**Question 4:**

*The test automation framework automatically updates the organisations test*

*management system with the result of testing via an API. Identify a suitable framework to*

*communicate with this API and explain how it might the be used.*

**Answer Option-1 :**

First of all, as you can see on my project, it can be created TestListener class.

In this class, it can override the chosen test runner adapter and call the Test Management System API in those methods for updates to test results.

In my project, override TestNG's methods and create my own test results for Extent Report.

If I know the payload for this API, I can feed the payload with the results of the test and send to Test Management System via API.

**Answer Option-2:**

If I have integrated the way for Test Management System with extensions or via JWT token, I can create an integration class(this class will use API payload with dynamic parameters, and call API via library) for use on TestListener class.

**Question 5:**

*The website talks to an underlying API which handles communications with back*

*end systems. Required backend functionality is not available when the website is deployed*

*for testing. Describe how you might simulate the bahaviour of the API and what tools you*

*might use.*

Actually, there is a lot of choice for this answer.

* Mock systems
* Virtual Service systems

**Option-1 :**

If I will manage the simulation of API via framework, I can create a library for the mock API’s and integrate this library into the target test framework and manage this mock library via environment parameters.

For example,

* I can use the RestAssuredMockMVC library in the test framework.
* I can use modify the RestSharpResponse in the test framework.

If I will not manage the simulation of API via framework, I can create a Project for the mock API’s and manage it via CI/CD tools like Jenkins, GitLab, Bamboo, TeamCity etc.

For example,

* Swagger
* PostMan
* Mockito
* MockMoon

**Option-2 :**

Virtual Service systems are my favorite actually. I can create an on-prem or cloud base virtual service system and use it in the framework via configuration parameters. The virtual Service system saves time for the creation of mock services. Because it has sniffer adapters for keeping to real API responses for re-use. And those reusable virtual services can modify parameterized via payload variables. This Virtual Service system can be used from any test framework or any backend and frontend project’s unit testing side.

Example tool,

* Hoverfly.io
* Microfocus Data Simulation software
* CA Service Virtualizations