**TestOps Task**

Write test cases using any programming language and framework (Python + Selenium is preferable)

1. Visit <https://useinsider.com/> and check Insider home page is opened or not
2. Select “Company” menu in navigation bar, select “Careers” and check Career page, its Locations, Teams and Life at Insider blocks are opened or not
3. Go to <https://useinsider.com/careers/quality-assurance/>, click “See all QA jobs”, filter jobs by Location - Istanbul, Turkey and department - Quality Assurance, check presence of jobs list
4. Check that all jobs’ Position contains “Quality Assurance”, Department contains “Quality Assurance”, Location contains “Istanbul, Turkey”
5. Click “View Role” button and check that this action redirects us to Lever Application form page

**Containerize Test Project**:  
Modify your test project to run in a Kubernetes environment using Docker images.

* Create Docker images for the test project (using a Dockerfile).
* Ensure that the necessary ports are exposed.
* Push the Docker images to a container registry (e.g., Docker Hub or Amazon ECR).

**Set up Kubernetes Pods for Selenium Test Execution**:

* Create two distinct pods for your Selenium tests:
  1. **Test Case Controller Pod**: This pod will be responsible for reading and managing test cases.
  2. **Chrome Node Pod**: This pod will run the actual Selenium tests in a headless Chrome browser.
* The **Test Case Controller Pod** should be able to pass test cases to the **Chrome Node Pod** for execution.
* Configure **Inter-Pod Communication** using Kubernetes **Services** or direct **DNS** to ensure that the Test Controller Pod can communicate with the Chrome Node Pod.

**Create Kubernetes Resources**:

* Use **Helm** or write **YAML files** for:
* Deployment of the Test Case Controller Pod.
* Deployment of the Chrome Node Pod (using selenium/node-chrome or similar images).

Ensure that the Chrome Node Pod is dynamically created based on the **node\_count** parameter (min = 1, max = 5).

**Write a Python Script for Kubernetes Deployment and Test Execution**:

* Write a **Python script** that:
* Deploys the Kubernetes resources (Test Controller Pod, Chrome Node Pod) based on the **node\_count** parameter.
* Passes the test cases from the Test Case Controller Pod to the Chrome Node Pod using inter-pod communication.
* Checks if the Chrome Node Pod is ready before attempting to run the test cases.Handles error handling and retries during the deployment and test execution process.

The script should execute Selenium test cases via terminal commands in the Chrome Node Pod using a headless browser (ensure the necessary browser binaries and drivers are included in the pod).

**Create Kubernetes Cluster on AWS**:

* Set up a free-tier **t2.micro** Linux instance on AWS EC2.
* Install **kubectl** and configure it to manage the Kubernetes cluster.
* Use **Amazon EKS** to manage the Kubernetes cluster.
* Deploy your Kubernetes resources and run the test project in the EKS cluster.

**Expected Deliverables**

Provide the following outputs:

* Output (screenshot) of kubectl from the **AWS EC2 instance** to show the deployments and status of the pods (Test Controller Pod and Chrome Node Pods)
* Output (screenshot) of logs showing the **test case collection** from the Test Controller Pod and **test execution** from the Chrome Node Pod.
* **GitHub Repo Link** containing:
* The Selenium tests and their structure.
* Dockerfile for both the Test Controller Pod and Chrome Node Pod.
* Kubernetes YAML files for deploying the resources.
* A **README** file in the GitHub repo with a brief **system overview**, including:
* How the Test Controller Pod collects and sends tests to the Chrome Node Pod.
* Steps to deploy the system to Kubernetes (both locally and on AWS EKS).
* How inter-pod communication works between the controller and node.