Luồng xử lí roll call chính:

1. Teacher gọi API rollCall để điểm danh
2. Server generate chuỗi IdentifyString (chuỗi thuộc về Class – có 10 kí tự) √
3. Server cập nhật lại chuỗi + tạo ra event để set chuỗi = null sau 10p (sử dụng event scheduler của mySQL để làm việc này) √
4. Server ghi lại time vào listRollCall của bảng TeacherClass (datetime) và vào field isChecked của Class (để chống việc teacher điểm danh 1 buổi học nhiều hơn 1 lần) √

*(Hoàn thành teacher roll call)*

1. Teacher thông báo cho Student về chuỗi (bên ngoài đời thực)
2. Student roll call √
3. Server ghi lại time vào listRollCall của bảng StudentClass và vào field isChecked của StudentClass (để chống việc student điểm danh 1 buổi học nhiều hơn 1 lần) √

*(Hoàn thành student roll call)*

1. Hết 10p, server set lại IdentifyString = null √
2. Hết ngày, admin gọi API CheckRollCallToday để quét

Server xử lí:

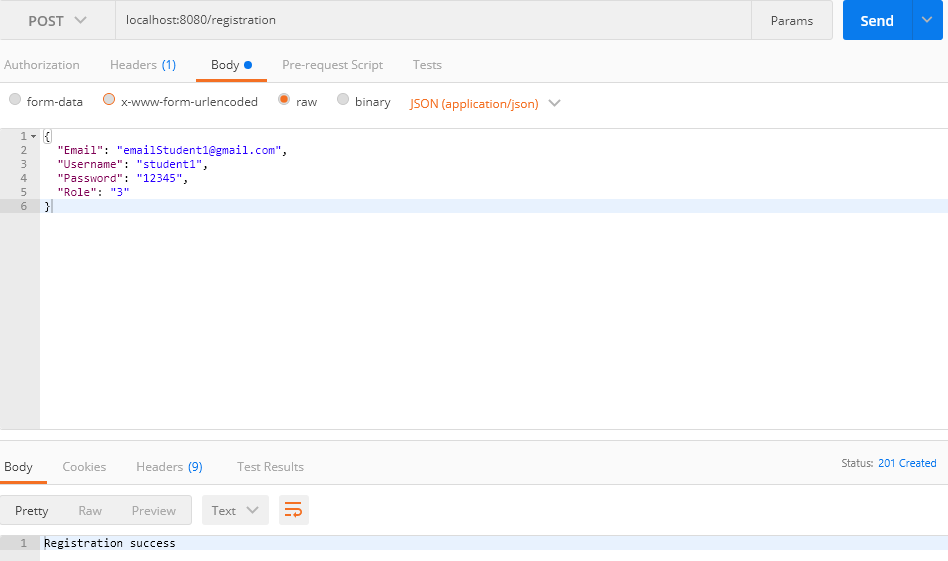
+ Đến cuối ngày, admin sử dụng API CheckRollCallToday:

* Các lớp đc dạy trong ngày -> đã đc lưu vào bảng ClassToday
* Lớp nào có teacher ko điểm danh => teacher.time = “day + x”, student ko quét

+ Nếu student nghỉ có phép và đc chấp nhận, teacher sd API TeacherGivePermission (chỉ sd khi đang trong giờ điểm danh)

**THÔNG TIN API**

**API Registration**

{

"Email": "emailStudent",

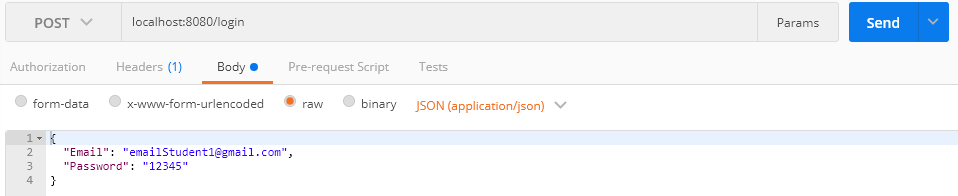
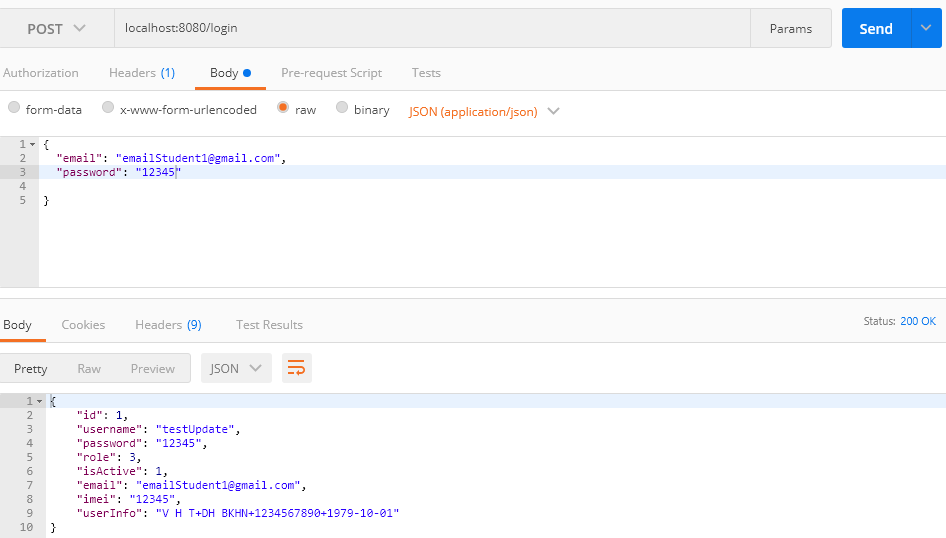
"Username": "studen",

"Password": "12345",

"Role": "4"

}

**API Login**



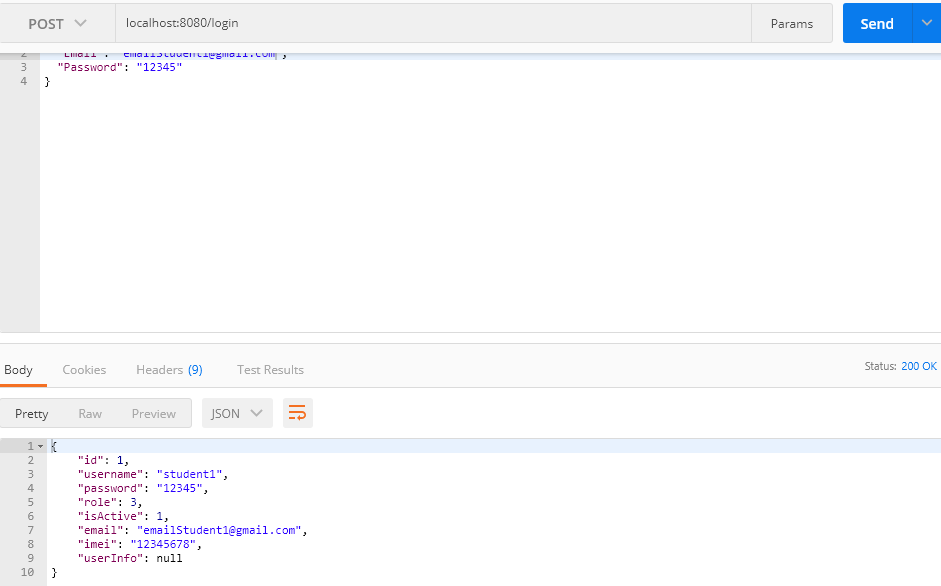
{

"email": "emailStudent1@gmail.com",

"password": "12345"

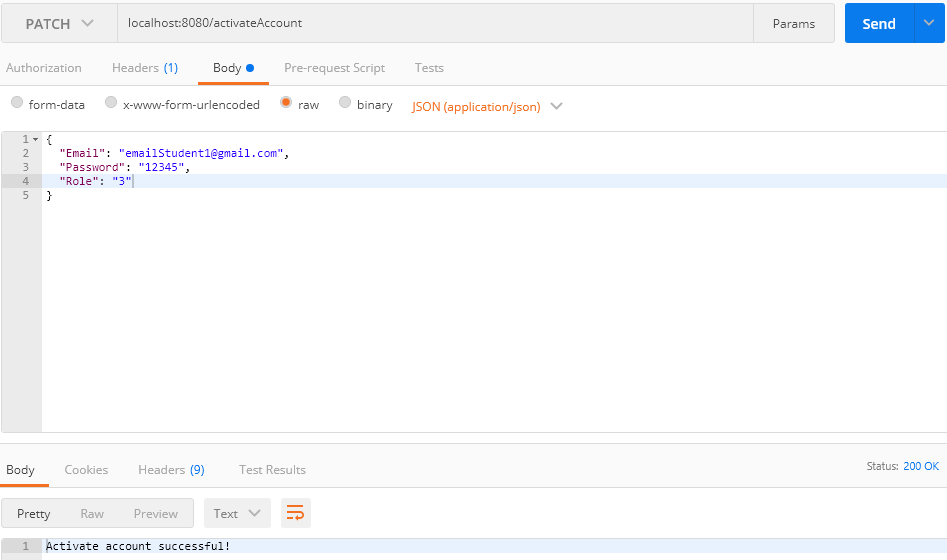
}

* Ở lần login đầu tiên và cả những TH login mà IMEI chưa đc cung cấp thì value của field imei trả về = null
* Đây là TH Imei đã đc cung câp



* Mẫu text:

**API Activate account**



* Mẫu text:

{

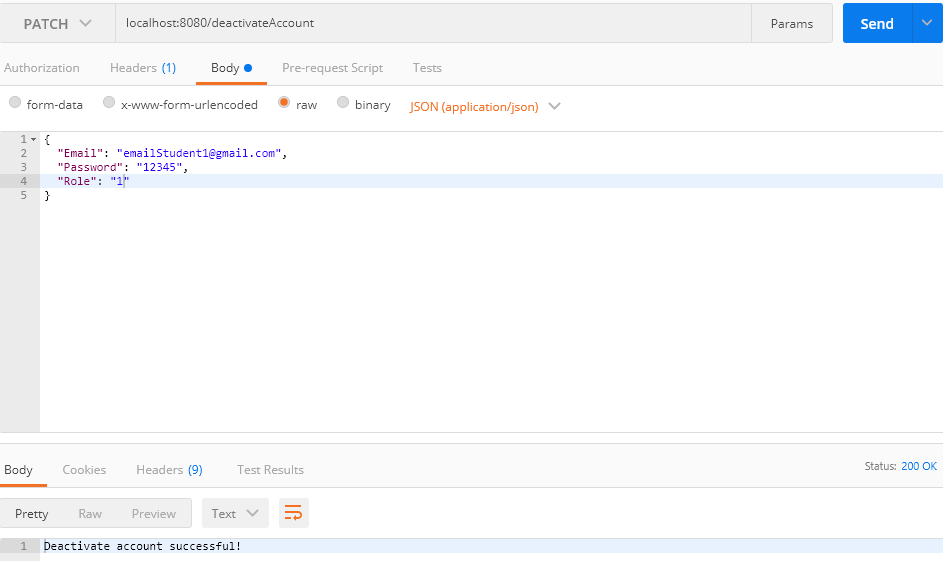
"Email": "emailStudent1@gmail.com",

"Password": "12345",

"Role": "3"

}

**API Deactivate account**



* Mẫu text:

{

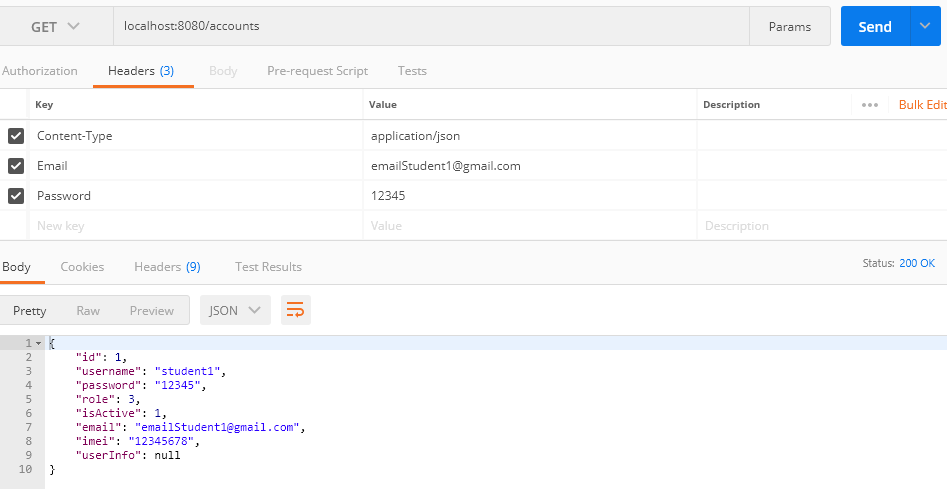
"Email": "emailStudent1@gmail.com",

"Password": "12345",

"Role": "2"

}

**API Get info account**

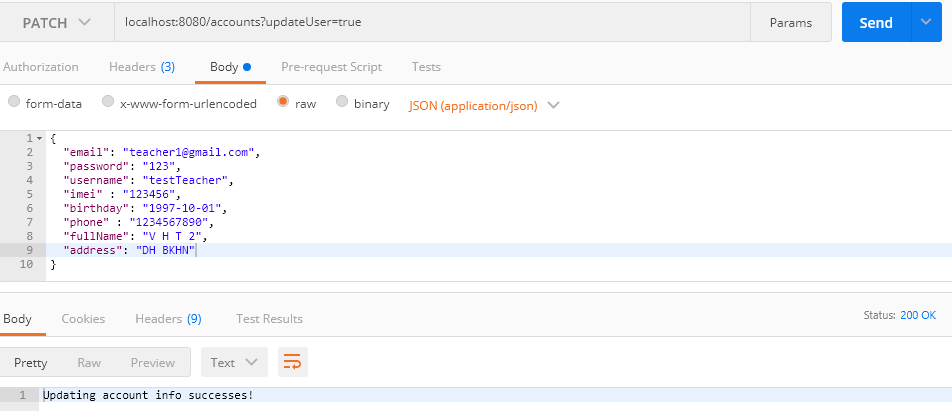


* Mẫu text:

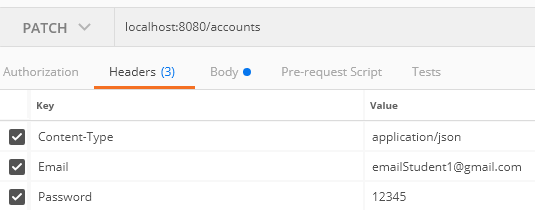
(chỉ có trong Headers như hình vẽ)

**API updateAccountInfo**

* phần Request Body



* Phần Header:



* **URL =** /accounts?updateUser=true

*(updateUser là biến chỉ định việc có thay đổi thông tin user hay ko; true là có và sẽ phải dùng mẫu text 2; false là ko)*

* Mẫu text khi ko muốn add/update userInfo

{

"email": "teacher1@gmail.com",

"password": "123",

"username": "testTeacher",

"imei" : "123456",

}

* Mẫu text khi muốn add/update userInfo:

{

"email": "teacher1@gmail.com",

"password": "123",

"username": "testTeacher",

"imei" : "123456",

"birthday": "1997-10-01",

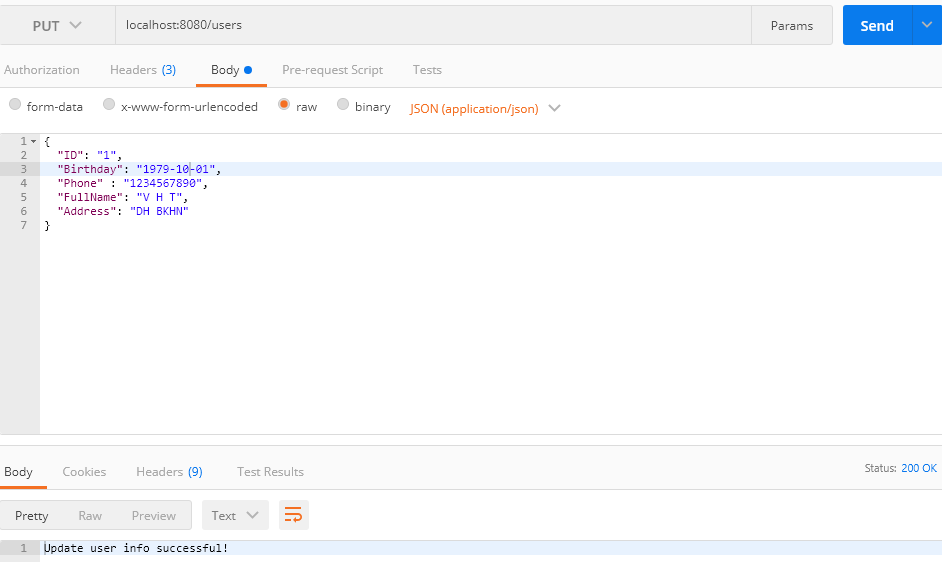
"phone" : "1234567890",

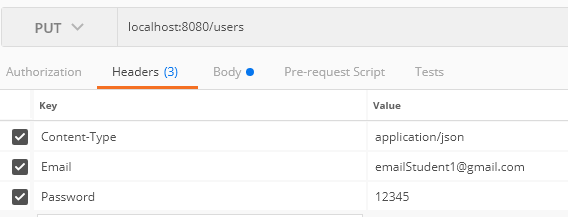
"fullName": "V H T 2",

"address": "DH BKHN"

}

**API UpdateUser**





* Mẫu text:

{

"ID": "1",

"Birthday": "1979-01-01",

"Phone" : "1234567890",

"FullName": "V H T",

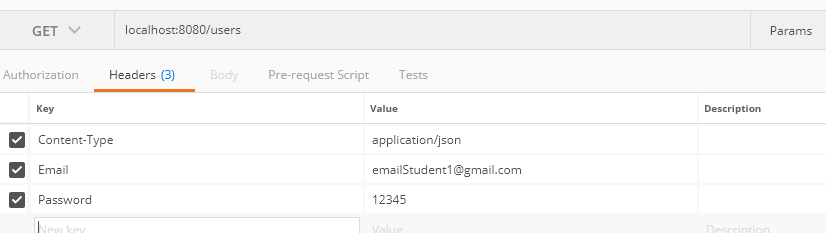
"Address": "DH BKHN"

}

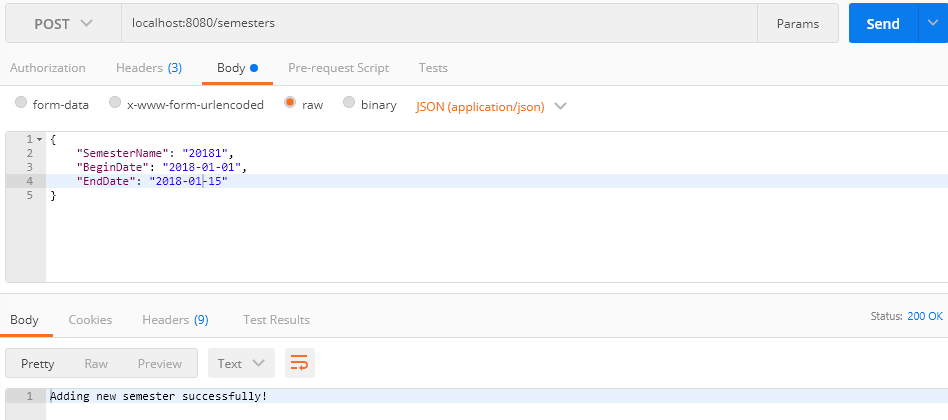
**API AddUserInfo**

**(Như UpdateUserInfo, chỉ khác là dùng POST và chỉ áp dụng khi userInfo = null)**

**API Get user info**



**API Add Semester**



* Mẫu text:

{

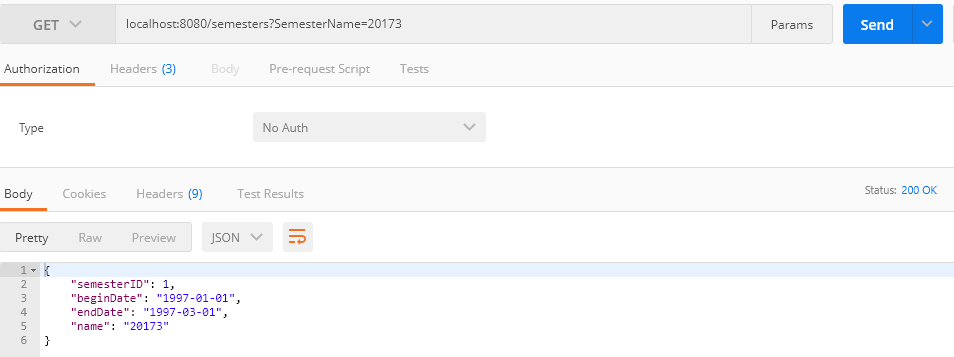
"SemesterName": "20181",

"BeginDate": "2018-09-01",

"EndDate": "2019-01-15"

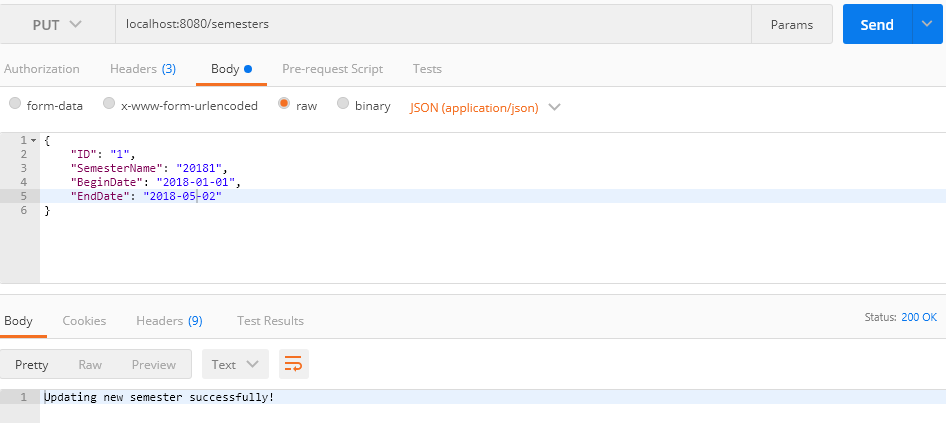
}

**API getSemesterInfo**



(Ko có mẫu text, đường link: localhost:8080/semesters?SemesterName=20173)

**API updateSemesterInfo:**



* Mẫu txt:

{

"ID": "1",

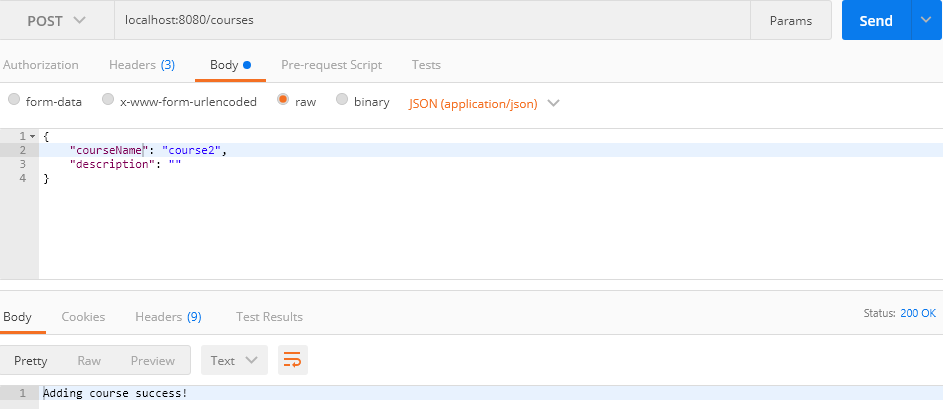
"SemesterName": "20181",

"BeginDate": "2018-01-01",

"EndDate": "2018-05-02"

}

**API addNewCourse**



* Mẫu txt:

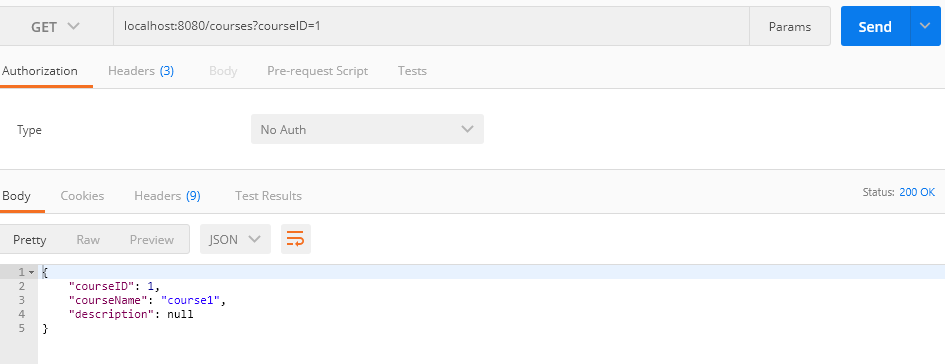
{

"courseName": "course2",

"description": ""

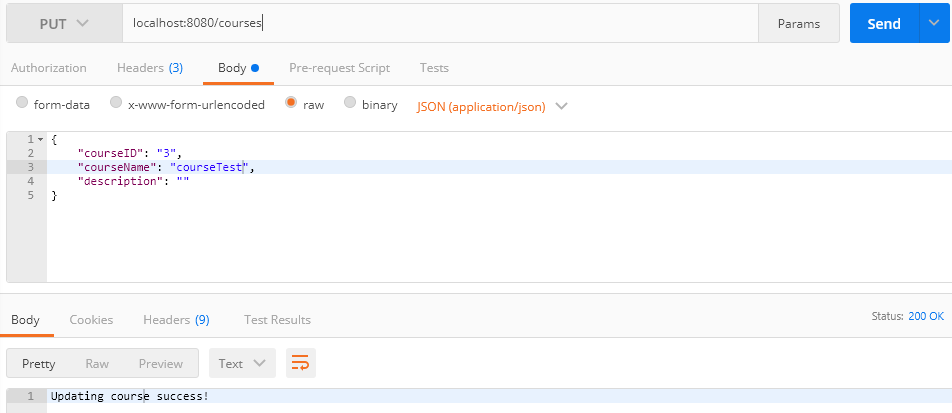
}

**API getCourseInfo**



* Đường link: localhost:8080/courses?courseID=1

**API updateCourseInfo**



* Mẫu txt:

{

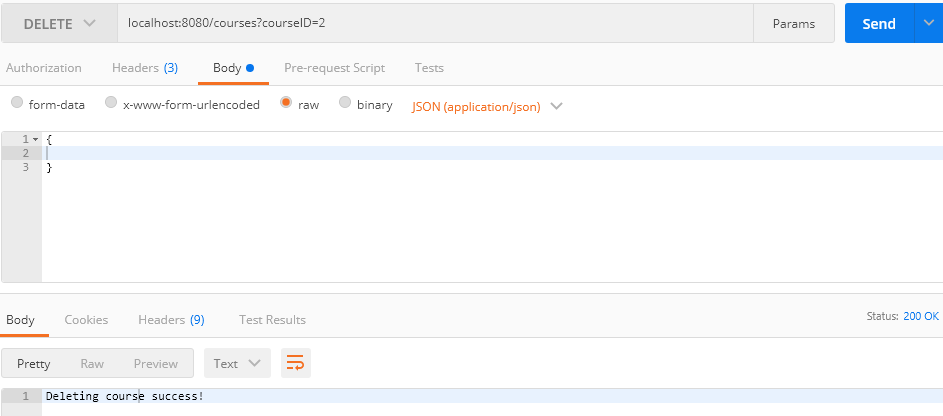
"courseID": "3",

"courseName": "course1",

"description": ""

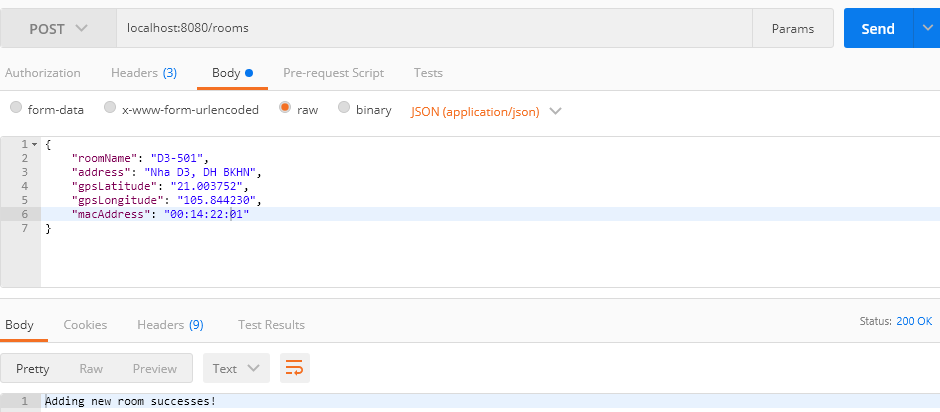
}

**API deleteCourse**



* Đường link: localhost:8080/courses?courseID=1

**API addNewRoom**



* Mẫu txt:

{

"roomName": "D3-501",

"address": "Nha D3, DH BKHN",

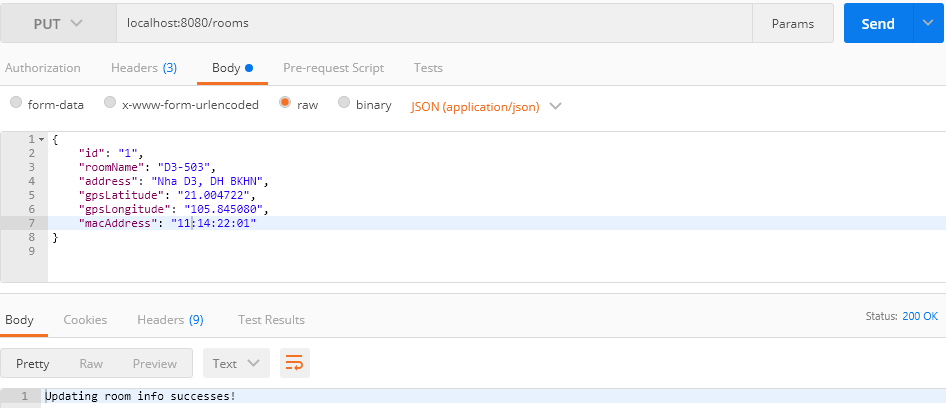
"gpsLatitude": "21.003752",

"gpsLongitude": "105.844230",

"macAddress": "00:14:22:01"

}

**API Update room info**



{

"id": "1",

"roomName": "D3-503",

"address": "Nha D3, DH BKHN",

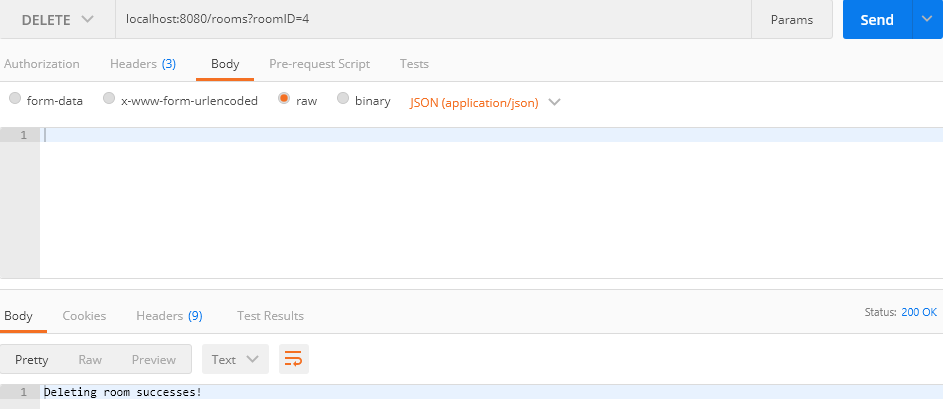
"gpsLatitude": "21.004722",

"gpsLongitude": "105.845080",

"macAddress": "11:14:22:01"

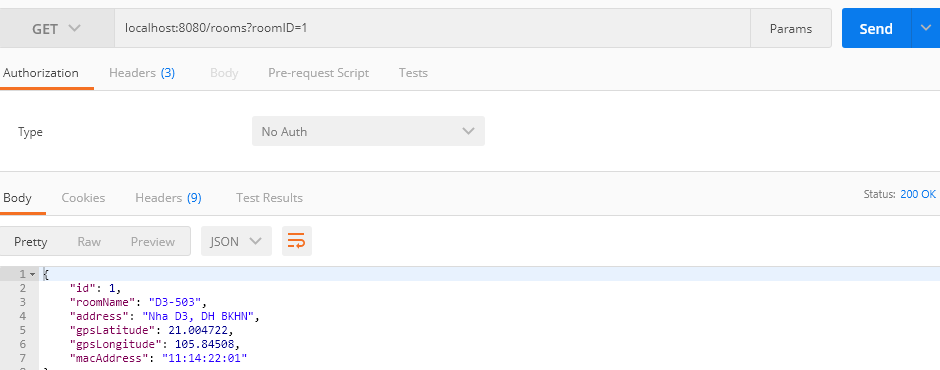
}

**API Delete room**



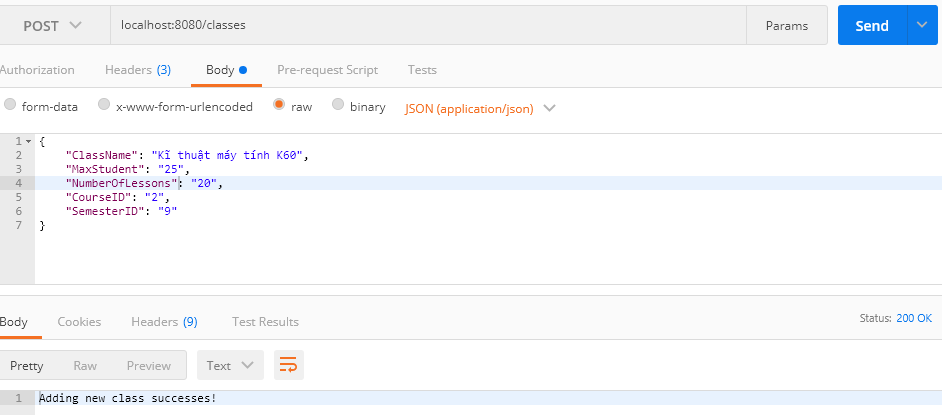
**URL =** localhost:8080/rooms?roomID=4

**API Get room info**



**URL =** localhost:8080/rooms?roomID=4

**API Add new class**



{

"ClassName": "Kĩ thuật máy tính K60",

"MaxStudent": "25",

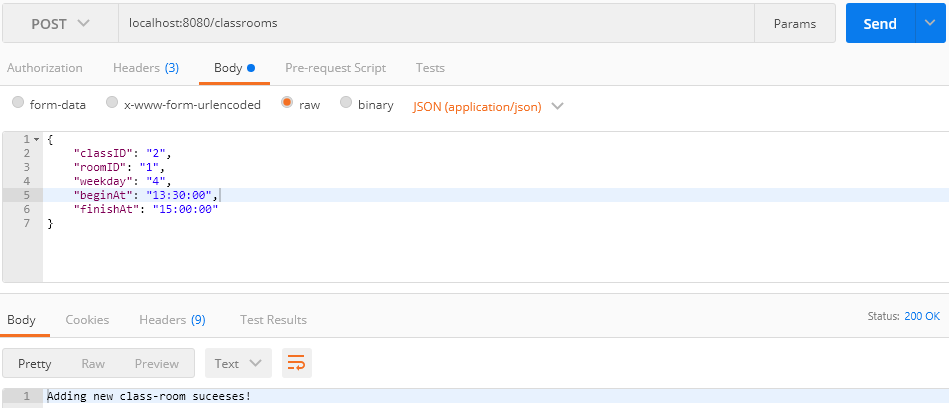
"NumberOfLessons": "20",

"CourseID": "2",

"SemesterID": "9"

}

**API Add new ClassRoom**



{

"classID": "2",

"roomID": "1",

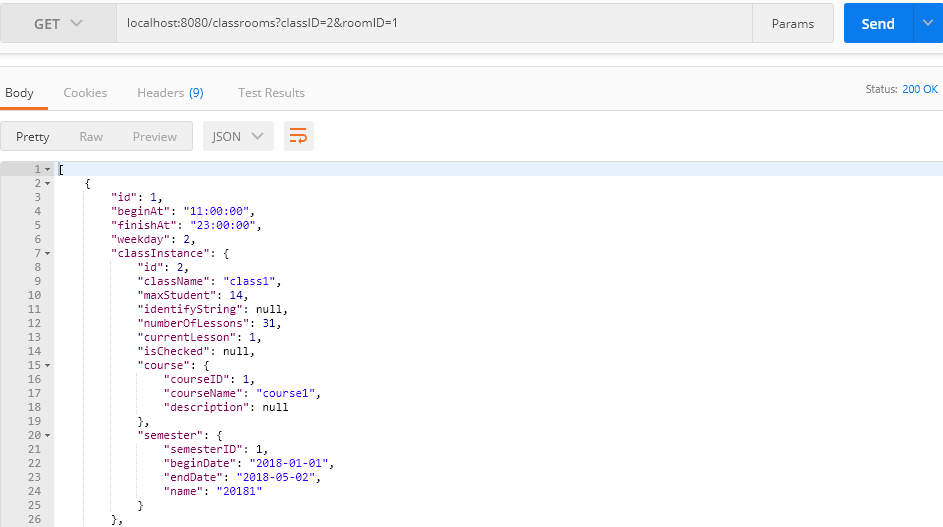
"weekday": "4",

"beginAt": "13:30:00",

"finishAt": "15:00:00"

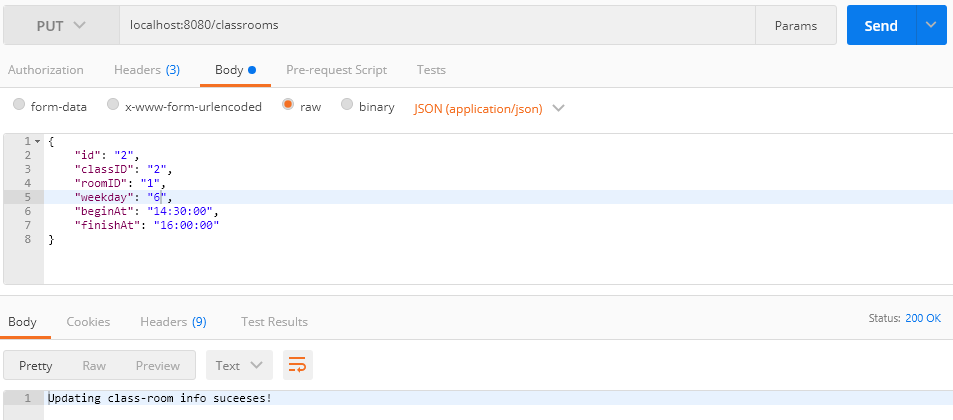
}

**API Get ClassRoom info**



**URL =** localhost:8080/classrooms?classID=2&roomID=1

**API Update ClassRoom info**



{

"id": "2",

"classID": "2",

"roomID": "1",

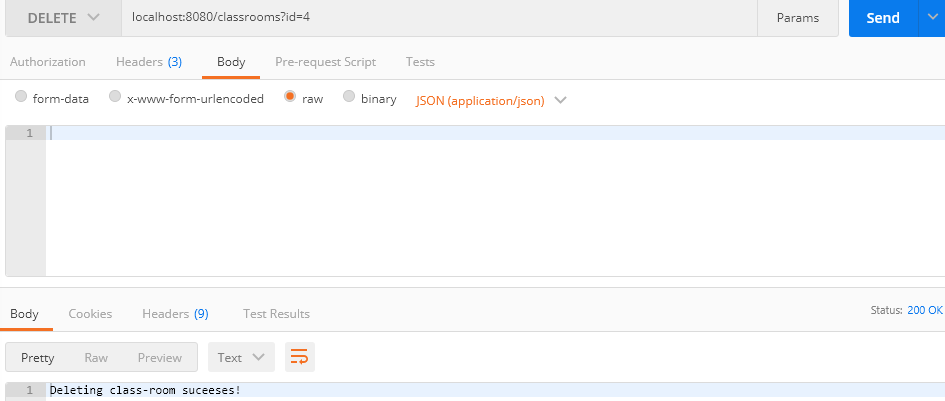
"weekday": "6",

"beginAt": "14:30:00",

"finishAt": "16:00:00"

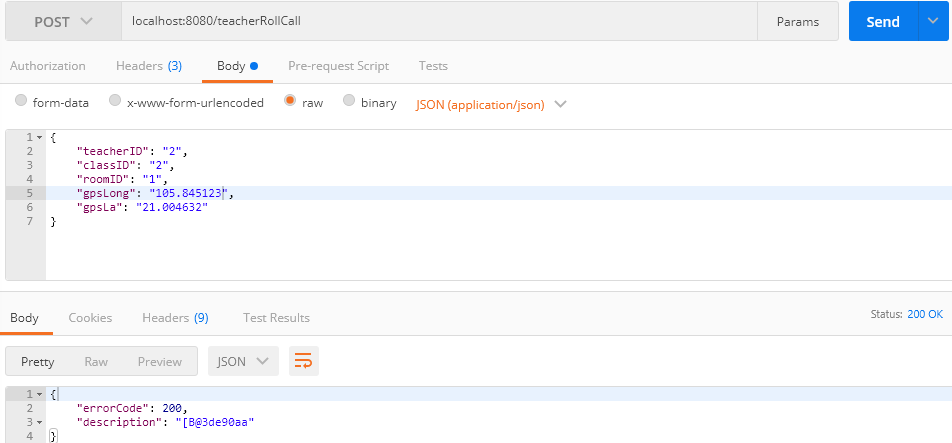
}

**API Delete ClassRoom**



**URL =** localhost:8080/classrooms?id=4

**API teacherRollCall**



{

"teacherID": "2",

"classID": "2",

"roomID": "1",

"gpsLong": "105.845123",

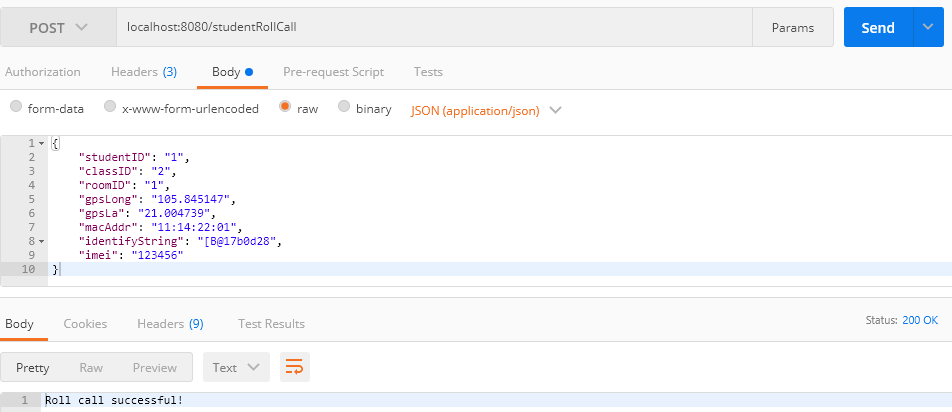
"gpsLa": "21.004632"

}+ TH thành công:

điều kiện:

* thời gian roll call nằm trong thời gian của buổi học và trước khi buổi học kết thúc 15 phút
* 1 buổi học chỉ được phép điểm danh 1 lần
* Trả về identifyString của lớp học đó

**API studentRollCall**



* 1 buổi học chỉ được phép điểm danh 1 lần

{

"studentID": "1",

"classID": "2",

"roomID": "1",

"gpsLong": "105.845147",

"gpsLa": "21.004739",

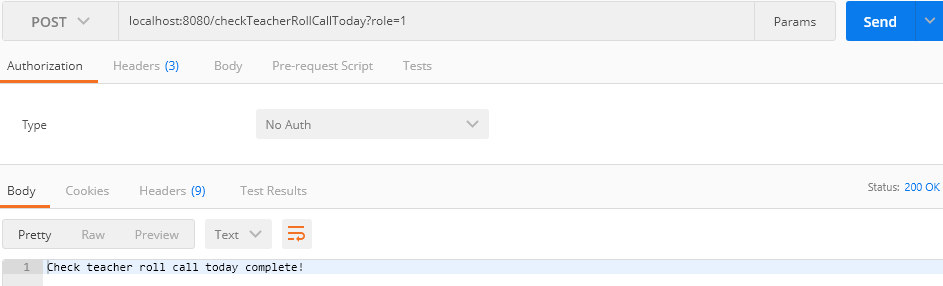
"macAddr": "11:14:22:01",

"identifyString": "[B@17b0d28",

"imei": "123456"

}

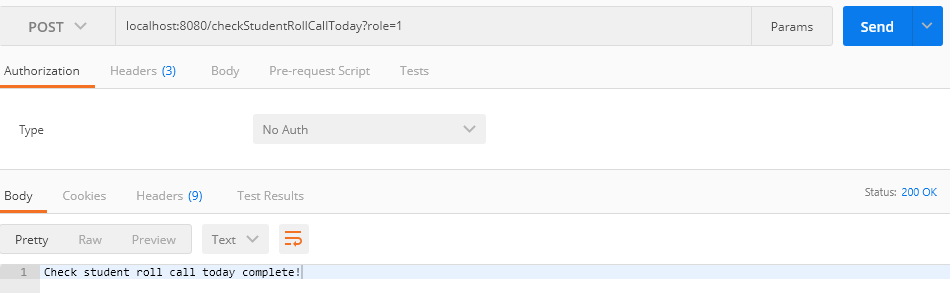
**API checkTeacherRollCallToday**



**URL =** localhost:8080/checkTeacherRollCallToday?role=1

*(role bắt buộc phải = 1 vì chỉ có admin đc phép sd API này; API này và API bên dưới 1 ngày chỉ đc phép gọi 1 lần => mỗi lần test phải reset lại data hoặc tạo data mới)*

**API checkStudentRollCallToday**



**URL =** localhost:8080/checkStudentRollCallToday?role=1

*(role bắt buộc phải = 1 vì chỉ có admin đc phép sd API này; API này chỉ đc gọi sau khi đã gọi checkTeacherRollCall thì mới có thể hoạt động chính xác)*

THAY ĐỔI TRONG DB

------------------ Room -----------------------------------

ID: int(++)

Name: varchar(20) Unique # số hiệu của phòng học. VD: D1-101

Address: nvarchar(200) # địa chỉ của phòng học

GPSLatitude: decimal(x,x) #(kinh độ)

GPSLongitude: decimal(x,x) #(vĩ độ)

MAC: char(8) # 8 chữ số đầu trong địa chỉ MAC của WAP

------------------ Class\_Room ----------------------------

ID: int(++)

IDRoom: int (FK to Room)

IDClass: int (FK to Class)

WeekDay: int #(ngày có tiết trong tuần. Giá trị: từ 2 - 6)

BeginAt: time

FinishAt: time

-------------------- Lop hoc - Class ------------------

ID: int(++) (PK)

Name: nvarchar(40)

SemesterID: INT #(FK toi ID cua Semester)

CourseID: #(FK toi ID cua Course)

MaxStudent: int

IdentifyString: char(10)

Weight: int #tổng số buổi học

CurrentLesson: int # thứ tự của tiết học hiện tại. VD: 2

-------------------- TeacherClass --------------------- (bảng lưu trữ dữ liệu RollCall)

ID: int++

ClassID: in #(FK toi ID cua Class)

TeacherID: int #(FK toi ID cua Teacher)

IsTeaching: int # 1 = TEACHING, 2 = TAUGHT

listRollCall: varchar(500)

*(Note: Data có dạng: 2018-129-1130,…*

*2018 = năm*

*129 = ngày trong năm*

*1130 = giây trong ngày*

*Lưu trữ thế để đỡ tốn space)*

-------------------------------------------------------------------------

Cấu trúc của String userInfo của Account:

“fullName+address+phone+birthDay”

(dấu “+” là để phân cách các trường; vì fullName có thể chứa dấu cách; còn birthDay chứa dấu “-“)

**LIST CÁC LỖI + ERROR CODE:**

* Về JSON mapping:

Lỗi 1:

{

"errorCode": 1,

"description": "Json dynamic map lacks necessary key(s)!"

}

Lỗi 2:

"error": "Bad Request",

"message": "{\"errorCode\" : 2, \"description\" : Error happened when jackson deserialization info!}",

* Về account:

{

"errorCode": 11,

"description": "Authentication has failed or has not yet been provided!"

}

* + **new** ReportError(12, "Registration failed because " + errorMessage);
  + **new** ReportError(13, "Registrantion failed because this email has already been used");
  + **new** ReportError(14, "Deactive account failed because " + errorMessage);
  + **new** ReportError(15, "Active account failed because " + errorMessage);
  + **new** ReportError(16, "Getting account info failed because " + errorMessage);
  + **new** ReportError(17, "Updating account info failed because " + errorMessage);
* Về user:

**new** ReportError(20, "Adding user info failed because" + errorMessage); √

report = **new** ReportError(21, "User's info cannot be overriden by this API !");√

**return** **new** ResponseEntity<>(report, HttpStatus.***CONFLICT***);

**new** ReportError(22, "Updating user info failed because" + errorMessage); √

report = **new** ReportError(23, "This user info has not existed yet!"); √

**return** **new** ResponseEntity<>(report, HttpStatus.***NOT\_FOUND***);

* Về semester:
* **new** ReportError(30, "Adding semester failed because " + errorMessage);
* **new** ReportError(31, "This semester name has already been used!");
* **new** ReportError(32, "This semester's duration violate another semester's duration!");

report = **new** ReportError(33, "This semester do not exist yet!");

**return** **new** ResponseEntity<>(report, HttpStatus.***NOT\_FOUND***);

* **new** ReportError(34, "This semester still has dependant!");
* **new** ReportError(35, "Updating semester failed because " + errorMessage);
* **new** ReportError(36, "Deleting semester failed because " + errorMessage);
* **new** ReportError(37, "Getting semester info failed because " + errorMessage);
* Về course:
* **new** ReportError(40, "Adding course failed because " + errorMessage);
* **new** ReportError(41, "Duplicate course name!");
* **new** ReportError(42, "Getting course failed because " + errorMessage);

report = **new** ReportError(43, "This course do not exist!");

**return** **new** ResponseEntity<>(report, HttpStatus.***NOT\_FOUND***);

* **new** ReportError(44, "Updating course failed because " + errorMessage);
* **new** ReportError(45, "This course still has dependant!");
* Về Room:
* **new** ReportError(50, "Adding room failed because " + errorMessage);
* **new** ReportError(51, "Duplicate room name!");
* **new** ReportError(52, "Duplicate MAC address!");

report = **new** ReportError(53, "This room do not exist!");

**return** **new** ResponseEntity<>(report, HttpStatus.***NOT\_FOUND***);

* **new** ReportError(54, "Deleting room failed because " + errorMessage);
* **new** ReportError(55, "This room still has dependant!");
* **new** ReportError(56, "Updating room failed because " + errorMessage);
* **new** ReportError(57, "Getting room failed because " + errorMessage);
* Về Class:
* ReportError(60, "Adding new class failed because " + errorMessage);
* report = **new** ReportError(61, "Adding new class do not work after semester begins 2 weeks " + errorMessage); => thêm class chỉ có thể thực hiện nếu semester chưa bắt đầu (vì các nghiệp vụ phức tạp liên quan đến điểm danh)
* *badRequest*().body("Error code: 62; Content: Getting class info failed because " + errorMessage);

report = **new** ReportError(63, "This class do not exist!");

**return** **new** ResponseEntity<>(report, HttpStatus.***NOT\_FOUND***);

* **new** ReportError(64, "Updating class info failed because " + errorMessage);
* **new** ReportError(65, "Updating class info do not work after semester begins 2 weeks! ");
* **new** ReportError(66, "This class still has dependant! ");

Về ClassRoom:

* **new** ReportError(70, "Adding new class-room failed because " + errorMessage);
* **new** ReportError(71, "This class is not available at this duration!");
* **new** ReportError(72, "This room is not available at this duration!");
* **new** ReportError(73, "Getting class-room info failed because " + errorMessage);

report = **new** ReportError(74, "This class-room do not exist!");

**return** **new** ResponseEntity<>(report, HttpStatus.***NOT\_FOUND***);

* **new** ReportError(75, "Updating class-room info failed because " + errorMessage);
* **new** ReportError(76, "Deleting class-room failed because " + errorMessage);
* **new** ReportError(77, "This class-room still has dependants");

Về TeacherClass:

* **new** ReportError(80, "Teacher roll call failed because " + errorMessage);
* **new** ReportError(81, "Teacher roll call failed because timing is not in valid range!");
* **new** ReportError(82, "Cannot generate identify string!");
* **new** ReportError(83, "Teacher roll call failed because an ID is not a number");
* report = **new** ReportError(84, "Longitude/Latitude is out of practical range!");
* **new** ReportError(85, "Device is out of valid distance to classroom!");

Về StudentClass:

* **new** ReportError(90, "Student roll call failed because " + errorMessage);
* **new** ReportError(91, "Longitude/Latitude is out of practical range!");
* **new** ReportError(92, "Missing MAC address");
* **new** ReportError(93, "Missing IMEI info");
* **new** ReportError(94, "Missing identifyString info");
* **new** ReportError(95, "MAC address is incorrect!");
* **new** ReportError(96, "Device is out of valid distance to classroom!");

Về Uncategorized (gồm timetable, checkRollCallToday):

* **new** ReportError(100, "Getting timetable failed because " + errorMessage);

report = **new** ReportError(101, "No record for this semester!");

**return** **new** ResponseEntity<>(report, HttpStatus.***NOT\_FOUND***);

* **new** ReportError(102, "Check roll call today failed because of current time is not valid! ");

report = **new** ReportError(103, "No class has lessons today!");

**return** **new** ResponseEntity<>(report, HttpStatus.***NOT\_FOUND***);

* **new** ReportError(104, "This API must be called after calling checkTeacherRollCallToday API!");

**INTERNAL CONSTRAINT CỦA DB**

* <https://stackoverflow.com/questions/34132644/sql-check-constraint-not-working-properly>
* mySQL not accept CHECK
* mọi constraint sẽ đc thực hiện ở phần validate của server