Test Analyst

Competition Task Project Mars

Introduction:

Welcome! The Mars Competition Task is designed to familiarize you with newer technologies, this should give you exposure and experience on how to use these technologies in practice, furthermore here you should be more focused on trying to meet the deadline.

The following document will describe what to do but It may not necessarily tell you how to carry out the task. This has been done deliberately as since now you are a part of MVP Studio and just like working for any company you have to find out solutions to problems you face on your own (of course you can always reach out to your mentor for guidance but given that you have followed all the right steps and have done your due diligence). We would like to highlight how important this is for your development as in the IT world you are required to work independently and drive to solve problems on your own. The good news is by now you should know your fellow interns and you could also reach out to them for help and tips.

You should have a good idea where to start as you have received 6 weeks of training and from the onboarding task, however if you have any questions we are always here to help. Please utilize Slack to develop a good network of communications with peers, to seek help on Google, use Question Hub and maybe even post questions there if you need too.

Now that we have gone through the housekeeping rules, let's get onto the fun stuff, starting your next task!

Competition Task Spec:

Overview:

The second task is all about introducing you to how we use test data within tests! It is a very important concept when it comes to automation testing. Having good test coverage is one thing to worry about but making sure you can supply your test with test data without hardcoding it within your test is another skill to master. As part of the competition task you will be required to master the art of test data. You will be looking at user stories that will help you learn exactly this along with other exciting technologies you should be familiar with.

Learning outcomes:

It is very important to understand the learning outcomes, what you will be learning and technologies you will be working with!

- Be able to analyse requirements and to come up with test scenarios/cases which will be used for writing manual and automation tests.

- Be able to write automation tests using Selenium WebDriver with C# by making use of NUnit framework if you are not familiar with this then see link for more information around what NUnit is and documentation around it (https://docs.nunit.org/articles/nunit/intro.html).
- Implement the POM pattern as done previously in the Onboarding task.
- Be able to implement Extent Report and incorporate this into your tests, please see the following link around what Extent Report is (https://www.extentreports.com/docs/versions/4/net/).
- Be able to understand JSON and implement JSON reader so that It can be used to retrieve test data to use within the automation tests.
- Be able to apply correct coding standards and design patterns into practice so that you write clean code. (see link for more information about this https://www.dofactory.com/csharp-coding-standards)
- Learn and apply best practices around GitHub and the use of Git commands this means working for branches and raising PRs. https://docs.github.com/en/pull-requests/collaborating-with-pull-requests/proposing-changes-to-your-work-with-pull-requests)

Now that we have gone through the learning outcomes, time to look at the user stories!

Note: you should now know how to start up project Mars via docker!

User Story 1:

Description:

Come up with test cases for education and certification around profile features in Project Mars.

Use case:

As a user I would be able to show what certification and education I have done.
So that the people that are interested in my profile can see the education and certifications I hold.

Acceptance Criteria:

- To come up with test cases, you should cover all test paths and most test cases are covered! (See tips).
- Provide an excel document that has the test cases you have come up with (should clearly be able to see the steps taken in order to carry out the test.
- The excel sheet should be pushed to GitHub with a pull request created.
- You should also carry out manual testing using the test cases you have come up with.

Best Practices/Tips:

Below are some best practices and tips to think about!

Tip 1:
Remember to play around with the application, try to explore the functionality around what the certification and education features have to offer in the profile page. Keep in mind the steps you are doing and also don't be scared to play around with some of the other features available!

Tip 2:
Think about authentication and incorporate that into your test cases

Tip 3:
Types of test cases

• Basic positive tests (happy paths)

• Extended positive testing with optional parameters
• Negative testing with valid input
• Negative testing with invalid input
• Destructive testing

Below are some examples of the above scenarios!
Happy path tests check basic functionality and the acceptance criteria of the API. We later extend positive tests to include optional parameters and extra functionality.

The next group of tests is negative testing where we expect the application to gracefully handle problem scenarios with both valid user input (for example, trying to add an existing username) and invalid user input (trying to add a username which is null).

Destructive testing is a deeper form of negative testing where we intentionally attempt to break the API to check its robustness (for example, sending a huge payload body to overflow the system).

User Story 2:

Description:

Automate the test cases for certification and education features. Remember to use the NUnit framework with Selenium web driver using C#, make sure you incorporate Extent Reporter, JSON Reader for test data using JSON file and POM pattern into your automation framework.

Use case:

As a user I would be able to show what certification and education I have done.
So that the people that are interested in my profile can see the education and certifications I hold.

Acceptance Criteria:

- Write automation tests from the test cases that were written for certification and education.
- Make sure all tests pass.

Best Practices/Tips:
Below are some best practices and tips to think about!

Tip 1:
You have used the correct technology stack and have mastered the use of test data. This task ensures if the tests are written using NUnit with Selenium web driver in C#, you have used a JSON reader and have used JSON files to incorporate test data which is being passed to the test. Extent Report is used and can see the report being generated, try to incorporate at least one screenshot functionality in one test case. POM pattern is implemented and correct coding standards are used.

Tip 2:
In the task you are not provided with a sample solution, therefore you can either create one from scratch or use the from your 6 weeks training with industry connect on NUnit. Remember that you can copy the code if you are reusing, remove unwanted code that not should not be there and create a new repo on your own GitHub account (ensure naming convention on repo is meaningful e.g. CompetionTaskMars). If you have trouble setting this up please feel free to message your master mentor for help and they will guide you further on how to move forward with this.

Tip 3:
Remember to work on feature branches and **do not work** on the main or master branch! The best thing to do is before implementing any test logic for automation, come up with the skeleton code and push that up onto your GitHub repository and then branch of the master/main branch. (Please see learning outcomes for more information around this).

Tip 4:
When converting JSON files into objects you might first need to create the object classes, make sure you create individual classes for any objects you create and not create them all in one file. You can use the following link for an example of a JSON reader (https://code-maze.com/csharp-read-and-process-json-file/amp/).

Tip 5:
Remember to use Google, Question hub or Stack Overflow if you are stuck on a problem, we know at times it can be frustrating to be stuck on a problem, but it is crucially important that you learn to solve problems and find solutions independently. If you seem like you are not getting anywhere and have also posted a question on Question Hub then you can seek help from your mentor or reach your master mentor the QA lead of MVP Studio.

Raising PR/Reviewing Code:

Working On Feature Branches And Pull Request:

It is very important that you learn to work on feature branches rather than main or master branch. Once you are ready to submit each task (user story 1 & 2), you will need to create a PR please follow the instruction on the following powerpoint on pages 22-23 which shows you how to raise a PR, the steps should very straight forward to follow https://docs.google.com/presentation/d/1KMg3UBtcl5DNzP0Ny6rmclr5FAETVyub/edit#slide=id.p58 .

Reviewing What You Will Be Accessed On:

The things your mentor will look out for when reviewing your code will check how well you have followed best coding practices such as

- Did you implement the POM pattern
- Did you implement the Extend Reporter, with one test case using screenshots.
- Did you use NUnit with Selenium web driver
- Did you use JSON Reader, JSON files to pass test data to the tests
- Object classes are created correctly and JSON binds to objects correctly.
- Are your code method sizes reasonable (30-40 lines any more you should break the methods into smaller methods).
- Are the tests cleanly written and can understand what each test is about by following the steps.
- Making use of constructors if needed.
- Are you using a common way of initialising the Web Driver as in using a CommonDriver class.
- Are you making use of Hooks. https://docs.nunit.org/articles/nunit/writing-tests/setup-teardown/index.html
- Are you asserting what makes sense and if your asserts are in the correct place; where the tests live or assertion helper classes and not in pages.
- Asserting the correct expected data with the actual data (using expected test data from the JSON files and not hardcoded).

- Class names, variables and method names are following the correct naming conventions (see learning outcomes).
- Make sure that you have captured all the different test scenarios such as Basic positive tests (happy paths, Extended positive testing with optional parameters, Negative testing with valid input, Negative testing with invalid input and Destructive testing).

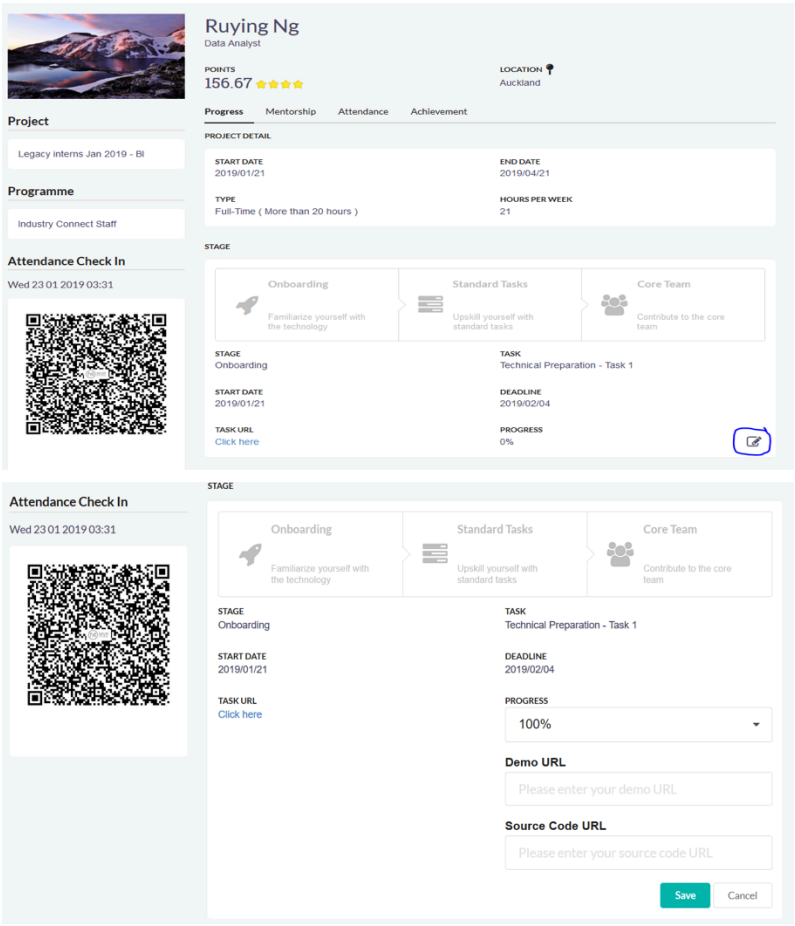
Please also let your mentor know you have submitted a task for review, and also send them the PR link!

Task Submission:

Follow the steps below

1. Once you have done all your user stories and you have a PR raised on GitHub.
2. Submit your GitHub link on the Internship Portal at the IndustryConnect.IO website. Click on the edit progress icon and enter the URLs.
    a. Demo URL : (ignore not required)
    b. Source Code URL: Google drive URL

Note: Your progress has to be at least 50% to submit the URLs.



Now That we have gone through everything it's time for you to dig in and have some fun! If you have any questions please feel free to get in touch with your mentor, master mentor or the MVP internship team.

All the best!!!

and if you need help refreshing basic C# fundamentals please see https://www.w3schools.com/cs/index.php.