Contents

[**1.** **Class Student** 2](#_Toc184255185)

[**2.** **Class Printer Configuration** 3](#_Toc184255186)

[**3.** **Class Printer** 5](#_Toc184255187)

[**4.** **Class Login** 6](#_Toc184255188)

[**5.** **Class SPSO** 6](#_Toc184255189)

[**6.** **Class Printing System (thấy không cần thiết lắm)** 7](#_Toc184255190)

[**7.** **Class Print Modification** 8](#_Toc184255191)

[**8.** **Class Buy Log** 8](#_Toc184255192)

[9. **Class PaperSetting** 9](#_Toc184255193)

[**10.** **Class Report** 11](#_Toc184255194)

[**11.** **Document** 12](#_Toc184255195)

[**12.** **Class Print Log** 13](#_Toc184255196)

**List API**

1. **Class Student**

**Body:**

{

“id”: MSSV (long),

“name”: Tên sinh viên (String),

“balance”: Số trang giấy còn lại (int) (số trang A4 hiện có),

“phoneNum”: số điện thoại (String),

“doB”: ngày sinh (Date),

“email”: email hcmut.edu.vn (String),

“login”: (object của class login)

{

“id”: tạo tự động (long),

“username”: tên đăng nhập (String),

“password”:mật khẩu (String)

}

}

* 1. ***GET: lấy danh sách tất cả sinh viên có trong database và thông tin của họ.***

API: <http://localhost:8080/api/v1/students>

* 1. ***GET: lấy thông tin của sinh viên ứng với mã số sinh viên***

API: [http://localhost:8080/api/v1/student/{id}](http://localhost:8080/api/v1/student/%7bid%7d)

* 1. ***GET: lấy thông tin đăng nhập của sinh viên***

*API:* <http://localhost:8080/api/v1/student>/{id}/login

* 1. ***POST: tạo một đối tượng sinh viên mới***

API: <http://localhost:8080/api/v1/student>

yêu cầu nhập vào thông tin cho một đối tượng sinh viên mới

Cần tạo login trước khi tạo đối tượng sinh viên.

Nếu nhập vào sinh viên đã có MSSV trong database rồi thì nó sẽ ném lỗi. Nên muốn chỉnh sửa thông tin sinh viên thì dùng Put Resquest.

* 1. ***PUT: cập nhật thông tin sinh viên bằng id sinh viên***

***(Thông tin login của sinh viên có thể được cập nhật bằng API này, hoặc bằng username của tài khoản sinh viên)***

***//cập nhật số trang A4 của sinh viên ở API này luôn nhe***

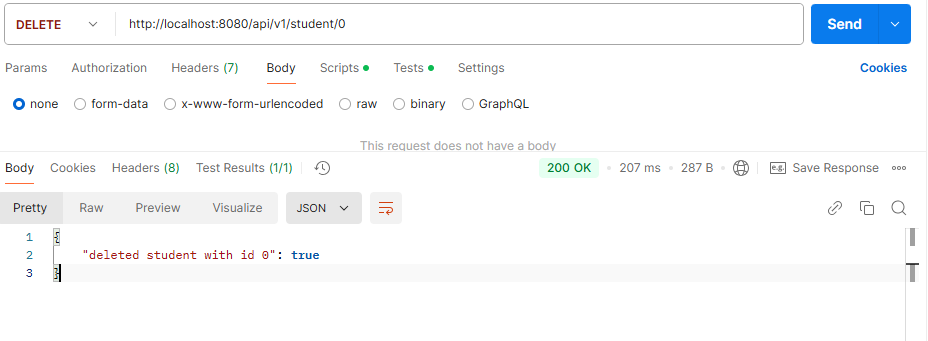
API: [http://localhost:8080/api/v1/student/{id}](http://localhost:8080/api/v1/student/%7bid%7d)

* 1. DELETE: xóa tất cả sinh viên

API: <http://localhost:8080/api/v1/students>

* 1. DELETE: xóa sinh viên có MSSV id

API: [http://localhost:8080/api/v1/student/{id}](http://localhost:8080/api/v1/student/%7bid%7d)



1. **Class Printer Configuration**

**Body:**

{

“id”: tạo tự động (long),

“model”: mẫu máy in (String).

“brand”: thương hiệu máy in (String),

“technology”: công nghệ in (laser, phun) (String)

“duplexPrinting”: in hai mặt thủ công hay tự động (automatic or manual) (String),

“printer”:{

“name”: tên máy in (key)(String),

“location”: vị trí (String),

“state”: trạng thái máy in (on/off) (String)

}

}

**Class Printer Configuration có lưu một đối tượng Printer, nhưng đối tượng Printer thì không.**

* 1. **GET: lấy danh sách tất cả cấu hình máy in**

**API:** http://localhost:8080/api/v1/configurations

* 1. **GET: lấy cấu hình máy in tương ứng với ID của cấu hình**

**API:** [http://localhost:8080/api/v1/configuration/{id}](http://localhost:8080/api/v1/configuration/%7bid%7d)

**Note: id của cấu hình máy in được tạo tự động**

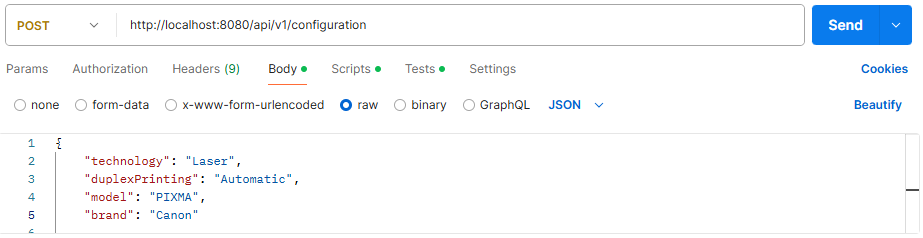
* 1. **GET: lấy cấu hình máy in tương ứng với Printer Name**

**API:** [http://localhost:8080/api/v1/configuration/printer/{printer\_name}](http://localhost:8080/api/v1/configuration/printer/%7bprinter_name%7d)

* 1. **POST: tạo một configuration**

**API:** <http://localhost:8080/api/v1/configuration>

**Example:**

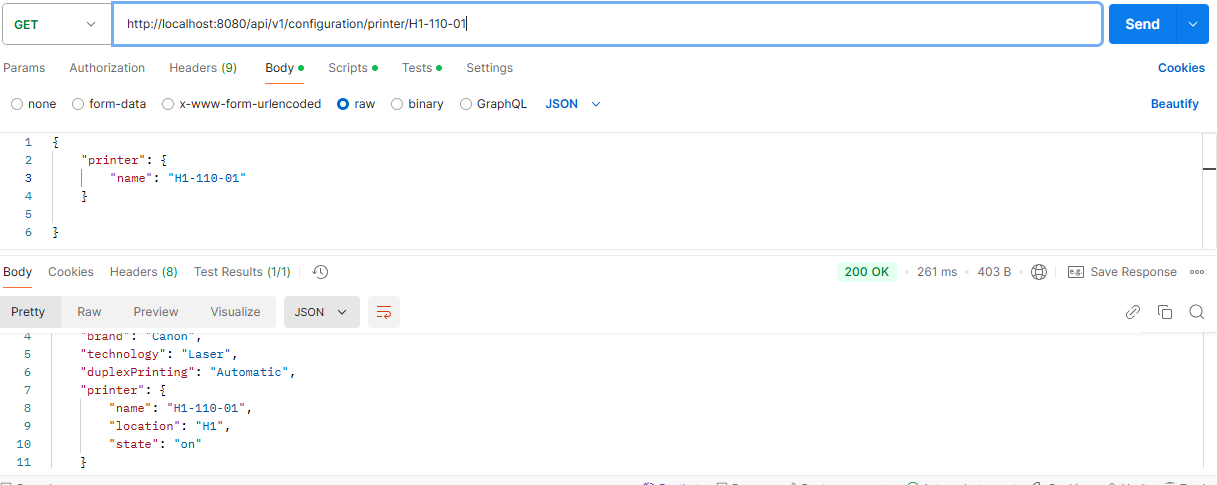


**2.5 PUT: sửa một cấu hình máy in tương ứng với id của nó**

API: [http://localhost:8080/api/v1/configuration/{id}](http://localhost:8080/api/v1/configuration/%7bid%7d)

* 1. PUT: sửa một cấu hình máy in tương ứng với máy in

API: [http://localhost:8080/api/v1/configuration/printer/{printer\_name}](http://localhost:8080/api/v1/configuration/printer/%7bprinter_name%7d)



* 1. DELETE: xóa tất cả cấu hình của máy in

API: <http://localhost:8080/api/v1/configurations>

* 1. DELETE: xóa cấu hình máy in tương ứng với id của nó

API: [http://localhost:8080/api/v1/configuration{id}](http://localhost:8080/api/v1/configuration%7bid%7d)

* 1. DELETE: xóa cấu hình máy in tương ứng với tên máy in

**API:** [http://localhost:8080/api/v1/configuration/printer/{printer\_name}](http://localhost:8080/api/v1/configuration/printer/%7bprinter_name%7d)

1. **Class Printer**

{

“name”: tên máy in (key)(String),

“location”: vị trí (String),

“state”: trạng thái máy in (on/off) (String)

}

* 1. GET: lấy danh sách tất cả máy in

API: <http://localhost:8080/api/v1/printers>

* 1. GET: lấy máy in tương ứng với tên máy in (tên máy in là key của máy in)

API: [http://localhost:8080/api/v1/printer/{printer\_name}](http://localhost:8080/api/v1/printer/%7bprinter_name%7d)

* 1. POST: thêm một máy in vào hệ thống

API: <http://localhost:8080/api/v1/printer>

* 1. PUT: Sửa thông số (bao gồm trạng thái) của một máy in dựa vào tên của máy in

API: [http://localhost:8080/api/v1/printer/{printer\_name}](http://localhost:8080/api/v1/printer/%7bprinter_name%7d)

* 1. DELETE: Xóa tất cả máy in

API: <http://localhost:8080/api/v1/printers>

* 1. DELETE: xóa máy in tương ứng với tên máy in

API: [http://localhost:8080/api/v1/printer/{printer\_name}](http://localhost:8080/api/v1/printer/%7bprinter_name%7d)

1. **Class Login**

Body:

{

“id”: id của login được tạo tự động, id ảo, không có giá trị sử dụng (long),

“username”: tên đăng nhập (String),

“password”: mật khẩu (String)

}

* 1. GET: lấy danh sách tất cả thông tin log in (bao gồm cả sinh viên và SPSO)

API: <http://localhost:8080/api/v1/logins>

* 1. GET: lấy thông tin login dựa vào id của login

API: [http://localhost:8080/api/v1/login/{id}](http://localhost:8080/api/v1/login/%7bid%7d)

* 1. POST: tạo một thông tin login (không phân biệt sinh viên hay SPSO)

API: <http://localhost:8080/api/v1/login>

Note: không thể tạo trùng username

* 1. PUT: cập nhật thông tin login dựa vào id của login (không phân biệt sinh viên hay SPSO)

API: [http://localhost:8080/api/v1/login/{id}](http://localhost:8080/api/v1/login/%7bid%7d)

Note: chỉ được cập nhật mật khẩu, không được cập nhật username

* 1. DELETE: xóa thông tin login dựa vào id của login

API: [http://localhost:8080/api/v1/login/{id}](http://localhost:8080/api/v1/login/%7bid%7d)

* 1. DELETE: xóa hết thông tin login

API: <http://localhost:8080/api/v1/logins>

1. **Class SPSO**

Body:

{

“id”: do người dùng nhập (long),

“name”: tên nhân viên (String),

“phoneNum”: số điện thoại (String),

“doB”: ngày sinh nhân viên (Date),

“email” : email nhân viên (String)

“login”: {

“id”: long

“username”: String,

“password”: String

}

}

* 1. GET: Lấy danh sách tất cả SPSO

API: <http://localhost:8080/api/v1/spsos>

5.2 GET: Lấy SPSO dựa vào id

API: : [http://localhost:8080/api/v1/spso/{id}](http://localhost:8080/api/v1/spso/%7bid%7d)

* 1. POST: Thêm 1 SPSO

API: : <http://localhost:8080/api/v1/spso>

Lưu ý: nhập trùng id với spso đã có sẽ báo lỗi, muốn update thì dùng put

5.4. PUT: Chỉnh sửa SPSO dựa vào id

API: : [http://localhost:8080/api/v1/spso/{id}](http://localhost:8080/api/v1/spso/%7bid%7d)

* 1. DELETE: Xóa SPSO dựa vào id

API: : [http://localhost:8080/api/v1/spso/{id}](http://localhost:8080/api/v1/spso/%7bid%7d)

* 1. DELETE: xóa tất cả SPSO:

API: : <http://localhost:8080/api/v1/spsos>

5.7: GET: lấy thông tin login của SPSO tương ứng với id

API: [http://localhost:8080/api/v1/spso/{id}/login](http://localhost:8080/api/v1/spso/%7bid%7d/login)

1. **Class Printing System (thấy không cần thiết lắm)**

Body:

{

“id”:

“printers”: list các máy in,

“printlogs”: list các print log

}

1. **Class Print Modification**

Body:

{

“id”: tự động tạo(long),

“paperSize”: cỡ giấy (String),

“copies”: số bản sao (in),

“doubleSize”: có in hai mặt hay không(boolean)

}

* 1. GET: lấy danh sách tất cả print modification

API: <http://localhost:8080/api/v1/printModifications>

* 1. GET: lấy print modification ứng với id

API: [http://localhost:8080/api/v1/printModification/{id}](http://localhost:8080/api/v1/printModification/%7bid%7d)

* 1. POST: tạo một print modification mới

API: <http://localhost:8080/api/v1/printModification>

* 1. PUT: cập nhật print modification

API: [http://localhost:8080/api/v1/printModification/{id}](http://localhost:8080/api/v1/printModification/%7bid%7d)

* 1. DELETE: xóa tất cả

API: <http://localhost:8080/api/v1/printModifications>

* 1. DELETE: xóa 1 print modification dựa vào id

API: [http://localhost:8080/api/v1/printModification/{id}](http://localhost:8080/api/v1/printModification/%7bid%7d)

1. **Class Buy Log**

**Body:**

{

“id”: buy log id (long)

“paperSize”: cỡ giấy (String)

“boughtPageNum”: số trang (int),

“price”: tổng giá,

“paymentTime”: thời gian thanh toán,

“student”: {

“id”:MSSV,

và các thông tin khác

}

}

* 1. GET: lấy tất cả buy log

API: http://localhost:8080/api/v1/buyLogs

* 1. GET: lấy buy log ứng với id của nó

API: http://localhost:8080/api/v1/buyLogs/{id}

* 1. GET: lấy tất cả buy log của một sinh viên

API: [http://localhost:8080/api/v1/student/{id}buyLogs](http://localhost:8080/api/v1/student/%7bid%7dbuyLogs)

* 1. POST: create một buy log

API: <http://localhost:8080/api/v1/buyLog>

8.5: PUT: sửa thông tin một buy log dựa trên ID của nó

API: <http://localhost:8080/api/v1/buyLog>

* 1. DELETE: xóa một buy Log

API: [http://localhost:8080/api/v1/buyLog/{id}](http://localhost:8080/api/v1/buyLog/%7bid%7d)

* 1. DELETE: xóa tất cả buy Log

API: <http://localhost:8080/api/v1/buyLogs>

* 1. Xóa tất cả buy log của 1 sinh viên

API: http://localhost:8080/api/v1/student/{id}/buyLogs

1. **Class PaperSetting**

Body:

{

“id”: tự động tạo (long),

“paperSize”: chuỗi các cỡ giấy được cho phép, cách nhau bởi “, “ (ví dụ: “A4, A3, letter”) (String)

“numPage”: số trang A4 mặc định (int),

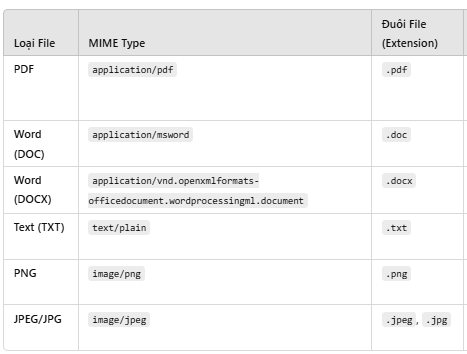
“validFileType”: chuỗi các loại file được cho phép, cách nhau bởi “, “ (ví dụ: pdf, pnj, jpeg, ”) (String)

“settingDate”: ngày thiết lập (Timestamp)

}

Bảng filetype:

(VD: nếu nhập file .pdf và .doc thì nhập vào: “pdf, doc” chứ không nhập dấu chấm nha, rồi t dựa vào chuỗi này mà so sánh cho phép tạo các document nha hii.



9.1 GET: lấy paper setting dựa trên id

API: [http://localhost:8080/api/v1/paperSetting/{id}](http://localhost:8080/api/v1/paperSetting/%7bid%7d)

9.2 GET: lấy paper setting mới nhất

API: <http://localhost:8080/api/v1/paperSetting>

9.3 POST: thêm 1 paper setting

API: <http://localhost:8080/api/v1/paperSetting>

9.4 PUT: update paper setting với id của nó

API: [http://localhost:8080/api/v1/paperSetting/{id}](http://localhost:8080/api/v1/paperSetting/%7bid%7d)

9.5 DELETE: xóa tất cả paper setting

API: <http://localhost:8080/api/v1/paperSettings>

9.6 DELETE: xóa paper setting bằng id

API: [http://localhost:8080/api/v1/paperSetting/{id}](http://localhost:8080/api/v1/paperSetting/%7bid%7d)

1. **Class Report**

Body:

{

“id”: tạo tự động,

“name”: tên của báo cáo (String),

“date”: Date,

“filePath” : đường dẫn đến vị trí lưu trữ của file trong local

}

* 1. GET: lấy danh sách tất cả báo cáo

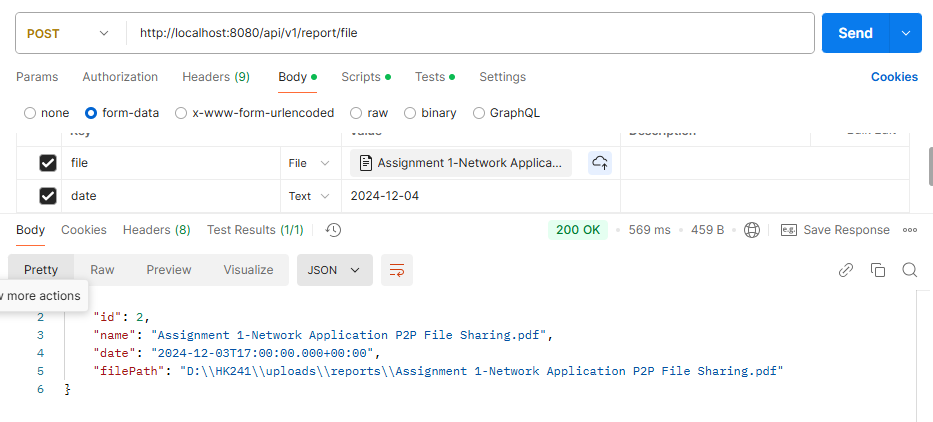
API: <http://localhost:8080/api/v1/reports>

* 1. GET: lấy báo cáo dựa trên id của nó

API: [http://localhost:8080/api/v1/report/{id}](http://localhost:8080/api/v1/report/%7bid%7d)

* 1. POST: tạo một đối tượng báo cáo mới với 2 tham số nhập vào là file (multipart file) và date (Date)

example:



API: <http://localhost:8080/api/v1/report>

(Nhập vào file, back end sẽ lưu file vào thư mục: D:/HK241/uploads/reports (đối với máy t, cấu hình lại thư mục trong class com.example.demo.Config.FileProperties )

* 1. PUT: sửa thông tin báo cáo

API: [http://localhost:8080/api/v1/report/{id}](http://localhost:8080/api/v1/report/%7bid%7d)

Sửa được tên và ngày. Các thông số như id và file path không được sửa.

* 1. DELETE: xóa một report bằng id của nó

API: [http://localhost:8080/api/v1/report/{id}](http://localhost:8080/api/v1/report/%7bid%7d)

Xóa cả file được lưu ở filepath.

* 1. Xóa tất cả report

API: <http://localhost:8080/api/v1/reports>

1. **Document**

**body:**

{

“id”: id của tài liệu (nhập thủ công nhe)

“fileName”: tên của tài liệu, được tạo từ tên file upload

“fileType”: kiểu file, được lấy từ file upload,

“filePath”: đường dẫn đến thư mục chứa file.

}

* 1. GET: lấy danh sách tất cả tài liệu

API: <http://localhost:8080/api/v1/documents>

* 1. GET: lấy tài liệu theo id

API: [http://localhost:8080/api/v1/document/{id}](http://localhost:8080/api/v1/document/%7bid%7d)

* 1. POST: tạo một tài liệu mới, với tham số đầu vào là student id và request param multipart file.

API: [http://localhost:8080/api/v1/document/{document\_id}/{student\_id}](http://localhost:8080/api/v1/document/%7bdocument_id%7d/%7bstudent_id%7d)

Tương tự như report, document sẽ được lưu vào thư mục. Nhưng document sẽ được lưu vào “D:/HK241/uploads/documents/student\_id”, do đó, yêu cầu mỗi document phải được tạo với id của sinh viên sở hữu nó.

* 1. PUT: chỉnh sửa một tài liệu dựa trên id của tài liệu

API: [http://localhost:8080/api/v1/document/{id}](http://localhost:8080/api/v1/document/%7bid%7d)

Chỉnh sửa được mọi thứ, nhưng không chỉnh được file path.

* 1. DELETE: xóa tất cả tài liệu

API: <http://localhost:8080/api/v1/documents>

* 1. DELETE: xóa tất cả tài liệu của 1 sinh viên

API: [http://localhost:8080/api/v1/documents/{studentid}](http://localhost:8080/api/v1/documents/%7bstudentid%7d)

* 1. DELETE: xóa tài liệu trên id của tài liệu

API: [http://localhost:8080/api/v1/document/{id}](http://localhost:8080/api/v1/document/%7bid%7d)

1. **Class Print Log**

Body:

{

“id”: tự động tạo (long),

“status”: trạng thái (String),

“startTime”: Timestamp,

“finishedTime”: Timestamp,

“document”:

{

“id”:

//các thuộc tính khác

}

“printModification”:

{

“id”

//các thuộc tính khác

}

“student”:

{

“id”:

}

“printer”:

{

“name”

}

}

* 1. GET: lấy danh sách tất cả print log

API: <http://localhost:8080/api/v1/printLogs>

* 1. GET: lấy print log ứng với id của nó

API: [http://localhost:8080/api/v1/printLog/{id}](http://localhost:8080/api/v1/printLog/%7bid%7d)

* 1. GET: lấy danh sách print log của sinh viên

API: [http://localhost:8080/api/v1/student/{student\_id}/printLogs](http://localhost:8080/api/v1/student/%7bstudent_id%7d/printLogs)

* 1. GET: lấy danh sách print log của máy in

API: [http://localhost:8080/api/v1/printer/{printer\_name}/printLogs](http://localhost:8080/api/v1/printer/%7bprinter_name%7d/printLogs)

* 1. POST: thêm 1 print log mới

API: <http://localhost:8080/api/v1/printLog>

* 1. PUT: chỉnh sửa thông tin print log

API: [http://localhost:8080/api/v1/printLog/{id}](http://localhost:8080/api/v1/printLog/%7bid%7d)

* 1. DELETE: xóa tất cả print log

API: <http://localhost:8080/api/v1/printLogs>

* 1. DELETE: xóa Print Log bằng id

API: <http://localhost:8080/api/v1/printLog>/{id}