SPECFLOW HYBRID FRAMEWORK

V1.0

Muhammad Haider Ejaz

Contents

SPECFLOW HYBRID FRAMEWORK DOCUMENTATION

1. Framework Introduction

This framework is designed for testing purposes so we can test our web application and API's easily. Mainly This Framework has more focused on Web & API's testing but yes we can add Mobile Testing as well with just one configuration file through which we can decide the devices, browsers etc.

2. Specflow Introduction

SpecFlow is a test automation solution for .NET built upon the BDD paradigm. Use SpecFlow to define, manage and automatically execute human-readable acceptance tests in .NET projects (Full Framework and .NET Core).

SpecFlow tests are written using Gherkin Language, which allows you to write test cases using natural languages. SpecFlow uses the official Gherkin parser, which supports over 70 languages. These tests are then tied to your application code using Bindings, allowing you to execute the tests using the testing framework of your choice. You can also execute your tests using SpecFlow's dedicated test runner, SpecFlow+ Runner.

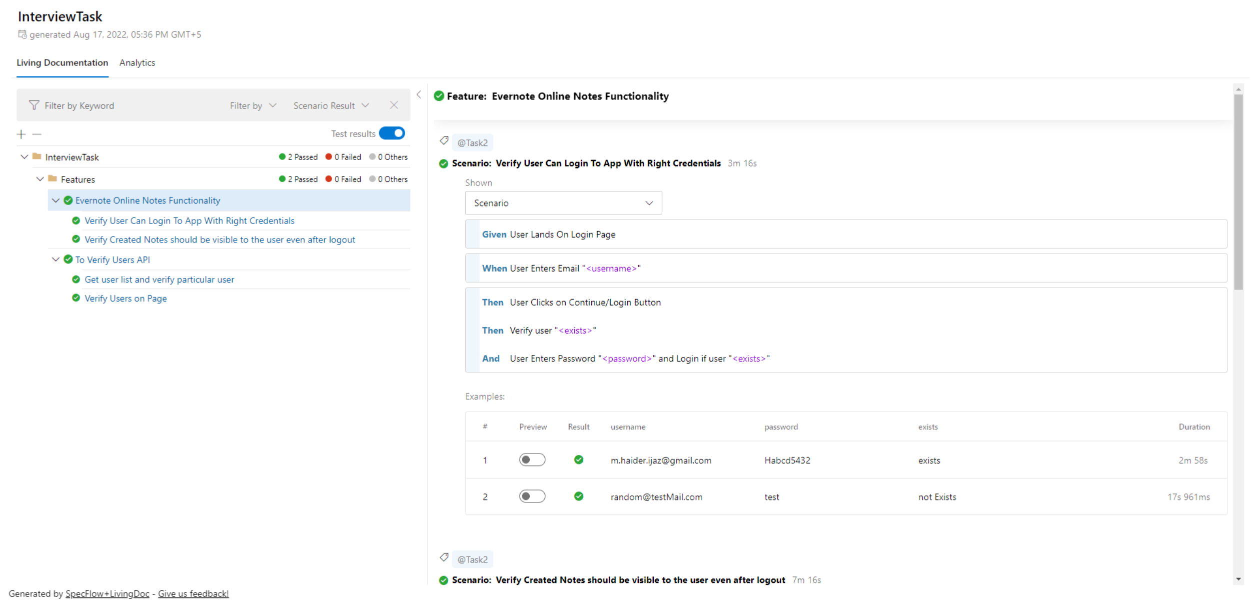
2.1 Why Specflow

- **Step definitions** - Step definitions provide the connection between Gherkin feature specifications and application interfaces for test automation.
- **Navigation to Step definitions** - Don't waste your time searching for the correct definition across your binding classes, just right-click and jump to the relevant code.
- **Hooks** - Hooks (event bindings) can be used to perform additional automation logic at specific times, such as any setup required before executing a scenario.
- **Context Injection** - SpecFlow supports a dependency injection framework that can instantiate and inject context for scenarios. This allows you to group the shared state in context classes, and inject them into every binding class that needs access to that shared state.

2.2 Specflow plus Living Doc

SpecFlow+LivingDoc is a set of tools that allows you to share and collaborate on Gherkin Feature Files with stakeholders who may not be familiar with developer tools and for technical team as well.

Living Documentation:



InterviewTask
generated Aug 17, 2022, 05:36 PM GMT+5

Living Documentation | Analytics

Filter by Keyword | Filter by | Scenario Result
Test results

- InterviewTask — 2 Passed, 0 Failed, 0 Others
  - Features — 2 Passed, 0 Failed, 0 Others
    - Evernote Online Notes Functionality
      - Verify User Can Login To App With Right Credentials
      - Verify Created Notes should be visible to the user even after logout
    - To Verify Users API
      - Get user list and verify particular user
      - Verify Users on Page

Generated by SpecFlow+LivingDoc - Give us feedback!

Feature: Evernote Online Notes Functionality

@Task2
Scenario: Verify User Can Login To App With Right Credentials 3m 16s

Shown: Scenario

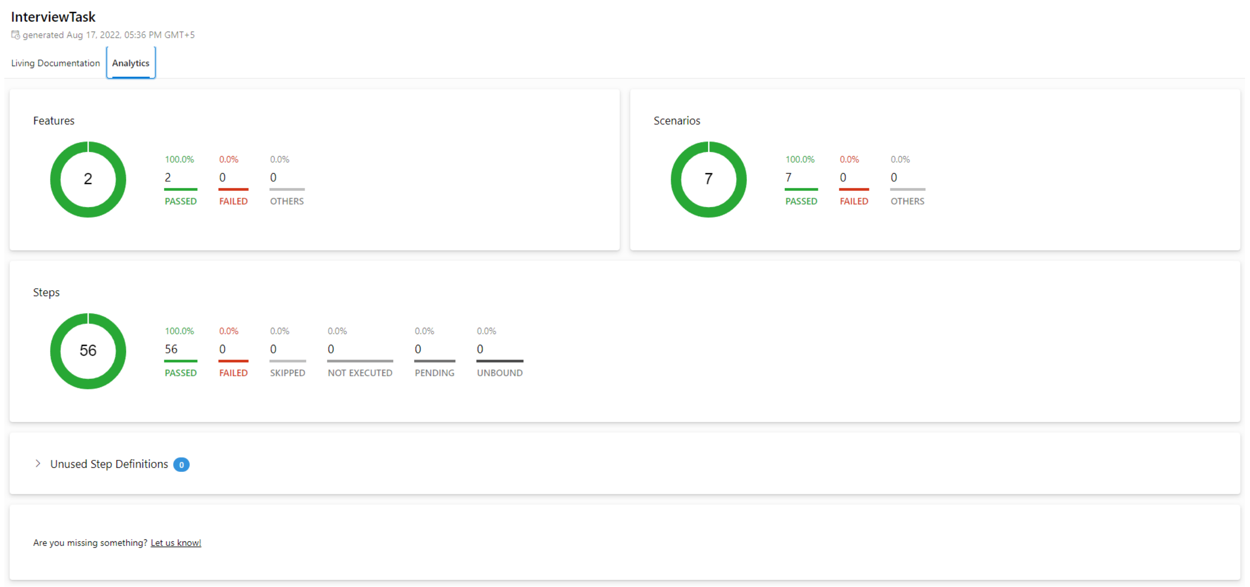
Given User Lands On Login Page
When User Enters Email "<username>"
Then User Clicks on Continue/Login Button
Then Verify user "<exists>"
And User Enters Password "<password>" and Login if user "<exists>"

Examples:

#	Preview	Result	username	password	exists	Duration
1		✓	m.haider.ijaz@gmail.com	Habcd5432	exists	2m 58s
2		✓	random@testMail.com	test	not Exists	17s 961ms

@Task2
Scenario: Verify Created Notes should be visible to the user even after logout 7m 16s

Analytics:



InterviewTask
generated Aug 17, 2022, 05:36 PM GMT+5

Living Documentation | Analytics

Features: 2

PASSED	FAILED	OTHERS
100.0%	0.0%	0.0%
2	0	0

Scenarios: 7

PASSED	FAILED	OTHERS
100.0%	0.0%	0.0%
7	0	0

Steps: 56

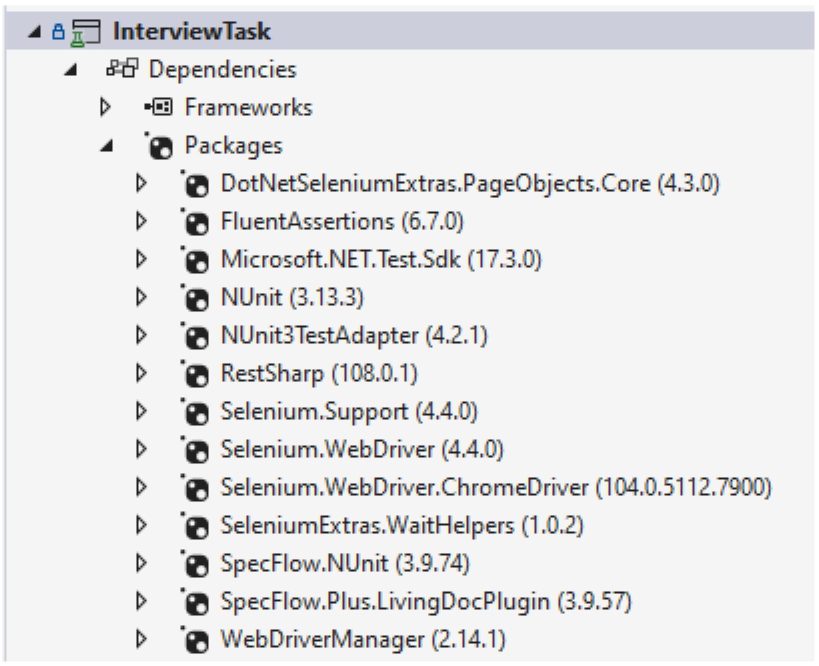
PASSED	FAILED	SKIPPED	NOT EXECUTED	PENDING	UNBOUND
100.0%	0.0%	0.0%	0.0%	0.0%	0.0%
56	0	0	0	0	0

Unused Step Definitions 0

Are you missing something? Let us know!

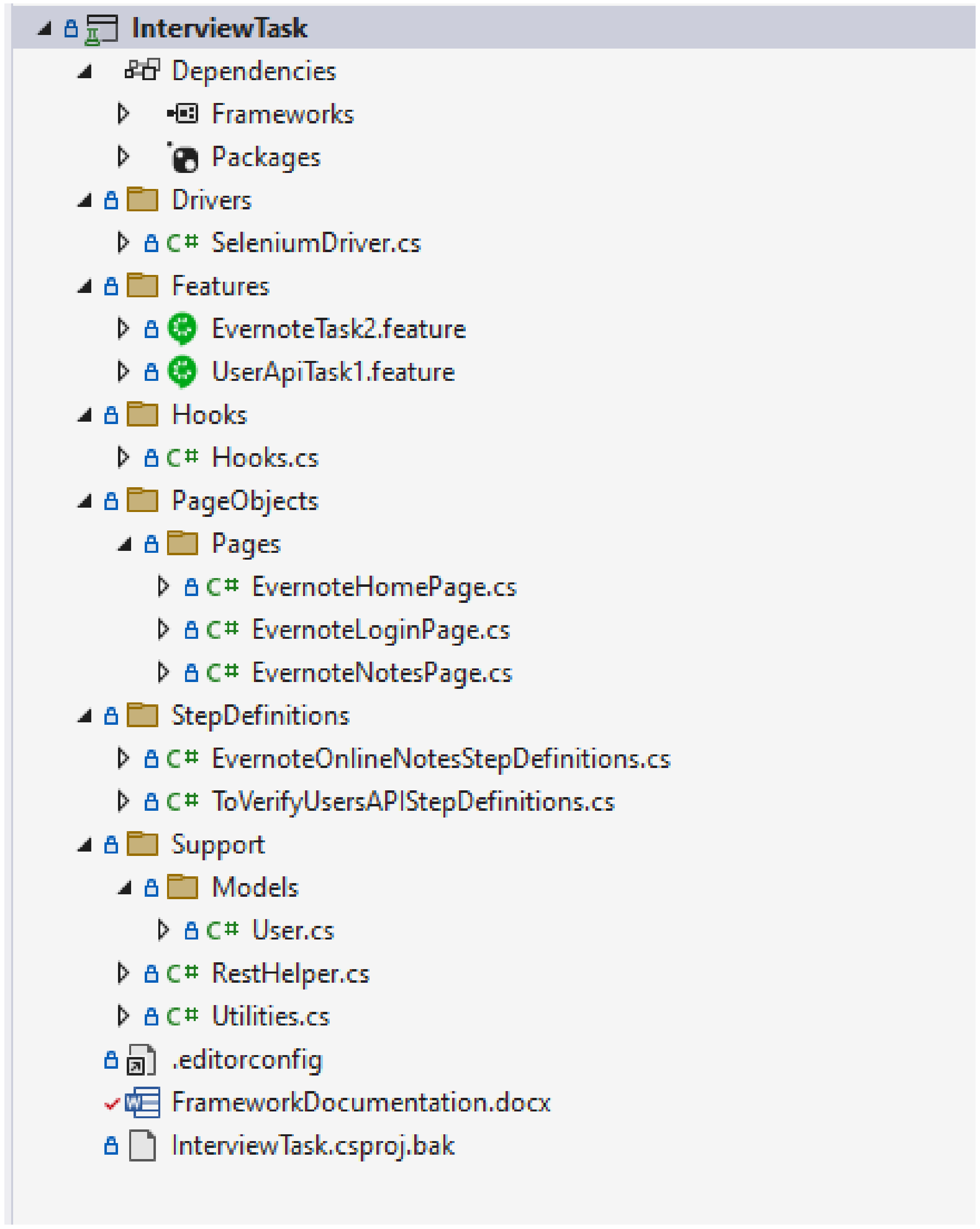
3. Specflow Framework Packages

- **Selenium.Webdriver.ChromeDriver:** Install Chrome Driver for Selenium WebDriver into your Unit Test Project, "chromedriver(.exe)" is copied to the bin folder from the package folder when the build process NuGet package restoring ready, and no need to commit "chromedriver(.exe)" binary into source code control repository. (This package automatically downloads the latest version of chrome driver so we don't need to use webdriver manager).
- **SeleniumExtras.PageObjects:** Page Factory is a class provided by Selenium WebDriver to support Page Object Design patterns and lazy initialization. In Page Factory, testers use `FindsBy` annotation. The initElements method is used to initialize web elements. `FindsBy`: An annotation used in Page Factory to locate and declare web elements using different locators.
- **NUnit**: NUnit features a fluent assert syntax, parameterized, generic, and theory tests and is user-extensible.This package includes the NUnit 3 framework assembly, which is referenced by your tests. You will need to install version 3 of the nunit3-console program or a third-party runner that supports NUnit 3 to execute tests. Runners intended for use with NUnit 2.x will not run NUnit 3 tests correctly.
- **RestSharp:** I've used RestSharp for API testing since RestSharp is probably the most popular HTTP client library for .NET. Featuring automatic serialization and deserialization, request and response type detection, variety of authentications and other useful features.


```
InterviewTask
  Dependencies
    Frameworks
    Packages
      DotNetSeleniumExtras.PageObjects.Core (4.3.0)
      FluentAssertions (6.7.0)
      Microsoft.NET.Test.Sdk (17.3.0)
      NUnit (3.13.3)
      NUnit3TestAdapter (4.2.1)
      RestSharp (108.0.1)
      Selenium.Support (4.4.0)
      Selenium.WebDriver (4.4.0)
      Selenium.WebDriver.ChromeDriver (104.0.5112.7900)
      SeleniumExtras.WaitHelpers (1.0.2)
      SpecFlow.NUnit (3.9.74)
      SpecFlow.Plus.LivingDocPlugin (3.9.57)
      WebDriverManager (2.14.1)
```

4. Project Structure

Page object model design pattern is used in this framework and the complete structure of this framework is shown below.



```
InterviewTask
    Dependencies
        Frameworks
        Packages
    Drivers
        SeleniumDriver.cs
    Features
        EvernoteTask2.feature
        UserApiTask1.feature
    Hooks
        Hooks.cs
    PageObjects
        Pages
            EvernoteHomePage.cs
            EvernoteLoginPage.cs
            EvernoteNotesPage.cs
    StepDefinitions
        EvernoteOnlineNotesStepDefinitions.cs
        ToVerifyUsersAPIStepDefinitions.cs
    Support
        Models
            User.cs
        RestHelper.cs
        Utilities.cs
    .editorconfig
    FrameworkDocumentation.docx
    InterviewTask.csproj.bak
```

5. Assignment Detail

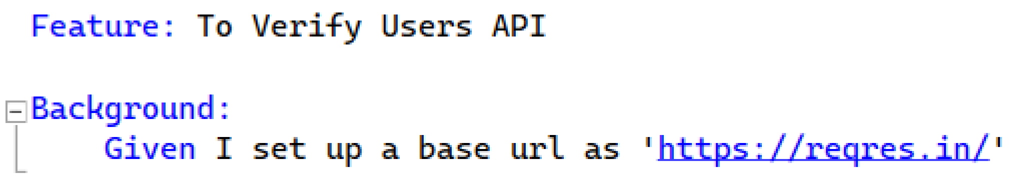
As QA Automation Engineer and as Solution Architect I've designed this Framework which can hold Web Browser Tests as well as API's Test's. So to achieve this target I've used above mentioned technology stack (Selenium, RestSharp with Specflow). Since as per requirement I've to implement it using BDD framework so every task has its own feature file and based on that feature we've different scenario's implemented, let's have a brief Introduction of each Scenario of both the Feature file.

✓ Solution

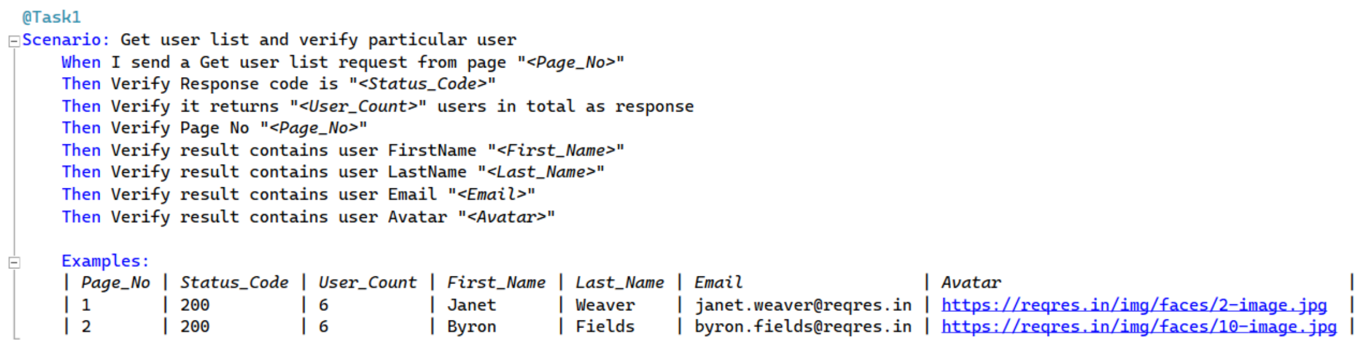
- I have created one feature file having two scenario's each Scenario has a Background where I'm setting up the base URL. Steps are defined and implemented in a way that every step has reusable capabilities and it can be run through data driven approach means any one can change or add the data to test more scenarios.
- I have used Tags approach to deal different conditions, e.g. we can group them and even if you discussed Task2 so there we have to initialize chrome browser and that we don't need in Task1, so created hooks with Tags means when Task2 trigger than only browser initializes else no need.
- Specflow Living Document and Detail Analysis also available view full execution report and also you can preview scenario behavior.

5.1 Task 1

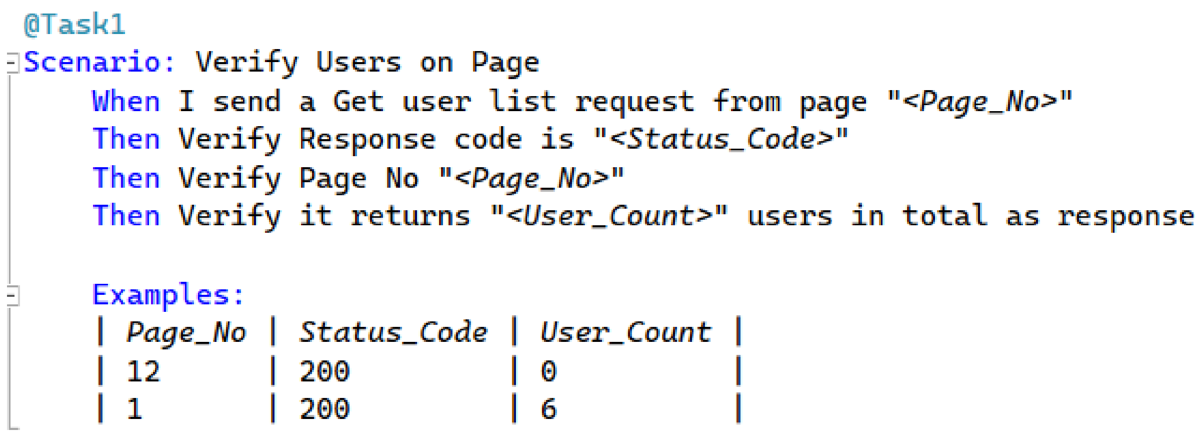
"As a QA engineer you are requested to test a particular API method with the following URL https://reqres.in/api/users?page=1. During the code analysis, 3 main scenarios to be tested were identified. Develop a .Net Test Framework to implement the below tests (ideally using a BDD language such as Gherkin)."



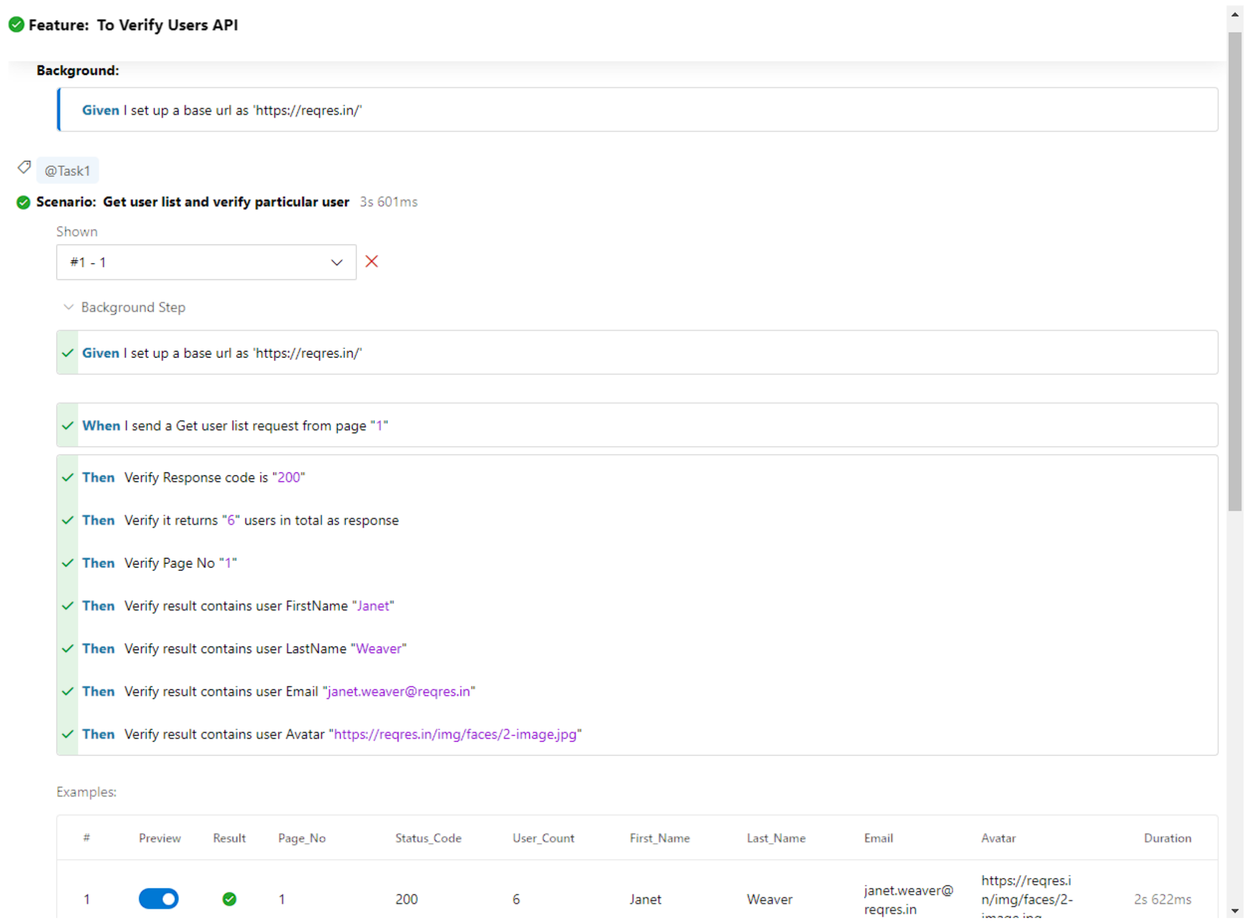
```
Feature: To Verify Users API

Background:
    Given I set up a base url as 'https://reqres.in/'
```



```
@Task1
Scenario: Get user list and verify particular user
    When I send a Get user list request from page "<Page_No>"
    Then Verify Response code is "<Status_Code>"
    Then Verify it returns "<User_Count>" users in total as response
    Then Verify Page No "<Page_No>"
    Then Verify result contains user FirstName "<First_Name>"
    Then Verify result contains user LastName "<Last_Name>"
    Then Verify result contains user Email "<Email>"
    Then Verify result contains user Avatar "<Avatar>"

    Examples:
    | Page_No | Status_Code | User_Count | First_Name | Last_Name | Email                  | Avatar                                  |
    | 1       | 200         | 6          | Janet      | Weaver    | janet.weaver@reqres.in | https://reqres.in/img/faces/2-image.jpg  |
    | 2       | 200         | 6          | Byron      | Fields    | byron.fields@reqres.in | https://reqres.in/img/faces/10-image.jpg |
```



```
@Task1
Scenario: Verify Users on Page
    When I send a Get user list request from page "<Page_No>"
    Then Verify Response code is "<Status_Code>"
    Then Verify Page No "<Page_No>"
    Then Verify it returns "<User_Count>" users in total as response

    Examples:
    | Page_No | Status_Code | User_Count |
    | 12      | 200         | 0          |
    | 1       | 200         | 6          |
```



Feature: To Verify Users API

Background:

Given I set up a base url as 'https://reqres.in/'

@Task1

Scenario: Get user list and verify particular user 3s 601ms

Shown

#1 - 1

Background Step

✓ Given I set up a base url as 'https://reqres.in/'

✓ When I send a Get user list request from page "1"

✓ Then Verify Response code is "200"

✓ Then Verify it returns "6" users in total as response

✓ Then Verify Page No "1"

✓ Then Verify result contains user FirstName "Janet"

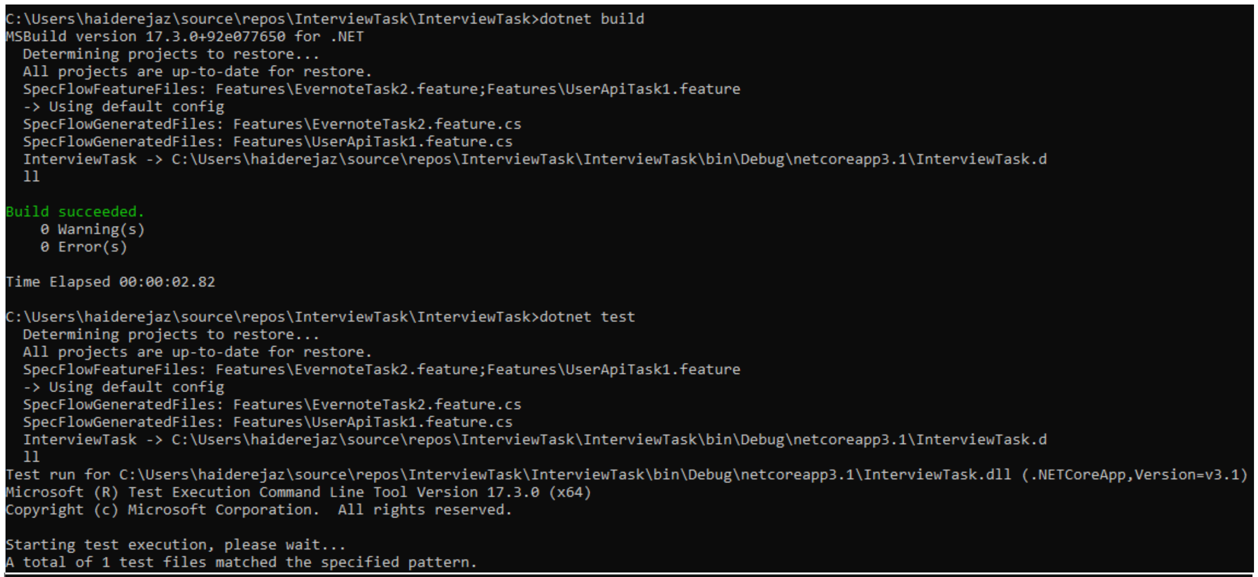
✓ Then Verify result contains user LastName "Weaver"

✓ Then Verify result contains user Email "janet.weaver@reqres.in"

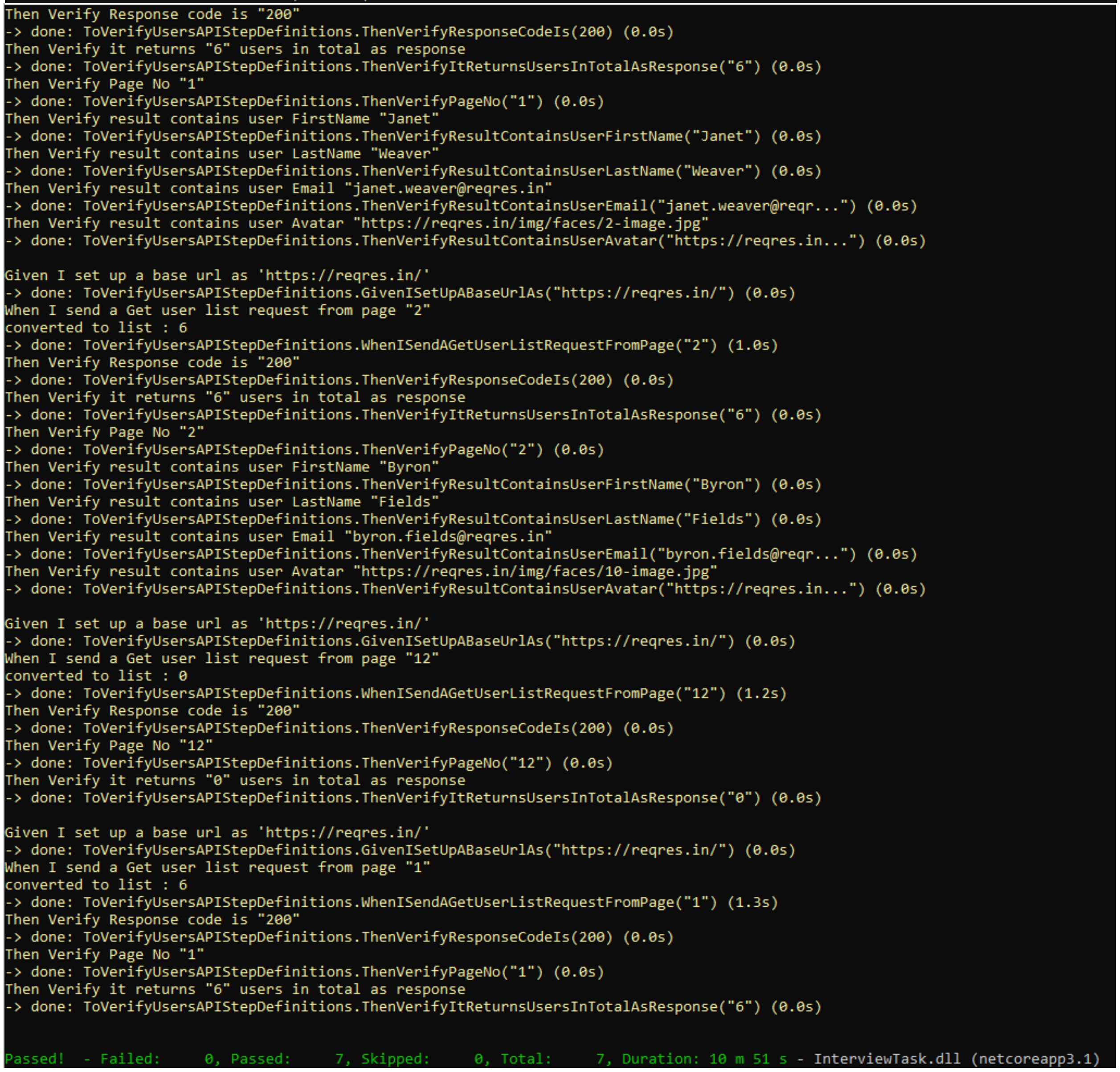
✓ Then Verify result contains user Avatar "https://reqres.in/img/faces/2-image.jpg"

Examples:

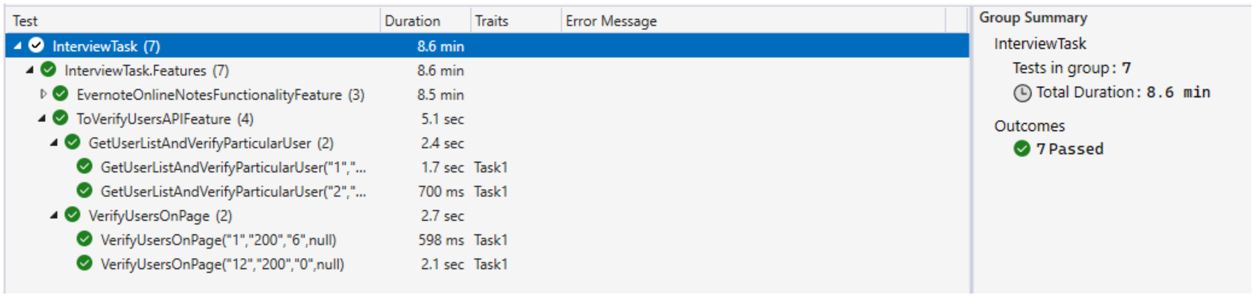
#	Preview	Result	Page_No	Status_Code	User_Count	First_Name	Last_Name	Email	Avatar	Duration
1		✓	1	200	6	Janet	Weaver	janet.weaver@reqres.in	https://reqres.in/img/faces/2-	2s 622ms



```
C:\Users\haiderejaz\source\repos\InterviewTask\InterviewTask>dotnet build
MSBuild version 17.3.0+92e077650 for .NET
  Determining projects to restore...
  All projects are up-to-date for restore.
  SpecFlowFeatureFiles: Features\EvernoteTask2.feature;Features\UserApiTask1.feature
  -> Using default config
  SpecFlowGeneratedFiles: Features\EvernoteTask2.feature.cs
  SpecFlowGeneratedFiles: Features\UserApiTask1.feature.cs
  InterviewTask -> C:\Users\haiderejaz\source\repos\InterviewTask\InterviewTask\bin\Debug\netcoreapp3.1\InterviewTask.d
  ll

Build succeeded.
    0 Warning(s)
    0 Error(s)

Time Elapsed 00:00:02.82

C:\Users\haiderejaz\source\repos\InterviewTask\InterviewTask>dotnet test
  Determining projects to restore...
  All projects are up-to-date for restore.
  SpecFlowFeatureFiles: Features\EvernoteTask2.feature;Features\UserApiTask1.feature
  -> Using default config
  SpecFlowGeneratedFiles: Features\EvernoteTask2.feature.cs
  SpecFlowGeneratedFiles: Features\UserApiTask1.feature.cs
  InterviewTask -> C:\Users\haiderejaz\source\repos\InterviewTask\InterviewTask\bin\Debug\netcoreapp3.1\InterviewTask.d
  ll
Test run for C:\Users\haiderejaz\source\repos\InterviewTask\InterviewTask\bin\Debug\netcoreapp3.1\InterviewTask.dll (.NETCoreApp,Version=v3.1)
Microsoft (R) Test Execution Command Line Tool Version 17.3.0 (x64)
Copyright (c) Microsoft Corporation.  All rights reserved.

Starting test execution, please wait...
A total of 1 test files matched the specified pattern.
```



```
Then Verify Response code is "200"
-> done: ToVerifyUsersAPIStepDefinitions.ThenVerifyResponseCodeIs(200) (0.0s)
Then Verify it returns "6" users in total as response
-> done: ToVerifyUsersAPIStepDefinitions.ThenVerifyItReturnsUsersInTotalAsResponse("6") (0.0s)
Then Verify Page No "1"
-> done: ToVerifyUsersAPIStepDefinitions.ThenVerifyPageNo("1") (0.0s)
Then Verify result contains user FirstName "Janet"
-> done: ToVerifyUsersAPIStepDefinitions.ThenVerifyResultContainsUserFirstName("Janet") (0.0s)
Then Verify result contains user LastName "Weaver"
-> done: ToVerifyUsersAPIStepDefinitions.ThenVerifyResultContainsUserLastName("Weaver") (0.0s)
Then Verify result contains user Email "janet.weaver@reqres.in"
-> done: ToVerifyUsersAPIStepDefinitions.ThenVerifyResultContainsUserEmail("janet.weaver@reqr...") (0.0s)
Then Verify result contains user Avatar "https://reqres.in/img/faces/2-image.jpg"
-> done: ToVerifyUsersAPIStepDefinitions.ThenVerifyResultContainsUserAvatar("https://reqres.in...") (0.0s)

Given I set up a base url as 'https://reqres.in/'
-> done: ToVerifyUsersAPIStepDefinitions.GivenISetUpABaseUrlAs("https://reqres.in/") (0.0s)
When I send a Get user list request from page "2"
converted to list : 6
-> done: ToVerifyUsersAPIStepDefinitions.WhenISendAGetUserListRequestFromPage("2") (1.0s)
Then Verify Response code is "200"
-> done: ToVerifyUsersAPIStepDefinitions.ThenVerifyResponseCodeIs(200) (0.0s)
Then Verify it returns "6" users in total as response
-> done: ToVerifyUsersAPIStepDefinitions.ThenVerifyItReturnsUsersInTotalAsResponse("6") (0.0s)
Then Verify Page No "2"
-> done: ToVerifyUsersAPIStepDefinitions.ThenVerifyPageNo("2") (0.0s)
Then Verify result contains user FirstName "Byron"
-> done: ToVerifyUsersAPIStepDefinitions.ThenVerifyResultContainsUserFirstName("Byron") (0.0s)
Then Verify result contains user LastName "Fields"
-> done: ToVerifyUsersAPIStepDefinitions.ThenVerifyResultContainsUserLastName("Fields") (0.0s)
Then Verify result contains user Email "byron.fields@reqres.in"
-> done: ToVerifyUsersAPIStepDefinitions.ThenVerifyResultContainsUserEmail("byron.fields@reqr...") (0.0s)
Then Verify result contains user Avatar "https://reqres.in/img/faces/10-image.jpg"
-> done: ToVerifyUsersAPIStepDefinitions.ThenVerifyResultContainsUserAvatar("https://reqres.in...") (0.0s)

Given I set up a base url as 'https://reqres.in/'
-> done: ToVerifyUsersAPIStepDefinitions.GivenISetUpABaseUrlAs("https://reqres.in/") (0.0s)
When I send a Get user list request from page "12"
converted to list : 0
-> done: ToVerifyUsersAPIStepDefinitions.WhenISendAGetUserListRequestFromPage("12") (1.2s)
Then Verify Response code is "200"
-> done: ToVerifyUsersAPIStepDefinitions.ThenVerifyResponseCodeIs(200) (0.0s)
Then Verify Page No "12"
-> done: ToVerifyUsersAPIStepDefinitions.ThenVerifyPageNo("12") (0.0s)
Then Verify it returns "0" users in total as response
-> done: ToVerifyUsersAPIStepDefinitions.ThenVerifyItReturnsUsersInTotalAsResponse("0") (0.0s)

Given I set up a base url as 'https://reqres.in/'
-> done: ToVerifyUsersAPIStepDefinitions.GivenISetUpABaseUrlAs("https://reqres.in/") (0.0s)
When I send a Get user list request from page "1"
converted to list : 6
-> done: ToVerifyUsersAPIStepDefinitions.WhenISendAGetUserListRequestFromPage("1") (1.3s)
Then Verify Response code is "200"
-> done: ToVerifyUsersAPIStepDefinitions.ThenVerifyResponseCodeIs(200) (0.0s)
Then Verify Page No "1"
-> done: ToVerifyUsersAPIStepDefinitions.ThenVerifyPageNo("1") (0.0s)
Then Verify it returns "6" users in total as response
-> done: ToVerifyUsersAPIStepDefinitions.ThenVerifyItReturnsUsersInTotalAsResponse("6") (0.0s)


Passed!  - Failed:     0, Passed:     7, Skipped:     0, Total:     7, Duration: 10 m 51 s - InterviewTask.dll (netcoreapp3.1)
```



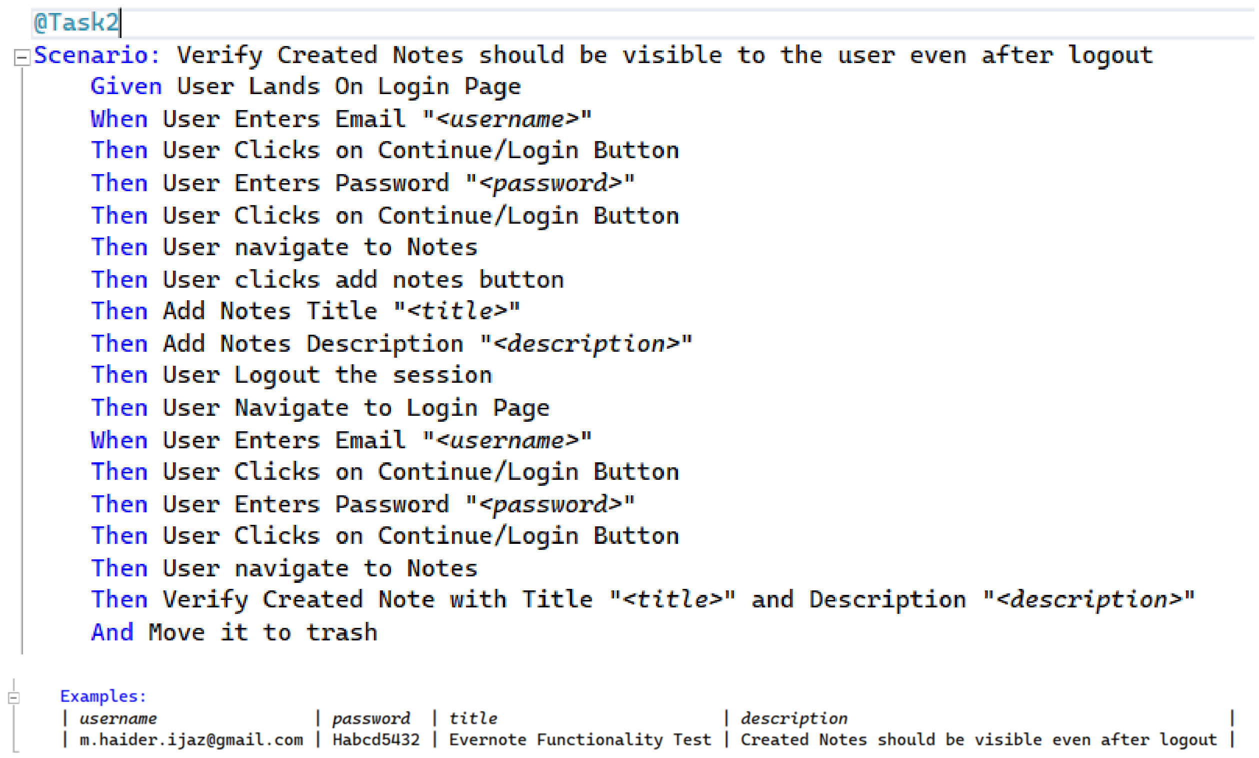
```
Test                                                  Duration  Traits  Error Message        Group Summary
◢ InterviewTask (7)                                   8.6 min                                  InterviewTask
  ◢ InterviewTask.Features (7)                        8.6 min                                    Tests in group: 7
    ▷ EvernoteOnlineNotesFunctionalityFeature (3)     8.5 min                                    Total Duration: 8.6 min
    ◢ ToVerifyUsersAPIFeature (4)                     5.1 sec                                  Outcomes
      ◢ GetUserListAndVerifyParticularUser (2)        2.4 sec                                    7 Passed
          GetUserListAndVerifyParticularUser("1","... 1.7 sec   Task1
          GetUserListAndVerifyParticularUser("2","... 700 ms    Task1
      ◢ VerifyUsersOnPage (2)                         2.7 sec
          VerifyUsersOnPage("1","200","6",null)       598 ms    Task1
          VerifyUsersOnPage("12","200","0",null)      2.1 sec   Task1
```

5.2 Task 2

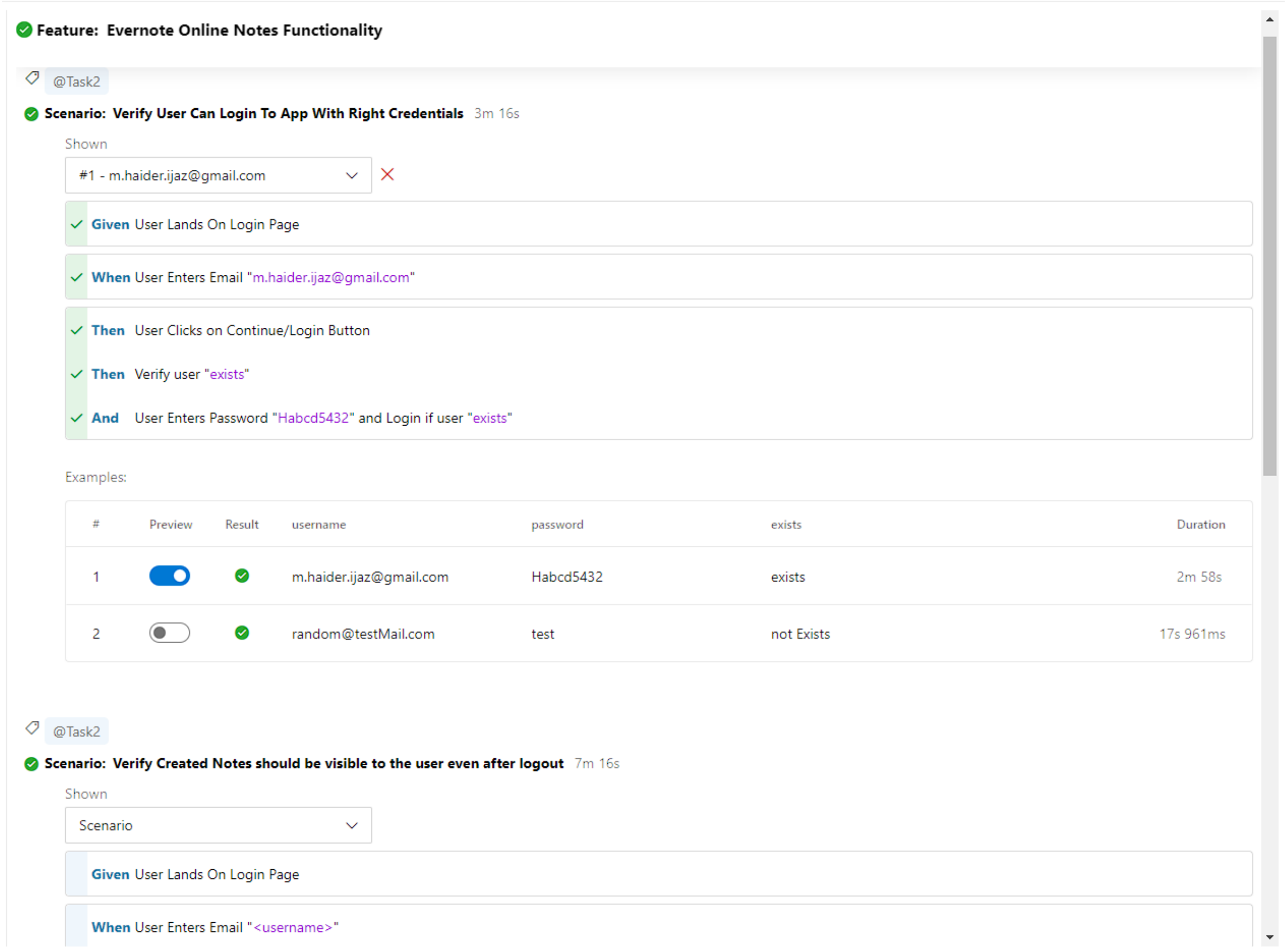
“Consider that you are working with a well-known online note taking company, https://evernote.com/, as QA Test Automation Engineer. You are given the task to develop a test automation framework using Selenium, together with a BDD language such as gherkin to implement UI automated tests on different core functionalities of the Website. These tests should cover:

1. *Unsuccessful login using email and assert that the proper error messages are returned*
2. *Successful login using email and assert that the user is granted access to the site*
3. *Login and write a note followed by a logout. Finally, login again and make sure you can open and view the note created previously. ”*

```
Feature: Evernote Online Notes Functionality

@Task2
Scenario: Verify User Can Login To App With Right Credentials
    Given User Lands On Login Page
    When User Enters Email "<username>"
    Then User Clicks on Continue/Login Button
    Then Verify user "<exists>"
    And User Enters Password "<password>" and Login if user "<exists>"

    Examples:
    | username                  | password  | exists     |
    | m.haider.ijaz@gmail.com   | Habcd5432 | exists     |
    | random@testMail.com       | test      | not Exists |
```



```gherkin
@Task2
Scenario: Verify Created Notes should be visible to the user even after logout
    Given User Lands On Login Page
    When User Enters Email "<username>"
    Then User Clicks on Continue/Login Button
    Then User Enters Password "<password>"
    Then User Clicks on Continue/Login Button
    Then User navigate to Notes
    Then User clicks add notes button
    Then Add Notes Title "<title>"
    Then Add Notes Description "<description>"
    Then User Logout the session
    Then User Navigate to Login Page
    When User Enters Email "<username>"
    Then User Clicks on Continue/Login Button
    Then User Enters Password "<password>"
    Then User Clicks on Continue/Login Button
    Then User navigate to Notes
    Then Verify Created Note with Title "<title>" and Description "<description>"
    And Move it to trash

  Examples:
  | username               | password  | title                        | description                                    |
  | m.haider.ijaz@gmail.com | Habcd5432 | Evernote Functionality Test | Created Notes should be visible even after logout |
```



Feature: Evernote Online Notes Functionality

@Task2

Scenario: Verify User Can Login To App With Right Credentials 3m 16s

Shown

#1 - m.haider.ijaz@gmail.com

✓ **Given** User Lands On Login Page

✓ **When** User Enters Email "m.haider.ijaz@gmail.com"

✓ **Then** User Clicks on Continue/Login Button

✓ **Then** Verify user "exists"

✓ **And** User Enters Password "Habcd5432" and Login if user "exists"

Examples:

#	Preview	Result	username	password	exists	Duration
1	●	✓	m.haider.ijaz@gmail.com	Habcd5432	exists	2m 58s
2	○	✓	random@testMail.com	test	not Exists	17s 961ms

@Task2

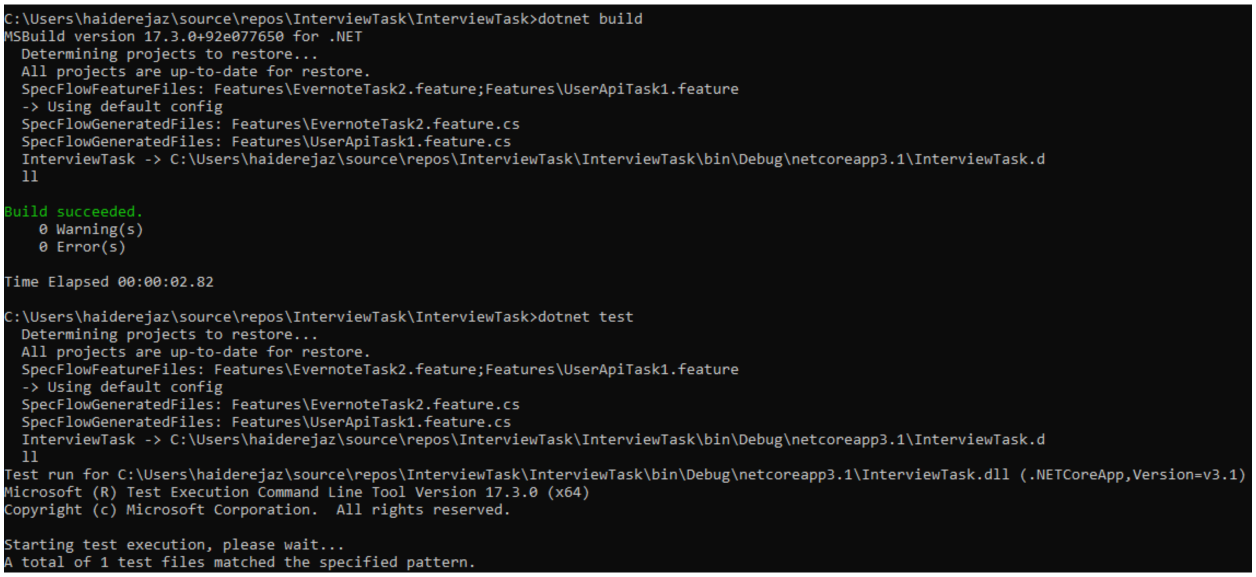
Scenario: Verify Created Notes should be visible to the user even after logout 7m 16s

Shown

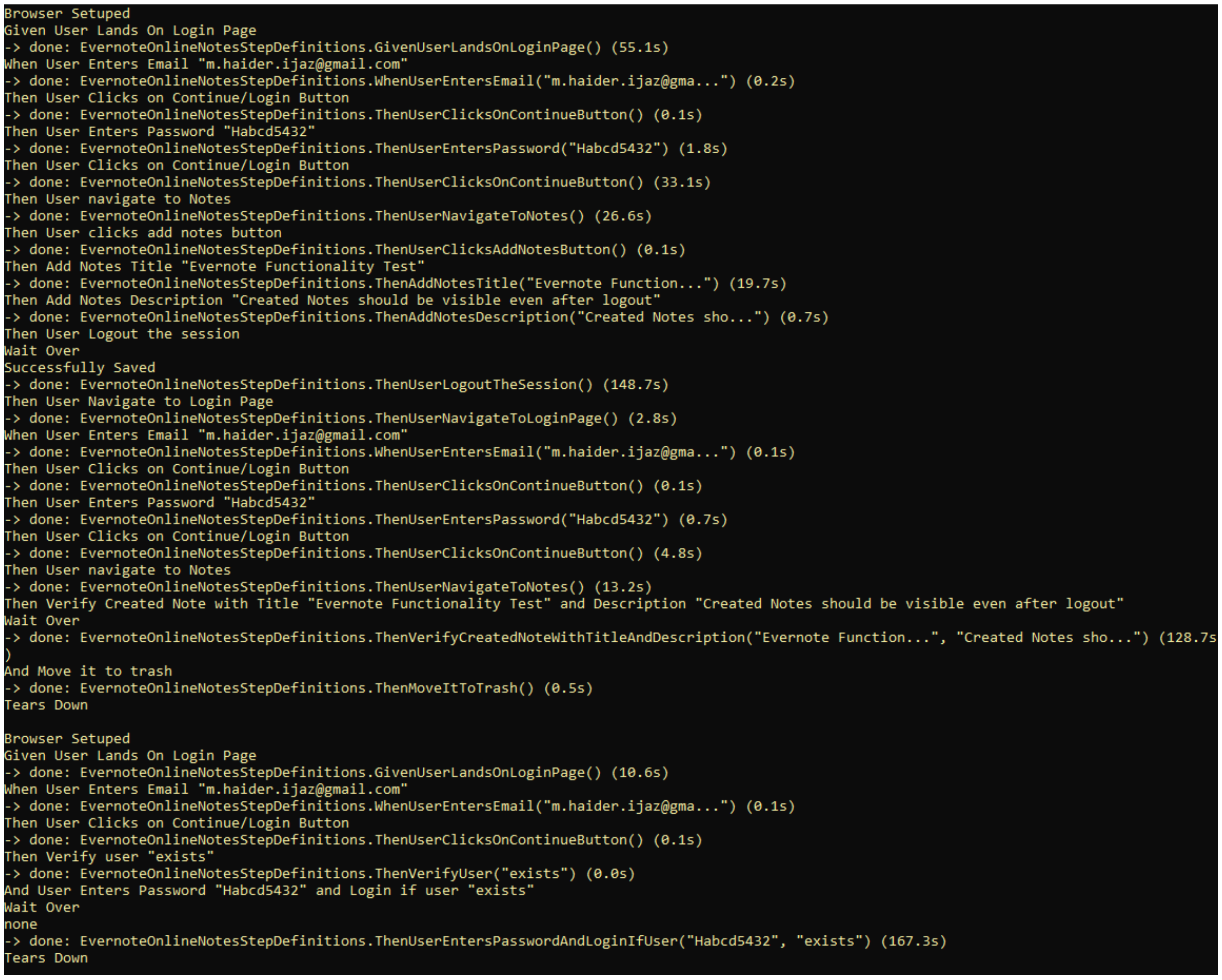
Scenario

Given User Lands On Login Page

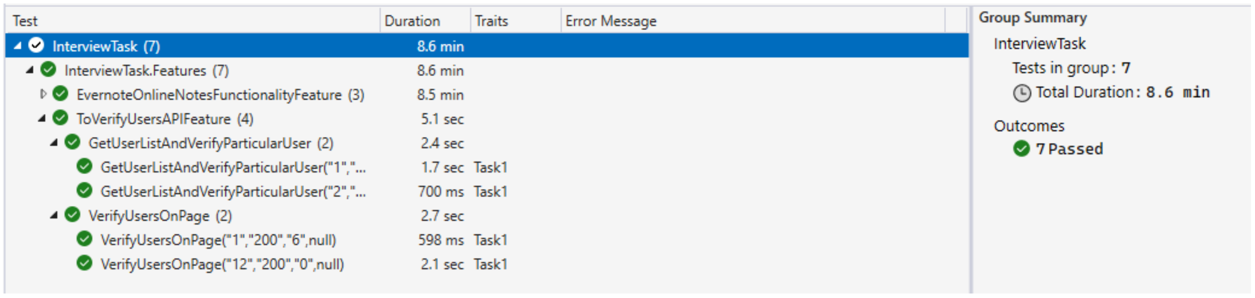
When User Enters Email "<username>"



```
C:\Users\haiderejaz\source\repos\InterviewTask\InterviewTask>dotnet build
MSBuild version 17.3.0+92e077650 for .NET
  Determining projects to restore...
  All projects are up-to-date for restore.
  SpecFlowFeatureFiles: Features\EvernoteTask2.feature;Features\UserApiTask1.feature
  -> Using default config
  SpecFlowGeneratedFiles: Features\EvernoteTask2.feature.cs
  SpecFlowGeneratedFiles: Features\UserApiTask1.feature.cs
  InterviewTask -> C:\Users\haiderejaz\source\repos\InterviewTask\InterviewTask\bin\Debug\netcoreapp3.1\InterviewTask.d
  ll

Build succeeded.
    0 Warning(s)
    0 Error(s)

Time Elapsed 00:00:02.82

C:\Users\haiderejaz\source\repos\InterviewTask\InterviewTask>dotnet test
  Determining projects to restore...
  All projects are up-to-date for restore.
  SpecFlowFeatureFiles: Features\EvernoteTask2.feature;Features\UserApiTask1.feature
  -> Using default config
  SpecFlowGeneratedFiles: Features\EvernoteTask2.feature.cs
  SpecFlowGeneratedFiles: Features\UserApiTask1.feature.cs
  InterviewTask -> C:\Users\haiderejaz\source\repos\InterviewTask\InterviewTask\bin\Debug\netcoreapp3.1\InterviewTask.d
  ll
Test run for C:\Users\haiderejaz\source\repos\InterviewTask\InterviewTask\bin\Debug\netcoreapp3.1\InterviewTask.dll (.NETCoreApp,Version=v3.1)
Microsoft (R) Test Execution Command Line Tool Version 17.3.0 (x64)
Copyright (c) Microsoft Corporation.  All rights reserved.

Starting test execution, please wait...
A total of 1 test files matched the specified pattern.
```



```
Browser Setuped
Given User Lands On Login Page
-> done: EvernoteOnlineNotesStepDefinitions.GivenUserLandsOnLoginPage() (55.1s)
When User Enters Email "m.haider.ijaz@gmail.com"
-> done: EvernoteOnlineNotesStepDefinitions.WhenUserEntersEmail("m.haider.ijaz@gma...") (0.2s)
Then User Clicks on Continue/Login Button
-> done: EvernoteOnlineNotesStepDefinitions.ThenUserClicksOnContinueButton() (0.1s)
Then User Enters Password "Habcd5432"
-> done: EvernoteOnlineNotesStepDefinitions.ThenUserEntersPassword("Habcd5432") (1.8s)
Then User Clicks on Continue/Login Button
-> done: EvernoteOnlineNotesStepDefinitions.ThenUserClicksOnContinueButton() (33.1s)
Then User navigate to Notes
-> done: EvernoteOnlineNotesStepDefinitions.ThenUserNavigateToNotes() (26.6s)
Then User clicks add notes button
-> done: EvernoteOnlineNotesStepDefinitions.ThenUserClicksAddNotesButton() (0.1s)
Then Add Notes Title "Evernote Functionality Test"
-> done: EvernoteOnlineNotesStepDefinitions.ThenAddNotesTitle("Evernote Function...") (19.7s)
Then Add Notes Description "Created Notes should be visible even after logout"
-> done: EvernoteOnlineNotesStepDefinitions.ThenAddNotesDescription("Created Notes sho...") (0.7s)
Then User Logout the session
Wait Over
Successfully Saved
-> done: EvernoteOnlineNotesStepDefinitions.ThenUserLogoutTheSession() (148.7s)
Then User Navigate to Login Page
-> done: EvernoteOnlineNotesStepDefinitions.ThenUserNavigateToLoginPage() (2.8s)
When User Enters Email "m.haider.ijaz@gmail.com"
-> done: EvernoteOnlineNotesStepDefinitions.WhenUserEntersEmail("m.haider.ijaz@gma...") (0.1s)
Then User Clicks on Continue/Login Button
-> done: EvernoteOnlineNotesStepDefinitions.ThenUserClicksOnContinueButton() (0.1s)
Then User Enters Password "Habcd5432"
-> done: EvernoteOnlineNotesStepDefinitions.ThenUserEntersPassword("Habcd5432") (0.7s)
Then User Clicks on Continue/Login Button
-> done: EvernoteOnlineNotesStepDefinitions.ThenUserClicksOnContinueButton() (4.8s)
Then User navigate to Notes
-> done: EvernoteOnlineNotesStepDefinitions.ThenUserNavigateToNotes() (13.2s)
Then Verify Created Note with Title "Evernote Functionality Test" and Description "Created Notes should be visible even after logout"
Wait Over
-> done: EvernoteOnlineNotesStepDefinitions.ThenVerifyCreatedNoteWithTitleAndDescription("Evernote Function...", "Created Notes sho...") (128.7s
)
And Move it to trash
-> done: EvernoteOnlineNotesStepDefinitions.ThenMoveItToTrash() (0.5s)
Tears Down

Browser Setuped
Given User Lands On Login Page
-> done: EvernoteOnlineNotesStepDefinitions.GivenUserLandsOnLoginPage() (10.6s)
When User Enters Email "m.haider.ijaz@gmail.com"
-> done: EvernoteOnlineNotesStepDefinitions.WhenUserEntersEmail("m.haider.ijaz@gma...") (0.1s)
Then User Clicks on Continue/Login Button
-> done: EvernoteOnlineNotesStepDefinitions.ThenUserClicksOnContinueButton() (0.1s)
Then Verify user "exists"
-> done: EvernoteOnlineNotesStepDefinitions.ThenVerifyUser("exists") (0.0s)
And User Enters Password "Habcd5432" and Login if user "exists"
Wait Over
none
-> done: EvernoteOnlineNotesStepDefinitions.ThenUserEntersPasswordAndLoginIfUser("Habcd5432", "exists") (167.3s)
Tears Down
```



Test	Duration	Traits	Error Message
InterviewTask (7)	8.6 min		
InterviewTask.Features (7)	8.6 min		
EvernoteOnlineNotesFunctionalityFeature (3)	8.5 min		
ToVerifyUsersAPIFeature (4)	5.1 sec		
GetUserListAndVerifyParticularUser (2)	2.4 sec		
GetUserListAndVerifyParticularUser("1","...	1.7 sec	Task1	
GetUserListAndVerifyParticularUser("2","...	700 ms	Task1	
VerifyUsersOnPage (2)	2.7 sec		
VerifyUsersOnPage("1","200","6",null)	598 ms	Task1	
VerifyUsersOnPage("12","200","0",null)	2.1 sec	Task1	

Group Summary
InterviewTask
Tests in group: 7
Total Duration: 8.6 min
Outcomes
✓ 7 Passed

5.3 Bonus Requirement

As I already mentioned on email, I'm trying to access those weather API's https://www.metaweather.com/api/ and these are not accessible and I've found "404 page not found" error more on that got the news from Google that they have closed their services.

But if you share with me anything new as the bonus requirement so yes, I'll submit it as soon as possible, in this repository only.