**Test Plan**

**Use-Cases:**

**GetUsers**: To Verify the list of Users

1. To verify users list with id, name and email when users list is available in respective service
2. To Verify empty users list when there are no users in system

**GetUser**: To verify user based on user Id

1. To Verify GetUser when valid User Id is passed to endpoint
2. To Verify GetUser when an invalid User Id is passed to end point
3. To Verify GetUser when an empty parameters are passed to end point

**CreateUser**: To verify creation of user

1. To Verify CreateUser when valid User Data set is passed to endpoint
2. To Verify CreateUser, when user data with Max character lendth(InBound-Outbound testing) is passed to endpoint
3. To Verify CreateUser when incorrect Data Set is passed to endpoint
4. To Verify CreateUser, when empty Data set is passed to endpoint

**UpdateUser**: To Verify update user with email & Name based on UserID

1. Verify updateUser when valid User Data set is passed to endpoint
2. Verify updateUser when valid user Data with no User ID
3. Verify UpdateUser, with invalid User Data set
4. Verify UpdateUser with boundary values with Max & min character length

**DeleterUser**: To Verify Delete User for an specified User ID

1. Delete User when an valid User ID is passed to endpoint
2. Delete User when an empty User ID is passed
3. Delete User when there is no Data available in System

In all above endpoint, we will be verifying Response codes along with UserID, Name, Email for create and update End points.

**Automation:**

***Framework Details:***

Here we have used BDD approach. We have user Cucumber, Rest assured for the given task. We can maintain all Features under Resources folder, Test Runner class under Test package. As of now all the tests are maintained under same class but we can create separate class field based on feature that we work on

**Coverage:**

* Automated one test case for each endpoint.

**Documentation for execution:**

1. Checkout to “ApiAutomationAssessment” branch and test files will be available under “src/test/java/com.test” and feature files will be available under “src/test/java/Resources”
2. Updated “build.gradle” file with all dependencies related to cucumber, rest-assured, junit
3. Run configuration for JUnit should be configured with selecting Project and class as TestRunner
4. Click on Run and tests will be executed. Test Results should be displayed under Junit Runner tab. Further we can export this xml field with Listener and convert it to html format as well.