A picture containing font, logo, text, symbol

Description automatically generated

**Test Analyst**

**Onboard Task Project Mars**

**Introduction:**

Welcome! The Mars Onboard Task is designed to familiarize you with technologies which are being used by MVP Studio and in fact any company nowadays.

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.

You should have a good idea where to start as you have received 6 weeks of training 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 first task!

**Onboarding Task Spec:**

**Overview:**

The first task is about introducing you to Project Mars! Project Mars is a system designed to be used by recruiters which offers various features which you will be testing and exploring! As part of the onboarding task you will be looking into smaller tasks which will be broken down into smaller user stories (User stories is what we give a tech term which defines that task specs for more information around this please feel free to look at the line below).

<https://www.visual-paradigm.com/guide/agile-software-development/what-is-user-story/>

**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 Specflow framework with making use of BBD. (see link for more information around what is Specflow and documentation around it <https://specflow.org/benefits/developers-code/> )
* 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.
* Learning how to work on feature branches and creating pull requests. (see link for more information about pull request

<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 set up Project Mars!

**Setup:**

To setup Project Mars please see the link on how to do this. <https://docs.google.com/document/d/1crUJuzOtaxP2fNaX4ipazLL3rvBrlcDSKbTdXBV7oU0/edit>

Once you have done this you should see the following (see the below image, remember the URL locally for Mars is <http://localhost:5000/> and make sure you create your own account so that you can login or use username: [mvpstudio.qa@gmail.com](mailto:mvpstudio.qa@gmail.com) password: SydneyQa2018 ). If you have trouble setting this up please message your mentor for help.

A screenshot of a computer

Description automatically generated with low confidence

Let's now have a look at the user stories which you will be working on. We have broken the onboarding task down to smaller user stories to make it easier to understand what is required to be done. Remember to do one user story first submit for review before moving onto the next!

**User Story 1:**

**Description:**

Come up with test cases around languages and skills around profile features in Project Mars.

**Use case:**

**As a** user I would be able to show what languages and skills I know.

**So that** the people seeking for skills and languages can look at what details 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.
* The excel sheet should be pushed to GitHub with a pull request created.
* You should also carry out manually 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 language and skills 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 language and skills. Remember to use the Specflow framework with BDD (Given, When, Then etc with step definitions and pages please refer to Specflow docs mentioned above for more info), you may use the following <http://git.mvp.studio/qa-examples/onboarding.specflow> provided. Note: please change the credentials in the solution file according to your details.

**Use case:**

**As a** user I would be able to show what languages and skills I know.

**So that** the people seeking for skills and languages can look at what details I hold.

**Acceptance Criteria:**

* Write automation tests from the test cases that were written for language and skills.
* Make sure all tests pass.

**Best Practices/Tips:**

Below are some best practices and tips to think about!

**Tip 1:**

Remember that once you download the solution project to create a new repo on your own GitHub account, as no MVP GitHub access will be provided. If you have trouble setting the solutions file you can use the one of the Specflow projects you did in your 6 weeks training but, remember to copy the files from the solution project provided so that you can setup the project in the same way how it is done in the solution project that is provide in the link above.

**Tip 2:**

Make sure when implementing feature files make sure the BDD syntax is correct, step definition and pages follow best coding practices, such as making use of the POM (Page Object Model) pattern, use of constructors and also correct coding standards.

**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:**

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
* Are your code method sizes reasonable (30-40 lines any more you should break the methods into smaller methods).
* Are your features following the correct syntax and are using a mixture of table and pass in values. <https://cucumber.io/docs/gherkin/reference/>
* Are your steps clean and if needed are you making use of constructors.
* Are you using a common way of initialising the Web Driver (see CommonDriver.class in the solution example project).
* Are you making use of Hooks.
* Are you asserting what makes sense and if your asserts are in the correct place; in the step definitions or assertion helper classes and not in pages.
* Class names, variables and method names are following the correct naming conventions (see learning outcomes).
* Make sure that you have captured all the different tests 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).

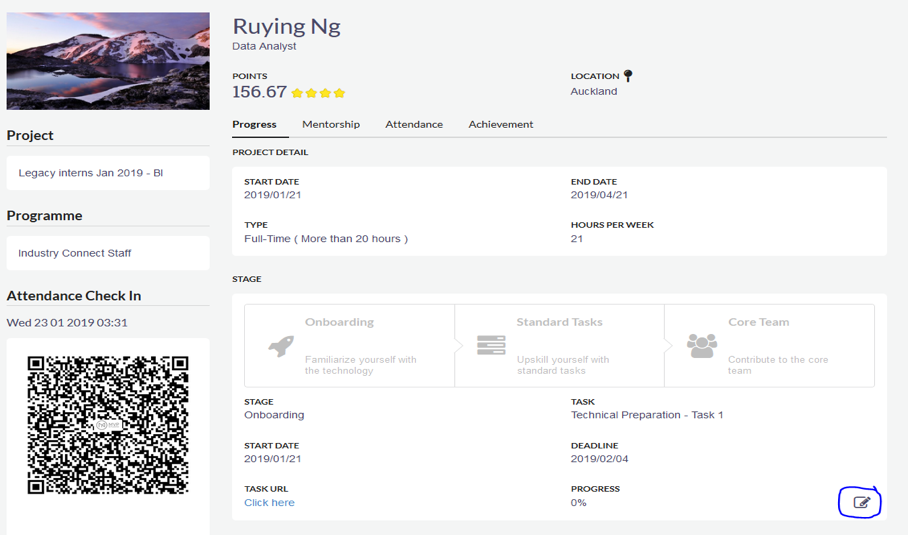
Please also let your mentor know you have submitted a task for review, and also send them the PR link!

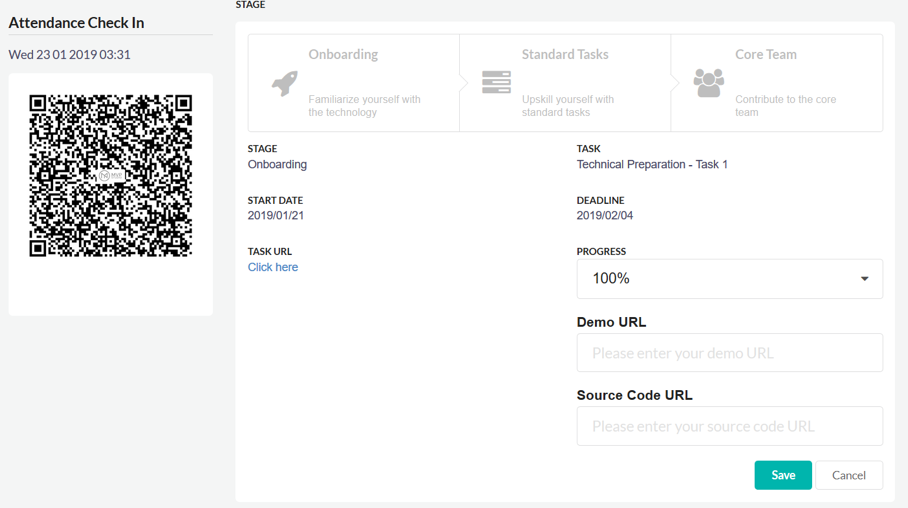
**Task Submission:**

**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.
   1. Demo URL : (ignore not required)
   2. 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>.