Project Title:
Demonstration of Automation Skills Learnt from Work Project

Contact Information:
Safwan Rahman
Safwanrahman207@gmail.com
(646) 575-9130

Supervisor:
Nithya Rajaraman
nrajaraman@loc.gov

Abstract:
Automation is the latest and most used method to test applications and websites over manual testing and has become an industry standard for most, if not all, QAs. This project is to demonstrate the skills I have acquired working as an automation engineer on my jobs project while also demonstrating tools being used in a real-life work environment, why they are used, and their pros/cons to using such tools. With the use of postman, I can demonstrate network calls that our everyday websites like YouTube and Facebook use to talk with their servers using APIs and showing how we can automate specific functions to a website so we can test its functionality while observing to see if any bugs/defects arise within the application with the use of Java, Cucumber, Junit, and Maven as the framework to our automation script(s).

Diagram:
To get an understanding of how the automation framework would work and how it would be structured.

```
Start → Project Structure → Dependencies → Feature Files → Step Definition    Test Runner
```
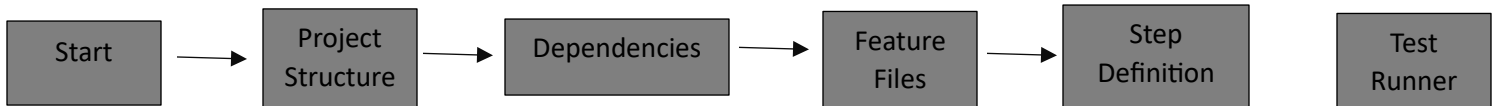
What this diagram is depicting is what the flow and structure will look for the automation tests going forward. The project structure is how we want to organize the entire project. Where we want the runner, the main step definitions, elements (page factory), dependencies, the feature files etc. From there we need the dependencies that will be used to make the automation test work. Dependencies are basically like adding the tools we want. Webdriver is the main dependency we will use to access different browsers without having us to set up drivers for every unique browser. We will also have junit in the dependency file since that is the framework we will be testing with, and even more once I have a list of all the tools required. Feature file comes from cucumber. This is how we write out step by step what we want our test to do using the coding language "Gherkin" which really is just simple sentences using the English language to demonstrate to our business owners/product owners what our tests are doing and so they can easily understand the scenario without having to look through lines and lines of code. The step definition aspect is the bread and butter of this framework. This is where all the code will be put in to verify data, locate buttons on the website, and more. This is where the actual process will take place through code. Finally, the test runner will be the end where it will dictate how we run the entire framework. Do we want reports generation? Do we want a regression run? Maybe a SmokeTest run? This test runner will help us with tags to identify what tests we want to run and whether we want reports generated or not.

Start → Create → Get/Read → Update → Delete

What we have above is the diagram/flow chart of how our postman/API test will go. We will follow the CRUD format which is what we use on my project for our API automation tests. What does CRUD stand for? Create, Read, Update, Delete. This is how we want to the flow to go for the API automation because after creating and messing around with the generated data, we want to clean up any useless and lingering information that is no longer required as we don't want to fill up our database.

Tentative Schedule:
The chart below will demonstrate what I will need to be working on for the remainder of the semester and how I would like to break it up the work. I will use the story points system we used at our job to show severity, importance, and time it will take and it follows the Fibonacci sequence. It goes from 1 point (3-6 hours of work) to 8 pointers (5-7 days of work). 2 points means 6 to 16 hours of work, 3 points means 2 to 3 days of work, and 5 points means 3 to 5 days of work.

Tasks	1	2	3	4	5	6	7	8	9	10	11	12	13
Setup project structure	2												
Figure out/Add all dependencies		5											
Set up feature files			8										
Test Runner		1											
Step Definitions				8	3								
Postman: Find Functional Swagger File For Testing						1							
Setup basic CRUD test using swagger file							5						
Add all verifications in API automation								8	2				
Update Repo to include all files										1			
Research/Begin setting up pipeline											8		
Get schedules pipeline runs												8	

Finalize full run, verify schedule runs send out emails, verify automation scripts run functionally													8

This is pretty much the complexity of the project and how long each task should take for the entire week. I can't determine the actual time and workload it will take due to working on my actual project at my job and finding space between work and these deadlines to work on each of these tasks. This is the scope I would like to follow from till the end of the semester and also whatever task lines up with the week of demos/presentations is what I would like to show when explaining what I am doing.

This about sums up my project and what too expect from me during check ins. My time logs will be an insight to what I am doing day to day at my workplace, but to manage this project I will create tickets for each task on the github board and track the progress there. I will dedicate 10 – 20 hours a week to focus on the project specifically and more when I am not working on workplace project and other classes as well. This is bare bones in my opinion compared to what a highly detailed document would look like as I joined my project midway through when it started so I don't have a good example of how we would show all this in my workplace. This is what I think explains best and shows how my day-to-day workflow would look like and how I would prioritize tasks before a sprint starts. Below will be my repo and my project board to check if my tasks are moving as expected.

Repo: https://github.com/rahmann00dles/4900Project
Project Board: https://github.com/users/rahmann00dles/projects/1