NOTE: In milestone 3, APIs can only be used upon successful login. There are a few exceptions such as login API and register customer API.

Admin:

POST:

http://*ip-address*:8080/api/admin/register

```
{
    "userName":"admin",
    "companyName ":"Fri-10-30-team3",
    "password":"123456",
    "confirmPassword":"123456",
    "fname":"test",
    "lname":"test"

}
```

Note: the length of the username should be greater than 5.

PUT:

http://*ip-address*:8080/api/ admin /put/{id}

```
{
    "userName":"c123456",
    "companyName ":"Fri-10-30-team3",
    "fname":"test",
    "lname":"test"

}
```

NOTE: This API could not change the user password even if you add the password field in JSON.

GET:

1. Get all admin: http://*ip-address*:8080/api/admin.
2. Get by id: http://*ip-address*:8080/api/admin/{id}

Customer:

POST:

http://*ip-address*:8080/api/customer/register

```
{
    "userName":"c123456",
    "fname":"test",
    "lname":"test",
    "password":"123456",
    "confirmPassword":"123456"
}
```

Note: the length of the username should be greater than 5.

http://*ip-address*:8080/api/customer/login

```
{
    "userName":"c123456",
    "password":"123456"
}
```

NOTE: this API can be used for all types of user, it will return the user type and token upon success.

PUT:

http://*ip-address*:8080/api/ customer /put/{id}

```
{
    "userName":"c123456",
    "fname":"test",
    "lname":"test",
}
```

NOTE: This API could not change the user password even if you add the password field in JSON.

GET:

1. Get all customer: http://*ip-address*:8080/api/customer.
2. Get by id: http://*ip-address*:8080/api/customer/{id}

Employee:

POST:

http://*ip-address*:8080/api/employee/ register

```
{
    "userName":"c1",
    "fname":"test",
    "lname":"test",
    "password":"123456",
    "confirmPassword":"123456"
}
```

We will create more fields in Employee class according to the specification. For now, we mainly focus on the functionality.

GET:

1. Get all employee: http://*ip-address*:8080/api/employee.
2. Get by id: http://*ip-address*:8080/api/employee/{id}

DELETE:

http://*ip-address*:8080/api/employee/delete/{id}

PUT:

http://*ip-address*:8080/api/employee/put/{id}

```
{
    "userName":"c1",
    "fname":"test",
    "lname":"test"
}
```

PUT request is similar to POST, but the {id} should be already in the database, otherwise, no record would be changed or created.

EmployeeSchedule:

POST:

http://*ip-address*:8080/api/schedule

```
{
    "employee":{"id":"1"},
    "skills":{"skillId":"1"},
    "availability":"2020-08-25",
    "capacity":"10",
    "length":"2"
}
```

GET:

1. Get all schedule: http://*ip-address*:8080/api/schedule.
2. Get by schedule id: http://*ip-address*:8080/api/schedule/{id}
3. Get by employee id: http://*ip-address*:8080/api/schedule/employee/{id}
4. Get by skills id: http://*ip-address*:8080/api/schedule/skill/{id}

DELETE:

http://*ip-address*:8080/api/schedule/delete/{id}

Enrollment:

POST:

http://*ip-address*:8080/api/enrollment

```
{
    "customer":{"id":"1"},
    "employeeSchedule":{"scheduleId":"2"},
    "info":"test"
}
```
GET:

1. Get all enrollment(booking): http://*ip-address*:8080/api/enrollment.
2. Get by enrollment id: http://*ip-address*:8080/api/enrollment /{id}
3. Get by customer id: http://*ip-address*:8080/api/enrollment /customer/{id}
4. Get by schedule id: http://*ip-address*:8080/api/enrollment/schedule/{id}

Skills:

POST:

http://*ip-address*:8080/api/skills

```
{
    "skills_name":"skill1"
}
```
NOTE: There are more fields in the Skills class.

GET:

1. Get all skills: http://*ip-address*:8080/api/skills.
2. Get by skills id: http://*ip-address*:8080/api/skills/{id}

DELETE:

http://*ip-address*:8080/api/skills/delete/{id}

PUT:

http://*ip-address*:8080/api/skills/put/{id}

```
{
    "skillsName":"skillchanged",
    "title":"t"
}
```
PUT is similar to POST.