VIETNAM NATIONAL UNIVERSITY, HO CHI MINH CITY
UNIVERSITY OF TECHNOLOGY
FACULTY OF COMPUTER SCIENCE AND ENGINEERING


## Công Nghệ Phần Mềm (CO3001)

---

### Bài Tập Lớn

# Hệ thống dịch vụ in ấn thông minh HCMUT-SSPS

---

Advisor: Trần Trương Tuấn Phát
Lớp: L02
Students: Nguyễn Hữu Huy Thịnh - 2213291
Nguyễn Anh Khoa - 2211612
Nguyễn Đình Nam - 2212136
Lê Thành Đạt - 2210683
Nguyễn Gia Thịnh - 2213286
Vương Quang Khải - 2211562
Trịnh Đình Khải - 2211561

THÀNH PHỐ HỒ CHÍ MINH, Tháng 11 2024

# Mục lục

# 1 Bảng phân công công việc

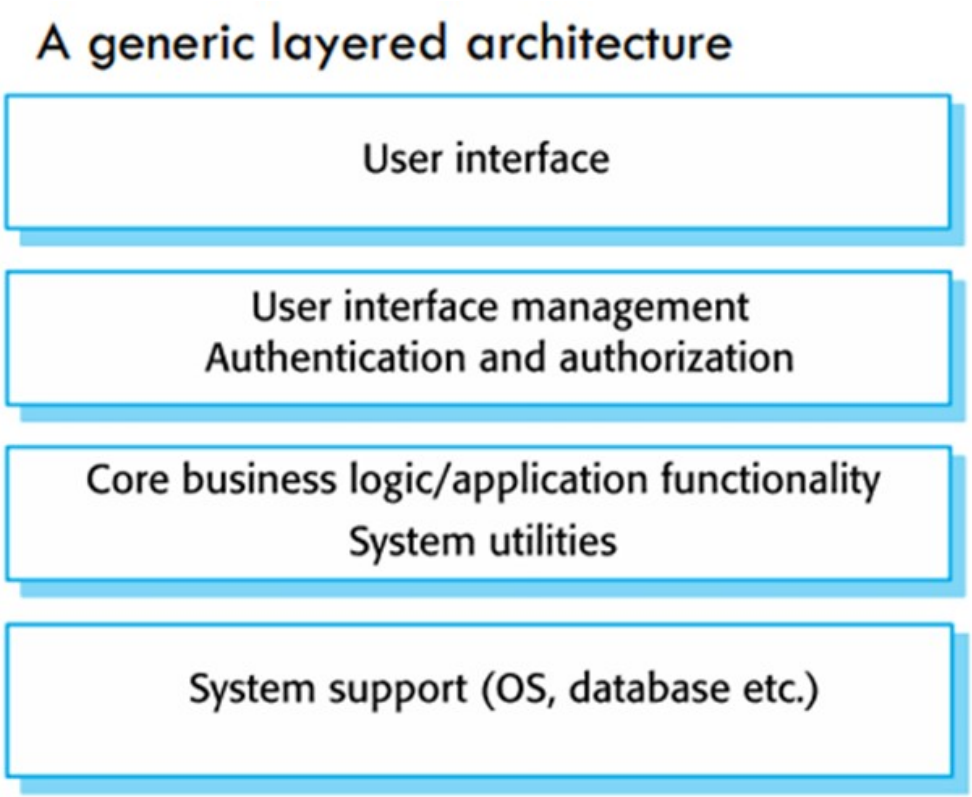
## BẢNG CÔNG VIỆC BÀI TẬP LỚN

| No. | Họ và tên | MSSV | Công việc | Mức độ hoàn thành |
|---|---|---|---|---|
| 1 | Nguyễn Đình Nam | 2212136 | - Layered Architecture<br>- Scrum Master | 100% |
| 2 | Nguyễn Gia Thịnh | 2213286 | - Presentation Strategy | 100% |
| 3 | Lê Thành Đạt | 2210683 | - API Management | 100% |
| 4 | Nguyễn Anh Khoa | 2211612 | - Component Diagram | 100% |
| 5 | Trịnh Đình Khải | 2211561 | - Layered Architecture | 100% |
| 6 | Vương Quang Khải | 2211562 | - Data Storage Approach | 100% |
| 7 | Nguyễn Hữu Huy Thịnh | 2213291 | - Component Diagram | 100% |

# 2 Architectural Design

## 2.1 Layered Architecture

Kiến trúc phân tầng (hay còn gọi là kiến trúc n-tier) - là một trong những mô hình phổ biến nhất và thường được xem là tiêu chuẩn không chính thức cho các ứng dụng Java EE. Mô hình này rất quen thuộc với các kiến trúc sư, nhà thiết kế và nhà phát triển phần mềm. Ý tưởng chính của kiến trúc phân tầng là tổ chức các modules và các thành phần có chức năng tương tự vào các tầng nằm ngang. Nhờ đó, mỗi tầng sẽ đóng một vai trò cụ thể, độc lập trong ứng dụng, giúp việc phát triển và sửa lỗi trong một tầng không ảnh hưởng đến các tầng khác.




Hình 1: Mẫu kiến trúc phân tầng chung (Nguồn: Slide bài giảng)

Tuy nhiên, kiến trúc phân tầng vẫn tồn tại một số hạn chế nhất định:

- Tính linh hoạt tổng thể: Do tính chất nguyên khối của kiến trúc, các hệ thống cần thay đổi thường xuyên sẽ gặp khó khăn và tốn nhiều thời gian khi muốn điều chỉnh. Đối với hệ

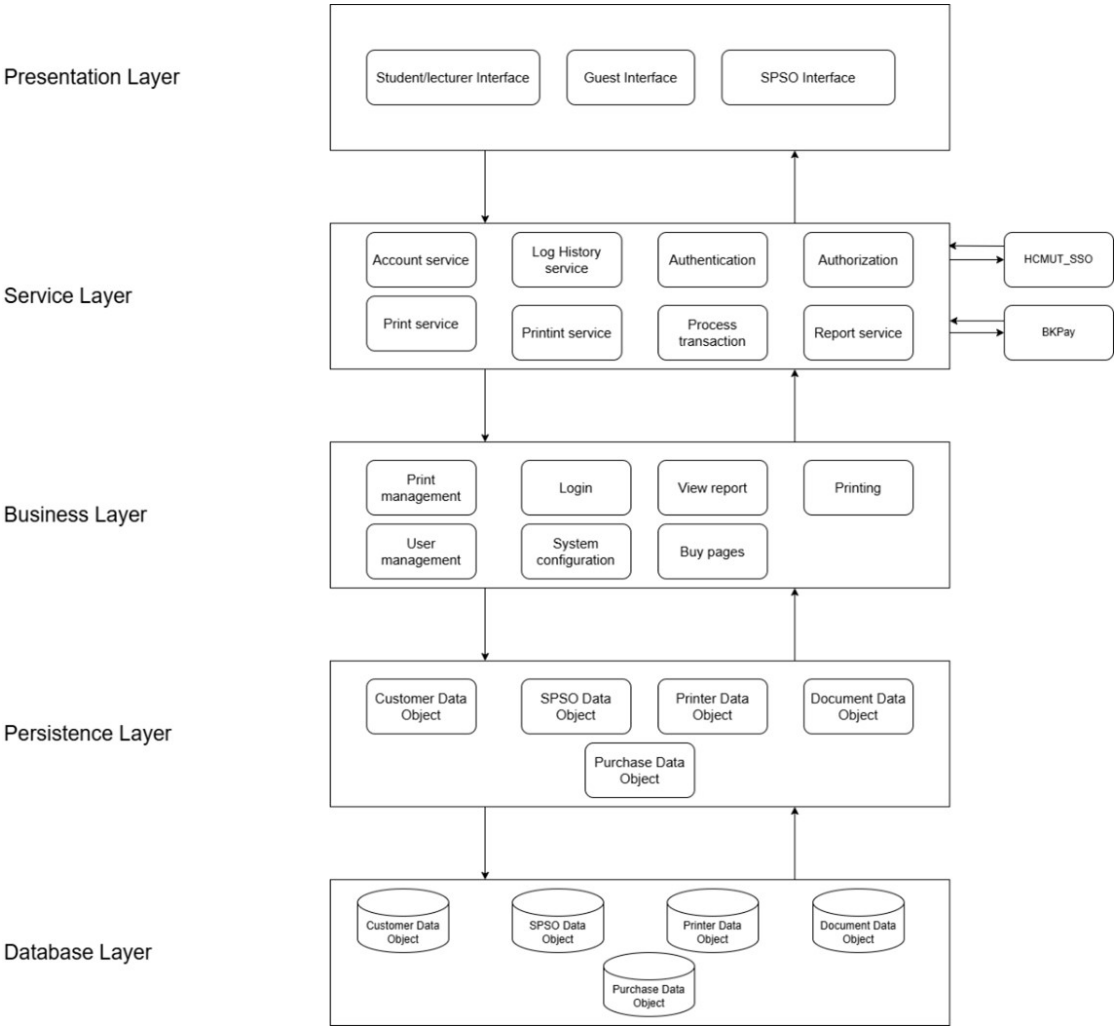
thống in ấn HCMUT_SSPS, vì đây là hệ thống đơn giản, cần triển khai nhanh chóng và chỉ tập trung vào chức năng chính là đáp ứng nhu cầu in ấn của sinh viên và cán bộ công nhân viên trong trường, nên có thể tạm thời bỏ qua vấn đề linh hoạt tổng thể.

- Hiệu năng: Kiến trúc phân tầng được coi là lựa chọn mặc định khi không có kiến trúc nào phù hợp hơn, nhờ vào tính đơn giản của nó. Tuy nhiên, trong các ứng dụng lớn, kiến trúc này có thể làm giảm hiệu năng. Nguyên nhân là do phải đi qua từng tầng để đáp ứng yêu cầu nghiệp vụ, dẫn đến giảm hiệu quả khi cần thực hiện qua nhiều tầng của kiến trúc để hoàn thành một tác vụ. Tuy nhiên, hệ thống HCMUT_SSPS hiện nay chưa có quy mô quá lớn, do đó nhược điểm này có thể chấp nhận được.

- Khả năng mở rộng: Kiến trúc phân tầng thường được triển khai theo hướng nguyên khối và có tính liên kết chặt chẽ, dẫn đến khó khăn trong việc mở rộng quy mô khi cần thiết. Tuy nhiên, vì HCMUT_SSPS chủ yếu phục vụ nhu cầu in ấn nội bộ trong trường, nên có thể chưa cần thiết để xem xét đặc điểm này ngay.

Ngoài những hạn chế nêu trên, kiến trúc phân tầng vẫn có một số ưu điểm khác. Từ các phân tích trên, chúng tôi quyết định triển khai hệ thống HCMUT_SSPS theo kiến trúc phân tầng để phù hợp với yêu cầu hiện tại.

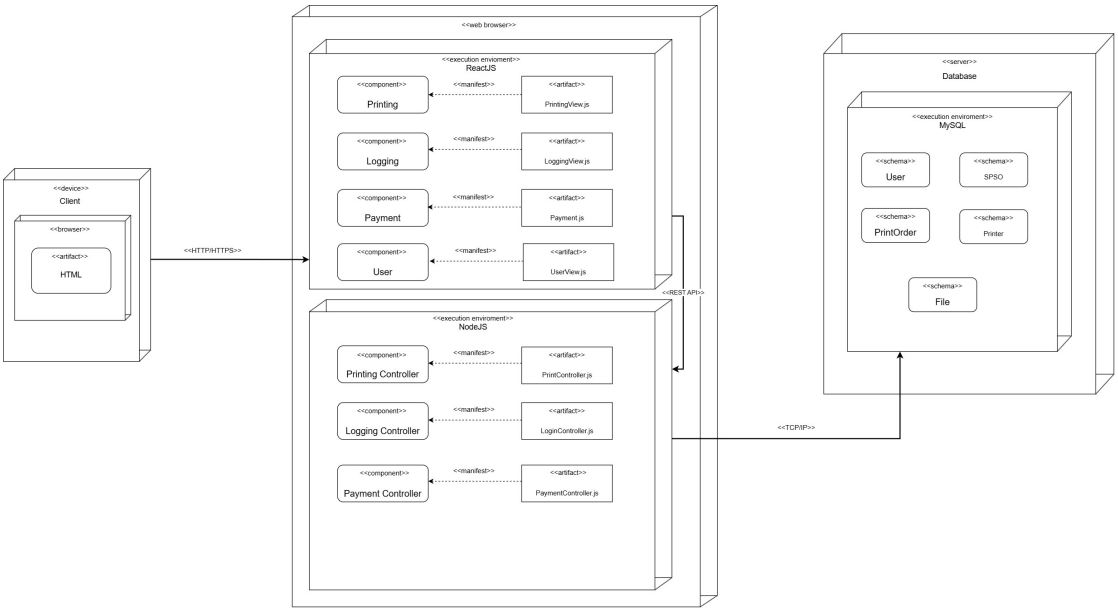
### 2.1.1 Box-Line Diagram




Hình 2: Box-line diagram của toàn hệ thống

Hệ thống được sử dụng bởi 3 đối tượng: sinh viên/giảng viên, khách và SPSO. Vì thế giao diện của hệ thống cũng được chia ra làm 3 giao diện. Giao diện của sinh viên/giảng viên có các chức năng như đăng nhập, in tài liệu, mua trang in, xem lịch sử in. Dịch vụ HCMUT_SSO chịu trách nghiệm xác thực người dùng. Chức năng mua trang in sẽ truy cập đến dịch vụ thanh toán của BKPay. Tương tự thì giao diện của SPSO thì sẽ có các chức năng như đăng nhập, xem lịch sử in của sinh viên/giảng viên, quản lí người dùng, quản lí máy in, chỉnh sửa cấu hình hệ thống. Đối với với người dùng khách thì sẽ chỉ các chức năng như hướng dẫn người dùng, giới thiệu chức năng hệ thống hoặc có thể đăng nhập vào tài khoản của mình.

### 2.1.2 Deployment Diagram




Hình 3: Deployment Diagram

Hình 4: Box-line diagram của toàn hệ thống

- Máy Client: Máy khách kết nối với hệ thống qua giao thức HTTP/HTTPS. Dữ liệu nhận được từ Server là các file hiện thực giao diện, được hiển thị trên trình duyệt thông qua HTML. Trên máy Client, trình duyệt sẽ xử lý và hiển thị các thành phần giao diện cho người dùng.
- Máy chủ Web (Server): Máy chủ này bao gồm hai môi trường thực thi (execution environment): ReactJS cho phía front-end và NodeJS cho phía back-end.
  - Phía front-end được xây dựng bằng ReactJS với các component chính bao gồm Printing, Logging, Payment và User. Mỗi component sẽ được hiện thực từ các file JavaScript tương ứng như PrintingView.js, LoggingView.js, Payment.js và UserView.js.
  - Phía back-end được xây dựng bằng NodeJS, sử dụng các controller như Printing Controller, Logging Controller, Payment Controller và User Controller. Được hiện thực từ các file như PrintController.js, LoginController.js, PaymentController.js và UserController.js.

Front-end và Back-end giao tiếp với nhau thông qua các REST API để xử lý tác vụ và cập nhật giao diện người dùng.

- Máy chủ Cơ sở Dữ liệu (Database Server): Máy chủ này sử dụng MySQL để lưu trữ dữ liệu. Các bảng dữ liệu chính bao gồm User, SPSO, PrintOrder, Printer và File. Máy chủ Web sẽ kết nối với Database Server thông qua giao thức TCP/IP để truy xuất và cập nhật dữ liệu khi có sự tương tác từ người dùng.
- Quá trình xử lý dữ liệu: Khi người dùng tương tác với ứng dụng trên Client, dữ liệu được gửi đến máy chủ Web qua HTTP/HTTPS. Tại đây, các REST API sẽ xử lý và trao đổi dữ liệu giữa Front-end và Back-end. Sau đó, dữ liệu sẽ được chuyển đến Database Server để lưu trữ hoặc lấy ra nếu cần.

## 2.2 Presentation Strategy

Chúng em sẽ tập trung vào sự đơn giản nhưng mang lại hiểu quả cao trong trải nghiệm của các nhóm người sử dụng. Vì thế chúng em quyết định sẽ sử dụng một số công nghệ sau đây:

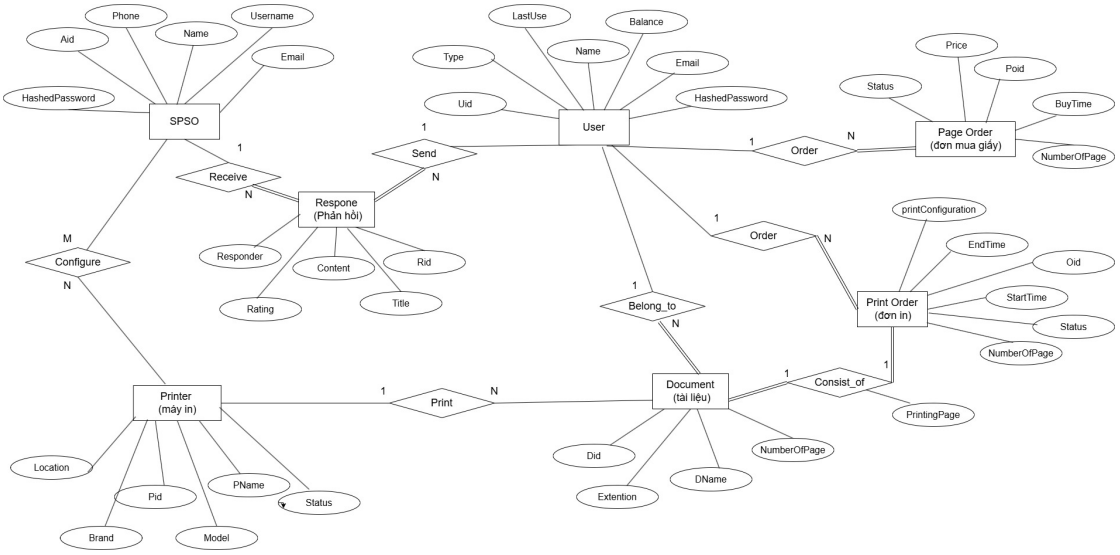
- Front-end: công nghệ phát triển giao diện người dùng như React. React cung cấp đầy đủ các công cụ để thiết kế giao diện đẹp mắt, thân thiện.
- Responsive Web Design là xu hướng mới theo đó quy trình thiết kế và phát triển web sẽ đáp ứng mọi thiết bị và môi trường của người dùng theo các tiêu chí kích thước và chiều của màn hình thiết bị, đáp ứng mọi thiết bị mà sinh viên và cán bộ công nhân viên nhà trường hiện có đảm bảo trải nghiệm.
- Các tính năng thân thiện với người dùng: Nhóm sẽ chú ý đến các yếu tố trực quan để tạo điều kiện cho người dùng có thể tiếp cận đến mà không gặp nhiều khó khăn như các nút bấm, các biểu mẫu, các menu điều hướng rõ ràng.

## 2.3 Data Storage Approach

Trong layered architecture, database sẽ là lớp ở dưới cùng và sẽ chịu trách nhiệm lưu trữ toàn bộ data. Các dữ liệu của ứng dụng sẽ được lưu trữ tại đây và các thao tác như create, retrive, update, delete sẽ được thực hiện thường xuyên để thao tác với dữ liệu thông qua hệ quản trị cơ sở dữ liệu. Đối với HCMUT-SSPS, database layer sẽ được lưu trữ bằng cơ sở dữ liệu quan hệ và được hiện thực hóa bằng hệ quản trị cơ sở dữ liệu MySQL, có nghĩa là dữ liệu của ứng

dụng sẽ được lưu trữ dưới dạng các bảng và các mối quan hệ giữa chúng. Đối với hệ thống HCMUT-SSPS, cơ sở dữ liệu sẽ bao gồm các thực thể sau:

- User: Lưu các thông tin của khách hàng như: Mã khách hàng (là Mã số sinh viên/Mã số cán bộ tại trường Đại học Bách Khoa), Họ và tên, Mật khẩu (được mã hóa), Loại (Sinh viên hay Cán bộ nhà trường), Email, Số dư (Số trang in người dùng đang có, đơn vị là 1 trang A4), và Lần sử dụng cuối.
- SPSO: Lưu các thông tin: Mã SPSO, Họ và tên, Tên tài khoản (dùng để đăng nhập), Mật khẩu (được mã hóa), Email, Số điện thoại.
- Máy in: Bao gồm các thông tin như: Mã máy in, Tên máy in, Nhãn hiệu, Mẫu máy, Mô tả, Vị trí (Cơ sở, Tòa, Phòng), Trạng thái.
- Tài liệu: Lưu các thông tin: Mã tài liệu, Tên tài liệu, Định dạng file, Số trang tài liệu.
- Đơn in: Bao gồm các thuộc tính như: Mã đơn in, Cấu hình in (Hướng giấy, Cỡ giấy, Loại mặt, Số trang/mặt in, Tỷ lệ), Thời gian bắt đầu, Thời gian kết thúc, Trạng thái, Số lượng trang sử dụng.
- Đơn mua trang: Bao gồm các thuộc tính như: Mã giao dịch, Thời điểm giao dịch, Số lượng trang mua, Giá mua, Trạng thái.
- Đơn phản hồi: Bao gồm các thuộc tính như: Mã phản hồi, Tiêu đề, Nội dung, Đánh giá, Người phản hồi.




Hình 5: Enhanced ER Diagram của hệ thống

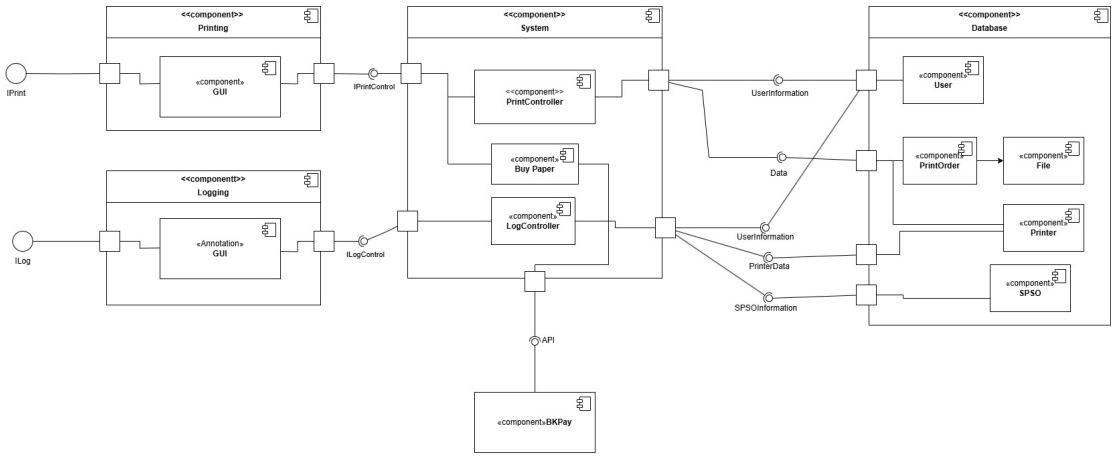
## 2.4 API Management

API (Application Programming Interface - Giao diên lập trình ứng dụng) là các phương thức, giao thức và công cụ cho phép các ứng dụng phần mềm giao tiếp và tương tác với nhau API cung cấp khả năng truy xuất đến một tập các hàm hay dùng, từ đó có thể trao đổi dữ liệu giữa các ứng dụng. Ngoài ra việc sử dụng API giúp các module "che giấu" các hàm không nên được module khác truy cập vào, từ đó tăng tính độc lập cho các module. Các API cho hệ thống in ấn tự động HCMUT_SSPS bao gồm:

- API giao diện người dùng: Cung cấp các chức năng hiển thị giao diện người dùng trên các thiết bị.
- API bảo mật và xác thực: Đảm bảo tính bảo mật và an toàn cho các giao tiếp và hoạt động in ấn bằng cách xác thực đăng nhập thông qua HCMUT-SSO và kiểm soát quyền truy cập bởi SPSO.
- API quản lý và điều khiển máy in: Cho phép ứng dụng hiển thị các máy in khả dụng và tương tác với máy in.
- API định dạng và xử lý dữ liệu đầu vào: Cho phép ứng dụng gửi dữ liệu đầu vào (như tệp in, hình ảnh, hoặc văn bản) và kiểm tra định dạng cho hệ thống.

- API quản lý công việc in: Cung cấp các API cho việc tạo, quản lý, và theo dõi các công việc in ấn. Điều này có thể bao gồm lập lịch in, theo dõi trạng thái của công việc in, kiểm tra số trang in khả dụng và hủy bỏ công việc in ấn.
- API thanh toán: Cung cấp khả năng tính toán chi phí cho mỗi lần in, hỗ trợ mua thêm trang thông qua BKPay.
- API báo cáo và thống kê: Cung cấp dữ liệu về công việc in ấn đã hoàn tất, thời gian in, số lượng trang in, và các thông tin liên quan khác để theo dõi và báo cáo về hoạt động in ấn.
- API lịch sử giao dịch: Cho phép cả người dùng và hệ thống truy xuất lịch sử in ấn của người dùng, giúp xem lại cái bản in cũ.

## 2.5 Component Diagram




Hình 6: Component Diagram của module in ấn

Mô tả Component Diagram của module in ấn:

- Component Diagram gồm các component: Printing, Logging, System, Database
- Printing: Đây là component chứa giao diện người dùng và các dịch vụ "in tài liệu". Bên trong chứa component con GUI để người dùng tương tác với hệ thống. Component này giao tiếp với các component con của System thông qua interface IprintControl để người dùng có thể thực hiện các dịch vụ in như đăng tải file, điều chỉnh và xác nhận cấu hình và thực hiện thanh toán.

- Logging: Đây là component chứa giao diện để SPSO có thể kiểm soát thông tin và quá trình in ấn của người dùng. Component này giao tiếp với System thông qua IprinterHandle, giúp SPSO có thể sử dụng dịch vụ kiểm soát Log.
- System: Bao gồm các component con PrintController, Buy Paper và Log Controller. Component PrintController và Buy Paper nhận lệnh thông qua interface IPrintControl từ component Printing. Đối với PrintController, component này sẽ tiến hành yêu cầu và cập nhật các dữ liệu cần thiết về người dùng và máy in thông qua interface UserInformation. Đối với Buy Paper, component sẽ yêu cầu dịch vụ từ component BKPay thông qua API được cung cấp. Đối với LogController, component được nhận lệnh từ GUI của component Logging, sau đó tiến hành truy cập dữ liệu thông qua các interface Userinformation, PrinterData và SPSOInformation để truy cập dữ liệu từ Database.
- BKPay: Component cung cấp API cho hệ thống để có thể xử lý thanh toán khi có yêu cầu mua giấy.
- Database: Chứa các component con User, PrintOrder, File, Printer và SPSO. Lưu trữ thông tin dữ liệu và truyền dữ liệu đi nếu được truy cập, cung cấp các interface cần thiết cho hệ thống để xử lý.

# Tài liệu