs to 1 minute), two browser should open executing a scenario within each. Output within your Eclipse console should also indicate all tests passed
4. Our test succeeded – do we understand how?

Micro Lab #2 – Run the Cucumber Test

Step 2 – Learn by analyzing a different method

Return to the “BrowsingAround.java” file and inspect a different method to get a better idea

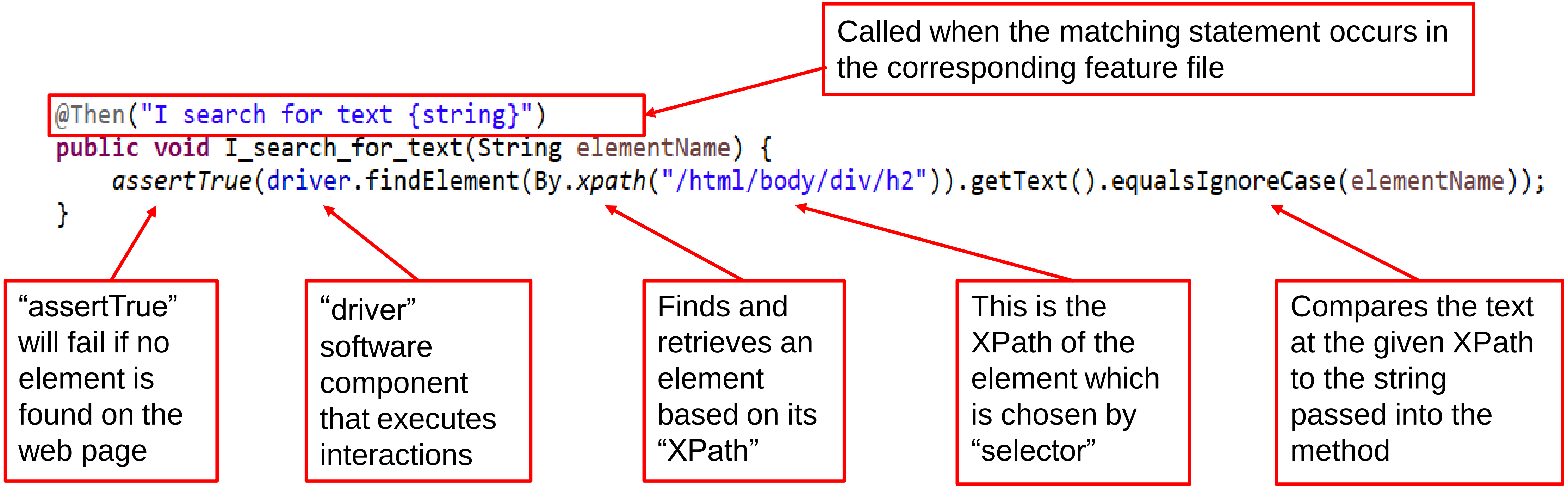


```
@Then("I search for text {string}")
public void I_search_for_text(String elementName) {
    assertTrue(driver.findElement(By.xpath("/html/body/div/h2")).getText().equalsIgnoreCase(elementName));
}
```

Micro Lab #2 – Run the Cucumber Test

Step 2 – Learn by analyzing a different method

Return to the “BrowsingAround.java” file and inspect a different method to get a better idea



Called when the matching statement occurs in the corresponding feature file

```
@Then("I search for text {string}")
public void I_search_for_text(String elementName) {
    assertTrue(driver.findElement(By.xpath("/html/body/div/h2")).getText().equalsIgnoreCase(elementName));
}
```

“assertTrue” will fail if no element is found on the web page

“driver” software component that executes interactions

Finds and retrieves an element based on its “XPath”

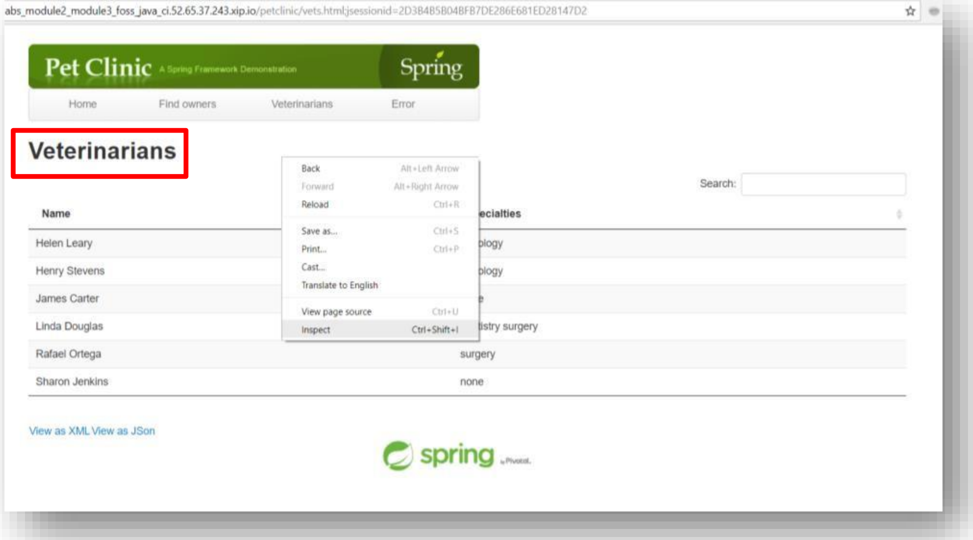
This is the XPath of the element which is chosen by “selector”

Compares the text at the given XPath to the string passed into the method

Micro Lab #2 – Run the Cucumber Test

Step 3 – Retrieving the XPath

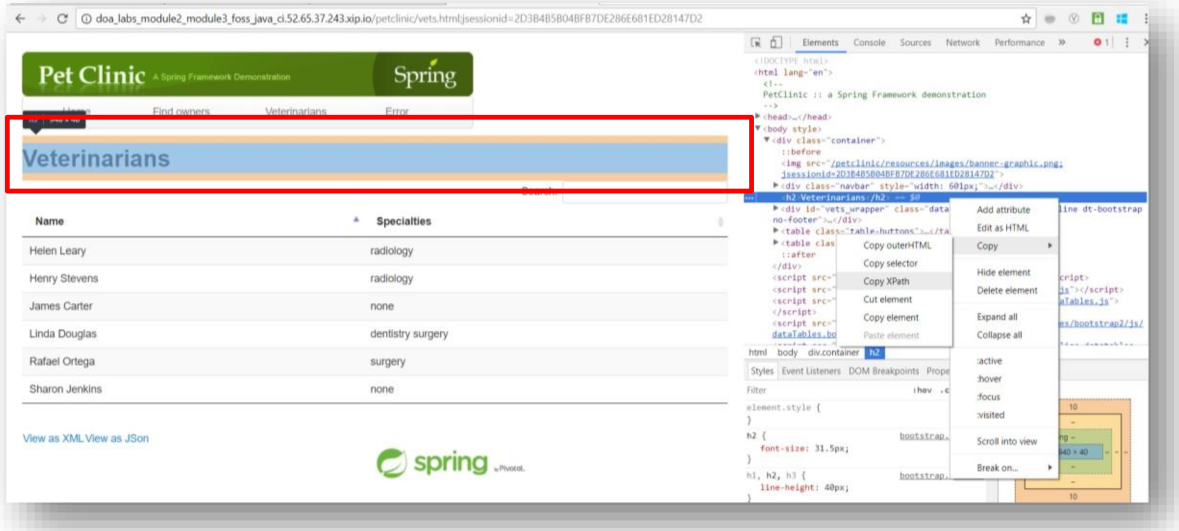
1. For the next step, ensure you’re using the Chrome browser to use the Chrome inspect tool
2. Navigate to the “Veterinarians” page, right click the “Veterinarians” title and select “inspect”
3. Right click the already highlighted text and select “Copy” > “Copy Xpath”
4. Paste the XPath into a text editor of your choice, i.e. Notepad. It should read: “/html/body/div/h2.”
   - Save this text for later use.
5. Observe how the XPath from the Pet Clinic matches the XPath in “**browsing_around.java**” file
6. See next slide for an enlarged imaged



abs_module2_module3_foss_java_ci.52.65.37.243.xip.io/petclinic/vets.html;jsessionid=2D3B4B5B04BFB7DE286E681ED28147D2

Veterinarians

Name	ecialties
Helen Leary	ology
Henry Stevens	ology
James Carter	e
Linda Douglas	istry surgery
Rafael Ortega	surgery
Sharon Jenkins	none



doa_labs_module2_module3_foss_java_ci.52.65.37.243.xip.io/petclinic/vets.html;jsessionid=2D3B4B5B04BFB7DE286E681ED28147D2

Veterinarians

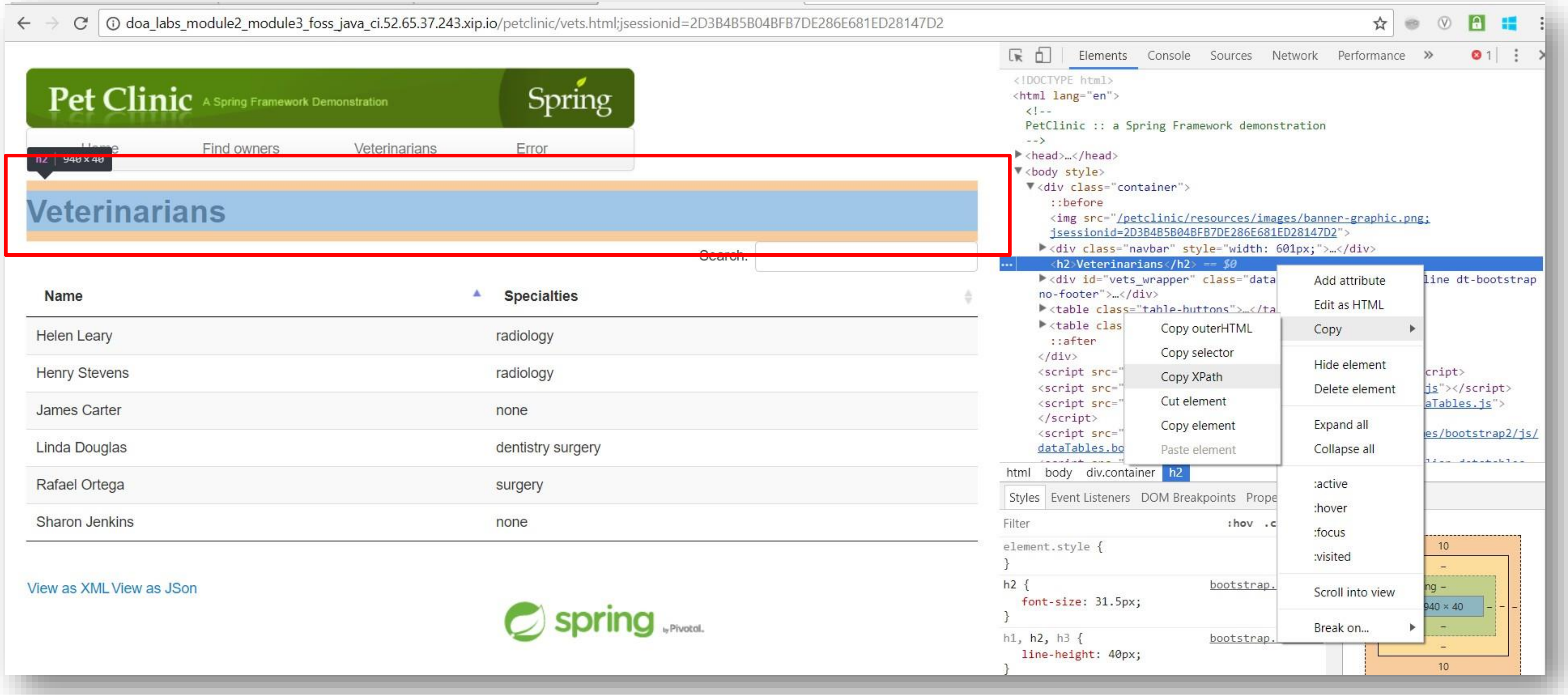
Name	Specialties
Helen Leary	radiology
Henry Stevens	radiology
James Carter	none
Linda Douglas	dentistry surgery
Rafael Ortega	surgery
Sharon Jenkins	none

```
<!DOCTYPE html>
<html lang="en">
  <!--
  PetClinic :: a Spring Framework demonstration
  -->
  <head>…</head>
  <body style>
    <div class="container">
      ::before
      <img src="/petclinic/resources/images/banner-graphic.png;jsessionid=2D3B4B5B04BFB7DE286E681ED28147D2">
      <div class="navbar" style="width: 601px;">…</div>
      <h2>Veterinarians</h2> == $0
```

```
element.style {
}
h2 {
  font-size: 31.5px;
}
h1, h2, h3 {
  line-height: 40px;
}
```

Micro Lab #2 – Run the Cucumber Test

Step 3 cont. – Retrieving the XPath



doa_labs_module2_module3_foss_java_ci.52.65.37.243.xip.io/petclinic/vets.html;jsessionid=2D3B4B5B04BFB7DE286E681ED28147D2

Veterinarians

Name	Specialties
Helen Leary	radiology
Henry Stevens	radiology
James Carter	none
Linda Douglas	dentistry surgery
Rafael Ortega	surgery
Sharon Jenkins	none

```
<!DOCTYPE html>
<html lang="en">
  <!--
  PetClinic :: a Spring Framework demonstration
  -->
▶<head>…</head>
▼<body style>
  ▼<div class="container">
      ::before
      <img src="/petclinic/resources/images/banner-graphic.png;
      jsessionid=2D3B4B5B04BFB7DE286E681ED28147D2">
    ▶<div class="navbar" style="width: 601px;">…</div>
      <h2>Veterinarians</h2> == $0
    ▶<div id="vets_wrapper" class="data                  line dt-bootstrap
    no-footer">…</div>
    ▶<table class="table-buttons">…</ta
    ▶<table clas
      ::after
    </div>
    <script src="                                        cript>
    <script src="                                        js"></script>
    <script src="                                        aTables.js">
    </script>
    <script src="                                        es/bootstrap2/js/
    dataTables.bo
```

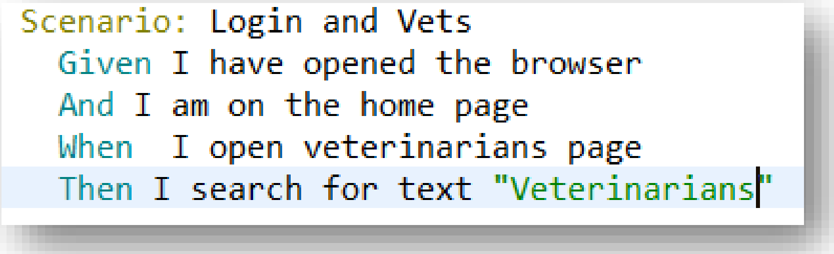
```
element.style {
}
h2 {                                  bootstrap.
  font-size: 31.5px;
}
h1, h2, h3 {                          bootstrap.
  line-height: 40px;
}
```

* Enlarged for reference

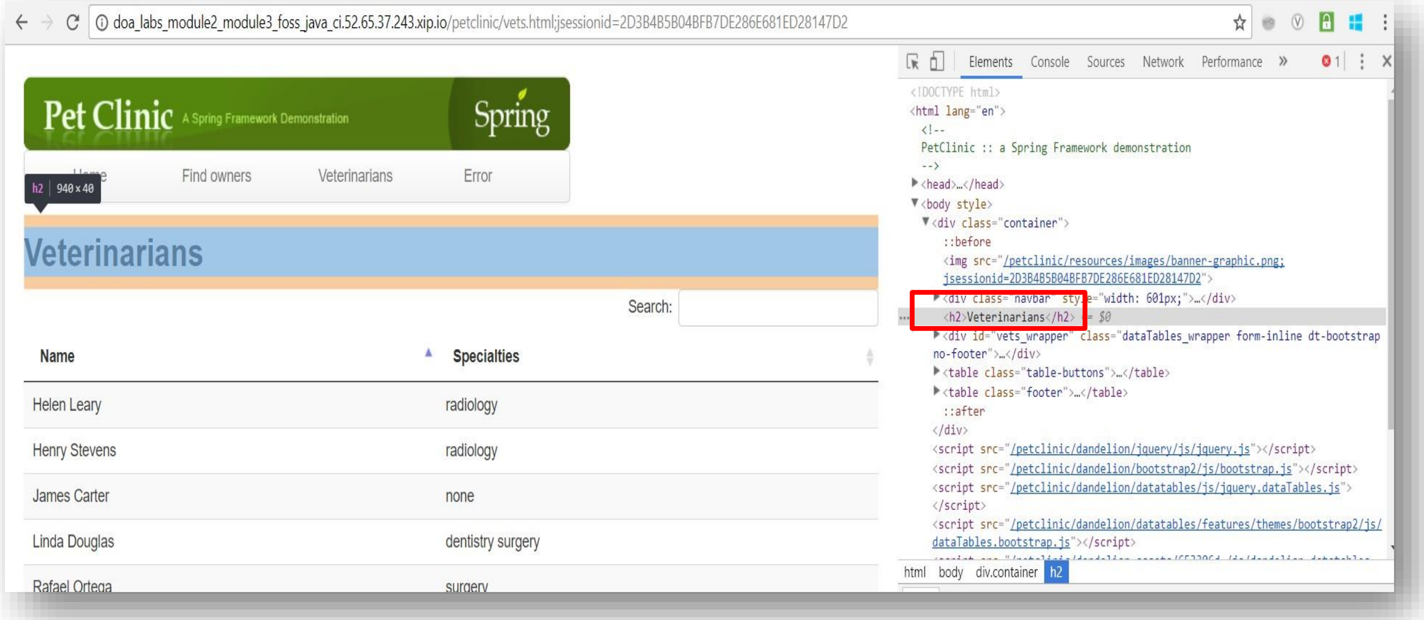
Micro Lab #2 – Run the Cucumber Test

Step 4 – How XPath is used within the Test

1. This XPath within our Step definition file directs the test to the element with the matching XPath on the current page. Each element on a given page has a different unique XPath.
2. The method then gets the text within that element and compares it to the string argument “arg1”.
3. In the “browsing_around.feature” feature file, the string “Veterinarians” is passed into the method and checked to see if it matches the string on the Pet Clinic website within the h2 header with the Xpath “/html/body/div/h2”
4. They match, so the test passes



```
Scenario: Login and Vets
  Given I have opened the browser
  And I am on the home page
  When  I open veterinarians page
  Then I search for text "Veterinarians"
```



doa_labs_module2_module3_foss_java_ci.52.65.37.243.xip.io/petclinic/vets.html;jsessionid=2D3B4B5B04BFB7DE286E681ED28147D2

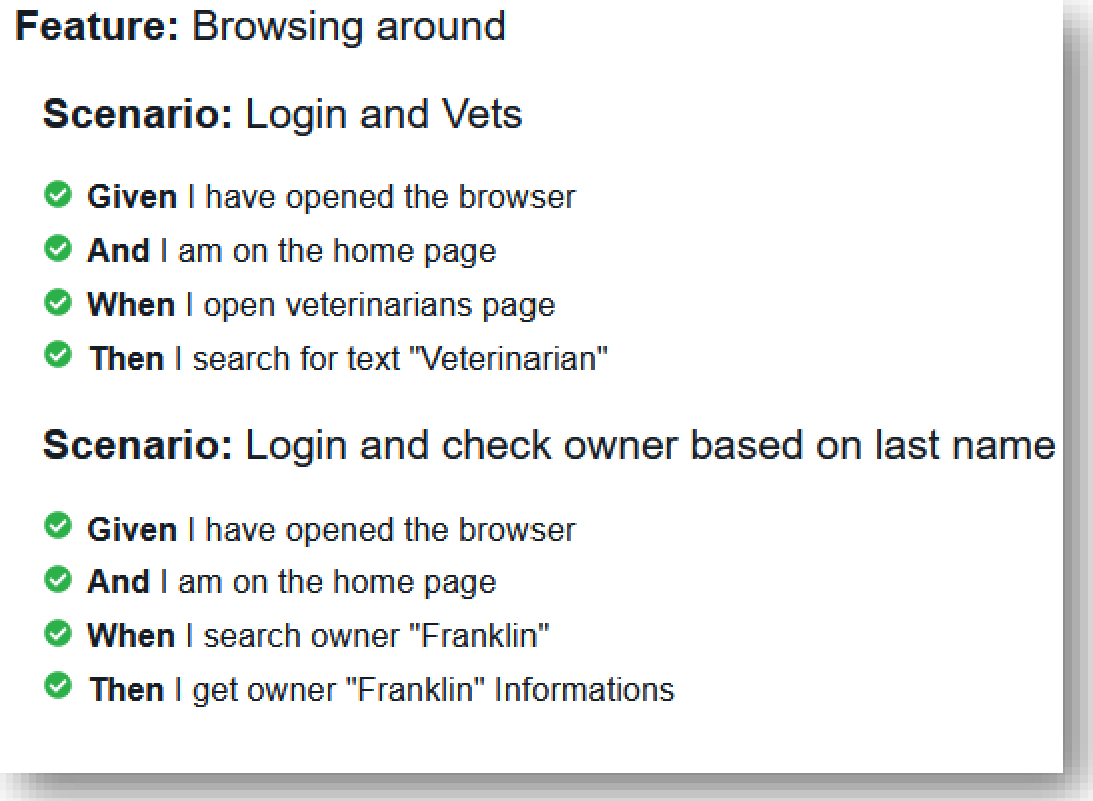
Name	Specialties
Helen Leary	radiology
Henry Stevens	radiology
James Carter	none
Linda Douglas	dentistry surgery
Rafael Ortega	surgery

```
<!DOCTYPE html>
<html lang="en">
  <!--
  PetClinic :: a Spring Framework demonstration
  -->
▶<head>…</head>
▼<body style>
  ▼<div class="container">
      ::before
      <img src="/petclinic/resources/images/banner-graphic.png;
      jsessionid=2D3B4B5B04BFB7DE286E681ED28147D2">
    ▶<div class="navbar" style="width: 601px;">…</div>
...   <h2>Veterinarians</h2> == $0
    ▶<div id="vets_wrapper" class="dataTables_wrapper form-inline dt-bootstrap
    no-footer">…</div>
    ▶<table class="table-buttons">…</table>
    ▶<table class="footer">…</table>
      ::after
    </div>
    <script src="/petclinic/dandelion/jquery/js/jquery.js"></script>
    <script src="/petclinic/dandelion/bootstrap2/js/bootstrap.js"></script>
    <script src="/petclinic/dandelion/datatables/js/jquery.dataTables.js">
    </script>
    <script src="/petclinic/dandelion/datatables/features/themes/bootstrap2/js/
    dataTables.bootstrap.js"></script>
```

Micro Lab #2 – Run and learn how it works

Step 5 – Open the Cucumber report

1. Using Windows Explorer navigate to the **cucumber-pretty.html** file on your workstation, located at *{workspace_location}*\target\cucumber-reports and open the cucumber-pretty.html file
2. You should see the below Cucumber report



Feature: Browsing around

Scenario: Login and Vets

✓ **Given** I have opened the browser

✓ **And** I am on the home page

✓ **When** I open veterinarians page

✓ **Then** I search for text "Veterinarian"

Scenario: Login and check owner based on last name

✓ **Given** I have opened the browser

✓ **And** I am on the home page

✓ **When** I search owner "Franklin"

✓ **Then** I get owner "Franklin" Informations

Lab #1 – Add a New Feature and Step definition

Lab #1 - Add a new feature and step definition

Step 1: Test Business Requirement - Create A New Owner Scenario

1. As a team, review the given JIRA user story and construct the new scenario using Gherkin syntax

JIRA-1234: Vets can add a new owner

Feature: Vets can add a new pet owner
  As a Veterinarian
  I Want To add a new pet owner to the Pet Clinic site
  So That I can easily retrieve the pet owners information

Gherkin Template
Scenario: Add new owner
  Given ???
  And ???
  When ???
  Then ???

2. Save your new scenario in the existing **browsing_around.feature** file or create a new file and call it **find_owner.feature**
3. Run the Test: Right-click testPetClinic project, select “Run As” > “Maven Test”

Lab #1 - Add a new feature and step definition

Step 2 – Add new scenario

```
Scenario: vets can add new pet owner
    Given I have opened the browser
    And I am on add New Owner form
    When I submit pet owner information as "Justin" "Bieber" "Hard Rock Cafe" "Ontario" "10183257"
    Then I see newly added pet owner information
```

* Example code provided. Yours will differ based on the Gherkin documentation your team came up with

Lab #1 - Add a new feature and step definition

Step 3 – Review Eclipse Console Output

1. Your new test will not run because you don’t have the step definition glue code. Eclipse will provide you with the skeleton to get started.
2. Find similar text in your output based on the Gherkin syntax your team came up with
3. Eclipse will not provide skeleton code for first step “**Given I have opened the browser**” as the code to open the browser is already present.

```
@And("I am on add New Owner form")
public void i_am_on_add_new_owner_form() {
// Write code here that turns the phrase above into concrete actions
throw new io.cucumber.java.PendingException();
}

@When("I submit pet owner information as {string} {string} {string} {string}
{string}")
public void i_submit_pet_owner_information_as(String string, String string2,
String string3, String string4, String string5) {
    // Write code here that turns the phrase above into concrete actions
    throw new io.cucumber.java.PendingException();
}
@Then("I see newly added pet owner information")
public void i_see_newly_added_pet_owner_information() {
// Write code here that turns the phrase above into concrete actions
throw new io.cucumber.java.PendingException();
}
```

* Example code provided. Yours will differ based on the Gherkin documentation your team came up with

Lab #1 - Add a new feature and step definition

Step 3 – Copy snippets provided in eclipse console to step definition file

Copy the “missing steps” to help you get started, and paste them into your **browsingAround.java** file OR if you created a **find_owner.feature**, then your **findOwner.java** file

```
@And("I am on add New Owner form")
public void i_am_on_add_new_owner_form() {
// Write code her that turns the phrase above into concrete actions
throw new io.cucumber.java.PendingException();
}

@When("I submit pet owner information as {string} {string} {string} {string} {string}")
public void i_submit_pet_owner_information_as(String string, String string2, String string3, String string4, String
string5) {
    // Write code here that turns the phrase above into concrete actions
    throw new io.cucumber.java.PendingException();
}
@Then("I see newly added pet owner information")
public void i_see_newly_added_pet_owner_information() {
// Write code here that turns the phrase above into concrete actions
throw new io.cucumber.java.PendingException();
}
```

* Example code provided. Yours will differ based on the Gherkin documentation your team came up with

Lab #1 - Add a new feature and step definition

Step 4 – Write logic to test “Add new owner” scenario in step definition file

Refer to Micro Lab #2, step #3 for instructions on how to retrieve the XPath.

```
@And("I am on add New Owner form")
public void i_am_on_add_new_owner_form() {

driver.navigate().to("http://localhost:8080/petclinic/owners/new");
}
```

* Example code provided. Yours will differ based on the Gherkin documentation your team came up with

Lab #1 - Add a new feature and step definition

Step 5 – Continue writing logic to test “Add new owner”

```
@When("I submit pet owner information {string} {string} {string} {string} {string}")
public void i_submit_pet_owner_information(String firstName, String lastName, String
address, String city, String telephone,) {
driver.findElement(By.xpath("//*[@id=\"firstName\"]")).sendKeys(firstName);
driver.findElement(By.xpath("//*[@id=\"lastName\"]")).sendKeys(lastName);
driver.findElement(By.xpath("//*[@id=\"address\"]")).sendKeys(address);
driver.findElement(By.xpath("//*[@id=\"city\"]")).sendKeys(city);
driver.findElement(By.xpath("//*[@id=\"telephone\"]")).sendKeys(telephone);

driver.findElement(By.xpath("//*[@id=\"add-owner-form\"]/div[6]/button")).click();
}
```

1. Example code provided. Yours will differ based on the Gherkin documentation your team came up with.
2. Notice we have renamed variable names to firstName, lastName, address, city and telephone

Lab #1 - Add a new feature and step definition

Step 6 – Continue writing logic to test “Add new owner”

Refer to Micro Lab #2, step #3 to find the corresponding XPath

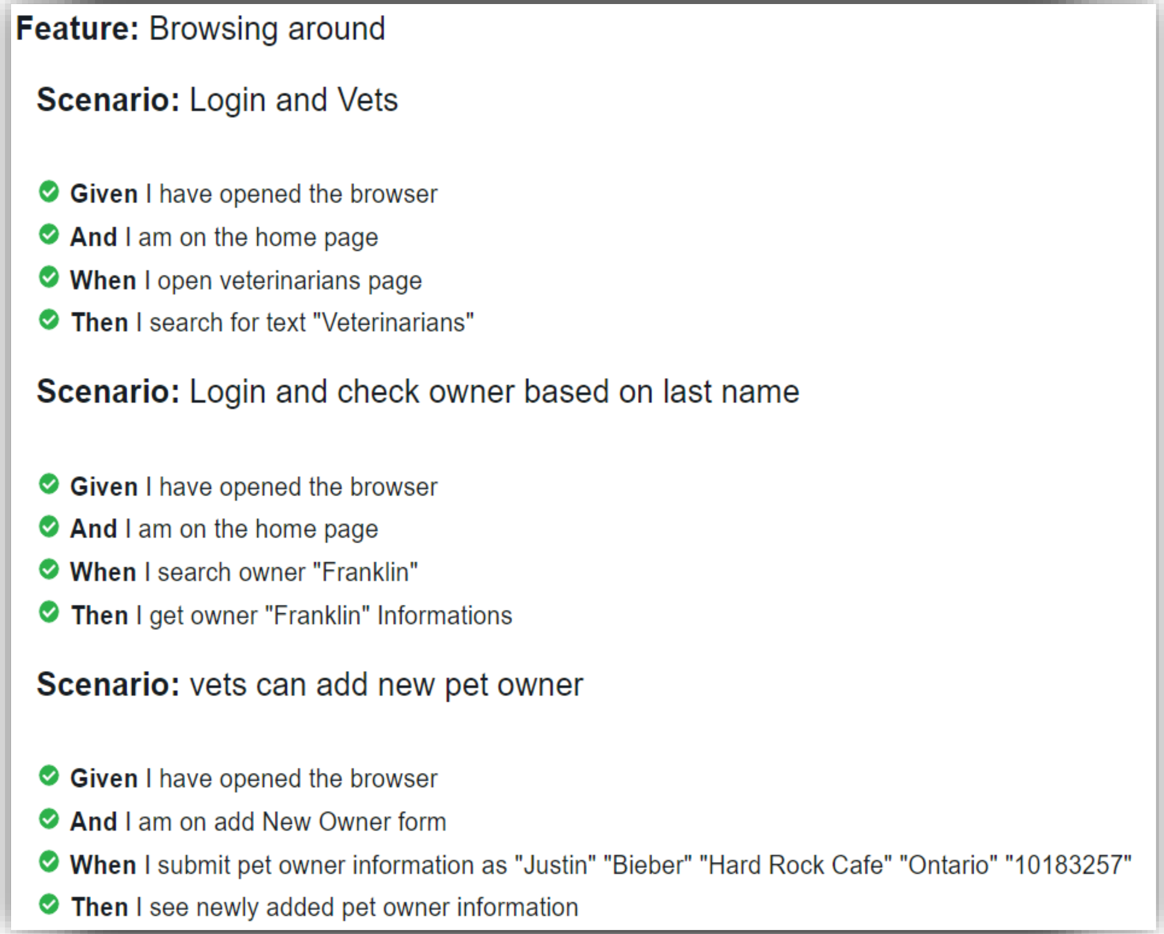
```
@Then("I see newly added pet owner information")
public void i_see_newly_added_pet_owner_information() {
String ownerName = driver.findElement(By.xpath("/html/body/div/table[1]/tbody/tr[1]/td/b")).getText();
assertTrue(ownerName.equalsIgnoreCase("Justin Bieber"));

String ownerAddress= driver.findElement(By.xpath("/html/body/div/table[1]/tbody/tr[2]/td")).getText();
assertTrue(ownerAddress.equalsIgnoreCase("Hard Rock Cafe"));

String ownerCity = driver.findElement(By.xpath("/html/body/div/table[1]/tbody/tr[3]/td")).getText();
assertTrue(ownerCity.equalsIgnoreCase("Ontario"));

String ownerTelephone = driver.findElement(By.xpath("/html/body/div/table[1]/tbody/tr[4]/td")).getText();
assertTrue(ownerTelephone.equalsIgnoreCase("10183257"));
}
```

When Complete, run the test: Right-click testPetClinic project, select “Run As” > “Maven Test”

Micro Lab #2 – Run and learn how it works

Step 5 – Open the Cucumber report

1. Using Windows Explorer navigate to the **Cucumber-pretty.html** file on your workstation, located at *{workspace_location}*\target\cucumber-reports and open the cucumber-pretty.html file
2. You should see the below Cucumber report



Feature: Browsing around

Scenario: Login and Vets

- ✓ **Given** I have opened the browser
- ✓ **And** I am on the home page
- ✓ **When** I open veterinarians page
- ✓ **Then** I search for text "Veterinarians"

Scenario: Login and check owner based on last name

- ✓ **Given** I have opened the browser
- ✓ **And** I am on the home page
- ✓ **When** I search owner "Franklin"
- ✓ **Then** I get owner "Franklin" Informations

Scenario: vets can add new pet owner

- ✓ **Given** I have opened the browser
- ✓ **And** I am on add New Owner form
- ✓ **When** I submit pet owner information as "Justin" "Bieber" "Hard Rock Cafe" "Ontario" "10183257"
- ✓ **Then** I see newly added pet owner information

End of Day 1 Document