**Quality Engineer Test Automation Take-Home Exercise**

This automated functional test coding task exercises a public-facing webpage. The suggested framework to use is Playwright. Please submit your code within 3 days of receiving the exercise.

**Instructions:**

1. Write an automated test that could function as an automated smoke test for the page

Test Case using POM

package test;

import java.util.concurrent.TimeUnit;

import org.openqa.selenium.WebDriver;

import org.openqa.selenium.firefox.FirefoxDriver;

import org.testng.Assert;

import org.testng.annotations.BeforeTest;

import org.testng.annotations.Test;

import pages.TrajectormedicalHomePage;

import pages.TrajectormedicalLogin;

public class TestTrajectormedicalLogin {

String driverPath = "C:\\geckodriver.exe";

WebDriver driver;

TrajectormedicalLogin objLogin;

TrajectormedicalHomePage objHomePage;

@BeforeTest

public void setup() {

System.setProperty("webdriver.gecko.driver", driverPath);

driver = new FirefoxDriver();

driver.manage().timeouts().implicitlyWait(10, TimeUnit.SECONDS);

driver.get("http://demo.trajectormedical.com/V4/");

}

@Test(priority=0)

public void test\_Home\_Page\_Appear\_Correct() {

objLogin = new TrajectormedicalLogin(driver);

String loginPageTitle = objLogin.getLoginTitle();

Assert.assertTrue(loginPageTitle.toLowerCase().contains("trajectormedical bank"));

objLogin.loginToTrajectormedical ("mgr123", "mgr!23");

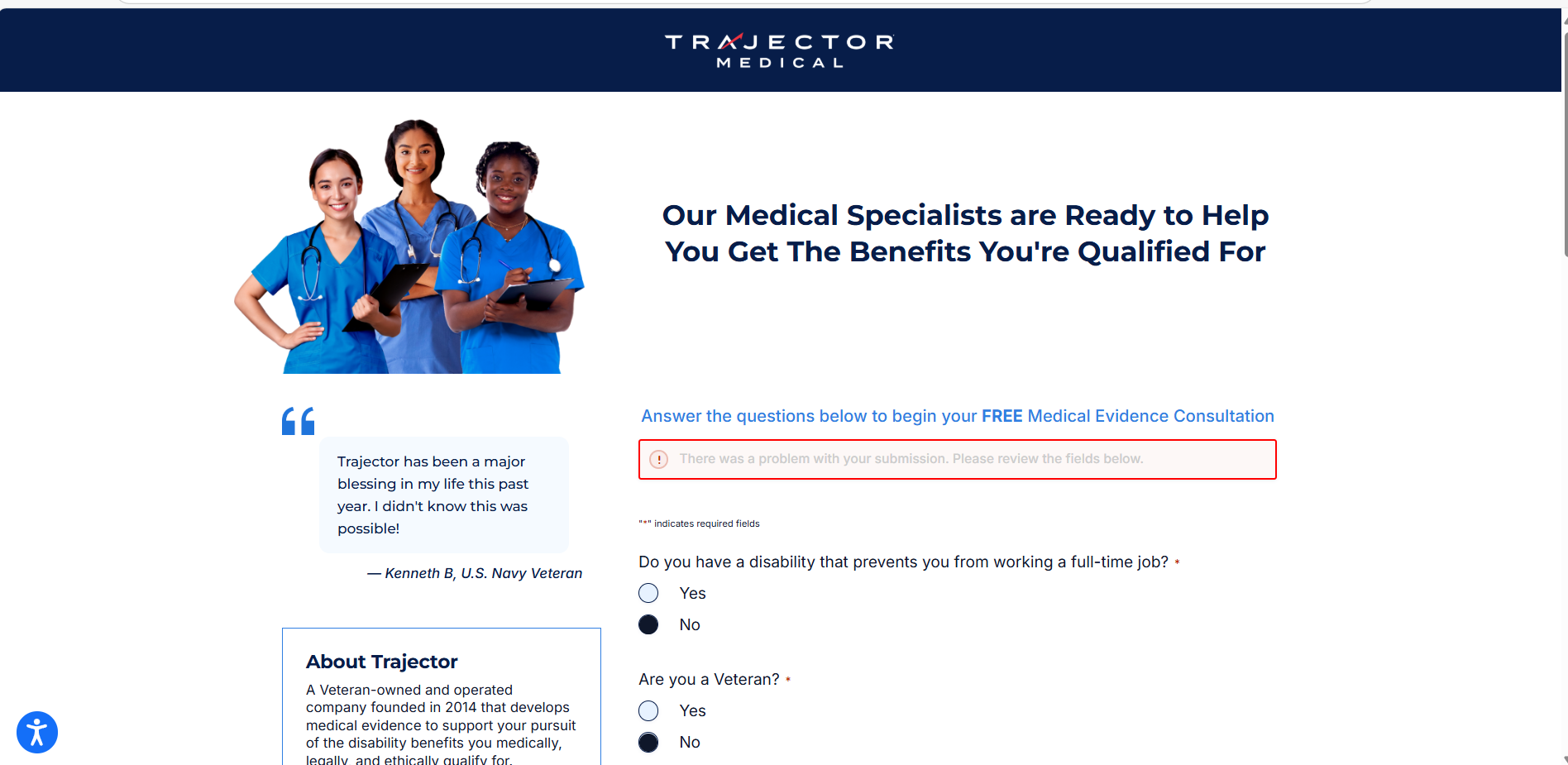
objHomePage = new TrajectormedicalHomePage(driver);

Assert.assertTrue(objHomePage.getHomePageDashboardUserName().toLowerCase().contains("manger id : mgr123"));

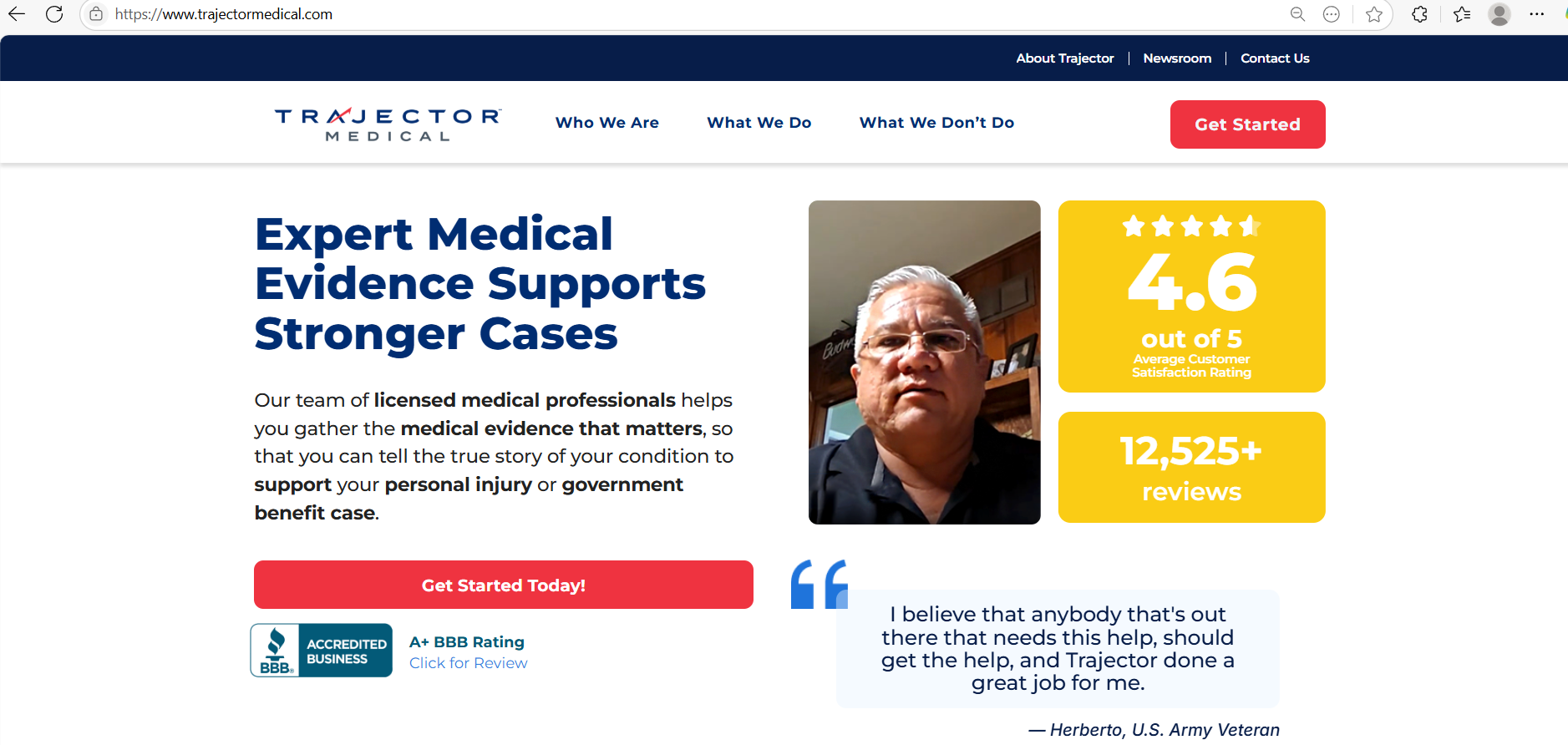
}

}

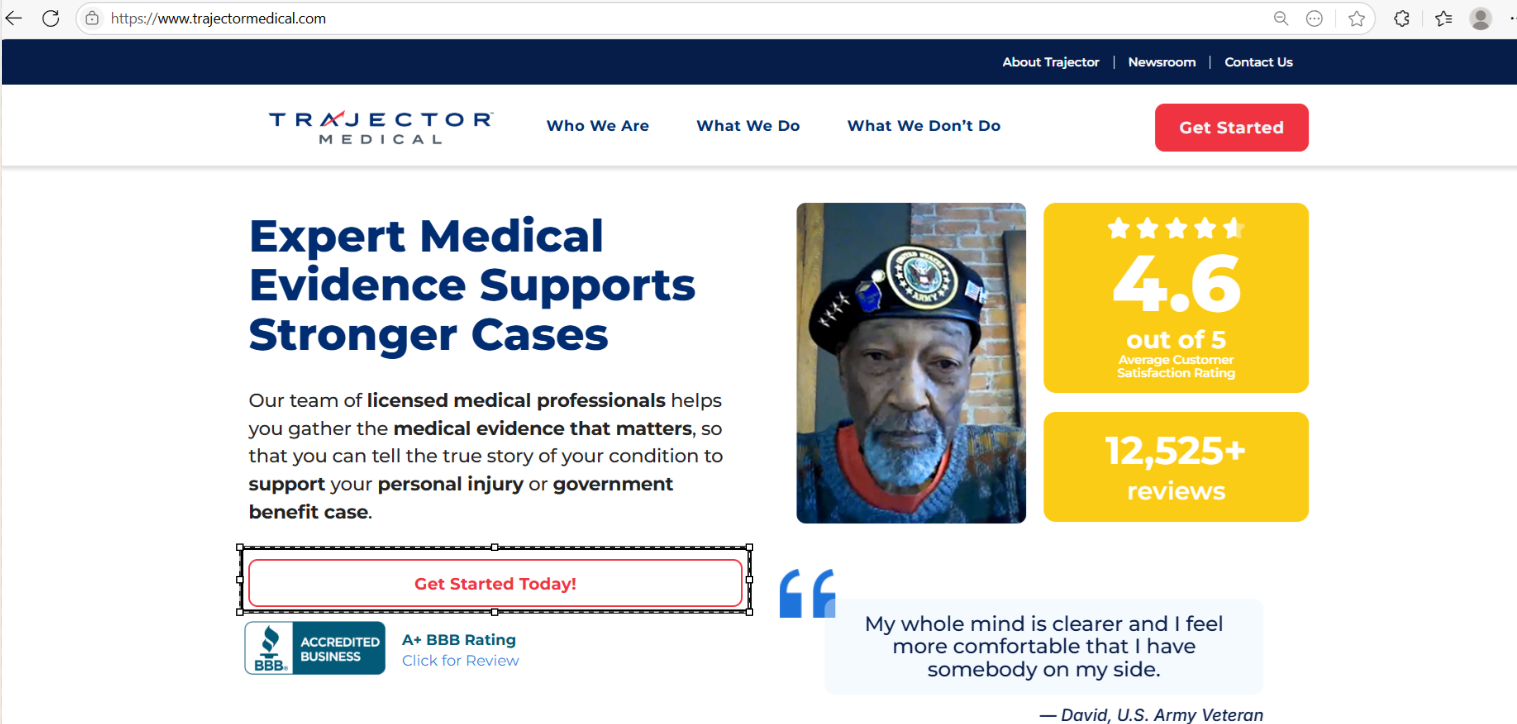
1. Navigate to **trajectormedical.com**



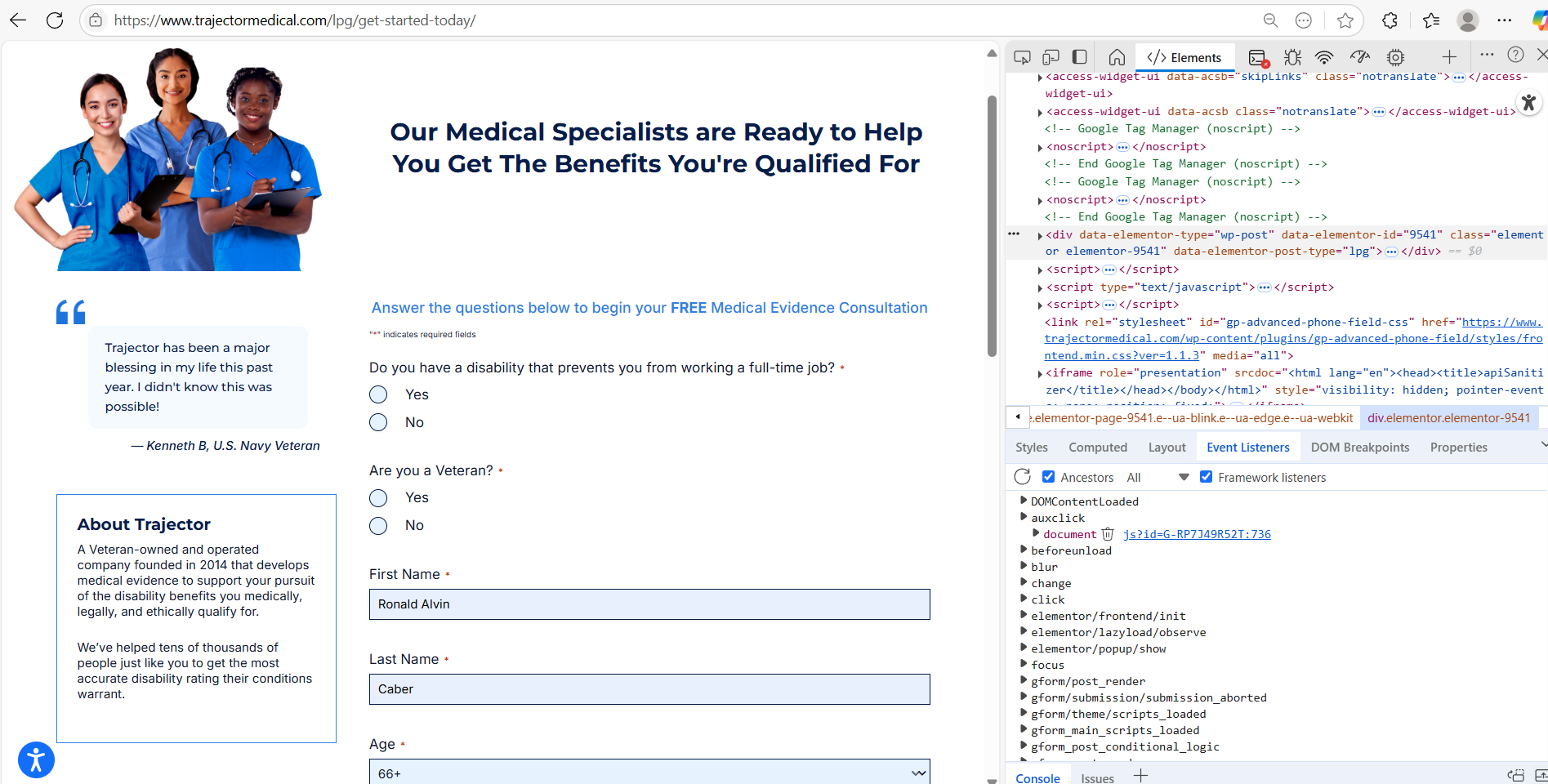
1. Confirm you've landed on **trajectormedical.com**



1. **Click the "Get Started Today!" button. (Please do not actually submit the web form to Trajector)**

****

1. Fill in the form using a randomly generated name each time the test runs



1. Create a page object model for the page

package pages;

>>Home Page POM<<

import org.openqa.selenium.By;

import org.openqa.selenium.WebDriver;

public class trajectormedicalHomePage {

WebDriver driver;

By homePageUserName = By.xpath("//table//tr[@class='heading3']");

public trajectormedicalHomePage(WebDriver driver) {

this.driver = driver;

}

public String getHomePageDashboardUserName() {

return driver.findElement(homePageUserName).getText();

}

}

1. Create a helper function that populates the form fields and call it from your script
2. Capture the test log in a file

[Issues · ZJU-BMI/Medical-Trajectory-Generation](https://github.com/ZJU-BMI/Medical-Trajectory-Generation)

1. Instead of hardcoded value, generate a random First and last name each time the test runs
2. Record a video of your test run and save it
3. Create a data fixture to use in populating the form fields

import { test, expect } from '@playwright/test';  
  
 test('basic test', async ({ page }) => {  
 await page.goto('https://playwright.dev/');  
  
 await expect(page).toHaveTitle(/Playwright/);  
 });

import { test as base } from '@playwright/test';  
import { TodoPage } from './todo-page';  
import { SettingsPage } from './settings-page';  
  
*// Declare the types of your fixtures.*  
type MyFixtures = {  
 todoPage: TodoPage;  
 settingsPage: SettingsPage;  
};  
  
*// Extend base test by providing "todoPage" and "settingsPage".*  
*// This new "test" can be used in multiple test files, and each of them will get the fixtures.*  
export const test = base.extend<MyFixtures>({  
 todoPage: async ({ page }, use) => {  
 *// Set up the fixture.*  
 const todoPage = new TodoPage(page);  
 await todoPage.goto();  
 await todoPage.addToDo('item1');  
 await todoPage.addToDo('item2');  
  
 *// Use the fixture value in the test.*  
 await use(todoPage);  
  
 *// Clean up the fixture.*  
 await todoPage.removeAll();  
 },  
  
 settingsPage: async ({ page }, use) => {  
 await use(new SettingsPage(page));  
 },  
});  
export { expect } from '@playwright/test';

1. Create a helper function that populates the form fields and call it from your script

>>Define the Helper Function<<

function populateFormFields(fieldData) { for (let fieldId in fieldData) { const field = document.getElementById(fieldId); // Fetch form field by ID if (field) { field.value = fieldData[fieldId]; // Set the value to the field } } }

>>Call the Helper Function<<

document.addEventListener('DOMContentLoaded', () => { const data = { username: "johndoe", email: "johndoe@example.com", password: "securepassword123" }; populateFormFields(data); // Call function with data object });

1. Change the screen size to a mobile viewport size

Setting The Viewport

<meta name="viewport" content="width=device-width, initial-scale=1.0">

Image

<img src="img\_girl.jpg" **style="width:100%;"**>

**When you are ready to submit your code:**

* Create a personal repo in GitHub for your code and push it to the repo
* Invite Trajector team reviewers as collaborators and provide appropriate level permissions to pull down the code:
  + Pfrancis-trajector
  + Gpurvis
  + Spoorthykakarla28
  + ts-ashishkumar

**Please do not actually submit the webform for evaluation**