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

SOFTWARE ENGINEERING (CO3001)

Software Requirements Specification for HCMUT - SPSO

Instructor(s):
Trương Tuấn Anh , CSE - HCMUT

Team name:
Class CC03 - Group CC03-01 - Semester 241

Student(s):
Cao Ngọc Lâm - 2252419 - Team Leader
Đặng Minh Khang - 2252287
Đặng Ngọc Phú - 2252617
Đoàn Anh Quang - 2252666

HO CHI MINH CITY, SEPTEMBER 2024

Contents

1 Task 1: Requirement eliciation (1.1, 1.2)

1.1 Domain Context

The HCMUT Student Smart Printing Service (HCMUT_SSPS) is an advanced printing solution specifically designed to address the document printing needs of students at Ho Chi Minh City University of Technology. The system operates via a distributed network of printers located across the university's multiple campuses, with each printer assigned a unique ID and location details (campus, building, room number). This setup provides students with convenient access to printers, enabling them to upload documents, select specific printers, and configure print settings such as paper size, number of copies, and one- or double-sided printing.

To ensure security and ease of use, HCMUT_SSPS is integrated with the university's Single Sign-On (SSO) authentication system, ensuring that only authorized students can access the service. The system also allocates a quota of free A4 pages to each student per semester, and students can purchase additional pages via an integrated online payment platform like BKPay (the payment platform of HCMUT). The system's tracking mechanism supports fair use by monitoring students' printing activities and maintaining a balance of available pages.

For system management, the Student Printing Service Officers (SPSO) have access to a range of administrative tools, allowing them to manage printers, configure system settings, and generate detailed reports. The system logs all printing activities, including student IDs, file names, and printing times, providing transparency and accountability. These logs, along with automated monthly and annual reports, allow SPSOs and the university administration to efficiently monitor resource usage and optimize the printing infrastructure.

In terms of scalability, HCMUT_SSPS is designed to adapt to future growth, with support for the addition of new printers and customization of accepted file types. The system could potentially be integrated with other university systems, such as a learning management system (LMS), enhancing the academic experience. By providing a reliable, accessible, and efficient printing service, HCMUT_SSPS ensures that students can meet their academic needs while allowing the university to manage resources effectively.

1.2 Stakeholders and Needs

The HCMUT Student Smart Printing Service (HCMUT_SSPS) serves three primary stakeholders: students, Student Printing Service Officers (SPSO), and the Information Technology (IT) Department, each with unique needs that the system addresses effectively.

1. **Students**: Students are the main users of HCMUT_SSPS, requiring convenient access to printers across the university's campuses. They benefit from a system that allows them to select the closest available printer and configure their print jobs according to their specific requirements, including paper size and printing options (e.g., single or double-sided).

   Students are also given a quota of free A4 pages each semester and can easily track their usage through the system. If they exceed their quota, the integrated online payment option (e.g., BKPay) enables them to purchase additional pages seamlessly. Transparency in usage is critical, so the system provides detailed logs of their printing history, helping them manage their printing activities efficiently.

2. **Student Printing Service Officers (SPSO)**: Student Printing Service Officers (SPSO) are responsible for the operational management of HCMUT_SSPS. They need comprehensive tools for printer management, allowing them to add or disable printers as necessary and monitor their status in real-time.

SPSOs are also tasked with configuring system settings based on university policies and student needs, such as adjusting the default number of pages allocated per semester. Access to detailed logs of all printing activities is essential for tracking usage patterns, identifying misuse, and generating reports on overall system performance.

The ability to monitor printer health and respond promptly to issues ensures that the printing service runs smoothly, minimizing disruptions for students. SPSOs are also tasked with overseeing the timely generation and submission of semester and annual logs. This crucial responsibility ensures the system's long-term viability, facilitates data-driven decision-making, and supports continuous improvement of the HCMUT_SSPS.

3. **Information Technology Department**: The Information Technology (IT) Department plays a crucial role in the implementation and maintenance of HCMUT_SSPS. Their needs involve ensuring the system's technical stability and security, integrating it with existing university systems (such as the SSO), and providing ongoing support for users and administrators.

   The IT department must also ensure that the system is scalable and can accommodate future upgrades or changes in technology. Access to data analytics and usage reports is vital for troubleshooting and enhancing system performance. In addition, they ensure secure and efficient payment processing for students purchasing additional printing pages. BKPay also requires access to transaction data to support reporting and compliance with financial regulations.

By addressing the diverse needs of each stakeholder, HCMUT_SSPS fosters a user-centric environment that enhances printing efficiency, accountability, and satisfaction across the university.

1.3 Benefits of the System

1. **Students**: Students can conveniently print documents from everywhere across two campuses, simply by uploading files onto the system and choosing the nearest printer. Additionally, the integration with online payment systems like BKPay allow the students better control for their expenses.

2. **Printing Service Provider**: Printing Service provider benefits economically through offering their services to students. They also can increase brand recognition, potentially attract more clients and customers.

3. **Student Printing Service Officer** (SPSO): They have centralized control over the entire system, thereby managing printers across two campuses effectively. Thanks to printing logs and detailed reports, SPSO can monitor student activity and resolve issues more efficiently.

4. **Online payment systems**: Online payment systems got a new stream of transactions from students, which increased their revenue. In addition, the partnership with universities helps them to strengthen their reputation and expand their market expenses.

5. **Ho Chi Minh University of Technology**: Enhancing the educational environment for students and strengthening the university's reputation for potential students.

1.4 Functional Requirements

1. **For Students:**

   (a) The system must allow students to log in using the university's Single Sign-On (SSO) system.

   (b) Students must be able to upload documents to the system for printing.

(c) The system must allow students to select a specific printer based on location and printer ID.

(d) Students must be able to configure print settings, such as paper size, number of copies, and print type (single- or double-sided).

(e) The system must provide students with detailed logs of their printing activities, including the number of pages used, time of print, and file names.

(f) Students must be able to monitor their remaining print quota and purchase additional pages through BKPay.

(g) The system should notify students when their print job is complete or if any issues occur.

(h) Students should have the option to cancel a print job before printing begins.

2. **For Student Printing Service Officers (SPSO):**

   (a) The system must allow SPSOs to monitor all printer statuses in real-time.

   (b) SPSOs must be able to add, configure, or disable printers across the university's campuses.

   (c) The system must provide SPSOs with tools to configure student print quotas each semester.

   (d) SPSOs must be able to generate detailed monthly and annual reports of printer usage, student activity, and system performance.

   (e) The system must allow SPSOs to track all printing activity logs, including student ID, file details, and timestamps.

3. **For the IT Department:**

   (a) The IT department must ensure the system integrates seamlessly with the university's SSO authentication platform.

   (b) IT must ensure secure data transmission between students, SPSOs, and BKPay.

   (c) The system must enable IT to monitor and optimize system performance through data analytics and logs.

   (d) IT should have administrative control over security features, ensuring user data protection and compliance with university policies.

   (e) The system must notify students and BKPay of successful transactions and quota updates.

1.5 Non - Functional Requirements

1. **Localization:** The system shall support multiple languages, with Vietnamese as the primary language, and be easily adaptable to include more languages as needed by the university.

2. **Accessibility:** The system shall be accessible through a web-based application by any browsers, ensuring the accessibility for all users.

3. **Security:** The system shall ensure secure authentication and data transmission, complying with the university's security policies.

4. **Integration:** The system shall integrate seamlessly with the HCMUT_SSO authentication service and the BKPay online payment system.

5. **Usability:** The system shall provide an intuitive and user-friendly interface for both students and the SPSO. The user interface shall be intuitive and accessible, allowing users to complete tasks like uploading documents and managing print settings.

6. **Availability:** The system shall have high availability and reliability, with minimal downtime to support continuous printing services.

7. **Performance:** The system is designed to efficiently manage simultaneous print requests from up to 1000 students across different printers, ensuring that there is no significant performance degradation. It will maintain optimal performance even under heavy concurrent usage, with print delays not exceeding 2 seconds.

8. **Responsiveness:** The system shall provide quick response times, with print jobs being processed promptly after submission.

9. **Data Accuracy:** The system shall ensure data accuracy and consistency in logging and reporting features.

10. **Maintainability:** The system shall be maintainable and extensible, allowing for future updates or integration with additional services.

11. **Privacy Compliance:** The system shall comply with relevant data protection regulations, ensuring the privacy and confidentiality of user data. It shall also comply with any accessibility standards applicable to educational software within the university.

2 Use-case Diagram (1.3)

2.1 Use-case Diagram for the Whole System

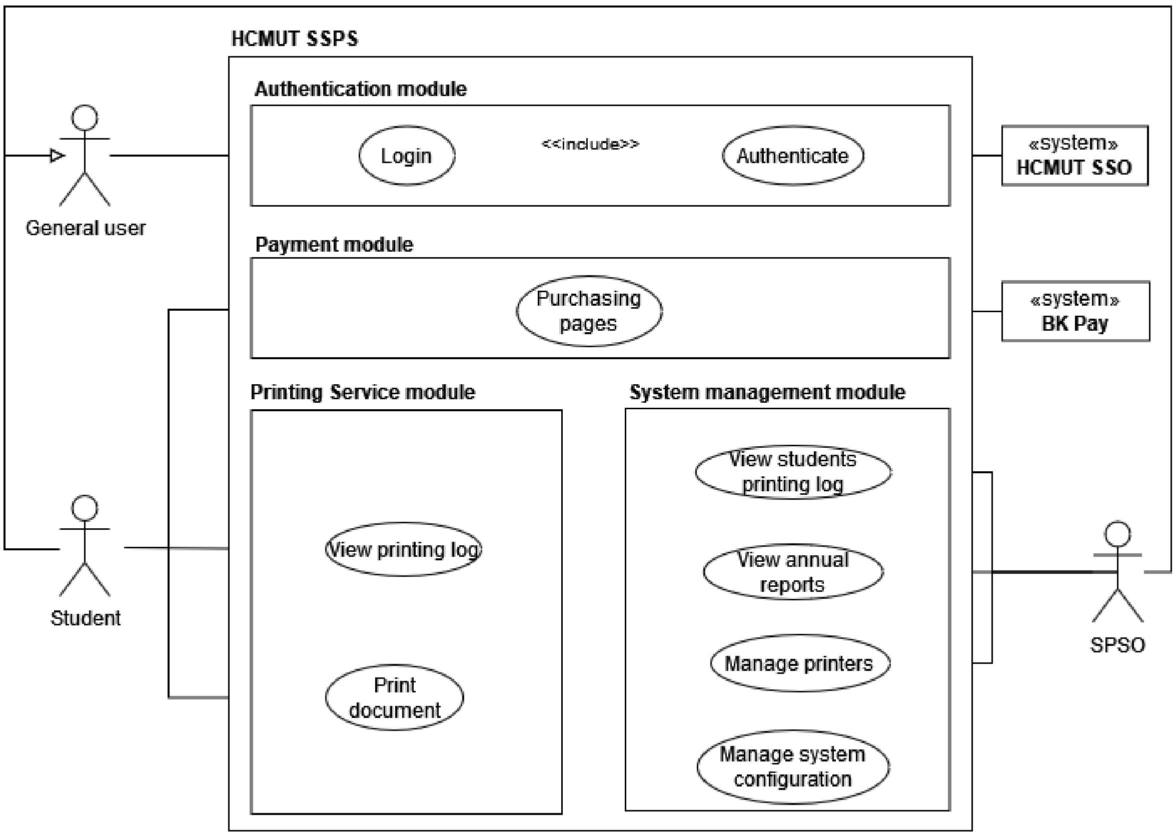


Figure 1: *Use-case Diagram for the Whole System*

2.2 Use-case Diagram for Printing Service Module

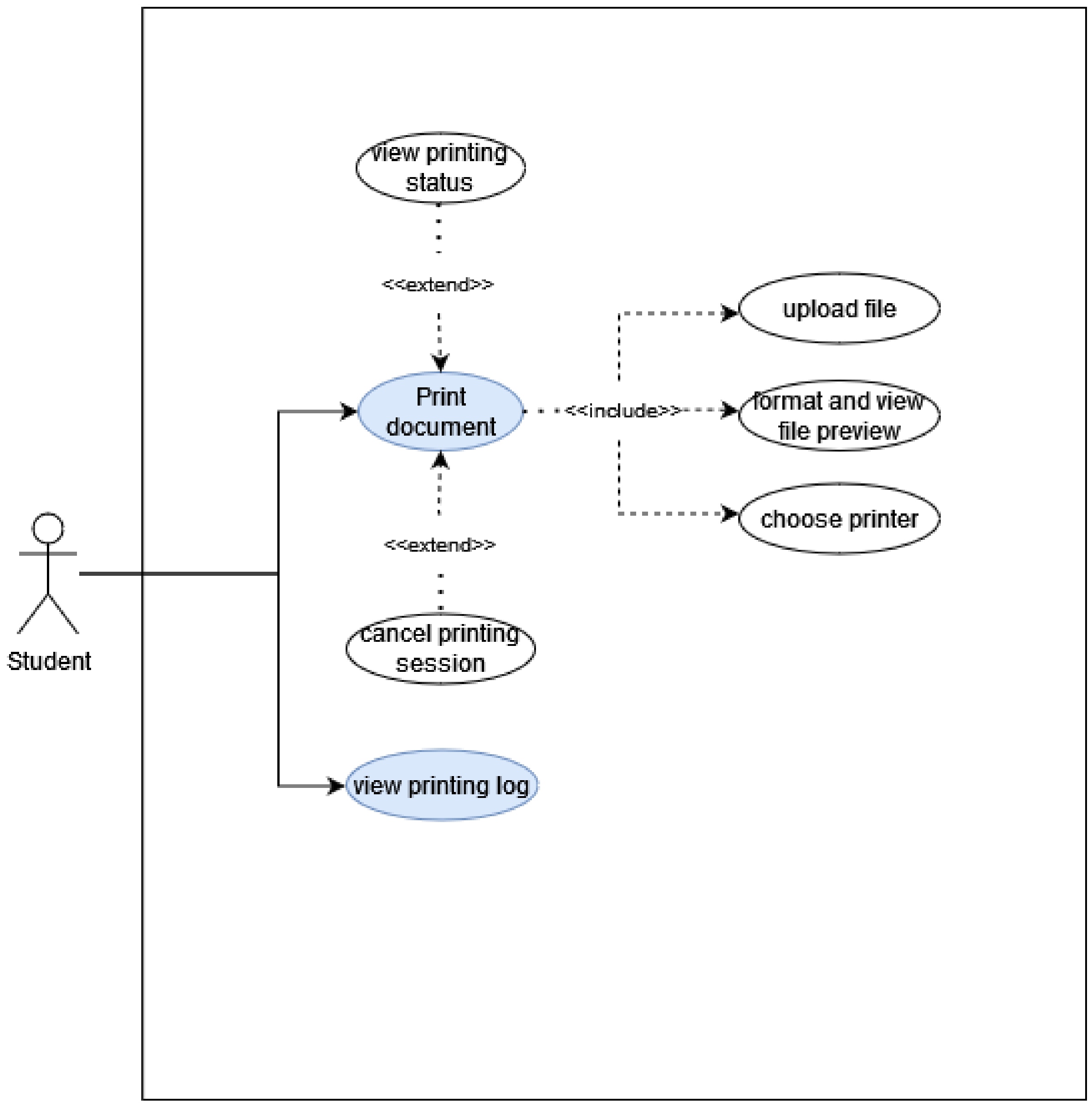


Figure 2: *Use-case Diagram for Printing Service Module*

2.3 The Details of Usecases in Printing Service Module

1. **Usecase Print document**

ID and Name	UC-1 Print Document
Date Created	29th September, 2024
Primary Actor	Student
Secondary Actor	Printer
Description	The student selects and prints a document via the smart printing system. The system allows the student to upload the file, format it, choose the printer, and send it for printing.
Trigger	The student selects the option to print a document
Preconditions	**PRE-1.** The student is authenticated. **PRE-2.** A printer is available in the system. **PRE-3.** The file is ready to be printed.
Postconditions	**POST-1.** The document is successfully sent to the printer. **POST-2.** The printer receives the job and starts the printing process.
Normal Flow	**1.0. Print Documents** (a) The student selects "Print Document". (b) The system prompts to upload the file. (c) The system shows file preview and formatting options. (d) The student chooses a printer. (e) The system processes the print request. (f) System queues the document for printing.
Alternative Flows	**1.1. Invalid file format, printers unavailable** (a) System shows an error message and requests a supported file. (b) System notifies the students that no printers are available.
Exceptions	**1.1 E1. Printer error** System displays an error message and prompts the student to try again later.

2. Usecase View printing status

ID and Name	UC-2 View Printing Status
Date Created	29th September, 2024
Primary Actor	Student
Secondary Actor	Print Management System
Description	The student checks the status of their print job (e.g., in queue. printing, completed).
Trigger	The student requests to view the print job status
Preconditions	**PRE-1.** The student is authenticated. **PRE-2.** There is an active print job in the system.
Postconditions	**POST-1.** The student views the real-time status of the print job. **POST-2.** Necessary actions (e,g., cancel, wait) are considered.
Normal Flow	**2.0 View printing status** (a) The student selects "View Printing Status". (b) The system displays the current status of the print job. (c) The student reviews the information.
Alternative Flows	**2.1 No active job** (a) System informs the student that there are no active print jobs. (b) System notifies the student if a print job is done, there is nothing to view.
Exceptions	**2.0 E1. System error in retrieving** System displays an error message and asks the student to try again later.

3. Usecase Cancel printing session

ID and Name	UC-3 Cancel Printing Session
Date Created	29th September, 2024
Primary Actor	Student
Secondary Actor	Print Management System
Description	The student cancels an active print session that they previously initiated.
Trigger	The students requests to cancel a printing job.
Preconditions	**PRE-1.** The student is authenticated. **PRE-2.** The print job is currently in progress.
Postconditions	**POST-1.** The print job is successfully canceled. **POST-2.** The print queue is updated to reflect the cancellation.
Normal Flow	**3.0. Cancel Printing Session** (a) The student selects "Cancel Printing Session". (b) Student selects the job to cancel. (c) The system confirms the request. (d) The system cancels the print job.
Alternative Flows	**3.1. Print job too far along** System notifies the student "Cannot proceed cancellation.The print job is already completed or in the final stages."
Exceptions	**3.1E1.** System error on canceling the print job. **3.1E2.** System cannot interfere the process and asks the student to try again later.

4. **Usecase Upload file**

ID and Name	UC-4 Upload File
Date Created	29th September, 2024
Primary Actor	Student
Secondary Actor	SPSO File Storage System
Description	The student uploads a document to the system to be printed. The system accepts the file and processes it for preview
Trigger	The student selects the option to upload a file for printing
Preconditions	**PRE-1.** The student is authenticated. **PRE-2.** The file is in an acceptable format.
Postconditions	**POST-1.** The file is successfully uploaded. **POST-2.** The file is available for preview and formatting.
Normal Flow	**4.0 Upload file to the system** (a) The student selects "Upload File". (b) The system prompts the student to select a file. (c) The student uploads the file. (d) The system confirms successful upload and prepares a preview.
Alternative Flows	**4.1. Invalid file or exceeds the allowed size** System requires the student to reupload the correct file type or smaller file size.
Exceptions	**4.1 E1.** System error during file upload. Prompt to ask the student to try again later.

5. **Usecase Format and View file preview**

ID and Name	UC-5 Format and view file preview
Date Created	29th September, 2024
Primary Actor	Student
Secondary Actor	Print Management System
Description	The student formats the document (e.g., adjusts margins, paper size, font size) and previews the file before printing.
Trigger	The student selects a document for formatting and initiates the preview process before printing.
Preconditions	**PRE-1.** The student is authenticated. **PRE-2.** The student must have uploaded a valid document. **PRE-3.** The system must support the document format for preview (e.g., PDF, DOCX).
Postconditions	**POST-1.** The student views the formatted document and approves or cancels the print job. **POST-2.** The system saves any formatting changes made by the student. **POST-3.** If the job is approved, the document is prepared for printing.
Normal Flow	**5.0 Format and view file preview** (a) Student selects and previews a document. (b) Adjusts formatting (margins, paper size, etc.). (c) Confirms changes and submits for printing. (d) System prepares the document for print.
Alternative Flows	**AF01 - Unsupported File Format:** System shows error that student uploads a supported file. **AF02 - Formatting Errors:** System shows formatting error; student adjusts and continues.
Exceptions	**E01 - System Failure During Preview:** Process halted, student notified, retry or report issue. **E02 - Invalid Document Content:** System blocks action, prompts for valid document upload.

6. **Usecase Choose printer**

ID and Name	UC-6 Choose Printer
Date Created	29th September, 2024
Primary Actor	Student
Secondary Actor	Print Management System
Description	The student selects a printer from a list of available printers on campus before proceeding with the print job.
Trigger	The student initiates the printing process and must choose a printer.
Preconditions	**PRE-1.** Student is authenticated. **PRE-2.** At least one printer is available and online. **PRE-3.** The system provides the location and status of each printer. **PRE-4.** Students has uploaded the document.
Postconditions	**POST-1.** The selected printer is assigned to the print job. **POST-2.** The system sends the document to the chosen printer for processing.
Normal Flow	**6.0 Choose printer** (a) The system displays a list of available printers with location and status. (b) The student chooses the nearest or preferred printer. (c) The system confirms the printer selection and prepares the job.
Alternative Flows	**AF01 - No Printer Available:** System notifies the student of no available printers; they can retry later or cancel. **AF02 - Printer Busy/Offline:** System alerts the student to choose another printer if the selected one is unavailable.
Exceptions	**E01 - System Error During Printer Selection:** Printer selection process is interrupted; the student is notified to retry or report the issue.

7. Usecase View printing log

ID and Name	UC-7 View Printing Log
Date Created	29th September, 2024
Primary Actor	Student
Secondary Actor	Printer Devices, Log Systems
Description	The student views their printing log, including the detail of printing history in summary or over a selected period of time.
Trigger	The student choose the option “Printing Log”.
Preconditions	**PRE-1.** Student must be logged into the system. **PRE-2.** The system must be online.
Postconditions	**POST-1.** The systems display the student’s printing history.
Normal Flow	**7.0 View Printing log summary** (a) Student selects the option to view the printing log. (b) Student chooses the view summary option. (c) System shows the list of print history, including date, file name and printing status of each file.
Alternative Flows	**7.1 View Printing log over selected range time** (a) Student selects the option to view the printing log. (b) Student chooses the “Detailed history” option. (c) System prompts the student to select the date range and optionally choose a specific printer. (d) Student enters his/her desired input. (e) System displays the detailed print history over the selected date such as: file name, page printed, status and page size,.... **7.2 Student view log with no printing history** (a) Student selects the option to view the printing log. (b) System displays a message indicating no log available.
Exceptions	**7.1.E1 System error in retrieving data** The system shows a message indicating error.

3 Task 2: Requirement eliciation (2.1, 2.2, 2.3, 2.4)

3.1 Activity Diagram (2.1)

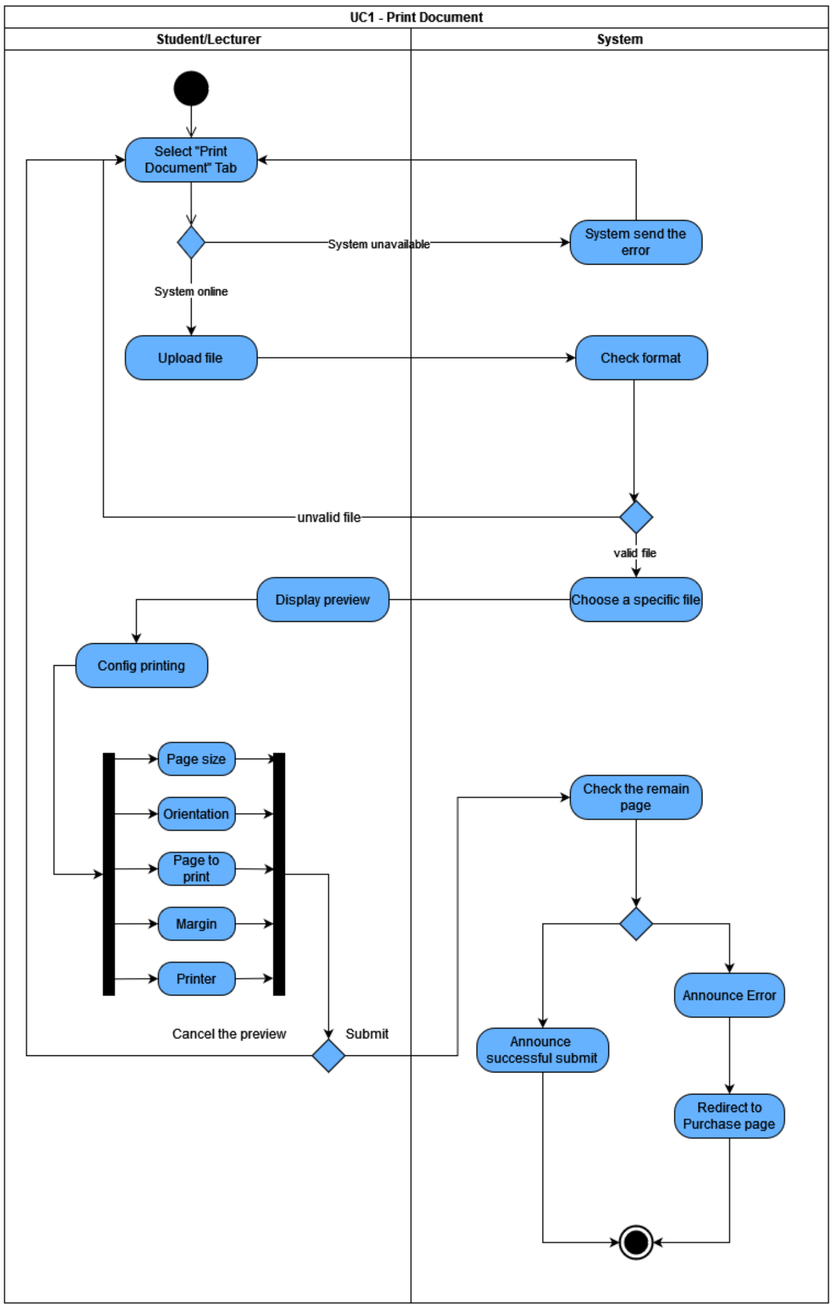


Figure 3: *Activity Diagram for Print Document Session*

Description: The Print Document use case for SSPS begins when a student or lecturer logs into the system, seeking to print a document from one of the university's available printers. The user initiates the process by navigating to the "Print Document" tab on the SSPS interface, which serves as the starting point of the use case. At this stage, the system performs a critical check to determine whether it is online and operational. If the system is unavailable due to technical reasons or maintenance, an error message is promptly displayed to inform the user, who must then wait until the service is restored to proceed with their printing task. However, if the system is online, the user is allowed to continue by uploading

the document file that they wish to print.
Once the file is uploaded, the system automatically checks its format to ensure it meets the acceptable file types specified by the system's administrator. In cases where the file format is unsupported (for example, a file type not permitted for printing), the system will return an error, prompting the user to correct the issue by uploading a valid file. If the file format is valid, the system moves forward by displaying a preview of the uploaded document, giving the user a chance to review the document before proceeding with the actual printing request. This preview helps the user ensure that the correct file has been selected and provides an opportunity to make any adjustments if necessary.
Following the document preview, the user must configure the printing settings, which include multiple parameters such as the paper size (e.g., A4, A3), the orientation of the pages (portrait or landscape), the range of pages they want to print (specific pages or the entire document), the margins, and the choice of printer from the list of available devices on campus. These settings allow for customized printing based on the user's specific needs for each job.
After configuring the printing preferences, the user submits the print job. The system then checks if the user has enough remaining pages in their account to complete the request. Each user has a set page quota per semester, which the system tracks. If the page balance is sufficient, the print job is processed, and a confirmation message is displayed. However, if there are not enough pages, the system halts the process and redirects the user to purchase more pages.
In either case, the system ensures that the user is always kept informed of the process. If successful, the user receives confirmation that their document is being printed, and they can check the status of the print job in the system interface. If there are issues, such as insufficient page balance or an invalid file, the system provides clear feedback, allowing the user to take the necessary steps to resolve the issue, whether by uploading a different file or purchasing more printing credits. This use case ensures a seamless and user-friendly experience for students and lecturers who need to print documents on campus, offering transparency, flexibility, and control over the entire printing process.

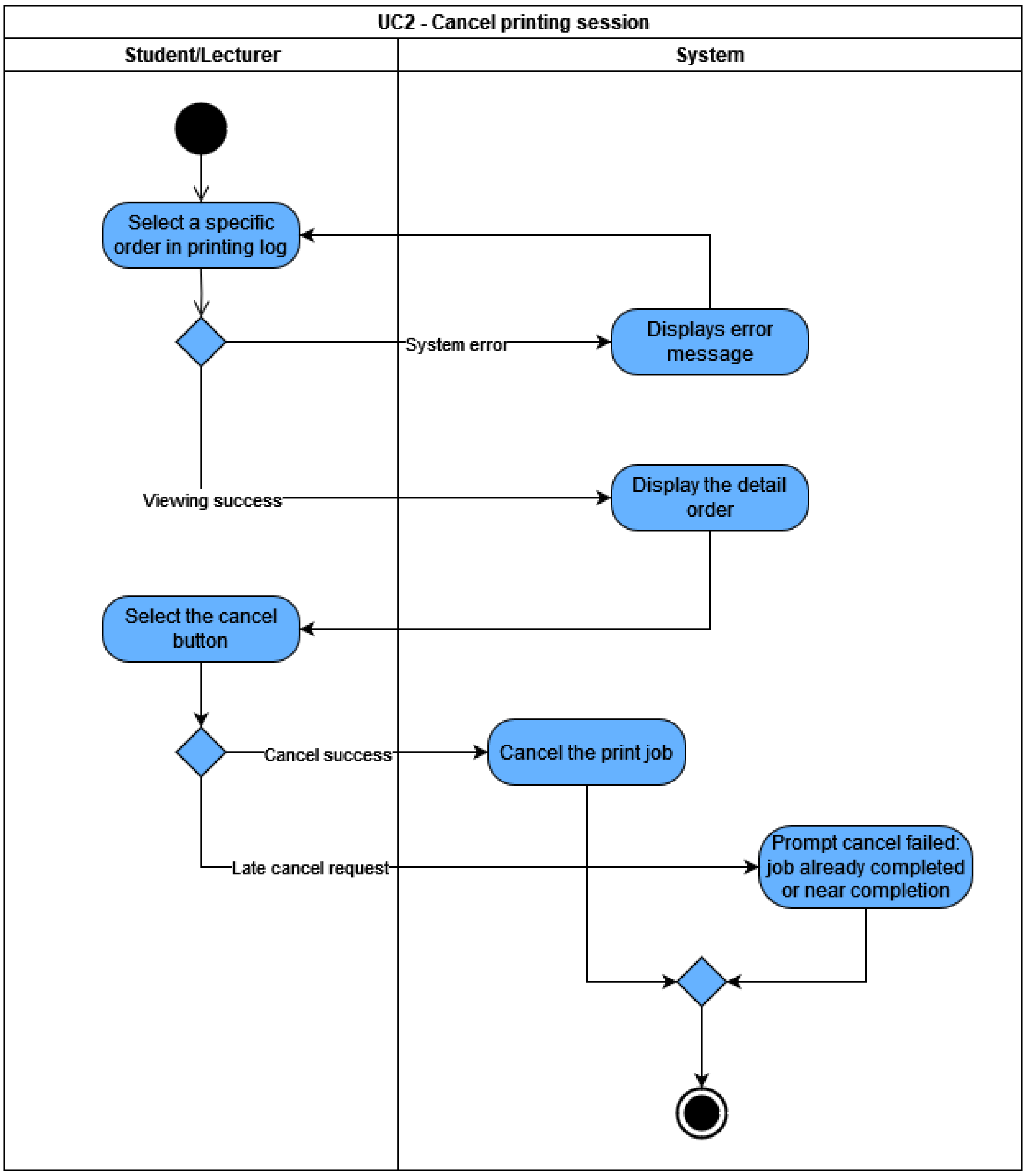


Figure 4: *Activity Diagram for Cancel Printing Session*

Description: The Cancel Printing Session use case allows a student or lecturer to cancel a print job they have previously submitted through the Student Smart Printing Service (SSPS). The process begins with the user selecting a specific print order from their printing log. The system then attempts to display the details of the chosen order. If a system error occurs during this process, an error message is shown to the user. If successful, the system displays the detailed information of the selected print job. The user can then proceed by clicking the "Cancel" button to request the cancellation of the print job. If the cancellation request is submitted in time, the system successfully cancels the print job and confirms this action to the user. However, if the print job is already completed or too close to completion, the system notifies the user that the cancellation has failed due to the late timing of the request. This use case ensures that users can manage their print jobs efficiently while also handling cancellation requests based on the job's status.

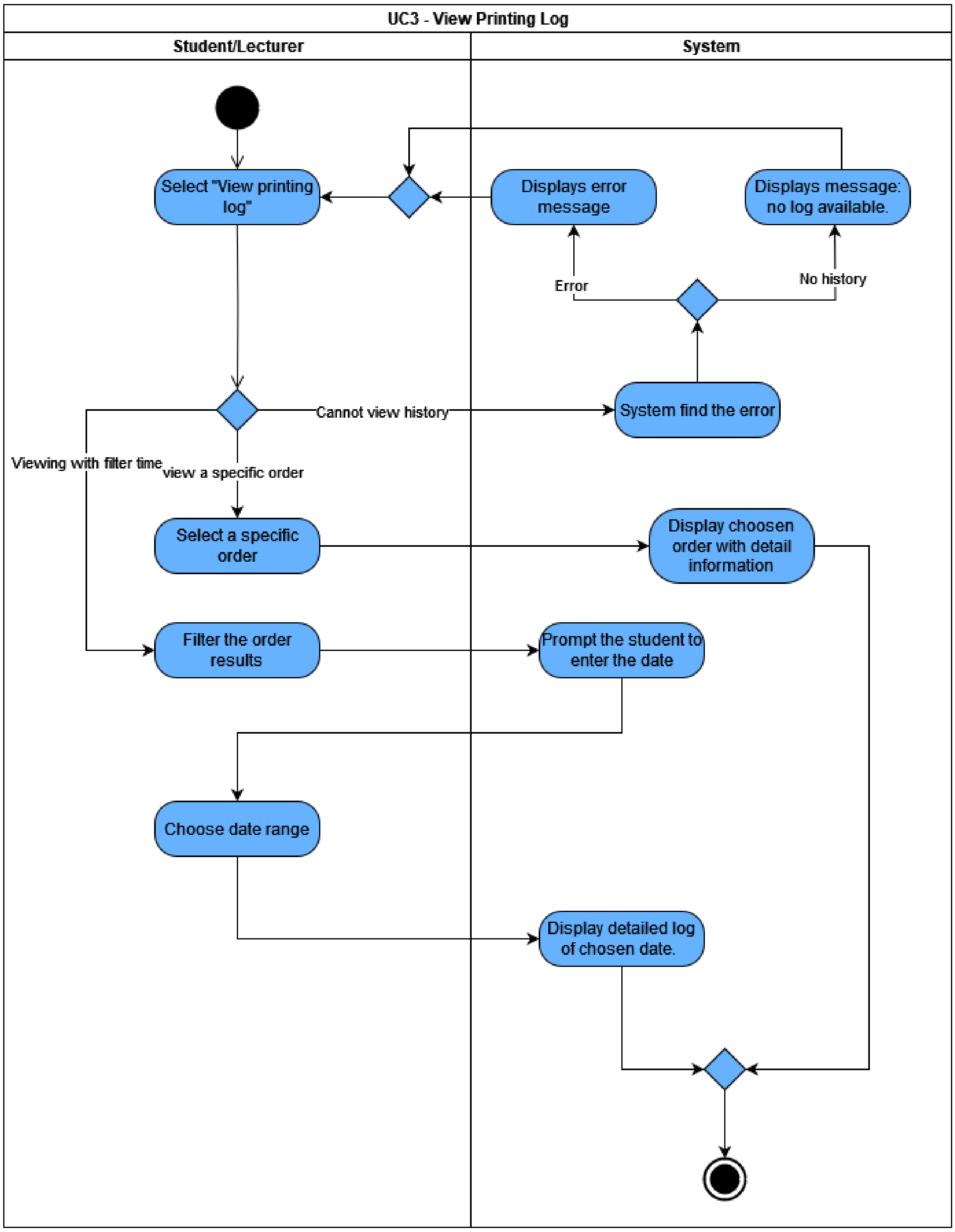


Figure 5: *Activity Diagram for View Printing Log Session*

Description: The View Printing Log use case provides students and lecturers with the ability to review their past printing activity within the Student Smart Printing Service (SSPS). The process begins when the user selects the "View Printing Log" option from the system's interface. At this point, the system

attempts to retrieve the user's printing history. If no logs are available. For example, if the user has not printed anything yet, or if there is a system error during the retrieval process, an error message is displayed, informing the user that the logs cannot be accessed at the moment.
However, if the user has printing history available, the system presents options for filtering the records based on specific criteria. The user can choose to either view the details of a specific print order or filter the history by specifying a date range. When selecting a specific order, the system will display detailed information about that print job, including key details such as the printer used, the number of pages printed, and the date and time of the print session. Alternatively, if the user opts to filter by date, the system prompts the user to enter a date range. Once the user specifies the range, the system displays a detailed log of all print jobs that occurred within the selected dates. This feature helps users track their printing activity, making it easier to review past prints, and resolve any discrepancies in their printing records.

3.2 Sequence Diagram (2.2)

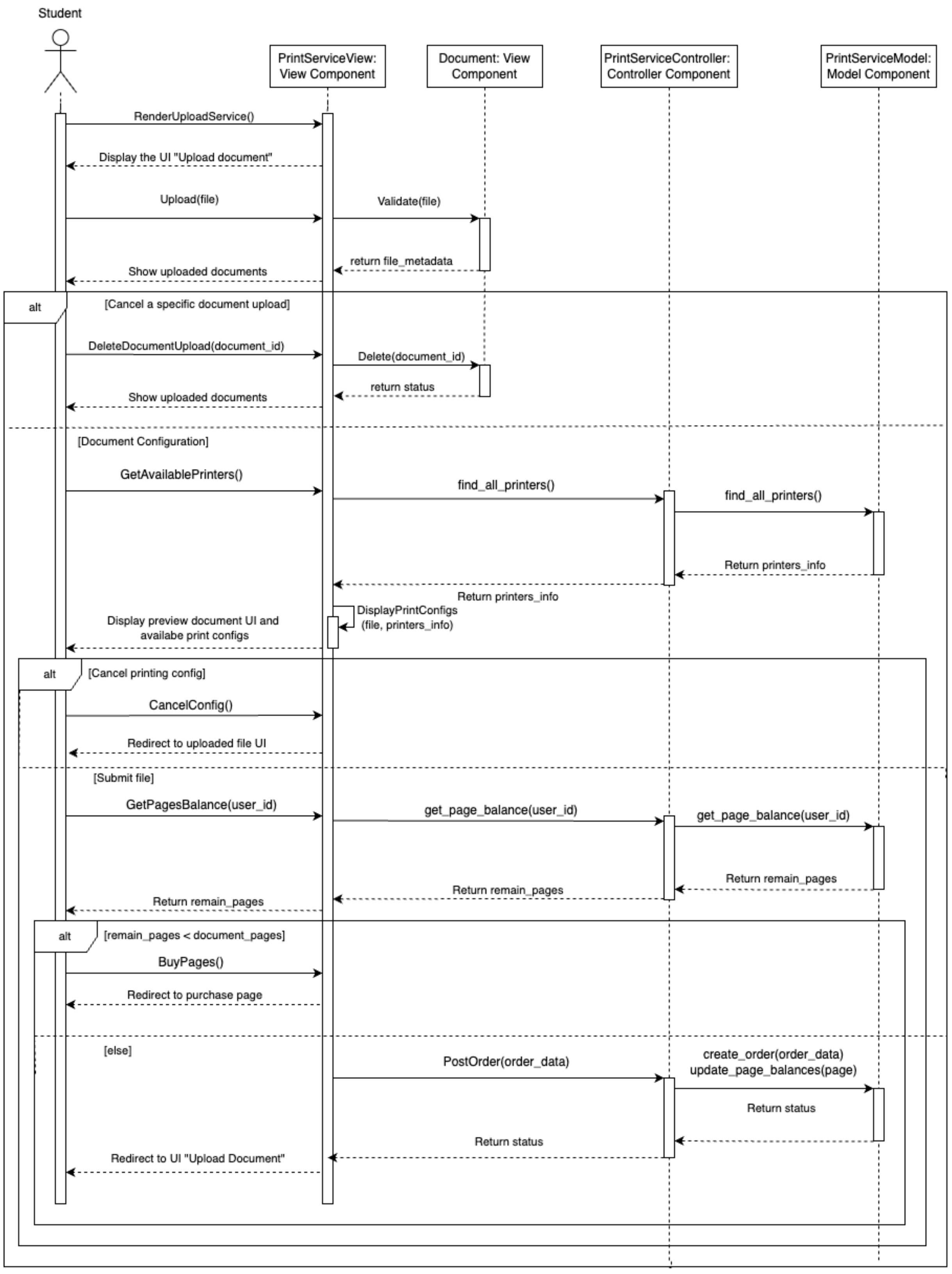


Figure 6: *Sequence Diagram for Print Document Session*

Description: When the student begins to use the print document service by clicking the tab "Print Document" bar, user will call the method UploadService() of PrintServiceComponent to display the UI component for uploading the document. In the Upload Interface, the student can choose to drag and drop the document or click the "Upload from computer" button to upload the document by calling the function Upload(file) of the PrintServiceView. When uploading the file, the PrintServiceView component will be validated by Validate(file) of Document, then return the file metadata to the PrintServiceView if the file is valid, or else a notification will be pop up, saying that the file is not valid. After receiving metadata from the Document, PrintServiceView will display the uploaded files on the screen.
After uploading the file, the user can have the option to delete a specific uploaded file by clicking the delete button on a specific file, which will call the method DeleteDocumentUpload(document_id) to delete that file and show the documents again.
For printing a specific file, the user will choose the print button in a specific file. After clicking, the method GetAvailablePrinters() of PrintServiceView will be called, the PrintServiceController will call the method $find\_all\_printers()$ to the PrinterModel, and the PrinterModel will call the method $find\_all\_printers()$ to the database to retrieve all the information of active printers. These information will be passed from the Database, through the Model and Controller layers to the PrintServiceView. The PrintServiceView will use these printer information details and the chosen file info file to render the configuration interface through the DisplayPrintConfigs(file ,info) and the preview session for the uploaded document. The user will choose his/her desired printing configuration for the document.
If the user wants to cancel this configuration session, he/she will choose the cancel button at the bottom, which will call CancelPrintConfig() of PrintServiceView to redirect the user to the uploaded file interface.
After configuring the printing option, the user can click on the submit button, then the PrintServiceView will call the method GetPagesBalance(user_id) to the Controller. The remaining pages of the user with user_id will come back to the PrintServiceView. If the remaining page number of the user is less than the document pages, PrintServiceView will create a pop-up notification through the method BuyPages() to redirect the user to the purchase interface. Otherwise, the PrintServiceView will call the function PostOrder($order\_data$) of the Controller, the Controller will create the order and update the page balance through the create_order($order\_data$) and $update\_page\_balance(page)$ of the model. After updating successfully, the user will receive a pop-up notification that the order has been successful, then redirect the user to the Upload Document User Interface.

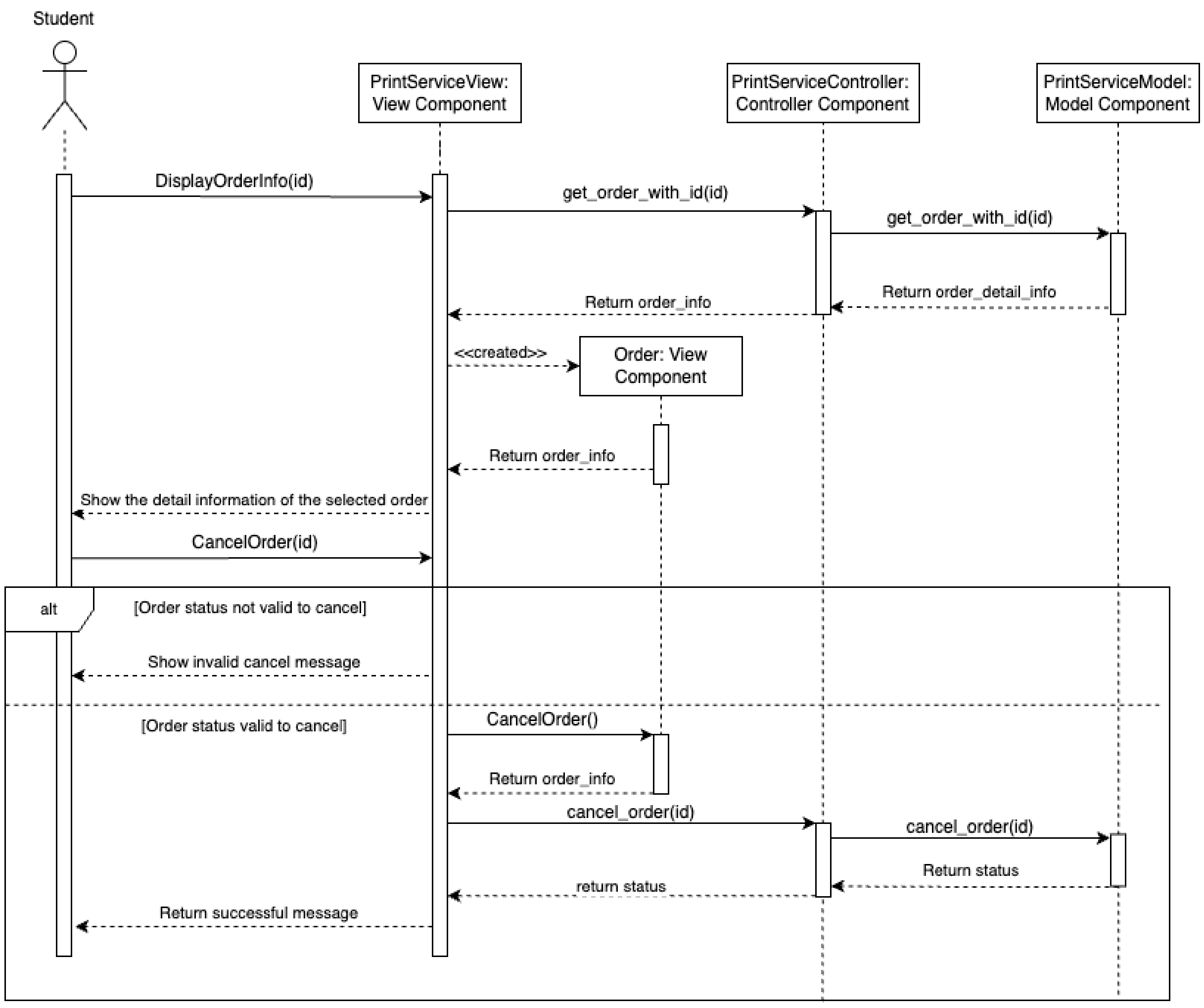

Student
PrintServiceView: View Component
PrintServiceController: Controller Component
PrintServiceModel: Model Component
DisplayOrderInfo(id)
get_order_with_id(id)
get_order_with_id(id)
Return order_detail_info
Return order_info
<<created>>
Order: View Component
Return order_info
Show the detail information of the selected order
CancelOrder(id)
alt
[Order status not valid to cancel]
Show invalid cancel message
[Order status valid to cancel]
CancelOrder()
Return order_info
cancel_order(id)
cancel_order(id)
Return status
return status
Return successful message

Figure 7: *Sequence Diagram for Cancel Printing Session*

Description: In case the user wants to cancel a specific printing order, he/she will call the function DisplayOrderInfo(id) of PrintServiceView, then PrintServiceView will call the method get_order_with_id(id) to Controller to get the order_info. In this process, Order object will be created, then return the detail information to the PrintServiceView, then the PrintServiceView will display the detail information of that Order as a result of method DisplayOrderInfo(id). If the status is invalid for canceling, the PrintServiceView will make a notification that the order is unable to cancel. If the status is valid for canceling, method CancelOrder(id) of PrintServiceView will be called to Controller, then pass through layers to the database for changing the status of that order to cancel. After that, the status of the action will be passed through the layer to the PrintServiceView, then a notification will pop up to show that the canceling action is successful. The user will be directed to the view printing log interface.

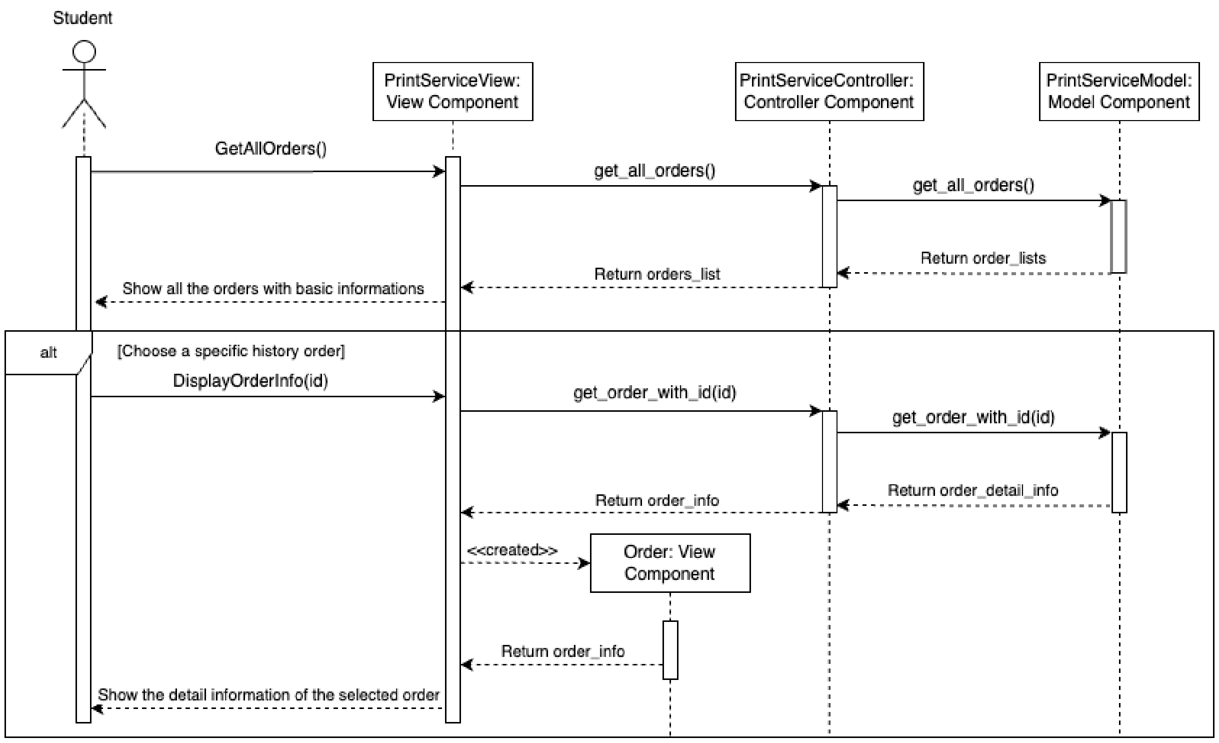


Figure 8: *Sequence Diagram for View Printing Log Session*

Description: In case the user wants to view their printing log, he/she can choose the tab "Printing Log" on the navigation bar, which will call the method GetAllOrders() of PrintServiceView. PrintServiceView will call the method get_all_orders() of Controller. The orders_data from the Database will be passed through the layers, going to the PrintServiceView. PrintServiceView will use this data to show all the orders with basic information as a result of method GetAllOrders(). In case the user wants to cancel a specific printing order, he/she will call the function DisplayOrderInfo(id) of PrintServiceView, then PrintServiceView will call the method get_order_with_id(id) to Controller to get the order_info. In this process, Order object will be created, then return the detail information to the PrintServiceView, then the PrintServiceView will display the detail information of that Order as a result of method DisplayOrderInfo(id).

3.3 Class Diagram (2.3)

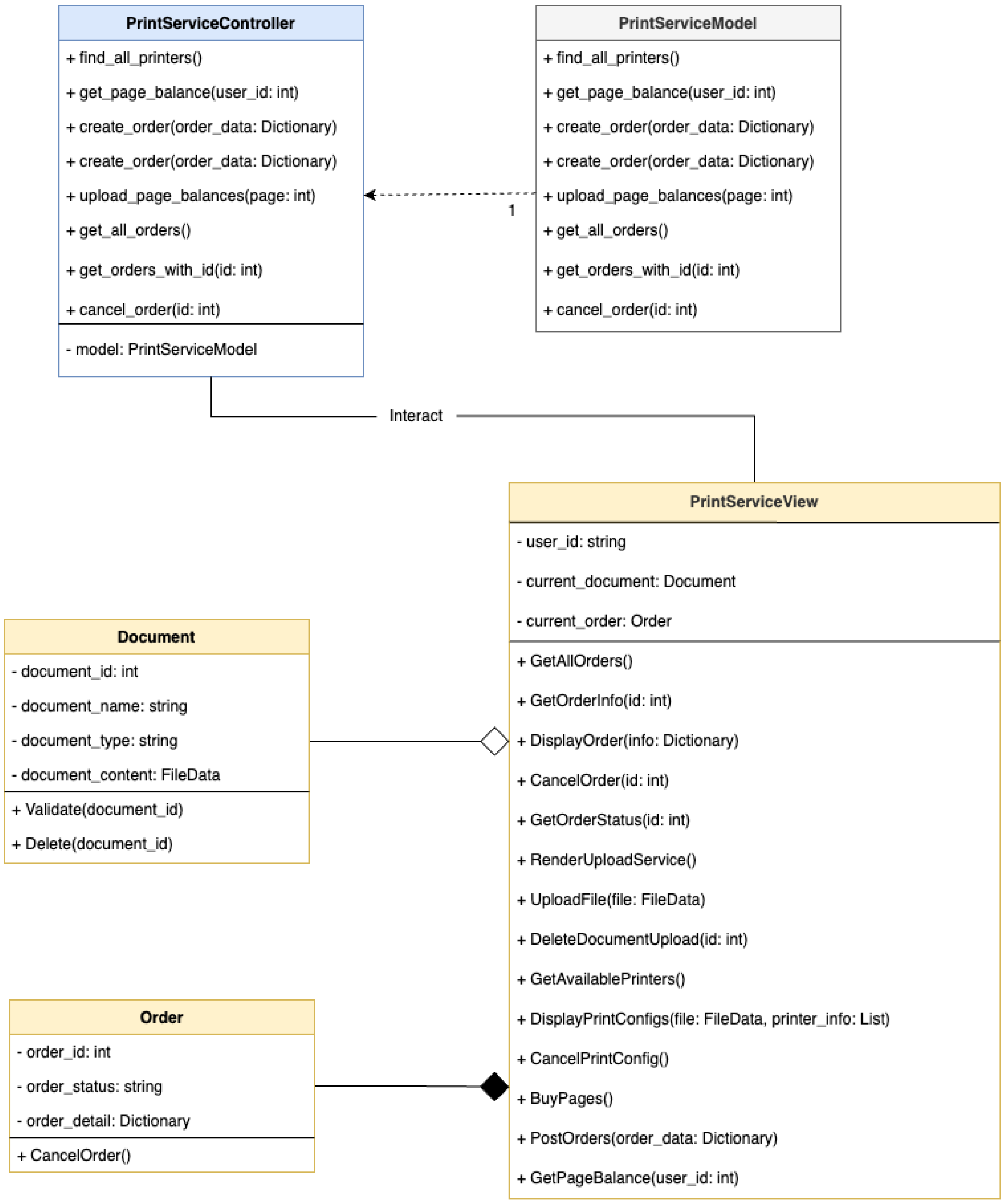
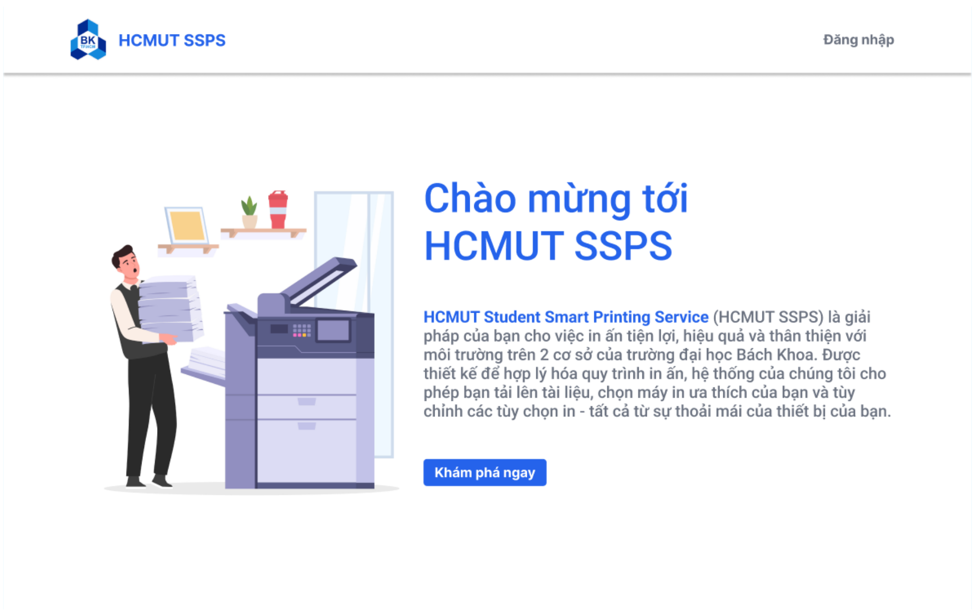

PrintServiceController
+ find_all_printers()
+ get_page_balance(user_id: int)
+ create_order(order_data: Dictionary)
+ create_order(order_data: Dictionary)
+ upload_page_balances(page: int)
+ get_all_orders()
+ get_orders_with_id(id: int)
+ cancel_order(id: int)
- model: PrintServiceModel
PrintServiceModel
+ find_all_printers()
+ get_page_balance(user_id: int)
+ create_order(order_data: Dictionary)
+ create_order(order_data: Dictionary)
+ upload_page_balances(page: int)
+ get_all_orders()
+ get_orders_with_id(id: int)
+ cancel_order(id: int)
1
Interact
PrintServiceView
- user_id: string
- current_document: Document
- current_order: Order
+ GetAllOrders()
+ GetOrderInfo(id: int)
+ DisplayOrder(info: Dictionary)
+ CancelOrder(id: int)
+ GetOrderStatus(id: int)
+ RenderUploadService()
+ UploadFile(file: FileData)
+ DeleteDocumentUpload(id: int)
+ GetAvailablePrinters()
+ DisplayPrintConfigs(file: FileData, printer_info: List)
+ CancelPrintConfig()
+ BuyPages()
+ PostOrders(order_data: Dictionary)
+ GetPageBalance(user_id: int)
Document
- document_id: int
- document_name: string
- document_type: string
- document_content: FileData
+ Validate(document_id)
+ Delete(document_id)
Order
- order_id: int
- order_status: string
- order_detail: Dictionary
+ CancelOrder()

Figure 9: *Class Diagram*

3.4 User Interface (2.4)

3.4.1 Pre log in Homepage



Chào mừng tới HCMUT SSPS

HCMUT Student Smart Printing Service (HCMUT SSPS) là giải pháp của bạn cho việc in ấn tiện lợi, hiệu quả và thân thiện với môi trường trên 2 cơ sở của trường đại học Bách Khoa. Được thiết kế để hợp lý hóa quy trình in ấn, hệ thống của chúng tôi cho phép bạn tải lên tài liệu, chọn máy in ưa thích của bạn và tùy chỉnh các tùy chọn in - tất cả từ sự thoải mái của thiết bị của bạn.

Figure 10: *Homepage Interface*

Description: The Pre Log in Homepage contains a short introduction about the system, with a navigation link and a button which will direct user to the login page.

3.4.2 Log in

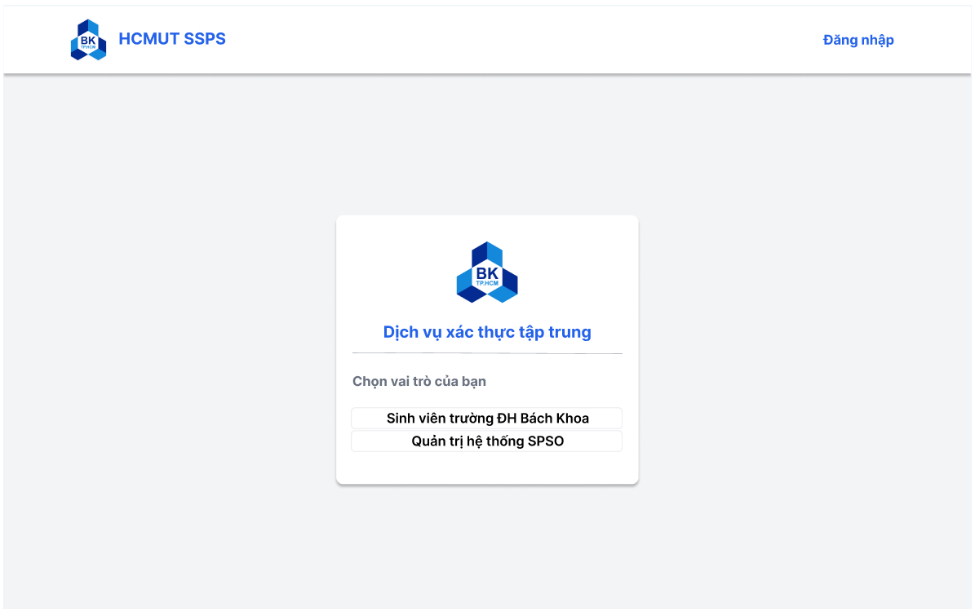


Figure 11: *Select roles*

Description: The Authenticative Service Page will contains two options, refering to two roles that the user can be login: Student of HCMUT or SPSO. User will choose their login role by clicking one option.

3.4.3 Log in as student

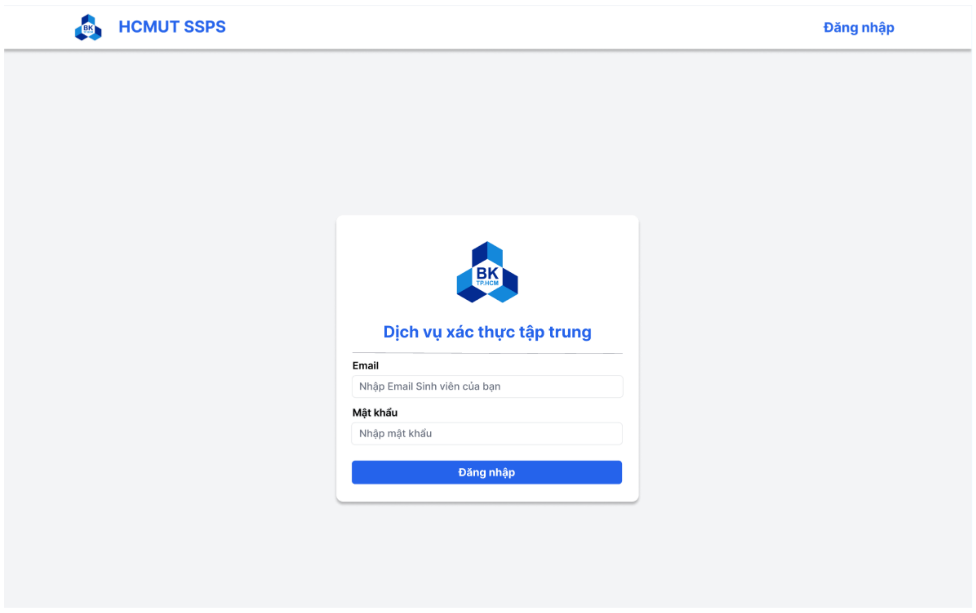

HCMUT SSPS
Đăng nhập
Dịch vụ xác thực tập trung
Email
Nhập Email Sinh viên của bạn
Mật khẩu
Nhập mật khẩu
Đăng nhập

Figure 12: *Student account*

Description: For student, the system requires HCMUT email and password for authentication.

3.4.4 Log in as Administrator

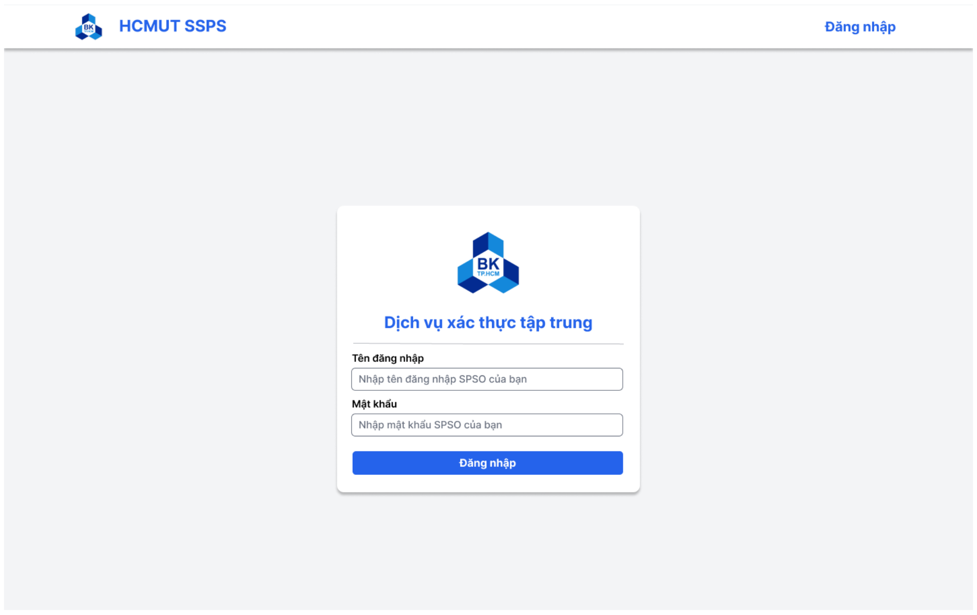
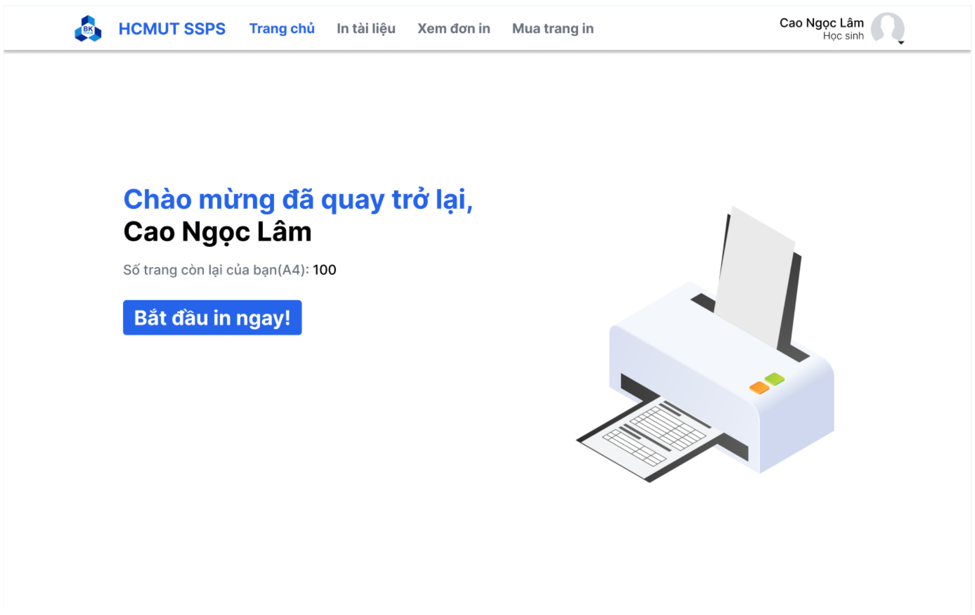

HCMUT SSPS
Đăng nhập
Dịch vụ xác thực tập trung
Tên đăng nhập
Nhập tên đăng nhập SPSO của bạn
Mật khẩu
Nhập mật khẩu SPSO của bạn
Đăng nhập

Figure 13: *Admin account*

Description: For SPSO, the system requires SPSO username and password for authentication.

3.4.5 Homepage



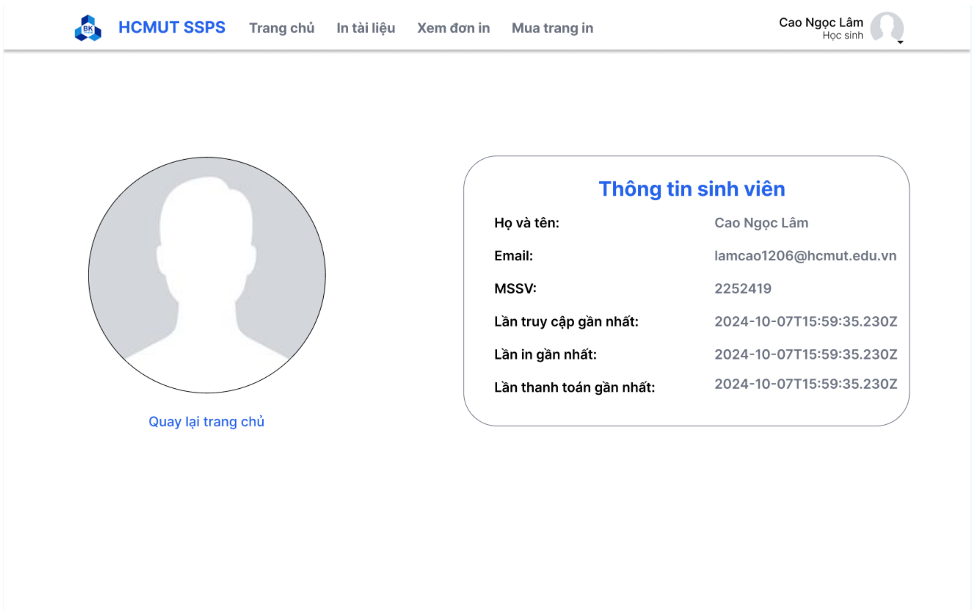
Chào mừng đã quay trở lại,
Cao Ngọc Lâm

Số trang còn lại của bạn(A4): 100

Figure 14: *Post log in homepage*

Description: When student login successfully, the system will render the avatar of student in the top right of the navigation, also rendering additional navigation tabs like Dashboard, Print Document, Printing Log and Purchase. When click the avatar, a drop down box will pop-up, showing 2 options for the student: "View profile", which will redirect them to the Profile Page, or "Log Out".

3.4.6 Profile



Thông tin sinh viên

Họ và tên:	Cao Ngọc Lâm
Email:	lamcao1206@hcmut.edu.vn
MSSV:	2252419
Lần truy cập gần nhất:	2024-10-07T15:59:35.230Z
Lần in gần nhất:	2024-10-07T15:59:35.230Z
Lần thanh toán gần nhất:	2024-10-07T15:59:35.230Z

Figure 15: *Information Summary*

Description: This page contain basic information of student. It also have the link in the bottom of the avatar to redirect the student to dashboard page.

3.4.7 Document Selection/Upload

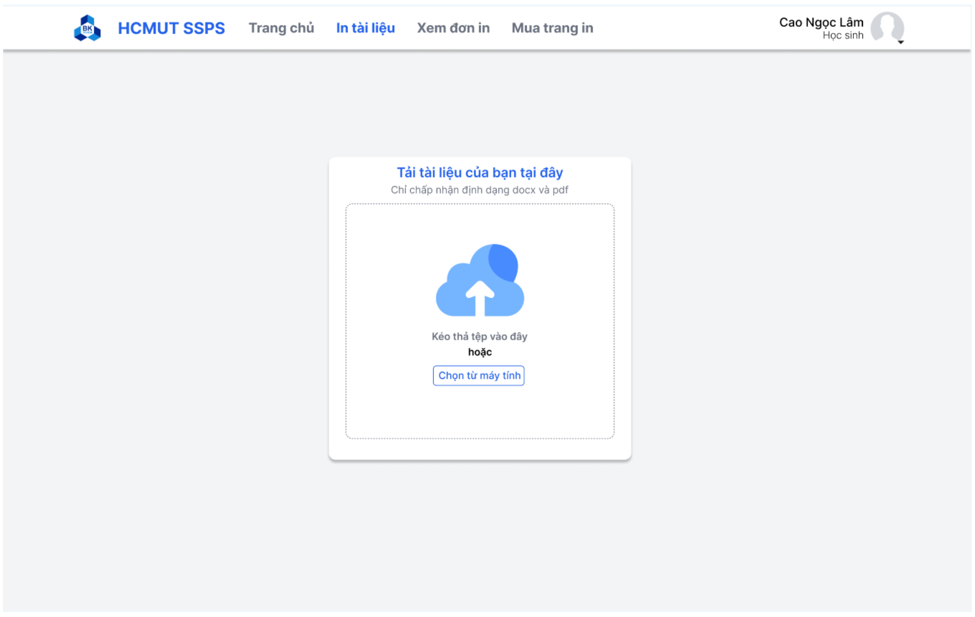


Figure 16: *Select desire document to print*

Description: When the student click on the "Print Document" tab, system will render the interface for upload document. Student can drag the document or choose desired document from the computer.

3.4.8 Document Selected/Uploaded

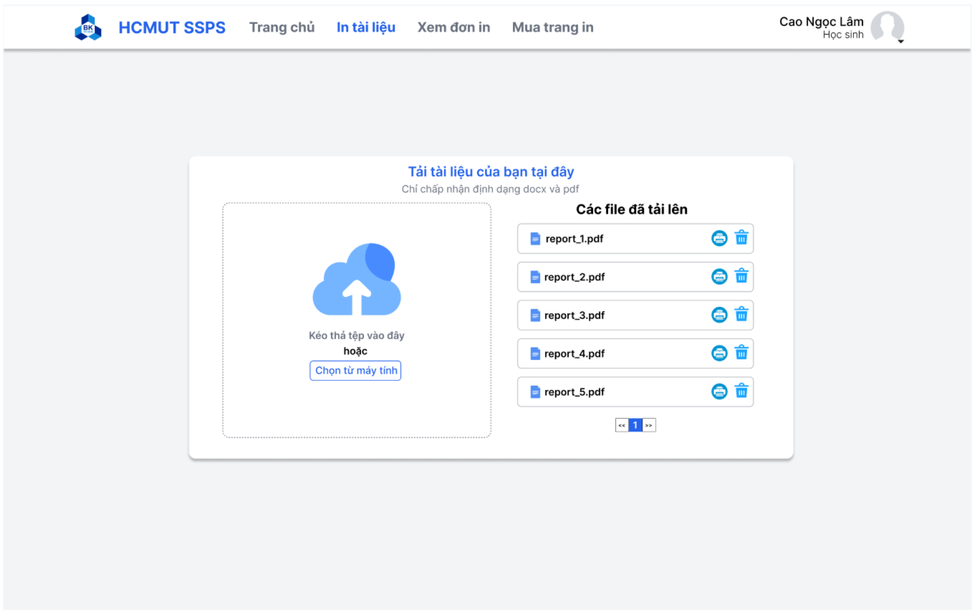


Figure 17: *Showing all the uploaded documents*

Description: After the students choose their desired document, the system will check the format of the document, then render the status of the document if the document is uploaded successfully, They can delete the uploaded document by click into the garbage icon.

3.4.9 Document Configuration

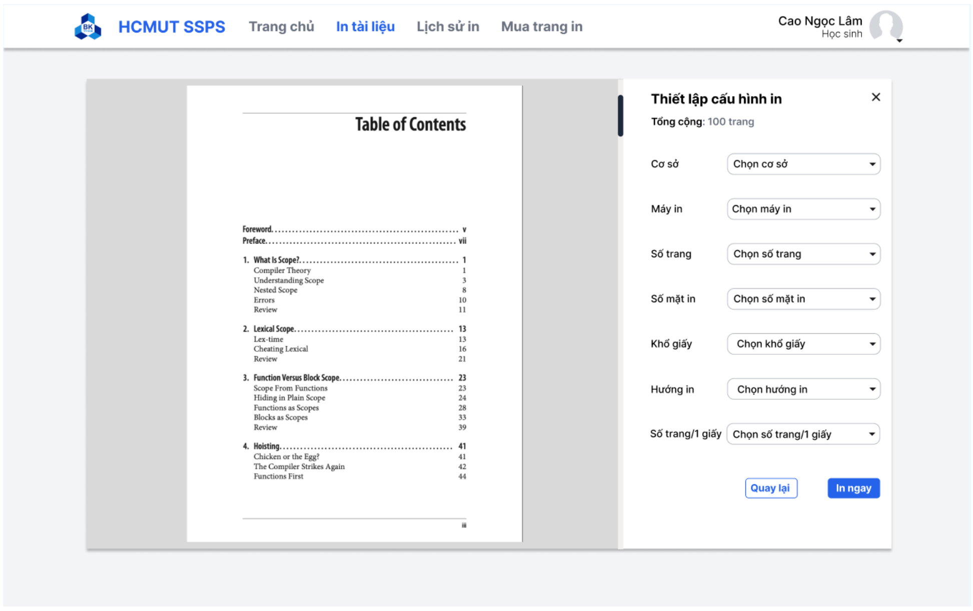


Table of Contents

iii

Thiết lập cấu hình in

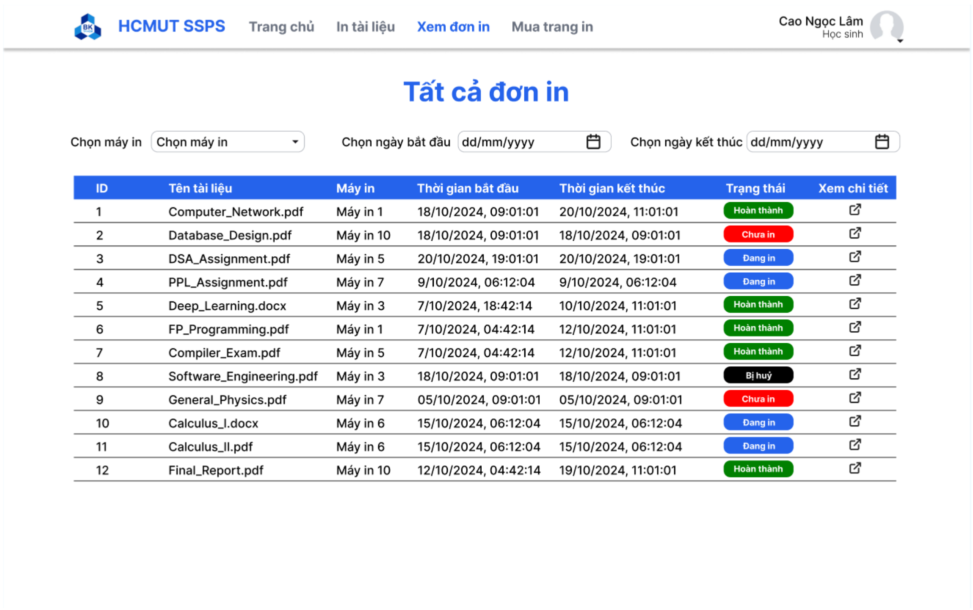
Tổng cộng: 100 trang

Cơ sở	Chọn cơ sở
Máy in	Chọn máy in
Số trang	Chọn số trang
Số mặt in	Chọn số mặt in
Khổ giấy	Chọn khổ giấy
Hướng in	Chọn hướng in
Số trang/1 giấy	Chọn số trang/1 giấy

Figure 18: *Different configurations to choose*

Description: When the students want to configure the printing option for an uploaded document, they can choose the print icon in a specific document. The system will render the configuration page for that document, contain the part for preview the document and another part for choosing desired configuration.

3.4.10 Printing Log



Tất cả đơn in

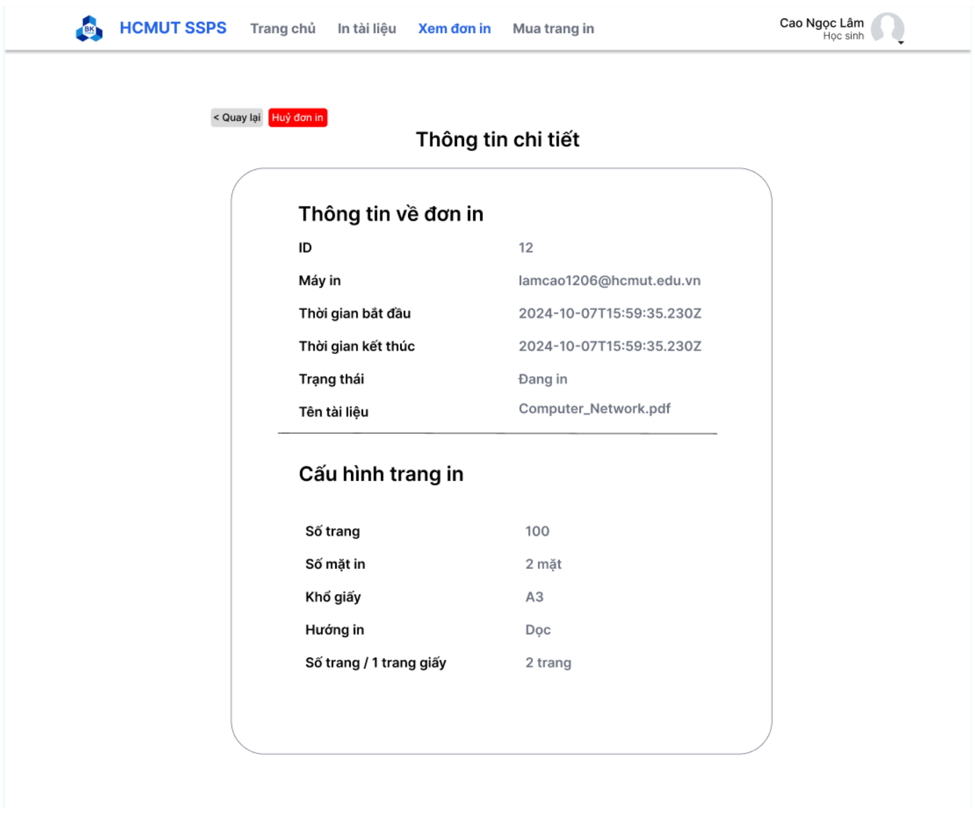
Chọn máy in: Chọn máy in
Chọn ngày bắt đầu: dd/mm/yyyy
Chọn ngày kết thúc: dd/mm/yyyy

ID	Tên tài liệu	Máy in	Thời gian bắt đầu	Thời gian kết thúc	Trạng thái	Xem chi tiết
1	Computer_Network.pdf	Máy in 1	18/10/2024, 09:01:01	20/10/2024, 11:01:01	Hoàn thành	↗
2	Database_Design.pdf	Máy in 10	18/10/2024, 09:01:01	18/10/2024, 09:01:01	Chưa in	↗
3	DSA_Assignment.pdf	Máy in 5	20/10/2024, 19:01:01	20/10/2024, 19:01:01	Đang in	↗
4	PPL_Assignment.pdf	Máy in 7	9/10/2024, 06:12:04	9/10/2024, 06:12:04	Đang in	↗
5	Deep_Learning.docx	Máy in 3	7/10/2024, 18:42:14	10/10/2024, 11:01:01	Hoàn thành	↗
6	FP_Programming.pdf	Máy in 1	7/10/2024, 04:42:14	12/10/2024, 11:01:01	Hoàn thành	↗
7	Compiler_Exam.pdf	Máy in 5	7/10/2024, 04:42:14	12/10/2024, 11:01:01	Hoàn thành	↗
8	Software_Engineering.pdf	Máy in 3	18/10/2024, 09:01:01	18/10/2024, 09:01:01	Bị huỷ	↗
9	General_Physics.pdf	Máy in 7	05/10/2024, 09:01:01	05/10/2024, 09:01:01	Chưa in	↗
10	Calculus_I.docx	Máy in 6	15/10/2024, 06:12:04	15/10/2024, 06:12:04	Đang in	↗
11	Calculus_II.pdf	Máy in 6	15/10/2024, 06:12:04	15/10/2024, 06:12:04	Đang in	↗
12	Final_Report.pdf	Máy in 10	12/10/2024, 04:42:14	19/10/2024, 11:01:01	Hoàn thành	↗

Figure 19: *Showing all the printing orders*

Description: If the students want to see their orders, they need to choose the tab "Printing Log" on the navigation bar. The system will render a table of all the orders and the option to filter the orders displayed on the screen. If the user want to see specific information of an order, they can choose the detail icon of a specific printing order.

3.4.11 Printing Order Detail



Thông tin chi tiết

Thông tin về đơn in

ID	12
Máy in	lamcao1206@hcmut.edu.vn
Thời gian bắt đầu	2024-10-07T15:59:35.230Z
Thời gian kết thúc	2024-10-07T15:59:35.230Z
Trạng thái	Đang in
Tên tài liệu	Computer_Network.pdf

Cấu hình trang in

Số trang	100
Số mặt in	2 mặt
Khổ giấy	A3
Hướng in	Dọc
Số trang / 1 trang giấy	2 trang

Figure 20: *Viewing printing order details including all specifications*

Description: This page will contain the necessary informations of an order, including the basic information of the order and the printing configurations. It also have two button on the top, the first one will navigate to the "printing log" page, and the order one will cancel the order.

3.5 Admin Interface

3.5.1 Dashboard

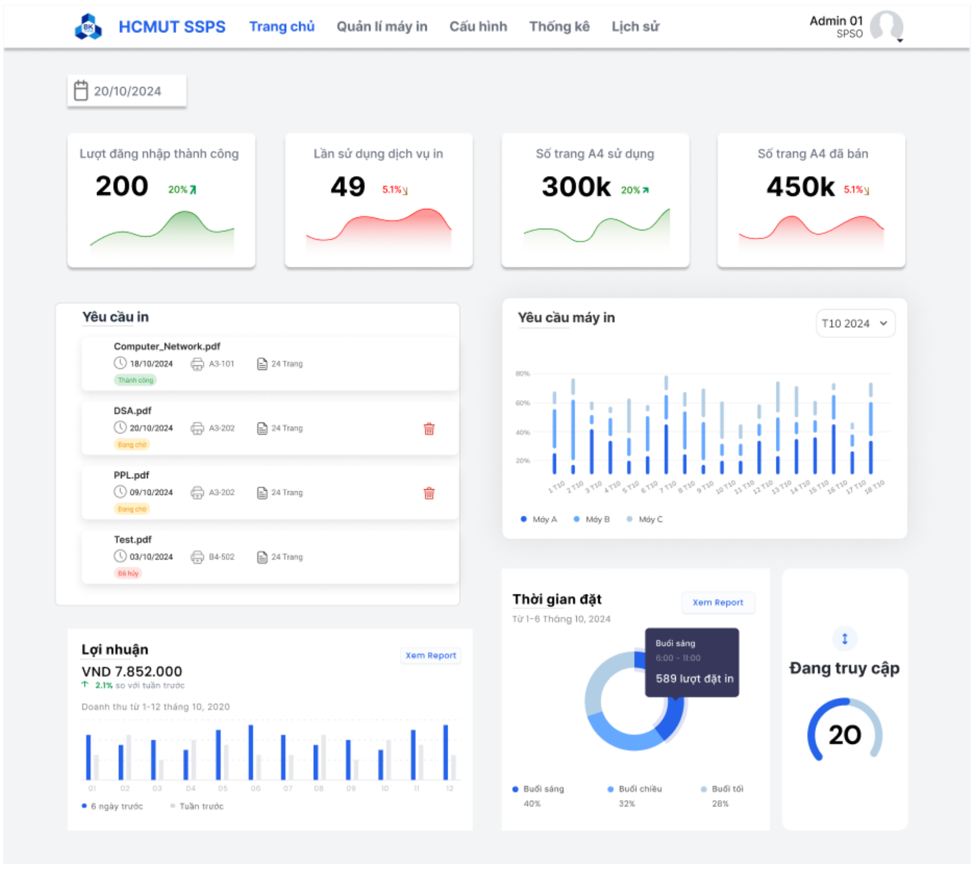
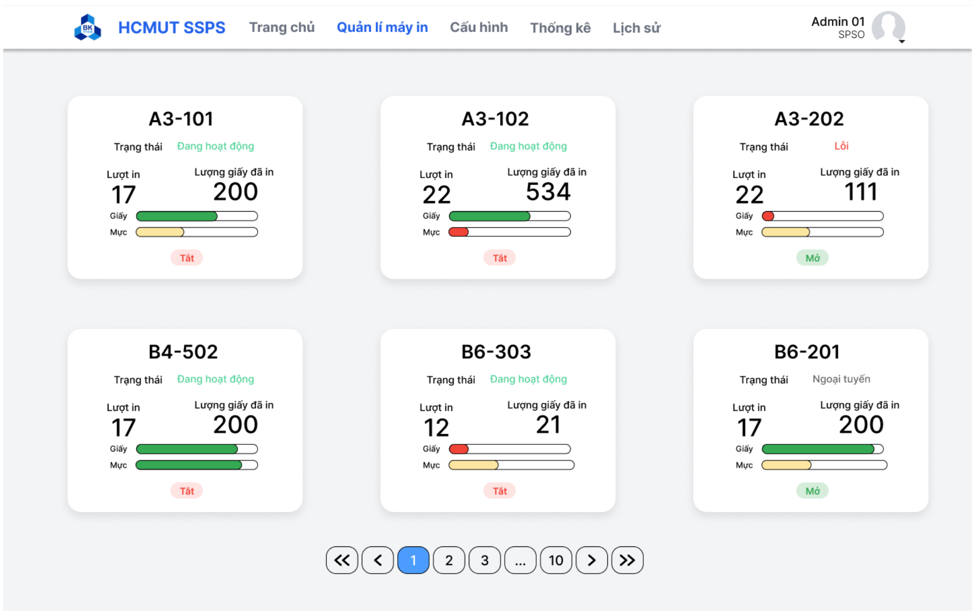


Figure 21: *Summarize all information of printers*

Description: After the SPSO login successfully, the system will render essential components for the SPSO dashboard such as print job volume, revenue, and active users, along with visual representations of key data points. The navigation for the SPSO also the same as students, except different tab contexts.

3.5.2 Printer Management



A3-101
Trạng thái Đang hoạt động
Lượt in 17
Lượng giấy đã in 200
Giấy
Mực
Tắt

A3-102
Trạng thái Đang hoạt động
Lượt in 22
Lượng giấy đã in 534
Giấy
Mực
Tắt

A3-202
Trạng thái Lỗi
Lượt in 22
Lượng giấy đã in 111
Giấy
Mực
Mở

B4-502
Trạng thái Đang hoạt động
Lượt in 17
Lượng giấy đã in 200
Giấy
Mực
Tắt

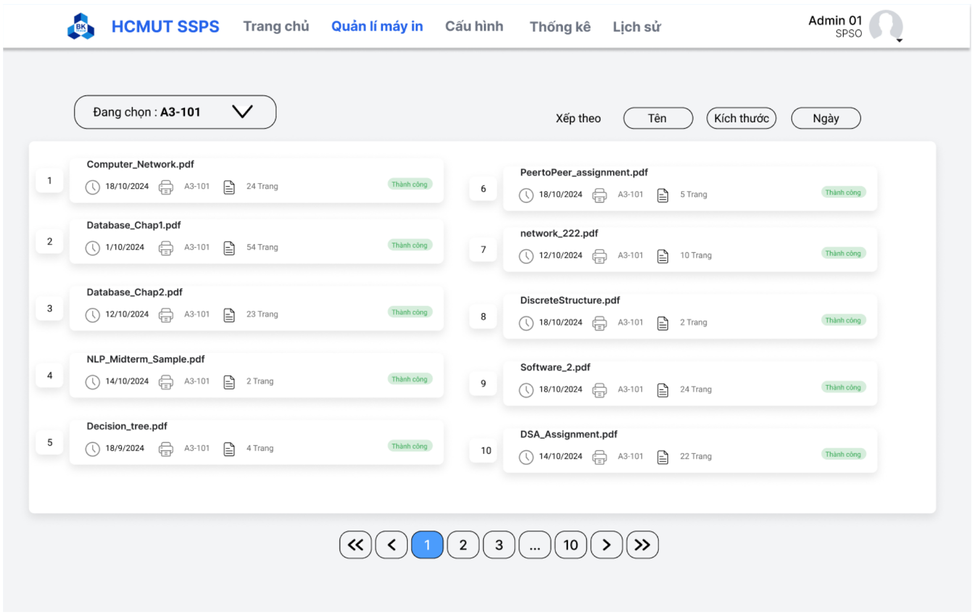
B6-303
Trạng thái Đang hoạt động
Lượt in 12
Lượng giấy đã in 21
Giấy
Mực
Tắt

B6-201
Trạng thái Ngoại tuyến
Lượt in 17
Lượng giấy đã in 200
Giấy
Mực
Mở

Figure 22: *Select printer to manage*

Description: When click on the tab "Printer management", the System will render the list of all printers, containing their basic information.

3.5.3 File management in printer



Đang chọn : **A3-101**

1	Computer_Network.pdf	18/10/2024	A3-101	24 Trang	Thành công
2	Database_Chap1.pdf	1/10/2024	A3-101	54 Trang	Thành công
3	Database_Chap2.pdf	12/10/2024	A3-101	23 Trang	Thành công
4	NLP_Midterm_Sample.pdf	14/10/2024	A3-101	2 Trang	Thành công
5	Decision_tree.pdf	18/9/2024	A3-101	4 Trang	Thành công
6	PeertoPeer_assignment.pdf	18/10/2024	A3-101	5 Trang	Thành công
7	network_222.pdf	12/10/2024	A3-101	10 Trang	Thành công
8	DiscreteStructure.pdf	18/10/2024	A3-101	2 Trang	Thành công
9	Software_2.pdf	18/10/2024	A3-101	24 Trang	Thành công
10	DSA_Assignment.pdf	14/10/2024	A3-101	22 Trang	Thành công

Figure 23: *Viewing history file of each printer*

Description: When click on a specific printer, the System will render all the documents has been uploaded to that specific printer, as well as their printing status.

4 Task 3: Requirement eliciation (3.1, 3.2)

4.1 Layered Architecture

The HCMUT SSPS will be implemented based on the layered architecture, with the combination of React in the Front-end and Django in the Back-end. However, in Django architecture, the framework follows the Model-View-Template (MVT) rather than the traditional pattern in layered architecture Model-View-Controller (MVC). Therefore, to align with the assignment requirements and simplify the diagrams, we will make some alias in the Django architecture to refer to its as MVC in our diagrams, since components between both architectures have the same functionalities:

- The **View** component in MVT of Django will be referred as **Controller** in MVC, responsible for interacting with the Component in Front-end.
- The **Model** component in MVT also has the same functionality in the MVC, therefore no alias is made in this component.
- The **Template** component in MVT represents the presentation layer, serving the same functionality as the View component in MVC. Therefore, the **Template** component will be referred to as **View**.

4.1.1 Box-Line Diagram

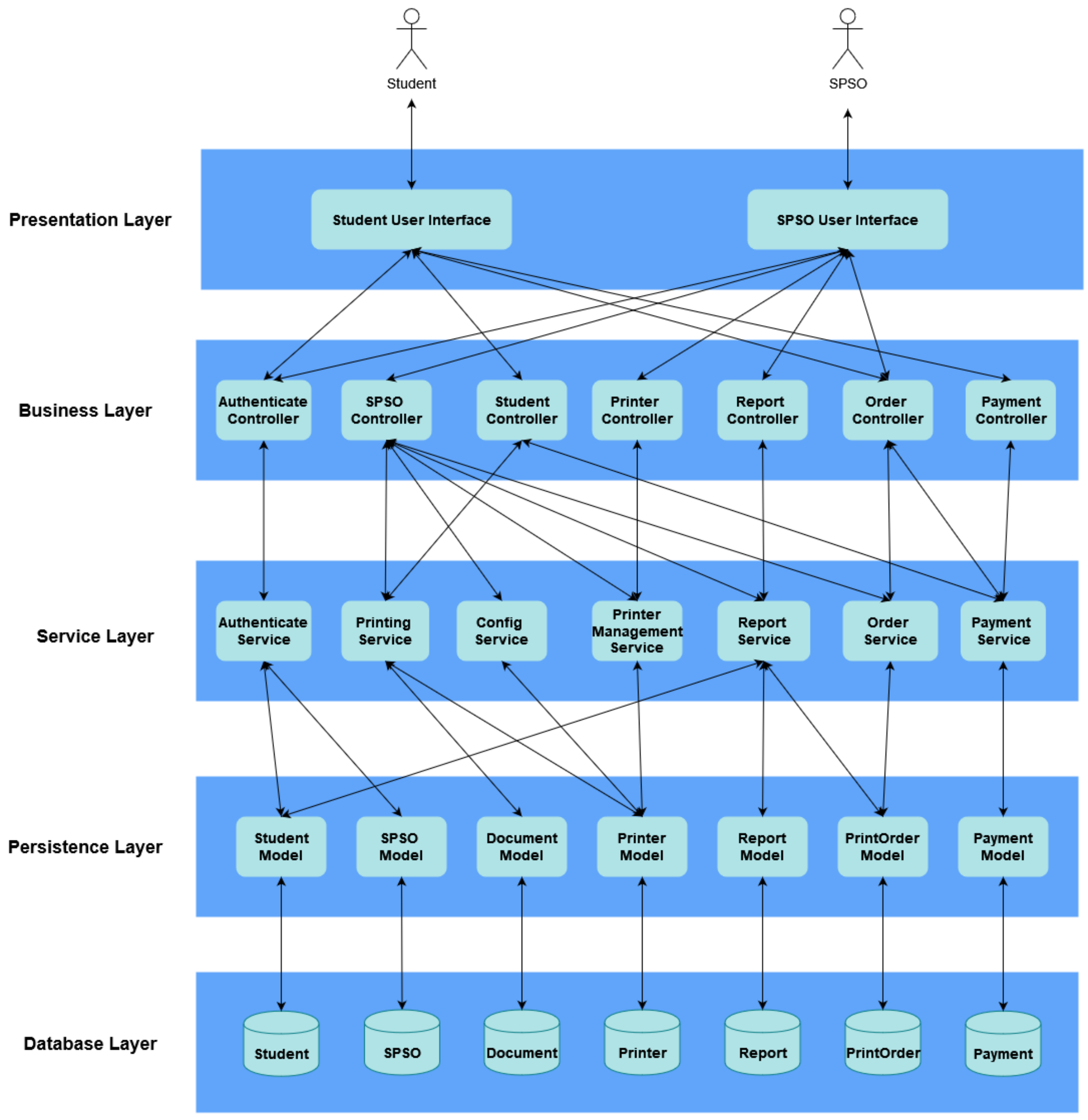


Figure 24: *Box-Line Diagram of all system*

Description for each layer:

- **Presentation layer**: is the User Interface for helping the Students and SPSO interact with the system. This layer will be divided into two large components UI is Student and SPSO. This layer represents the Front end of the system, developed by React library and other related libraries.
- **Business layer**: is the layer containing the Controller components, responsible for interacting with the Presentation Layer. It processes the request from the Presentation layer, validates them and chooses the proper correct business logic to apply to those requests.
- **Service layer**: is the intermediary between Business and Persistence layer. It containings services for specific actions that can be used by the Business layer.

- **Persistence layer**: is the layer responsible for managing the Model Objects and mapping them to database schemas in the Database Layer, containing the following models: **Student**, **SPSO**, **Document**, **Printer**, **Report**, **PrintOrder**, and **Payment**. These models define the entities in the system and establish the rules for how the data is structured and validated.
- **Database layer**: is the foundation layer for storing data of the HCMUT SSPS system.

4.1.2 Deployment Diagram

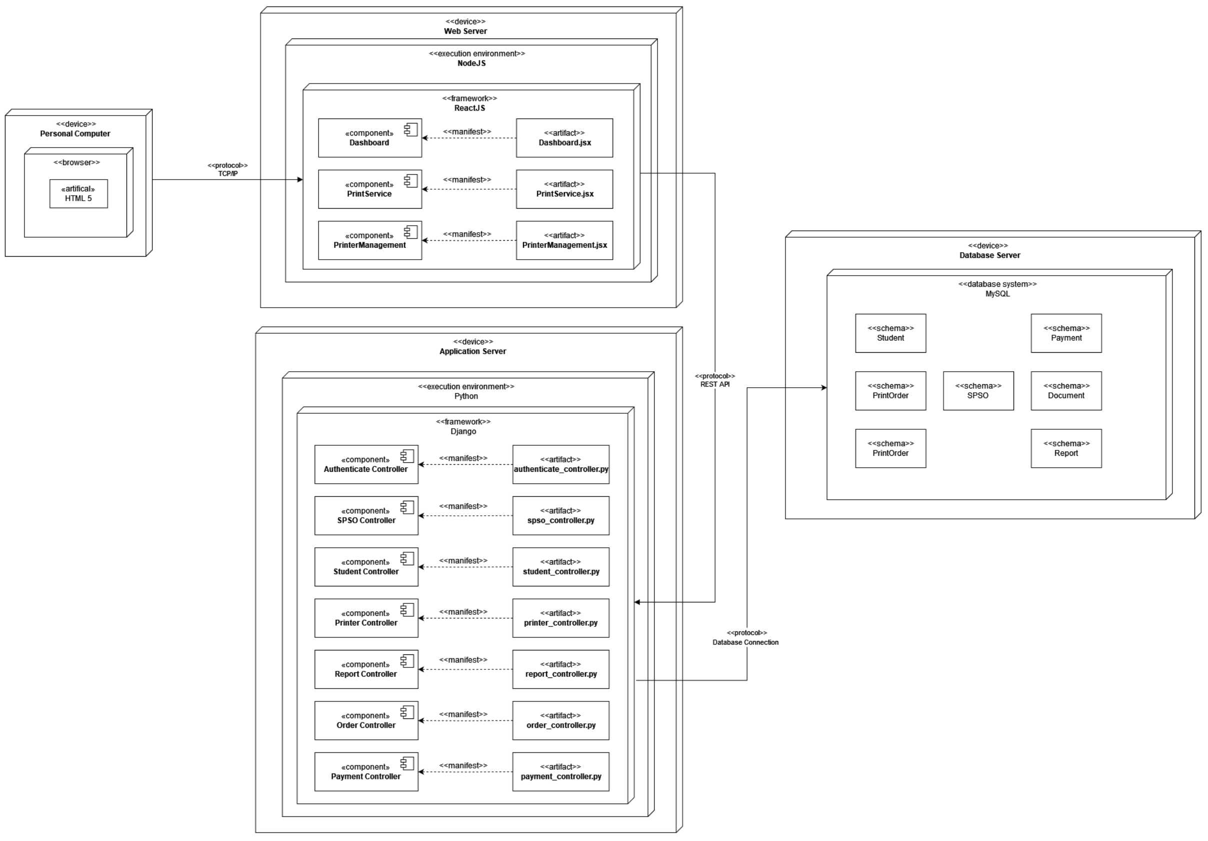


Figure 25: *Deployment Diagram*

Description for Deployment Diagram:

(a) Back-end Execution Environment: Python and Django

- The **main application server** operates within a **Python execution environment** using the **Django framework**.
- Within this environment, **seven primary components** are represented as **controllers** in Django, each responsible for distinct functionalities:
  - **Authenticate Controller**: Manages user authentication.
  - **SPSO Controller**: Handles specific SPSO-related tasks.
  - **Student Controller**: Manages student-related information and interactions.
  - **Printer Controller**: Supports printing services.
  - **Report Controller**: Facilitates report generation and management.
  - **Order Controller**: Manages order-related functions.
  - **Payment Controller**: Overseas payment processing.

Each component is mapped to a specific Python file (artifact) in the Django application.

(b) Front-End Execution Environment: ReactJS

- The front-end environment is built using **ReactJS** and hosted on Web Server with **three primary components**. These components handle client-side rendering and interact with the Django back-end via **REST APIs**.
- The front-end components are compiled from their respective JavaScript source files (artifacts), facilitating a dynamic user interface that updates and responds to user inputs.

(c) Client-Server Communication

- The **Client application** connects to the **Server** over the **TCP/IP protocol**.
- Data transferred from the Server to the Client includes interface-related files, which are displayed through an embedded **HTML5 reader** on the Client application.
- This setup allows users to interact with the application's functionality seamlessly on their local devices.

(d) Database Server

- The **Database Server** hosts a **MySQL relational database** that stores persistent data for the application.
- It communicates with the main application server over the **TCP/IP protocol** for data retrieval and updates. This ensures that data displayed on the client interface remains consistent and up-to-date based on user interactions.

(e) Data Flow and Interaction

- **Client-Server Interaction**: The client (user browser) sends user actions to the Server, which processes them, updates the interface, and fetches data from the Database Server when needed.
- **Server-Database Communication**: The Server retrieves or updates data stored in the Database Server, ensuring data persistence and consistency.

4.2 Presentation Strategy

For developing the User Interface for the HCMUT SSPS, we will use two technology:

- **React**: React is a library for developing web applications and native user interface. It focuses on building dynamic and reusable components, which can help for designing HCMUT SSPS more straightforwardly.

Figure 26: *(source: https://www.fullstackpython.com/react.html)*

- **TailwindCSS**: TailwindCSS is a CSS framework packed with classes, which can be composed directly into React Components. Utility classes from TailwindCSS will help us to style our application more quickly and effectively.

Figure 27: *(source: https://tailwindcss.com/)*

Beside the technology used for building User Interface, we made some criterias for our implementation:

- **Responsible Design**: We will try to create a responsive design for HCMUT SSPS, ensuring that our application looks and functions optimally across many designs like mobile phone and laptop.
- **User-friendly Interface**: The Interface should be easy to navigate between pages, with all essentials and easy-to-use components for both the students and SPSO like menu to upload document or managing printers.

4.3 Data Storage Approach

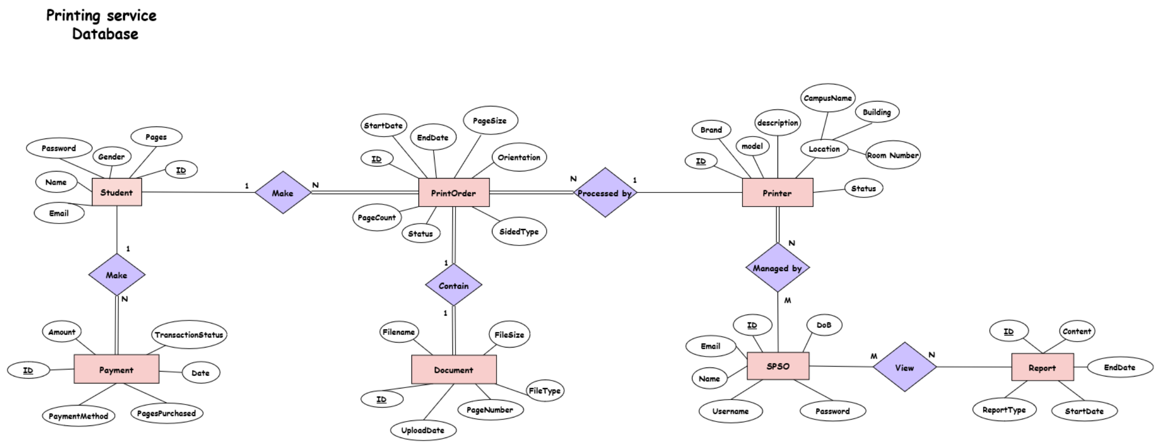


Figure 28: *(E)ERD of Data Storage Approach*

Description for (E)ERD of Data Storage Approach:
To store the data as defined by the ERD for the HCMUT Student Smart Printing Service, we convert each entity into a database table, where the attributes become the columns of the table. Each table is assigned a primary key to uniquely identify its records. Relationships between entities are established using foreign keys, which reference primary keys in related tables. Converting Entities to Tables:
Each entity in the ERD represents a real-world object or concept, such as a student, printer, or print order. We begin by creating a table for each entity:

- **Student Table**: Stores information about students using the service, including personal details and their page balance for printing.

```sql
-- Create the Student table
CREATE TABLE Student (
  StudentID INT PRIMARY KEY AUTO_INCREMENT,
  Name VARCHAR(100) NOT NULL,
  Email VARCHAR(100) NOT NULL UNIQUE,
  Password VARCHAR(255) NOT NULL,
  Gender ENUM('Male', 'Female', 'Other') NOT NULL,
  PageBalance INT NOT NULL DEFAULT 0 CHECK (PageBalance >= 0),
  RegistrationDate DATETIME DEFAULT CURRENT_TIMESTAMP
);
```

- **SPSO Table**: Stores information about the Student Printing Service Officers who manage the system.

```sql
CREATE TABLE SPSO (
  SPSOID INT PRIMARY KEY AUTO_INCREMENT,
  Name VARCHAR(100) NOT NULL,
  Email VARCHAR(100) NOT NULL UNIQUE,
  Username VARCHAR(50) NOT NULL UNIQUE,
  Password VARCHAR(255) NOT NULL,
  DateOfBirth DATE
);
```

- **Printer Table**: Contains details about the printers available in the system.

```sql
-- Create the Printer table
CREATE TABLE Printer (
  PrinterID INT PRIMARY KEY AUTO_INCREMENT,
  Brand VARCHAR(50) NOT NULL,
  Model VARCHAR(50) NOT NULL,
  Status ENUM('Active', 'Inactive') NOT NULL,
  CampusName VARCHAR(100),
  Building VARCHAR(100),
  RoomNumber VARCHAR(50), Description VARCHAR(255)
);
```

- **Manage**: Contains details relationship about the printers and spso.

```sql
CREATE TABLE Manage (
  PrinterID INT NOT NULL,
  SPSOID INT NOT NULL,
  AssignedDate DATETIME DEFAULT CURRENT_TIMESTAMP,
  PRIMARY KEY (PrinterID, SPSOID),
  FOREIGN KEY (PrinterID) REFERENCES Printer(PrinterID) ON DELETE CASCADE,
  FOREIGN KEY (SPSOID) REFERENCES SPSO(SPSOID) ON DELETE CASCADE
);
```

- **Document Table**: Holds information about documents uploaded for printing.

```sql
CREATE TABLE Document (
  DocumentID INT PRIMARY KEY AUTO_INCREMENT,
  StudentID INT NOT NULL,
  Filename VARCHAR(255) NOT NULL,
  FileSize INT NOT NULL CHECK (FileSize > 0),
  FileType VARCHAR(50) NOT NULL,
  PageNumber INT NOT NULL CHECK (PageNumber > 0),
  UploadDateTime DATETIME DEFAULT CURRENT_TIMESTAMP,
  FOREIGN KEY (StudentID) REFERENCES Student(StudentID) ON DELETE CASCADE
);
```

- **PrintOrder Table**: Represents each print order made by a student, containing details about the print job.

```
CREATE TABLE PrintOrder (
  PrintOrderID INT PRIMARY KEY AUTO_INCREMENT,
  StudentID INT NOT NULL,
  DocumentID INT NOT NULL UNIQUE,
  PrinterID INT,
  StartDateTime DATETIME DEFAULT CURRENT_TIMESTAMP,
  EndDateTime DATETIME,
  PageCount INT NOT NULL CHECK (PageCount > 0),
  Status ENUM('Pending', 'Completed', 'Cancelled') NOT NULL DEFAULT 'Pending',
  PageSize ENUM('A4', 'A3') NOT NULL,
  Orientation ENUM('Portrait', 'Landscape') NOT NULL,
  SidedType ENUM('Single-sided', 'Double-sided') NOT NULL,
  FOREIGN KEY (StudentID) REFERENCES Student(StudentID) ON DELETE CASCADE,
  FOREIGN KEY (DocumentID) REFERENCES Document(DocumentID) ON DELETE CASCADE,
  FOREIGN KEY (PrinterID) REFERENCES Printer(PrinterID) ON DELETE SET NULL
);
```

- **Payment Table**: Records transactions where students purchase additional printing pages.

```
-- Create the Payment table
CREATE TABLE Payment (
  PaymentID INT PRIMARY KEY AUTO_INCREMENT,
  StudentID INT NOT NULL,
  Amount DECIMAL(10,2) NOT NULL CHECK (Amount > 0),
  PaymentDateTime DATETIME DEFAULT CURRENT_TIMESTAMP,
  PaymentMethod VARCHAR(50) NOT NULL,
  PagesPurchased INT NOT NULL CHECK (PagesPurchased > 0),
  TransactionStatus ENUM('Successful', 'Failed') NOT NULL,
  FOREIGN KEY (StudentID) REFERENCES Student(StudentID) ON DELETE CASCADE
);
```

- **Report Table**: Contains system usage reports generated automatically.

```
CREATE TABLE Report (
  ReportID INT PRIMARY KEY AUTO_INCREMENT,
  Content TEXT NOT NULL,
  ReportType ENUM('Monthly', 'Yearly') NOT NULL,
  StartDate DATE NOT NULL,
  EndDate DATE NOT NULL
);
```

- **View**: Junction table for the many-to-many relationship between SPSO and Report

```
CREATE TABLE View (
  SPSOID INT NOT NULL,
  ReportID INT NOT NULL,
  ViewDateTime DATETIME DEFAULT CURRENT_TIMESTAMP,
  PRIMARY KEY (SPSOID, ReportID),
  FOREIGN KEY (SPSOID) REFERENCES SPSO(SPSOID) ON DELETE CASCADE,
  FOREIGN KEY (ReportID) REFERENCES Report(ReportID) ON DELETE CASCADE
);
```

Many-to-many relationships, such as between Printer and SPSO (Student Printing Service Officer), are managed using junction tables. A junction table like Printer_SPSO contains foreign keys referencing both PrinterID and SPSOID, effectively linking printers and officers. This approach ensures that all relationships in the ERD are accurately represented in the database schema. By defining appropriate data types, constraints, and indexes, we maintain data integrity and optimize query performance.
Implementing this schema in a database management system like MySQL involves writing Data Definition Language (DDL) commands to create the tables and define their relationships. These commands specify the structure of each table, the primary and foreign keys, and any necessary constraints. This structured storage of data according to the ERD facilitates efficient data management and supports the functionalities required by the printing service system.

4.4 API Accessing

In the HCMUT SSPS system, external APIs from two following systems are meant to be used: **BKPay** (responsible for handling payment transactions within the system) and **HCMUT SSO** (responsible for user authentication and authorization). However, for simplicity in this demonstration, we will not use any external APIs to connect to these third-party systems. Instead, we will build simple components on the back-end to handle these functionalities internally: Payment Controller for handling payment transactions and Authentication Controller for handling user authentication and authorization.
For the APIs to communicate between client and server, we categorized those APIs into:

- **Security APIs**: these APIs manage role-based access control and permissions, ensuring that users can only access resources and perform actions according to their roles (Student and SPSO).
- **Report APIs**: Responsible for generating, retrieving, and managing reports related to print orders, user activity, and system usage.
- **Printer Management APIs**: These APIs handle operations related to printer setup and management within the system. This includes adding, updating, or removing printers, as well as retrieving printer status or details
- **Print Service APIs**: Handle the related functionality of print service: document validation, document printing configuration.
- **Order APIs**: These APIs handle the retrieval of print orders for students and provide them with the ability to cancel their orders if needed.

4.5 Component Diagram

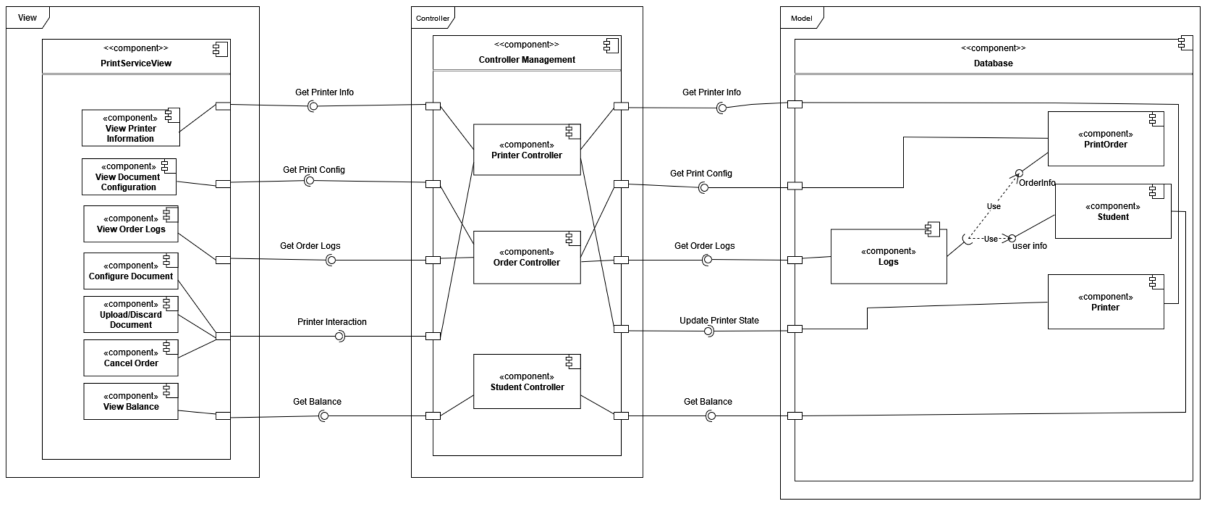


Figure 29: *Component Diagram*

Description for Component Diagram:
This component diagram illustrates a sophisticated printing service system using a three-tier architecture (View-Controller-Model). The system organizes its functionality through specialized controllers - Printer Controller, Order Controller, and Student Controller - each managing specific aspects of the printing service. The View layer provides user interface components for tasks like viewing printer information, configuring documents, managing orders, and checking balances. These user interactions are channeled throuh appropriate controllers to access and manipulate data in the Model layer, which contains core components like Database, PrintOrder, Student, Printer, and Logs. The system demonstrates clear separation of concerns and efficient data flow management, allowing students to interact with printing services while maintaining organized control over printer operations, order processing, and student-related functions.

References

[1] Ian Sommerville (2015), Software Engineering (10th ed.), ISBN 978-0133943030, Pearson