## RESTFul API ( Provided - PFB details to use the APIs)

Student Management REST api with available functionality to

1. Add new student
2. Update existing student
3. Delete existing student
4. Fetch student data 4.a can fetch all students 4.b can fetch student based on student id 4.c can fetch all students in a class 4.d can fetch with student id and class together

\*\* More details below on the APIs. also can check the swagger for api documentation.

## Instructions to use the API

### Get the APIs:

Download the project executable JAR from ‘Technical Test’ folder

Filename: studentmgmt-0.0.1-SNAPSHOT.jar

### Run the APIs

* + You should have java installed on your system
  + run the following command where the executable is present to run the project in localhost

**java -jar studentmgmt-0.0.1-SNAPSHOT.jar**

* + swagger document for api is available at : <http://localhost:9080/swagger-ui.html#/>
  + Base url for the APIs in test: [**http://localhost:9080/**](http://localhost:9080/)

## Technical **TEST:**

#### API Test:

Run above API on your machine following the instructions provided above

TEST above Rest API using **Cucumber** test scripts, BDD approach. Generate a suitable reporting system to present the results/identify issues in the application/tests.

#### 

#### WEBUI Test:

Navigate to [http://the-internet.herokuapp.com](http://the-internet.herokuapp.com/), click on ‘[Form Authentication](http://the-internet.herokuapp.com/login)’ and test the login feature available in that website using **Cucumber** test scripts, BDD approach. Generate a suitable reporting system to present the results/identify issues in the application/tests.

**You will be evaluated on**

* Framework, test coverage
* Reporting
* Code style and reusability
* Logging and code comments/ documentation
* Proper Readme to test the code

### Test Submission:

Submit your solution to github and share the repository link along with documentation ( if any) to our HR representative.

**\*\* API details:**

**New student enrolment:**

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)

1. Fetch student record by student id

GET Request, param in uri (http://yourdomain/fetchStudents? Id=223444)