QA Automation Challenge

This project is a web test automation framework using Cypress version 13, written in JavaScript. It utilizes Mochawesome for generating stylish and informative test reports. It tests the webpage PetSwagger Version 2 for API and Load Testing using **Cypress** for API tests and **K6** for load tests. To run locally please modify .env JSON and URL under K6.

API Testing

For API Testing, 3 scenarios were automated:

1. **Create a User**
2. **Search for the created user**
3. **Update the name and email of the user and assert the changes**

API Reference

This project automates different API responses from https://petstore.swagger.io/ using the Cypress framework and the cypress-plugin-api created by Filip Hric, generating reports and running in GitHub Actions as CI.

List User Details

```
GET /users/{username}
```

Request type	Endpoints	Expected Response Code
`GET`	`/users/{username}`	`200`

Create User

```
POST /users/{username}
```

Request type	Endpoints	Request Body	Expected Response Code
`POST`	`/users/{username}`	`"id": 0, "username": "string", "firstName": "string", "lastName": "string", "email": "string", "password": "string", "phone": "string", "userStatus": 0`	`200`

Modify Users

```
PUT /users/{username}
```

Request type	Endpoints	Request Body	Expected Response Code
`PUT`	`/users/{username}`	`"id": 0, "username": "string", "firstName": "string", "lastName": "string", "email": "string", "password": "string", "phone": "string", "userStatus": 0`	`200`

DELETE

```
DELETE /users/{username}
```

Request type	Endpoints	Expected Response Code
`DELETE`	`/users/{username}`	`200`

Load Testing with K6

In addition to API testing, **K6** has been integrated into the project for load testing. **K6** is a powerful tool for testing performance under heavy load. The following test cases were automated using **K6**:

1. **Create User Load Test**
2. **Search User Load Test**
3. **Modify User Load Test**
4. **Delete User Load Test**

These tests simulate concurrent users performing these operations to assess the performance and reliability of the API under load.

How to Run K6 Load Tests

To run the K6 load tests, use the following command:

```
npm run k6:run
```

This command will execute the load tests and generate reports for analysis.

Load Test Results

After running the load tests, you can find the **result analysis** in the `loadTests` folder in **PDF format** for a detailed view of the performance metrics and observations.

Tech Stack

- JavaScript
- Node.js
- Cypress.io
- cypress-plugin-api
- K6
- GitHub Actions
- cypress-mochawesome-reporter

Run Locally

Required to run the project

- Node.js

Steps to Run the Project

1. **Clone the repository**:

```
git clone https://github.com/edgarysabel/e2e-api-automation-test.git
```

2. **Install dependencies**:

```
npm install
```

3. **Run Cypress tests in headless mode**:

```
npm run cy:run
```

4. **Run Cypress tests in headed mode**:

```
npm run cy:open
```

5. **Run K6 load tests**:

```
npm run k6:run
```

This will execute all the tests and generate reports at the end of the execution.

CI with GitHub Actions

CI has been configured with GitHub Actions for ease of use and integration since the project is already hosted on GitHub. To run it, just go to **Actions** and trigger the workflow `Run QA Integration Tests` under your preferred branch. Additionally, the pipeline runs automatically whenever there is a new commit.

Note: To commit or run workflows, please contact me at edgarysabel@gmail.com.

Test Reports

- **Mochawesome** is used to generate standalone HTML reports after test execution. You can find the report in the `cypress/reports/mochawesome-report` directory. Open `mochawesome.html` in your browser to view the report.
- **Allure Report** is also configured and stored with GitHub Pages. After the pipeline runs, the reports are generated. To access the reports, visit https://edgarysabel.github.io/e2e-api-automation-test/.

For K6 load testing, the results are stored in **PDF format** under the `loadTests` directory.

Project Structure

UI Testing Configuration

The file `cypress.env.json` under the root directory contains the necessary credentials for UI testing. Working credentials are included with the project.

```
{
  "FRONTEND_URL": "https://demoblaze.com/",
  "API_ENDPOINT": "https://petstore.swagger.io/v2"
}
```

Must Know

- For invalid API scenarios, `failOnStatusCode: false` is used, allowing tests to continue and perform assertions.
- The project uses a custom **Page Object Model (POM)** pattern with JavaScript and Cypress.
- Static data for test cases is stored under the `fixture` directory.