**E-Portfolio Application**

**Testing Report**

**Emerging Trends in IT**

**Continuous Assessment**

Kevin Toh Zheng Ying

S10179142B

Table of Contents

[**Test Case Scenarios** 3](#_Toc25506062)

[**Test Case Scenario 1: Projects Section (Home Page)** 3](#_Toc25506063)

[**Test Case Scenario 2: Contact Form Section (Home Page)** 3](#_Toc25506064)

[**Test Case Scenario 3: Login** 4](#_Toc25506065)

[**Test Case Scenario 4: User Registration** 5](#_Toc25506066)

[**Test Case Scenario 5: Log Out** 6](#_Toc25506067)

[**Test Case Scenario 6: Blog Index** 6](#_Toc25506068)

[**Test Case Scenario 7: Blog Category** 6](#_Toc25506069)

[**Test Case Scenario 8: Comments Feature (Blog Details)** 7](#_Toc25506070)

[**Test Results** 8](#_Toc25506071)

[**Screenshot of Results** 8](#_Toc25506072)

[**Justification** 8](#_Toc25506073)

[**Coverage Results** 9](#_Toc25506074)

[**Screenshot of Results** 9](#_Toc25506075)

[**Justification** 9](#_Toc25506076)

# **Test Case Scenarios**

## **Test Case Scenario 1: Projects Section (Home Page)**

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Scenario** | **Test Description** | **Test Values** | **Expected Outcome** |
| 1.1 | This is to test the projects section of the “Home” page if clicking the “Read More” button for the project “BackToGoal” will redirect the user to the project details page for “BackToGoal”. | Click the “Read More” button for the project “BackToGoal” | User is redirected to a project details page with the project details for the project “BackToGoal”. |

## **Test Case Scenario 2: Contact Form Section (Home Page)**

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Scenario** | **Test Description** | **Test Values** | **Expected Outcome** |
| 2.1 | This is to test the submission of the contact message form when no input is given. | Name: None  Email: None  Message: None | Error message showing “Please fill up empty” fields should appear. |
| 2.2 | This is to test the submission of the contact message form when an invalid email is supplied for the email field. | Name: Jack  Email: jack#mail.com  Message: Hi there! | Error message showing invalid email format should appear. |
| 2.3 | This is to test the submission of the contact message form when a name of more than 50 characters is given. | Name: <Type the letter ’A’ 51 times>  Email: [ray@mail.com](mailto:ray@mail.com)  Message: “Hi! Would like to meet up?” | An error showing that name should not exceed 50 characters is shown. |
| 2.4 | This is to test the submission of the contact message form when an email of more than 70 characters is given. | Name: Ray  Email: [<Type the letter ‘A’ 62 times>@mail.com](mailto:ray123456@mail.com)  Message: “Hi! Would like to meet up?” | An error showing that name should not exceed 70 characters is shown. |
| 2.5 | This is to test the submission of the contact message form when a message of more than 2000 characters is given. | Name: Ray  Email: [ray@mail.com](mailto:ray@mail.com)  Message: <Type the letter ‘A’ 2001 times> | An error showing that message should not exceed 2000 characters is shown. |
| 2.6 | This is to test the contact form when valid input is provided to all fields and that a valid email is supplied for the email field as well as all input are kept within the maximum allowed character lengths. | Name: Ray  Email: [ray@mail.com](mailto:ray@mail.com)  Message: “Hi! Would like to meet up?” | A pop up showing that contact message was successfully sent should appear. |

## **Test Case Scenario 3: Login**

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Scenario** | **Test Description** | **Test Values** | **Expected Outcome** |
| 3.1 | This is to test the login function when no input is given | Username: None  Password: None | An error message appears on the same page showing “Please fill up the empty fields.” |
| 3.2 | This is to test the login function when a wrong password is given for a registered user. | Username: David3k  Password: red50blue | An error message appears on the same page which shows “Incorrect Username or Password” |
| 3.3 | This is to test the login function when an unregistered user provides a username and password. | Username: Dred6  Password: password | An error message appears on the same page which shows “Incorrect Username or Password” |
| 3.4 | This is to test the login function when a registered user provides a username and correct password. | At “Home” page  Click “Login” on the navigation bar  At “Login” page  Username: David3k  Password: Cyan30red | User is redirected to the previous page where they clicked “Login” on the navigation bar.  The “Login” option on the navigation bar should be changed to “Log Out”.  A welcome message  “Welcome, David3” is shown on an alert bar. |
| 3.5 | This is to test whether clicking on the back button of the login form will redirect the user back to the previous page where they came from.  **Note:** The login form has its own back button which is separate from the browsers back button. | At “Blog” page  Click “Login” on the navigation bar  At “Login” page  Click the back button on the top left of the login form. | User is redirected back to “Blog” page. |
| 3.6 | This is to test whether clicking on “Don’t have an account? Click here to register” will bring the user to the Registration page. | At “Login” page  Click the hyperlink “Don’t have an account? Click here to register” below the login button. | User is redirected to “Registration” page. |

## **Test Case Scenario 4: User Registration**

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Scenario** | **Test Description** | **Test Values** | **Expected Outcome** |
| 4.1 | This is to test the registration function when no input is given. | Email: None  Username: None  Password: None  Confirm Password: None | An error message appears on the same page showing “Please fill up the empty fields.” |
| 4.2 | This is to test the registration function when an invalid email is given. | Email: jack#mail.com  Username: Jack20  Password:  P@ssw0rd  Confirm Password: P@ssw0rd | An error message “Invalid email format.” is shown. |
| 4.3 | This is to test the registration function when an existing user email is used. | Email: d3k@mail.com  Username: Jack20  Password:  P@ssw0rd  Confirm Password: P@ssw0rd | An error message “An account with this email already exists” is shown. |
| 4.4 | This is to test the registration function when an existing user name is used. | Email: jack@mail.com  Username: David3k  Password:  P@ssw0rd  Confirm Password: P@ssw0rd | An error message “An account with this username already exists” is shown. |
| 4.5 | This is to test the registration function when a password that is less than 8 characters is given. | Email: jack@mail.com  Username: Jack20  Password:  Passwd  Confirm Password: Passwd | An error message “Password must be at least 8 characters long.” is shown. |
| 4.6 | This is to test the registration function when different password and confirm password are given | Email: jack@mail.com  Username: Jack20  Password:  P@ssw0rd  Confirm Password: Password | An error message “Password and Confirm Password do not match.” Is shown |
| 4.7 | This is to test the registration function when all input fields are valid. | Email: jack@mail.com  Username: Jack20  Password:  P@ssw0rd  Confirm Password: P@ssw0rd | User registration is successful.  User is redirected to “Blog Index” page. |
| 4.8 | Test back button on the registration form would redirect back to the login page.  **Note:** The registration form has its own back button which is separate from the browsers back button. | At “Registration” page  Click the back button on the top left of the registration form. | User is redirected back to “Login” page. |

## **Test Case Scenario 5: Log Out**

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Scenario** | **Test Description** | **Test Values** | **Expected Outcome** |
| 5.1 | This is to test the log out function when a user who is logged in clicks “Log Out” on the navigation bar. | Log in with the following registered user.  Username: David3  Password: Cyan30red  Click “Log Out” on the navigation bar. | “Log Out” option on the navigation bar changes to “Log In”.  The alert bar with the welcome message should disappear. |

## **Test Case Scenario 6: Blog Index**

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Scenario** | **Test Description** | **Test Values** | **Expected Outcome** |
| 6.1 | This is to test if clicking on a blog post title at the Blog Index Page would redirect the user to the blog details page for that specific post. | At Blog Index:  Click on “Cycling Journey: Changi towards the City” | Redirected to blog details page for “Cycling Journey: Changi towards the City” |
| 6.2 | This is to test if clicking on a category at the Blog Index Page would filter out blog posts with the following category. | At Blog Index:  Click on “School Project” | Blog posts with the “School Project” category are shown. |

## **Test Case Scenario 7: Blog Category**

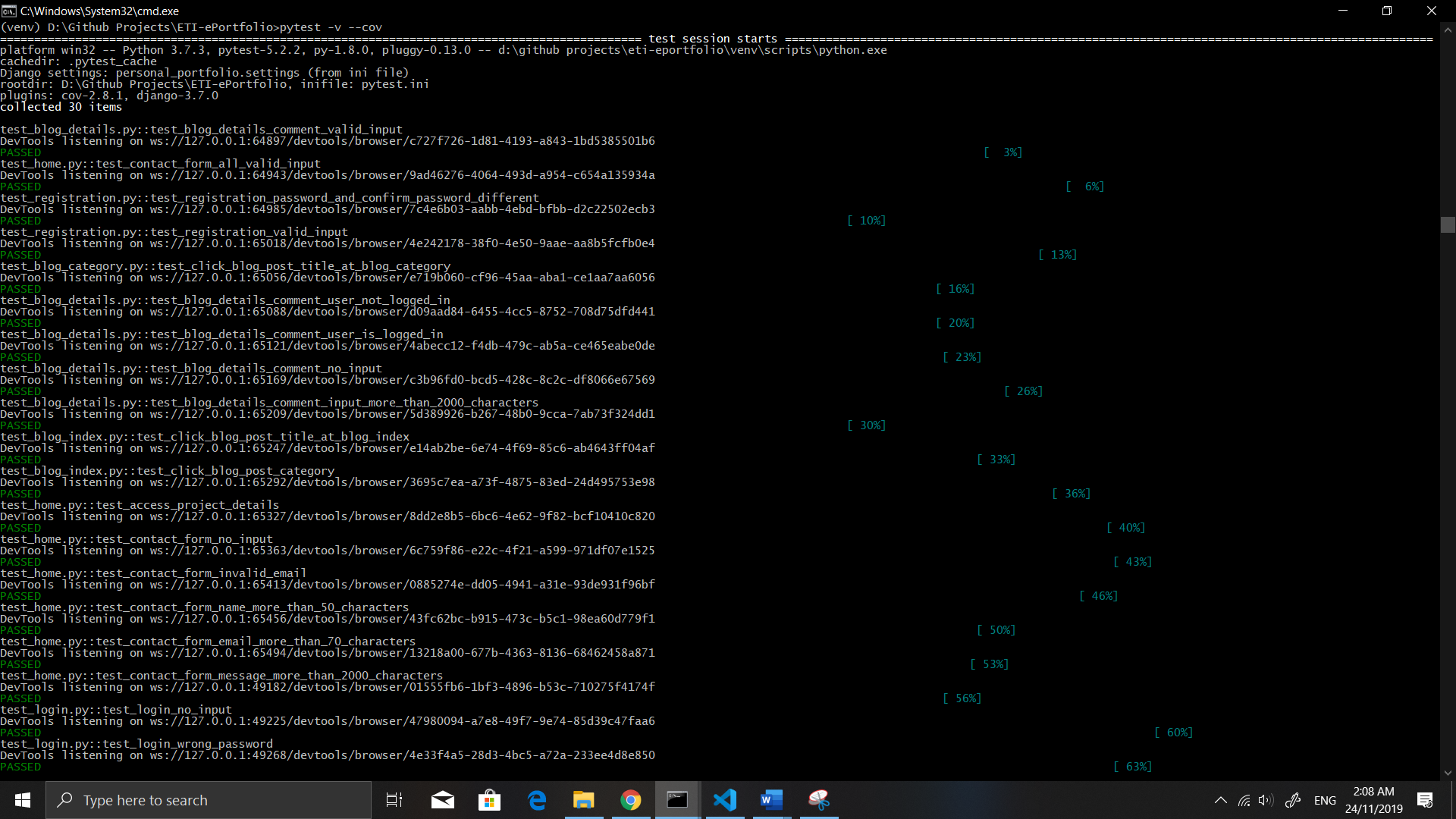
|  |  |  |  |
| --- | --- | --- | --- |
| **Test Scenario** | **Test Description** | **Test Values** | **Expected Outcome** |
| 7.1 | This is to test if clicking on a blog post title at the Blog Category Page would redirect the user to the blog details page for that specific post. | At Blog Category:  Category: “Personal Project”  Click on “Shopify” | Redirected to blog details page for “Shopify” |

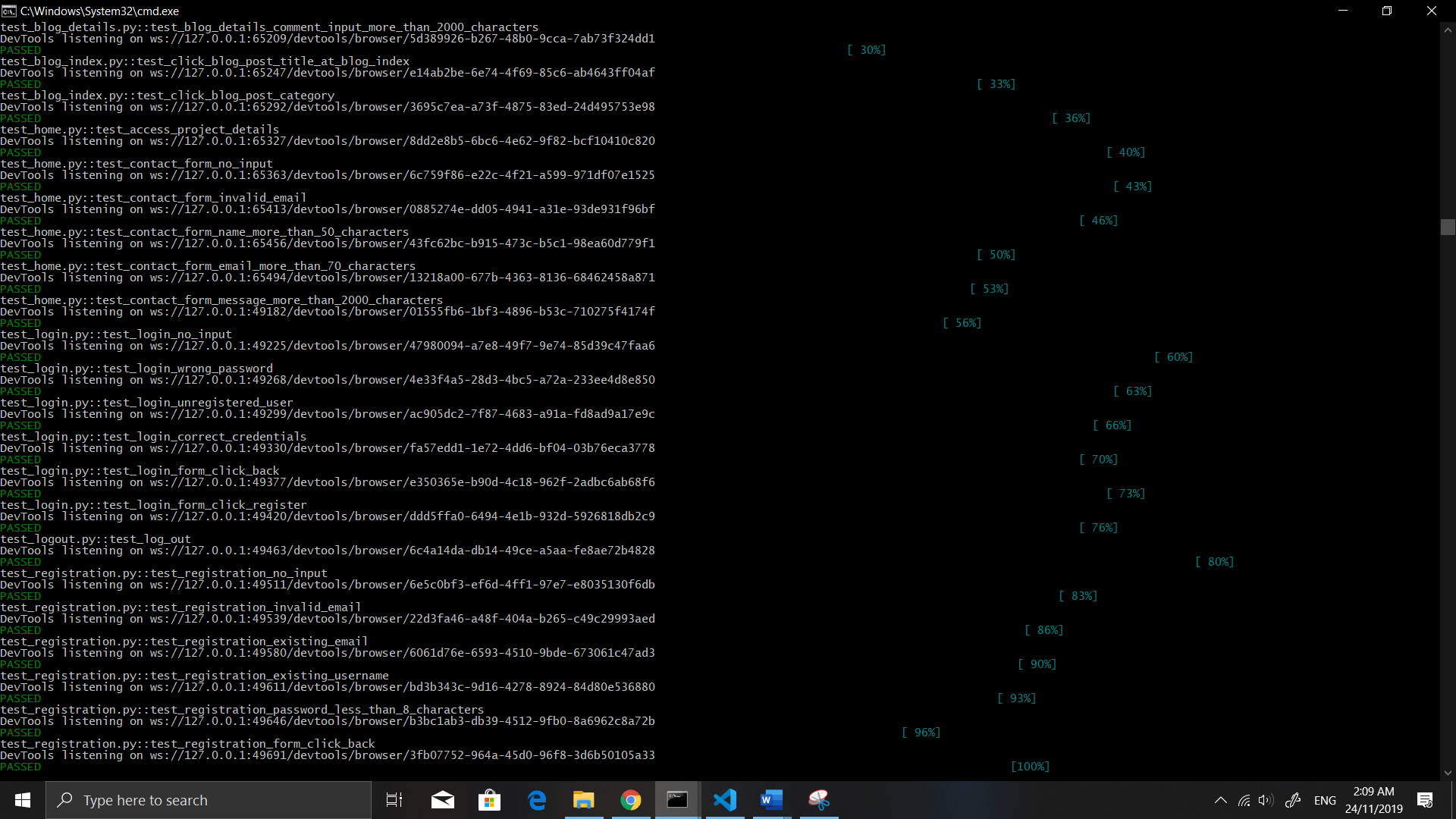
## **Test Case Scenario 8: Comments Feature (Blog Details)**

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Scenario** | **Test Description** | **Test Values** | **Expected Outcome** |
| 8.1 | This is to test the comments feature of the blog details page when the user that visits the page is not logged in. | If not logged out click “Log Out” on the navigation bar. | An alert dialog “Login to comment” is shown.  Only comments can be seen |
| 8.2 | This is to test the comments feature of the blog details page when the user that visits the page is logged in. | At “Home” page  Click “Login” on the navigation bar  At “Login” page  Username: David3  Password: Cyan30red | The user should be able to see the details of the blog post as well as the list of comments below.  The user would be able to post a comment as well. |
| 8.3 | This is to test the comments feature of the blog details page when the user who is logged in posts a comment with 0 characters. | Comment: None | An error about the comment being empty is shown. |
| 8.4 | This is to test the comments feature of the blog details page when the user who is logged in posts a comment that has more than 0 characters and less than 2000 characters | Comment: This project looks good. | Comment is posted and is shown as the latest comment on the top of the list of comments for the blog post. |
| 8.5 | This is to test the limits of the comments feature by posting a comment with more than 2000 characters. | Comment: <No space to type 2000 characters here> | An error about exceeding the character limit for comments is shown. |

# **Test Results**

## **Screenshot of Results**





## **Justification**

Although every written unit test for each test scenario passed, I would like to mention that the expected results for every single unit test were based on the changes that will happen to the user interface that is rendered at the front end due to a user’s action. For example, in Test Case Scenario 2.6, the scenario tests whether when valid input is provided to all fields in the contact form whereby a valid email is supplied for the email field and that all input are kept within the maximum allowed character lengths. The view function in the Django application will check if all the input fields contain valid values and will trigger the pop up showing that contact message was successfully sent at the front end when re-rendering the view.

I feel that it would be better if I was able to base the expected results on the actual database records but was unable to directly access them even with the pytest-django plugin even when forced running migrations is enabled. I later found out that it was because pytest-django creates a separate empty database which is used for unit testing. Therefore, I was not able to access the database that was used by the application on the development server and could not assert the outcome of any data creation or updating tests with the actual database records.

# **Coverage Results**

## **Screenshot of Results**



## **Justification**

While the coverage of codes in the files are 100% as shown in the screenshot above, the views.py files of each Django application in the project are not reflected on the coverage results. The Selenium WebDriver alone can only test the interactivity of the web pages and check if certain HTML elements appear as a result of an action. Although the codes of the view functions would run, it is not directly included into each written unit test and therefore not reflected in the coverage results.

It is possible to test the view functions of a Django project with the Django Client test tool as well as reverse()utility function which works similarly to the url template used in Django web templates.

An example below shows a unit test which tests if the “blog\_index” view loads successfully by checking if the HTTP response status code from the web server is 200 which signifies “OK”.

|  |
| --- |
| def test\_blog\_index\_load(self):  client = Client()  response = client.get(reverse('blog\_index'))  assert response.status\_code == 200 |

By doing a view load test for every page of the Django application, the coverage results should technically now include the views.py file of each Django application. Unfortunately, this has to be tested using Django’s own test execution framework at the tests.py file that resides within the folder of each registered Django application of the project.