Test 1: Build a REST API

Use your preferred language to develop REST API for student enrollment project for a school (Recommended Javascript/ Java Springboot / Python). You can persist data in AWS MySQL (preferred) or Azure MySQL or in data structure (in memory).

You can send following JSON as POST, PUT, DELETE and GET requests to REST endpoints

New student enrollment:

This will add a new record

POST request:

```
{
        “id”:223445,
        “firstName”: “Mike”,
        “lastName”: “Wong”,
        “class”:”3 A”,
        “nationality”: “Singapore”
}
```

Update student record:

This will update an existing record on the basis of unique ID

PUT request

```
{
        “id”:223445,
         “class”:”3 C”
}
```

Delete student record

This will delete an existing record on the basis of unique ID

DELETE request

```
{
         “id”: 223445
}
```

Fetch students record

1) Fetch bulk record: all students in database for that class
   GET Request, param in uri (http://yourdomain/fetchStudents? class= 3 A)

2) Fetch student record by student id
   GET Request, param in uri (http://yourdomain/fetchStudents? Id=223444)

You will be evaluated on

- Code style and reusability
- Unit test cases with code coverage
- Logging and code comments/ documentation
- **Proper Readme to build, run and test the code**

Test 2: Continuous deployment of above Rest API or code of your choice:

Though we prefer you to code above REST API endpoints, which will be a big plus, but if you choose already developed project(**code of your choice**), make sure the code runs after deployment and we are able to do some smoke tests.

Trigger pipeline (of your choice):

On code commit from CLI/IDE GitHub 📫☮Choice of your pipeline (ex: Shell /Python script / Azure Pipeline / Jenkins /Capistrano) 📫☮EC2 / Azure VMs / EKS / AKS

Please draw an architecture / flow diagram to explain

You will be evaluated on

- Flow diagram
- Completeness of pipeline
- Proper documentation/ Readme
- Choice of implementation using docker or direct artifact deployment
- Choice of code review tools (it is not mandatory, but a plus if you implement)
- Pipeline performance
- **Using Kubernetes is a big plus**

Test 3: Architecture (not mandatory but a plus)

Just a suggestion (you can also choose your preferred architecture, with proper explanation)

Design a highly available and scalable architecture using AWS services (preferred) or other Cloud provider of your choice.

1) Trigger rest call via postman or command prompt 📫☮API Gateway 📫☮NLB 📫☮

EC2 to host REST API (Auto Scaling /launch configuration/AMI static 2 instances) 📫☮MySQL DB

Above is preferred, else you can also implement the simpler version as below:

2) Trigger rest call via postman or Command prompt 📫☮EC2/Fargate/k8 to host REST API 📫☮MySQL DB

For option (1) try to secure EC2 instance and MySQL Db by blocking direct internet access by configuring inbound security group rule. For option (2) just allow your ipv4 public IP to access EC2 instance over internet.

Tools /Services to use:

1) For **TEST 1 :** We want you to host this REST API over EC2 or Fargate or Kubernetes EKS. Docker image/container is preferred but you can also deploy directly using artifacts ex: jar
2) For **TEST 2 :** Use your choice of pipeline (ex: AWS CodePipeline/Jenkins/TravisCI/GitlabCI), we want you to trigger pipeline on code commit from IDE or CLI
3) **EXTRA: You can use infrastructure as a code using any tool (e.g: ansible,chef, CloudFormation, terraform, etc), this is not mandatory but a plus.**
