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

SOFTWARE ENGINEERING (CO3001)

PROJECT
SMART PRINTING SERVICE FOR STUDENT

Advisor:	Phan Trung Hiếu	
Students:	Tạ Nguyễn Tiến Dũng	2110965
	Nguyễn Minh Điềm	2111056
	Phan Thanh Hải	2110154
	Đoàn Nguyễn Duy Cường	2110068
	Nguyễn Doãn Hoàng	2111238
	Nguyễn Đức Anh Tuấn	2115177
	Nguyễn Tất Linh	2113915

HO CHI MINH CITY, 9/2023

Contents

1 Member list & Workload

No.	Fullname	Student ID	Percentage of work
1	Đoàn Nguyễn Duy Cường	2110068	100%
2	Phan Thanh Hải	2110154	100%
3	Tạ Nguyễn Tiến Dũng	2110965	100%
4	Nguyễn Minh Điềm	2111056	100%
5	Nguyễn Doãn Hoàng	2111238	100%
6	Nguyễn Tất Linh	2113915	100%
7	Nguyễn Đức Anh Tuấn	2115177	100%

2 Description of HCMUT_SSPS System

2.1 Domain context of HCMUT_SSPS at HCMUT

The Ho Chi Minh City University of Technology (HCMUT) is currently working on establishing the Smart Printing Service for students (HCMUT-SSPS) as part of their efforts to improve academic services and convenience within the campus. HCMUT-SSPS is a convenient and dedicated printing system for students, with the goal of providing a convenient and efficient way for students to print documents for their academic and research needs. This system will integrate printers available on the university campus and allow students to experience fast and time-saving printing services.

2.2 Stakeholders in SPSS System

1. **Direct users:** Students are the primary group in this project, which also means they are the people who use the service the most frequently. In addition, instructors will also have access to this system.
2. **HCMUT_SSPS Administrator:** This is the department responsible for managing the entire SSPS system directly. The SSPS Admin department can be structured into several sub-departments, and two important ones should be mentioned:
    - **Regarding the project management department:** This is the department in charge of finances, providing directions, and planning for the project.
    - **The development and system installation department:** This department receives guidance and directions from the management department and develops the project according to this plan (coding). This department is responsible for writing, managing, modifying, maintaining, and updating the system's functions to align with the management department's directions, ensuring security and more.
3. **Student Printing Service Officers (SPSO):** This is the department where staff members are responsible for managing and configuring settings, monitoring students' printing tasks during the printing process, ensuring they align with the provided plan and standards set by the Admin.
4. **Authentication for Printing Web:** This is the department responsible for authenticating user accounts when logging into the SSPS system.
5. **Onlinepayment system:** "This is the department responsible for handling payment-related issues within the system."

2.3 Stakeholders' needs and benefit of HCMUT_SPSS

1. **For direct users:**
    - Communicating speed with SSPS fast, flexible, and with minimal steps to avoid confusion.
    - Available to print (with the ability to change the attributes of the pages to be printed), upload various file types, and support additional printing (if allocated pages run out) through extra payment.

- Have a "user-friendly" interface, which has basic functions such as uploading files; selecting printers; printing tasks such as paper size, single or double-sided, number of copies, remaining pages, and adding credit to continue using SSPS (when additional pages are needed).
- Since SSPS is built right on the campus, it provides convenience for users as they don't need to search for external print shops, saving both time and effort. SSPS also simplifies and modernizes the printing process: users can upload files from anywhere using smart devices, saving time and reducing the hassle of searching for printers. They also have full control over their printing process, from uploading files to selecting printers and configuring printing properties.
- Using SSPS ensures the security of users' information and documents, as they are processed safely. The SSPS service aligns with the needs of modern education for seamless digital access, which is currently trending.

2. **For the HCMUT_SSPS Administrator:**
    - This department is essentially the creative hub responsible for conceiving and developing ideas for HCMUT_SSPS with the goal of improving intelligent learning services that provide convenience for students to enhance their learning experience. HCMUT_SSPS will assist the SSPS Administrator in efficiently managing students' print jobs, reducing reliance on manual printing processes, and ensuring compliance with university regulations. This simplifies their work and makes it more efficient.
    - Managing printing resources and paper consumption sustainably, while simultaneously improving educational services for students. It also contributes to raising awareness about environmental conservation.
3. **For SPSO (Student Printing Service Officers):**
    - Manage and configure setting of students' printing tasks and ensuring that the printing process aligns with the provided plan from higher authorities, with the aim of enhancing modernization and simplification in the students' learning process. HCMUT-SSPS is the system that will support SPSO in achieving this.
4. **Authentication for Printing Web:**
    - Ensure that all users are under the management of the university and act in the collective interest of the institution.
    - Create a trustworthy service environment, minimizing disruptions from external parties and reducing risks to the system.
    - Easily manage financial resources and impose penalties for any acts of sabotage or misconduct.
5. **Online payment system:**
    - Users should be able to transparently manage their account balances, feeling secure and trustworthy when conducting any transactions with the system.
    - Ensures consistency in tracking the funds entering the system, making it easier to manage resources, generate capital, and allocate funds for system development, maintenance, or other university activities.
    - Can be combined with free document printing for individuals through the university's scholarship programs to incentivize students.

3 Describe all functional and non-functional requirements

Requirements that are well thought through and clearly documented are essential to any successful software engineering project. There are two main different types of system requirements that should be gathered by those working on software projects. System requirements can be categorized as either functional requirements or non functional requirements.
Functional Requirements: Specific features and functions that a system must possess in order to meet the needs and expectations of its stakeholders. They define what the system must do and are usually described in terms of inputs, processes, and outputs.
Non-functional Requirements: Specific attributes or qualities that a system must possess, but they are not directly related to a particular function or feature of the system.

3.1 Functional requirements

1. ***Functional Requirements for End Users (Adjusted for Teachers)***

    - File Upload: Students and teachers should be able to upload document files to the system.
    - Printer Selection: Students and teachers should be able to choose a printer from the available options.
    - Printing Properties: Students and teachers should be able to specify printing properties such as paper size, pages to be printed, one-/double-sided printing, and the number of copies.
    - File Type Validation: The system should validate the uploaded file to ensure it is of a permitted file type.
    - Printing History and Log Viewing: End users should have the ability to view their printing history and logs within the system. The printing history and log should provide details such as the timestamp of each print job, the printer used, and any associated costs.
    - Print Confirmation: Students and teachers should receive a confirmation message or notification after successfully submitting a print job.
    - Queue Status Display: The SPSO should have a function to display the number of orders in each queue and the estimated time to complete for each printer.
    - Priority Queue: Implement two types of priority queues for print jobs:
        - Regular Queue (MLQ - First Come First Serve): All print jobs, both from students and teachers, will be added to the regular queue and processed based on the FCFS principle.
        - Premium Queue: Teachers requiring urgent printing will have their print jobs prioritized in the premium queue, ensuring they are processed with higher priority than regular print jobs. Students would be able to use this premium queue if they pay additional money.
    - Adjustments for Teachers:
        - Pricing Adjustment: Teachers should be able to purchase paper at a discounted price compared to students.
        - Paper Allocation Adjustment: Teachers should have a higher allocation of free papers compared to students.

2. ***Functional Requirements for Student Printing Service Officer (SPSO)***

    - File Type Configuration: The SPSO should be able to configure and manage the permitted file types for printing.
    - Printer Management: The SPSO should have the ability to add, remove, or update printer options in the system.
    - Printing Properties Configuration: The SPSO should be able to define and modify the available printing properties, such as paper size options and supported double-sided printing.
    - Printing History and Log Viewing for SPSOs: The system should enable the SPSO to view the printing history and logs of all students/teachers or a specific student/teacher for a given time period (date to date). The printing history and log should provide detailed information, including the timestamp, student name, printer used, number of printed pages, and page sizes for each print job.
    - Access Control: The SPSO should have administrative privileges to manage student accounts, monitor print jobs, and generate usage reports.
    - Error Handling: The SPSO should be notified or provided with error logs in case of any system failures or issues.

3. ***Functional Requirements for HCMUT_SSPS Administrator:***

    - SPSO Management: The SSPS administrator should be able to manage other Service Provider System Operators (SPSOs) within the system. It should have the ability to create, modify, delete SPSO accounts, and the capability to view and monitor the activities and performance of individual SPSOs. The main duty of SSPS administrator is to assign roles and permissions to SPSOs based on their responsibilities and access requirements
    - Reporting and Analysis: The SSPS should have the capability to collect reports from SPSOs regarding the usage, performance, feedback of the system, and be able to generate and submit reports that provide insights and analysis on various aspects of the system, such as user trends, printing patterns, and functionality requirements.
    - Source Code Customization: The SSPS administrator should have the capability to adjust the source code of the system to cater to specific customization requirements. They should be able to modify the system's codebase to introduce new features, enhance existing functionalities, or integrate with external systems.

4. ***Functional Requirements for HCMUT_SSO authentication service:***

    - Authentication Integration: The system should integrate with the HCMUT's internal authentication system or Single Sign-On (SSO) solution to perform user authentication.
    - Internal User Access: The system should only allow users within the internal network of the Bach Khoa University (HCMUT) to log in.
    - User Authentication: All users must undergo authentication before being granted access to the system. The authentication process should verify the user's identity and credentials.

5. ***Functional Requirements for Online Payment System:***

    - Available Pages: The system should display the number of available print pages for each user account. Users should be able to view the remaining print pages associated with their account.
    - Buy Page Feature: The system should provide a Buy Page feature that allows users to purchase additional print pages. The Buy Page feature should calculate and display the specific amount of money required to add a certain number of pages.
    - Integration with digibank: The system should integrate with a digibank platform, to enable students to make payments via bank accounts (or other electronic wallets).

3.2 Non-functional Requirements

- User-Friendly Interface: The system should have an intuitive and easy-to-use interface for both end users (students) and SPSOs to navigate and interact with. To evaluate, we are using Usability Score (e.g., measured through user surveys or usability testing), and the target is at least 80% of users should rate the interface as "easy to use" or "very easy to use" in user satisfaction surveys.

- Fast Response Time: The system should provide quick responses for actions such as file uploads, printer selection, and specifying printing properties. The metric to evaluate it would be using Average Response Time (measured in milliseconds). The system should respond to user actions within 5 seconds on average.

- Reliability: The system should be available and reliable for both end users (students) and SPSOs to use at any time without frequent downtime or disruptions. For evaluation, we will use Uptime Percentage Metric (measured over a specific time period). The system should have an uptime of at least 99.9% during a given period.

- Security: The system should ensure the confidentiality and integrity of uploaded documents and student data. By using Vulnerability Assessment Score (e.g., based on penetration testing results), we can ensure that the system will undergo regular security assessments, with no critical security vulnerabilities identified.

- Scalability: The system should be able to handle a large number of concurrent print requests, especially during peak times. It should be able to handle, for example, 500 concurrent print requests without significant degradation in performance. To satisfy this requirement, we can use some metrics, for example, Concurrent Request Handling Capacity (measured as the maximum number of simultaneous print requests),...

- Administrative Interface: The system should provide a secure and user-friendly interface for SPSOs to configure settings, manage printers, and monitor print jobs. To evaluate, we can use Usability Score, and to satisfy this requirement, we expect that at least 80% of SPSOs should rate the administrative interface as "easy to use" or "very easy to use" in user satisfaction surveys.

- Performance: The system should handle the SPSO's administrative tasks efficiently, ensuring fast response times and minimal delays. It should be capable of handling concurrent administrative actions without compromising performance. This requirement ensures that the SPSO can accomplish their tasks in a timely manner, enhancing productivity and user satisfaction. Using Average Response Time for Administrative Tasks, we expect that the system would provide a fast response time, with administrative tasks completing within 3 minutes on average.

- Auditability: The system should log and track the SPSO's actions for accountability and auditing purposes. This requirement ensures that the SPSO's activities within the system can be monitored and reviewed if needed. By using Audit Log Completeness Metric, we can ensure that the system would accurately log at least 95% of the SPSO's actions for auditing and accountability purposes.

- Backup and Recovery: The system should have regular data backups and a disaster recovery plan to prevent data loss and ensure business continuity. This requirement safeguards the system's data and provides a mechanism to restore it in case of any unforeseen incidents. We can use both Backup Frequency and Recovery Time to evaluate for the system. The system should have regular data backups, with a recovery time objective (RTO) of 4 hours or less in case of a failure.

- Compatibility: The system should be compatible with different web browsers and devices to accommodate the SPSO's preferences and work environment. This requirement ensures that the SPSO can access and use the system seamlessly from their preferred browsers and devices. The system should be compatible with the latest versions of popular web browsers (e.g., Chrome, Firefox, Safari) and support at least 90% of commonly used devices. By using Browser and Device Compatibility, we would be able to ensure that.

4 Use-case diagram and use-case description table

4.1 Use-case diagram

4.1.1 For the whole system

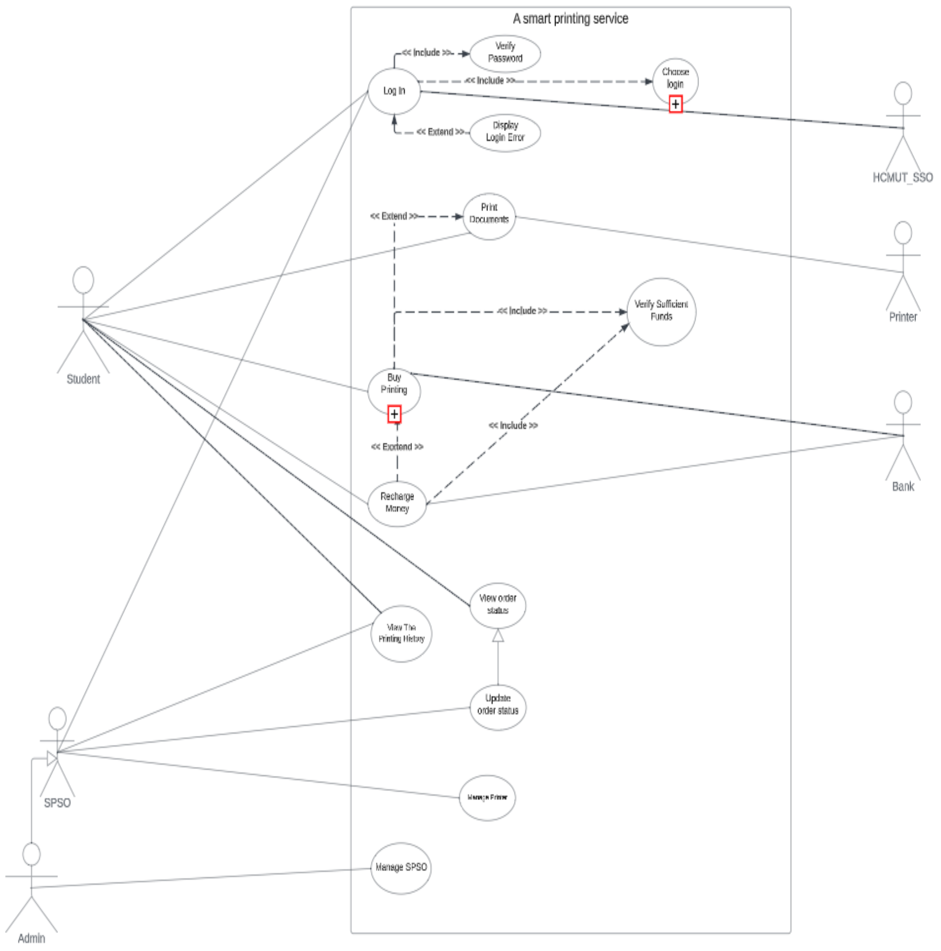


Figure 4.1: Use-case diagram for the whole system

4.1.2 Print Documents Module

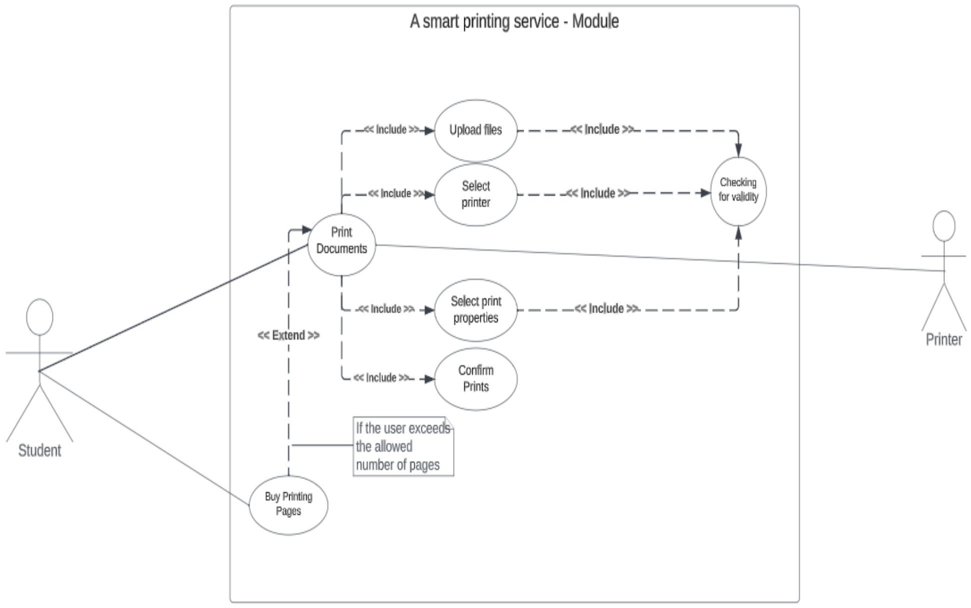


Figure 4.2: Use-case diagram for the Print Documents module

4.2 Use-case description table

4.2.1 Login

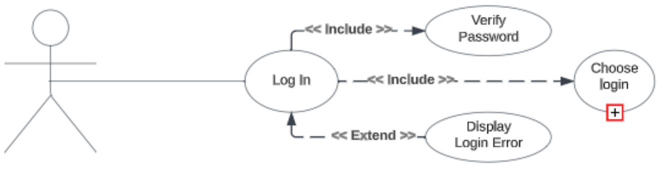


Figure 4.3: Login use case

Use case name	Login
Actor	Student, SPSO, Admin
Descriptions	The "Login" function allows users to verify their identity by providing a login name and password. After successfully logging in, they can access the system and perform specific operations.
Pre-conditions	• The system has been started. • The user has registered an account and has login information (login name and password).
Trigger	Users want to access the system or perform activities that require authentication.
Post-conditions	• The user has successfully logged in and has access to the system. • The system has determined the user's identity.
Normal Flow	**Step 1**: User accesses the system's login page. **Step 2**: The system displays the login screen with the following information fields: User name Password Login status (Student or SPSO or Admin) **Step 3**: Users enter their login information in the corresponding fields. **Step 4**: The user presses the "Login" button. **Step 5**: The system checks the login information with the database to verify the validity of the login name and password. If the login information is correct and valid, the system determines that the user has logged in successfully. **Step 6**: The system redirects the user to the home screen or page they want to access. **Step 7**: End Use Case
Alternative Flow	None
Exception Flow	**Step 5(Exception)**: If the login information is incorrect or invalid, the system displays an error message and requires the user to re-enter the login information.

4.2.2 Print Documents

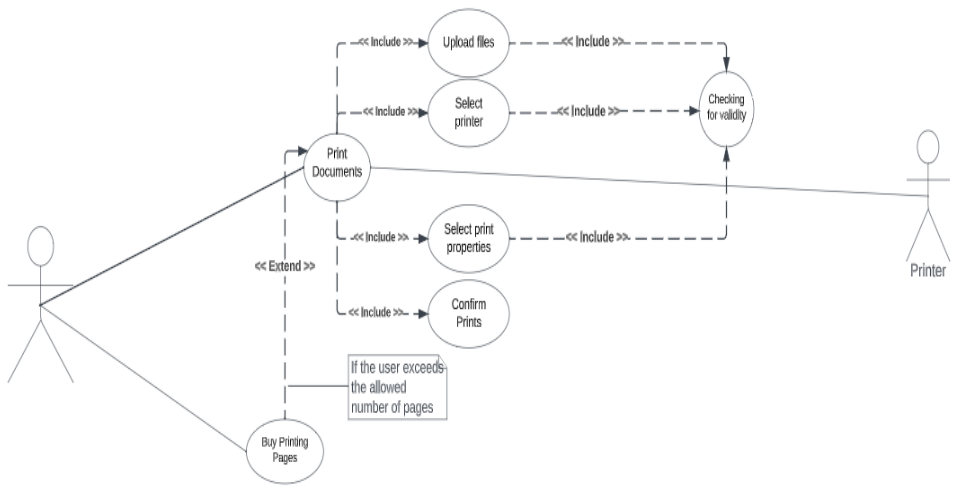


Figure 4.4: Print Documents use case

Use case name	Print Documents
Actor	Student
Descriptions	Print documents when users request.
Pre-conditions	• User logs in as a student.
Trigger	Clicks on "Print Document"
Post-conditions	• Printer confirm print request. • Update print queue. • Update print history. • Set the order to "waiting" status.

Normal Flow	**Step 1**: Users access the app's home screen and find the "Print Documents" option. The user clicks on "Print Document." **Step 2**: The system redirects the user to the "Upload File" screen for the user to upload the required documents. **Step 3**: Users upload documents by selecting files from their computer and pressing the "Upload" button. **Step 4**: The system checks the uploaded file format. If the format is valid, the system will continue. **Step 5**: The system displays the "Print Preview" screen including the printer selection function and print properties. **Step 6**: User selects printer. **Step 7**: User selects print properties. **Step 8**: The "Print Preview" screen displaying the preview document after selecting print properties. **Step 9**:The system checks the validity. If valid, the system will continue. **Step 10**: The system redirects the user to the specified screen to check again before entering. **Step 11**: User defines printing information, limits such as number of copies and other settings (if any). **Step 12**:The user has received a confirmation message that the document has been sent to the server. **Step 13**:The use case ends when the document is confirmed to be printed.
Alternative Flow	**Step 10(Alternative):** If the user wants to cancel the request after validating the information on the authentication screen, they can select the "Cancel request" option. **Step 11(alternative):** The use case ends when the user requests no requests and no documents are executed.

Exception Flow	**Step 4(Exception):** If the uploaded file format is invalid, the system displays an error notifying the user and asking them to upload a new file. **Step 9.1(Exception):** If the print attribute is invalid, the system displays an error message to the user and asks them to change the print properties. **Step 9.2(Exception):** If the user's available printing paper runs out, the system displays a message asking the user to buy more printing paper.

4.2.3 Buy Printing Pages

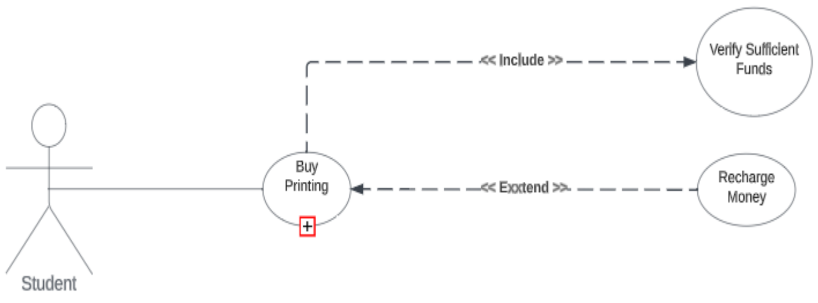


Figure 4.5: Buy Printing Page use case

Use case name	Buy Printing Pages
Actor	Student
Descriptions	Buy additional printing paper when needed or as required by the system.
Pre-conditions	• User logs in as a student • The account has enough money
Trigger	• Clicks on "Buy Printing Pages" • Click the OK button on the notification screen when the system requests to buy more printing paper.
Post-conditions	• Update wallet balance • Update the account of available printing paper. • Update history.
Normal Flow	**Step 1**: Users access the app's home screen and find the "Buy Printing Pages" option. The user clicks on "Buy Printing Pages." **Step 2**: The system transfers the user to the "Buy printed pages" screen for the user to select quantity of printing paper to buy. **Step 3**: Users enter the amount of printing paper to buy. **Step 4**: The system checks wallet balance. If enough, the system continues. **Step 5**: The use case ends when the purchase of printing paper is successful.
Alternative Flow	**Step 1(Alternative):** When the system displays the screen asking to buy more printing paper and the user clicks OK.
Exception Flow	**Step 4(Exception)**: If the account balance is insufficient, the system displays a request to add more money.

4.2.4 Recharge Money

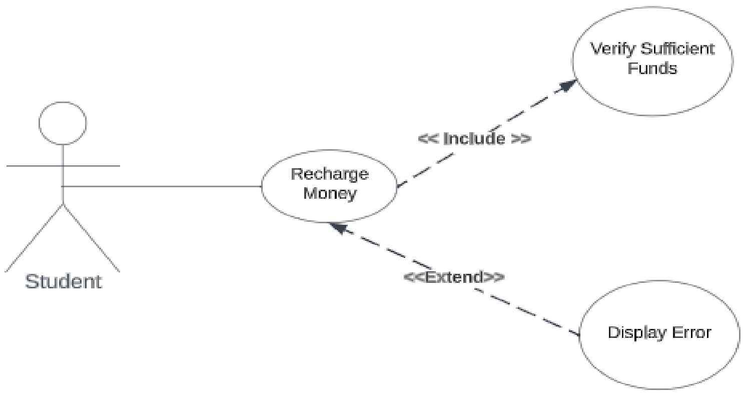


Figure 4.6: Recharge Money use case

Use case name	Recharge Money
Actor	Student
Descriptions	Deposit money into e-wallet
Pre-conditions	• User logs in as a student. • User account with bank connection
Trigger	Clicks on "Recharge Money". Click the OK button on the notification screen when the system.
Post-conditions	• Update wallet balance. • Update history.

Normal Flow	**Step 1**: Users access the app's home screen and find the "Recharge Money" option. The user clicks on "Recharge Money." **Step 2**: he system redirects the user to the "Recharge Money" screen. **Step 3**: The user enters the amount to deposit and selects the bank. **Step 4**: The system displays a screen with information about the deposit amount and selected bank and asks the user to confirm. **Step 5**: The system checks the balance of the bank wallet. If enough, the system continues. **Step 6**: Use case ends when the deposit is successful.
Alternative Flow	**Step 1 (Alternative):** When the system displays the screen requesting additional money and the user clicks OK.
Exception Flow	**Step 5 (exception):** If the bank account balance is insufficient, the system will display an error screen. **Step 7 (exception):** If the user enters the wrong OTP code, the system will ask to re-enter it. If more than 5 times, the system will cancel the transaction and Use case end.

4.2.5 Manage Printer

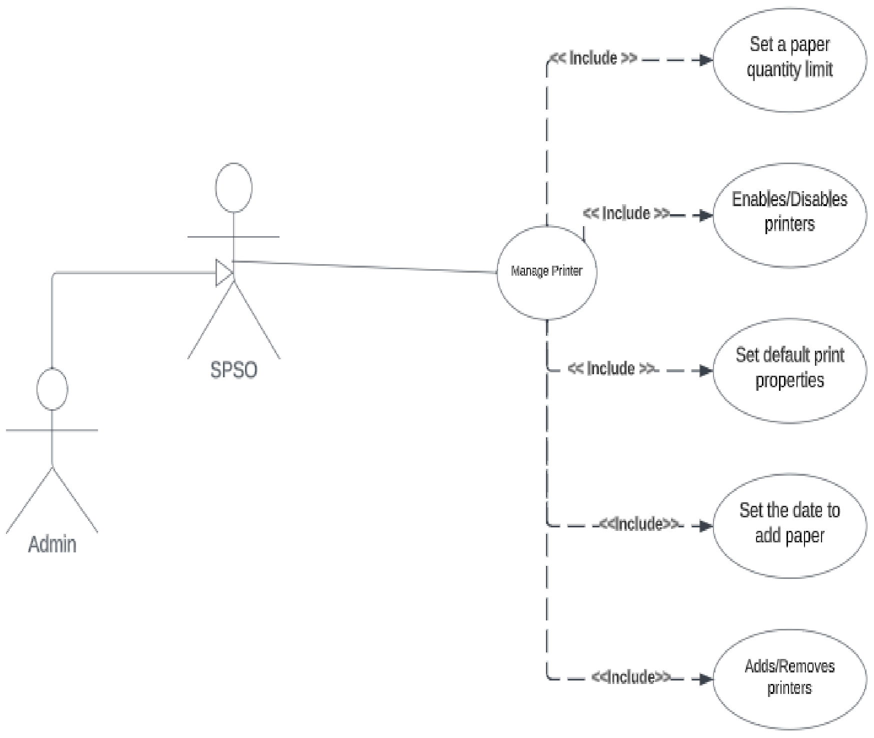


Figure 4.7: Manage Printer use case

Use case name	Manage Printer
Actor	SPSO or Admin
Descriptions	Manage Printer when Users request
Pre-conditions	User logs in as a SPSO or Admin
Trigger	clicks on "Manage Printer"
Post-conditions	• Set a limit on the amount of printing paper or manage the allowed printing properties that have been applied to the printer. • Changes are saved and the printer is updated.
Normal Flow	**Step 1**: Users access the app's home screen and find the "Manage Printer " option. The user clicks on "Manage Printer ". **Step 2**: The system displays an options screen that allows the system administrator to perform the following sub-functions: Set print paper limit: Users enter the print paper limit for the printer. Set default print properties: Users sets allowed data file types, default printing paper Enable/Disable printers Set the date to add paper: Set the date to receive additional free printing paper. Add/Remove Printer: If the user is logged in as an administrator. **Step 3**: Users chooses to save the changes. **Step 4**: The system updates the printer with the saved changes. End Use Case.
Alternative Flow	**Step 3 (Alternative):** If the Users does not want to make changes and returns to the main screen, the Use Case also ends with no changes applied.
Exception Flow	**Step 4 (Exception):** If the system encounters a technical error or cannot update the printer , the system will display an error message and the printer management process will not be completed. Users may need to try again or contact technical support.

4.2.6 View Order Status

View order status

Figure 4.8: View Order Status use case

Use case name	View order status
Actor	Student, SPSO
Descriptions	View order status when users request
Pre-conditions	User logs in as a student or SPSO
Trigger	Clicks on "Order Status"
Post-conditions	None
Normal Flow	**Step 1**: Users access the app's home screen and find the "Order Status" option. The user clicks on "Order Status" **Step 2**: The system displays a list of orders and the student's order status including two statuses: waiting (not printed) and finished (printed). **Step 3**: User checks order status. **Step 4**: The use case ends when the user closes the "Order Status" screen.
Alternative Flow	None
Exception Flow	**Step 3 (Exception):** If the system encounters a technical error or cannot find the order, the system will display an error message. Student or SPSO may need to try again or contact technical support.

4.2.7 Update Order Status

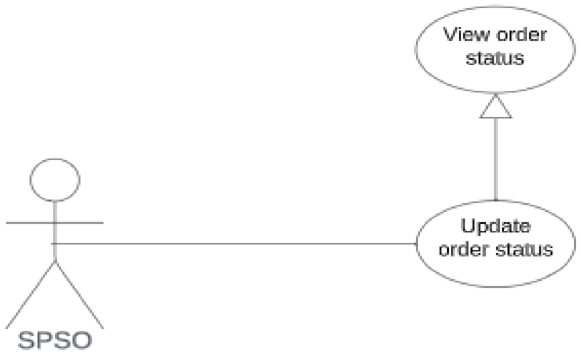


Figure 4.9: Update Order Status use case

Use case name	Update order status
Actor	SPSO
Descriptions	View order status when SPSO request
Pre-conditions	User logs in as the SPSO
Trigger	Clicks on "Order Status"
Post-conditions	None
Normal Flow	**Step 1**: Users access the app's home screen and find the "Order Status" option. The user clicks on "Order Status" **Step 2**: The system displays a list of orders and the student's order status including two statuses: waiting (not printed) and finished (printed). **Step 3**: SPSO updates order status when: If there is a printed notification from the printer, SPSO will change the order status from waiting to finished. If the order has been received by the student, SPSO will change the order status from finished to taken. **Step 4**: SPSO presses the "update" button. **Step 5**: The system updates order status. If successful, the system displays the new status and deletes orders with taken status from the "Order Status" screen and saves those orders in the transaction history. **Step 6**: The use case ends when the user closes the "Order Status" screen.
Alternative Flow	None

Exception Flow	**Step 4 (Exception):** If the system encounters a technical error or cannot find or update the order, the system will display an error message. SPSO may need to try again or contact technical support.

4.2.8 View the Printing History

View The Printing History

Figure 4.10: View the Printing History use case

Use case name	View The Printing History
Actor	Student, SPSO
Descriptions	View The Printing History when users request
Pre-conditions	User is logged in (Student or SPSO)
Trigger	Clicks on "History"
Post-conditions	Displays the history screen

Normal Flow	**Step 1**: Users access the app's home screen and find the "History" option. The user clicks on "History". **Step 2**: The system redirects the user to the "History" screen appropriate to the user account type (Student or SPSO): For Students: provides date search toolbar(date start - date end) , history type selection list (including print history, transaction history). For SPSO: provides object search toolbar, object type selection list (including students, printers) **Step 3**: User conducts search. **Step 4**: The system performs data search. If yes, the system will continue. **Step 5**: The use case ends when the screen displays the history successful.
Alternative Flow	None
Exception Flow	**Step 4(Exception):** The system displays a message that the history is not found, asking the user to re-enter it.

4.2.9 Manage SPSOs

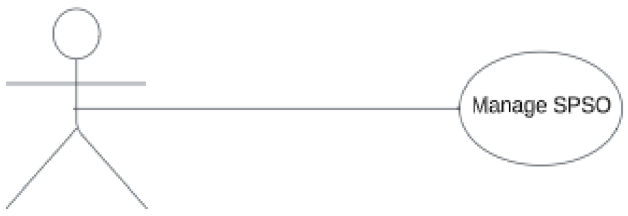


Figure 4.11: Manager SPSOs use case

Use case name	Manage SPSOs
Actor	Admin
Descriptions	Manage the list of SPSOs
Pre-conditions	User logs in as a Admin
Trigger	Clicks on "Manage SPSOs"
Post-conditions	Update list of SPSOs (if any changes)
Normal Flow	**Step 1**: Users access the app's home screen and find the "Manage SPSOs " option. The user clicks on "Manage SPSOs ". **Step 2**: The system displays an options screen that allows the system administrator to perform the following sub-functions: View SPSO list. Add/Remove SPSO. **Step 3**: Users choose to save the changes (if any). **Step 4**: The system updates the printer with the saved changes (if any). The use case ends when the user clicks the "Cancel" button.
Alternative Flow	**Step 2 (Alternative):** If the Users does not want to make changes and returns to the main screen, the Use Case also ends with no changes applied.
Exception Flow	**Step 2 (Exception):** If the system encounters a technical error or cannot display the SPSO list, the system will display an error message. The user may need to try again or contact technical support.

5 Activity diagram

5.1 Login

Firstly, the system will display the login site, the user enters the login information and the system will verify that information. If the information is valid, compare it with the database, otherwise an error will be displayed. Similarly, when comparing with the database, if the information is valid, the login is successful, otherwise an error is displayed.

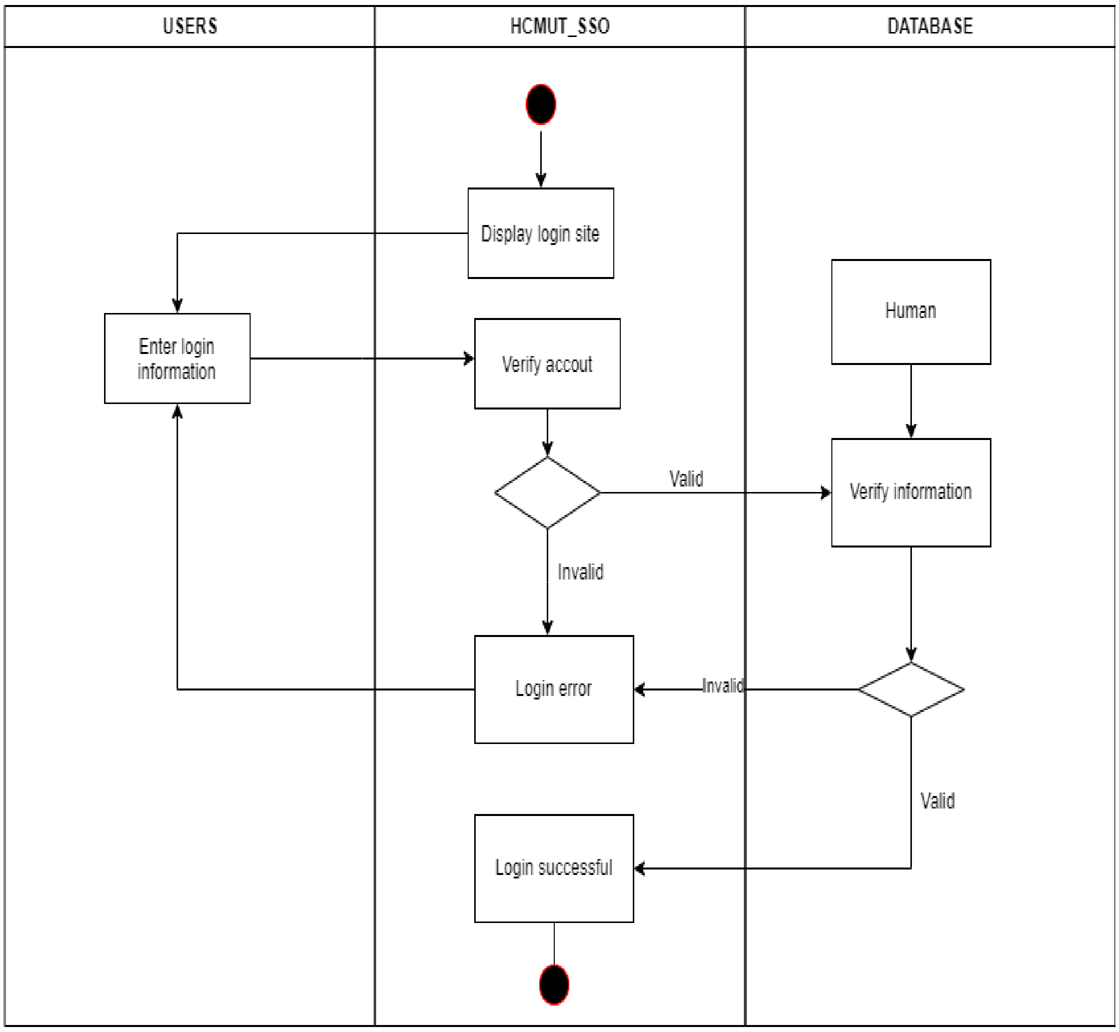


Figure 5.1: Activity diagram of login

5.2 Buying Printing Pages

Firstly, the user will choose to buy printing papers, the system displays the purchase site and verify sufficient funds. If the information is valid, compare it with the database, otherwise an error will be displayed. Similarly, when comparing with the database, if the information is valid, the payment will be successful, otherwise an error will be displayed.

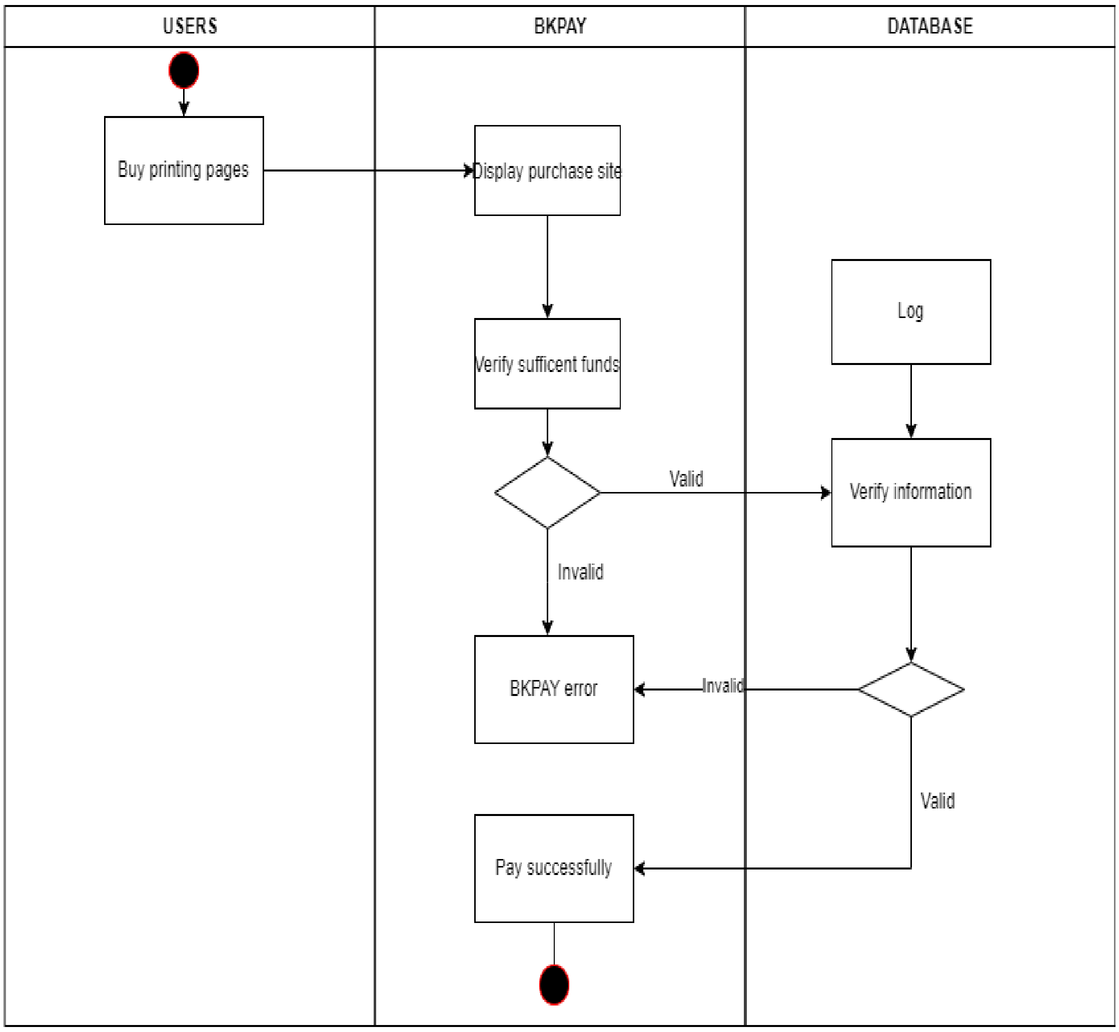


Figure 5.2: Activity diagram of buying printing pages

5.3 Print Documents

First, the user will choose to print the document, the system displays the printing site and the user will upload files then the system will check for validity. If the information is valid, save the data and continue, otherwise display an error. Continue with the select printers and select print properties steps and do the same. After saving the data, the system will request a print and the printer will queue the request to wait for its turn to print.

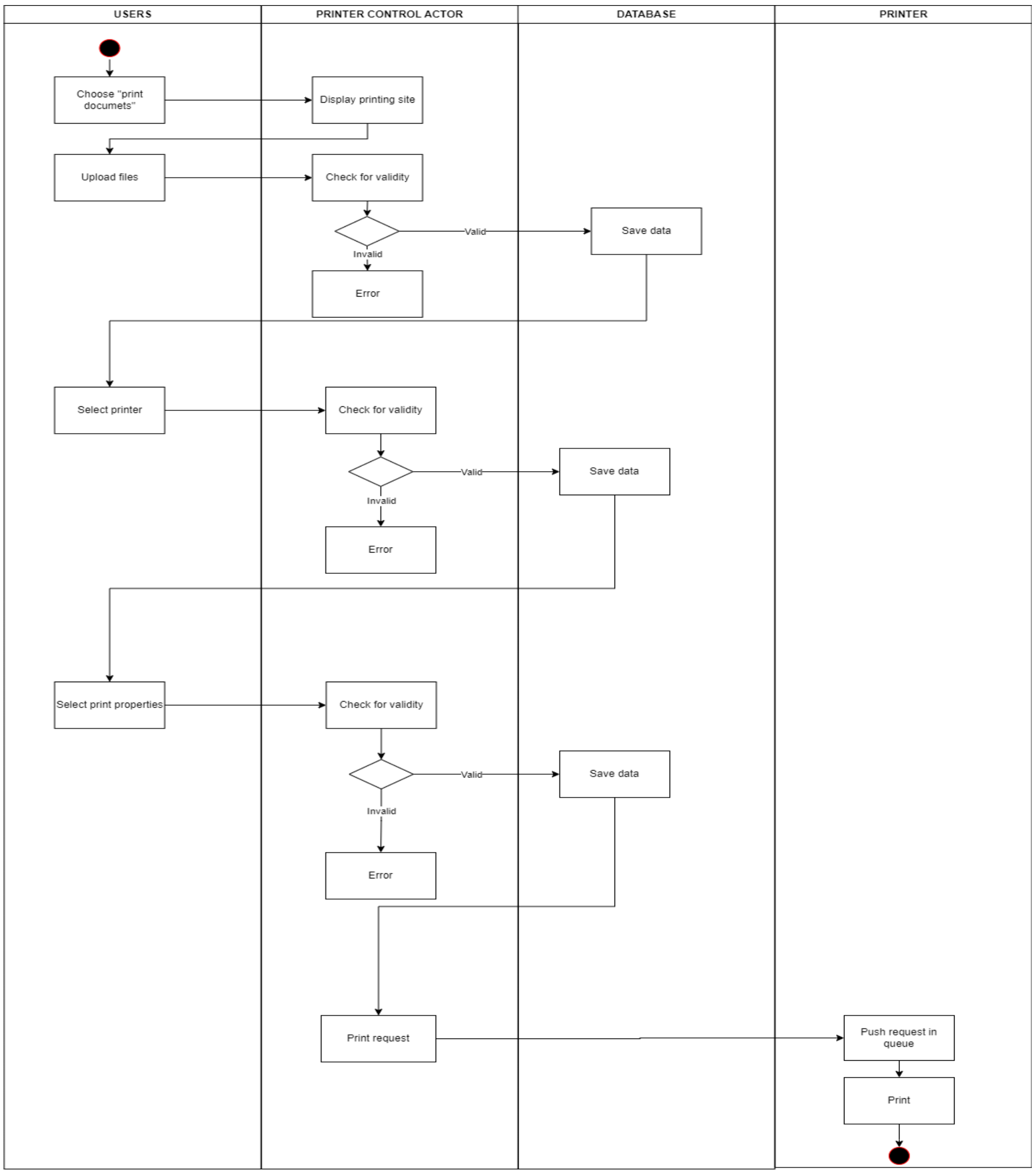


Figure 5.3: Activity diagram of printing documents

6 Sequence diagram

Follow this link for a better view of the diagram: Sequence diagram for the module

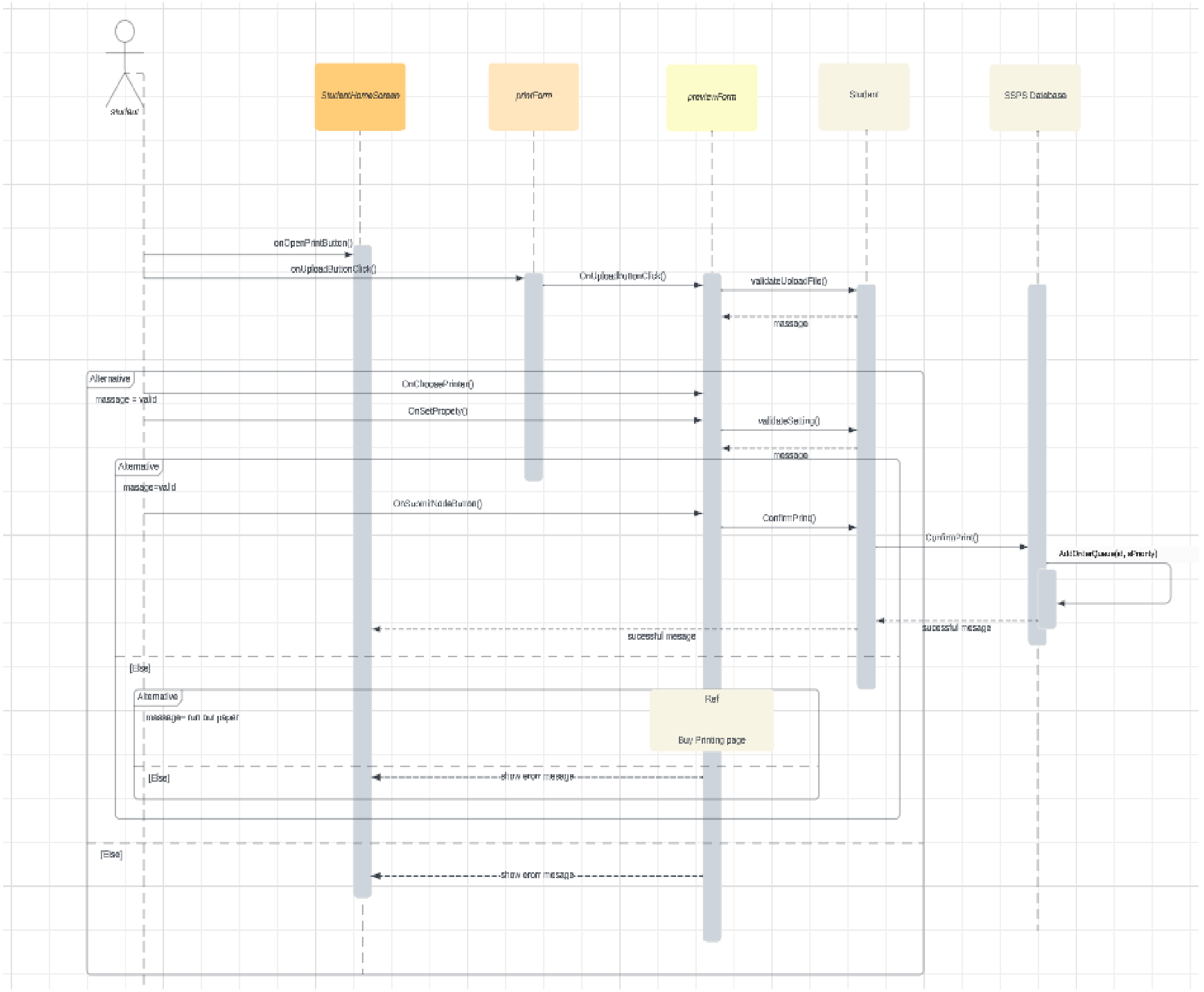


Figure 6.1: Sequence diagram for this module

This sequence diagram is about the Print Documents module. This is used primarily to show the interactions between objects in the sequential order that those interactions occur.

7 Class diagram

We have a class diagram for an overview of Print Documents module, as follows:

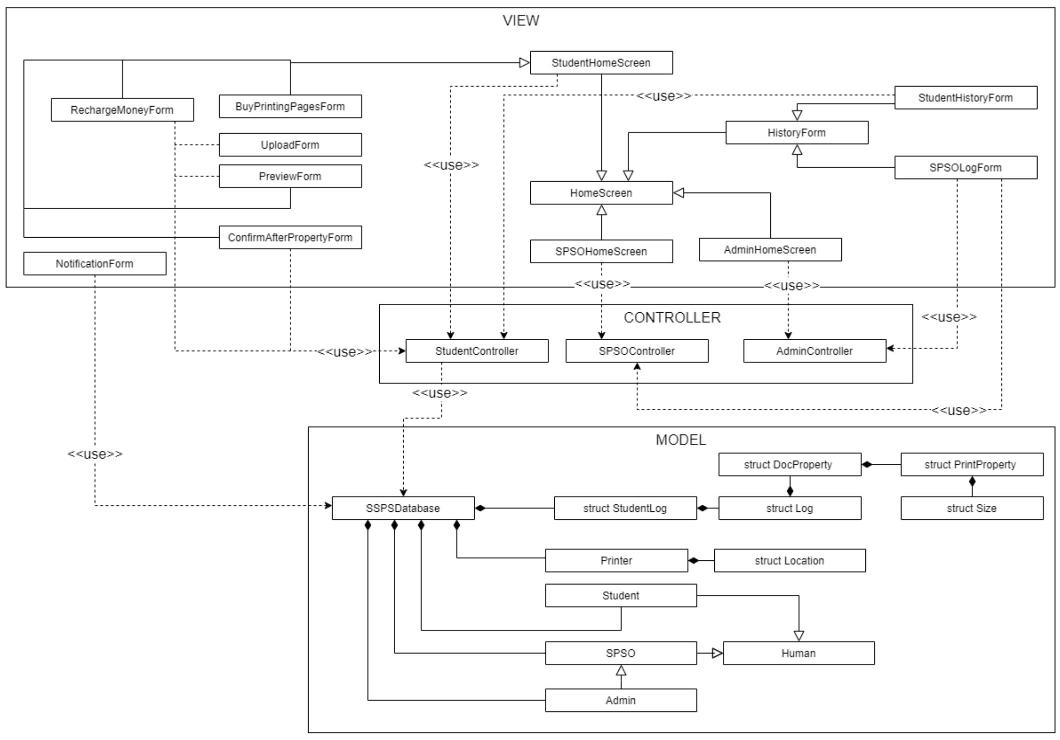


Figure 7.1: Overview Print Documents module class diagram

This class diagram is follow the MVC modul, with 3 main component: Controller, Model, View. Specifically, more details about the entities in the class diagram are as followed in link Print Documents module class diagram or these pictures:

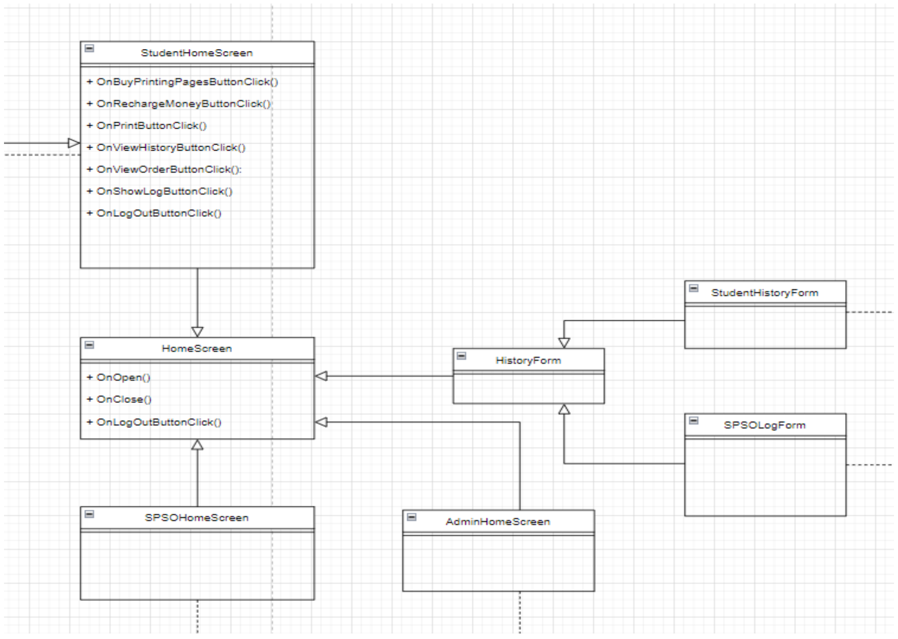


Figure 7.2: View part of module class diagram part 1

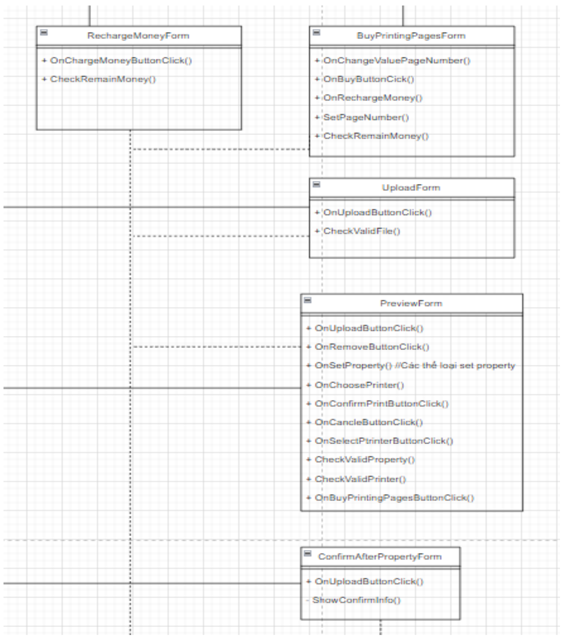


Figure 7.3: View part of module class diagram part 2

In the two images above, they depict the main forms used within this module as "View". These forms are controlled and retrieve data from the "Controller".

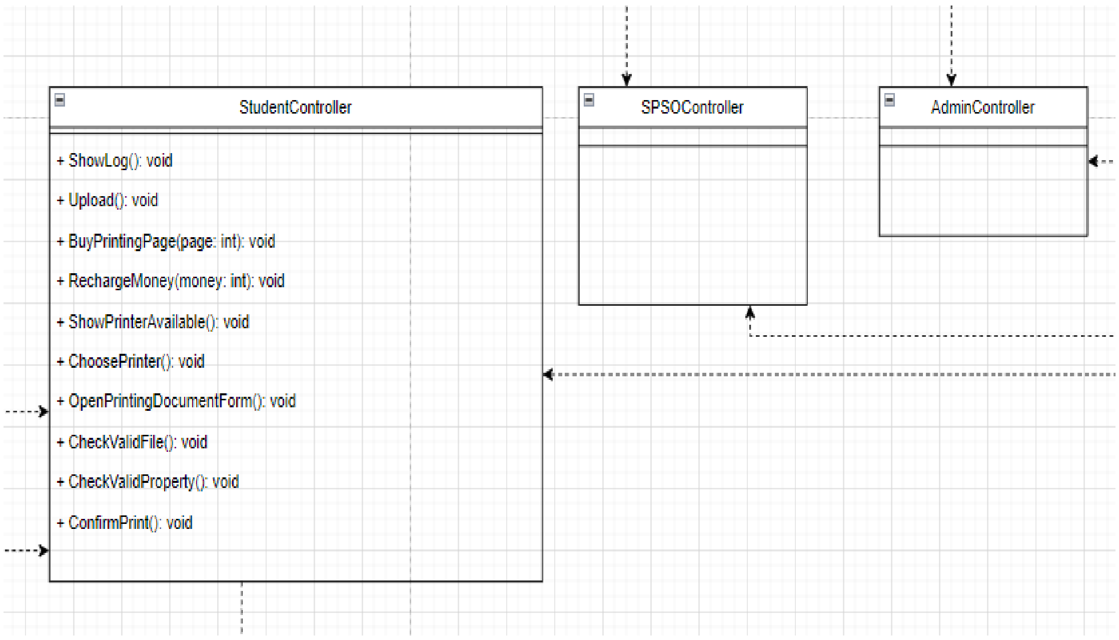

StudentController
+ ShowLog(): void
+ Upload(): void
+ BuyPrintingPage(page: int): void
+ RechargeMoney(money: int): void
+ ShowPrinterAvailable(): void
+ ChoosePrinter(): void
+ OpenPrintingDocumentForm(): void
+ CheckValidFile(): void
+ CheckValidProperty(): void
+ ConfirmPrint(): void
SPSOController
AdminController

Figure 7.4: Controller class in module

The image above represents the "Controller". It manages the "View" and retrieves data provided by the "Model".

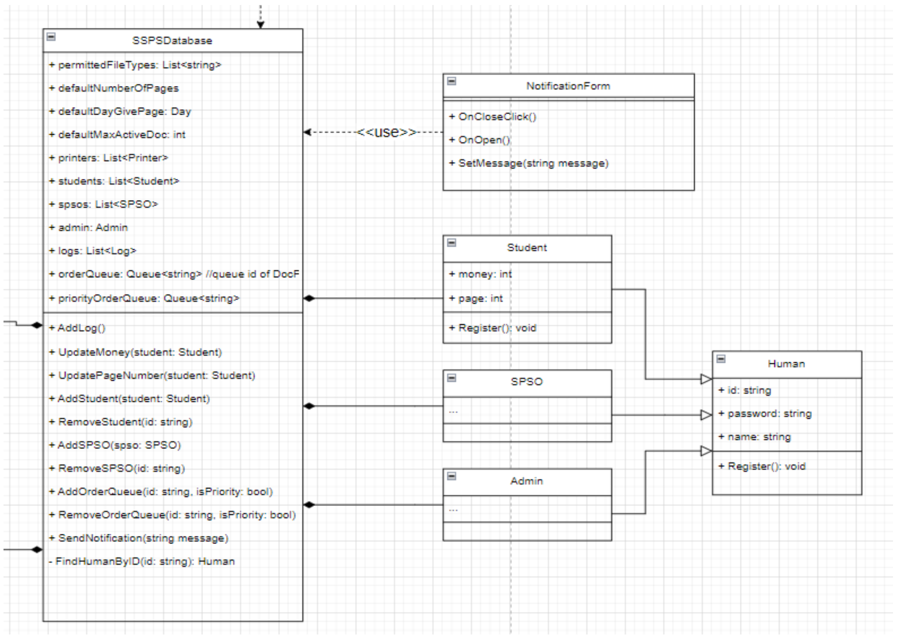

SSPSDatabase
+ permittedFileTypes: List<string>
+ defaultNumberOfPages
+ defaultDayGivePage: Day
+ defaultMaxActiveDoc: int
+ printers: List<Printer>
+ students: List<Student>
+ spsos: List<SPSO>
+ admin: Admin
+ logs: List<Log>
+ orderQueue: Queue<string> //queue id of DocF
+ priorityOrderQueue: Queue<string>
+ AddLog()
+ UpdateMoney(student: Student)
+ UpdatePageNumber(student: Student)
+ AddStudent(student: Student)
+ RemoveStudent(id: string)
+ AddSPSO(spso: SPSO)
+ RemoveSPSO(id: string)
+ AddOrderQueue(id: string, isPriority: bool)
+ RemoveOrderQueue(id: string, isPriority: bool)
+ SendNotification(string message)
- FindHumanByID(id: string): Human
<<use>>
NotificationForm
+ OnCloseClick()
+ OnOpen()
+ SetMessage(string message)
Student
+ money: int
+ page: int
+ Register(): void
SPSO
...
Admin
...
Human
+ id: string
+ password: string
+ name: string
+ Register(): void

Figure 7.5: Database controller class in module

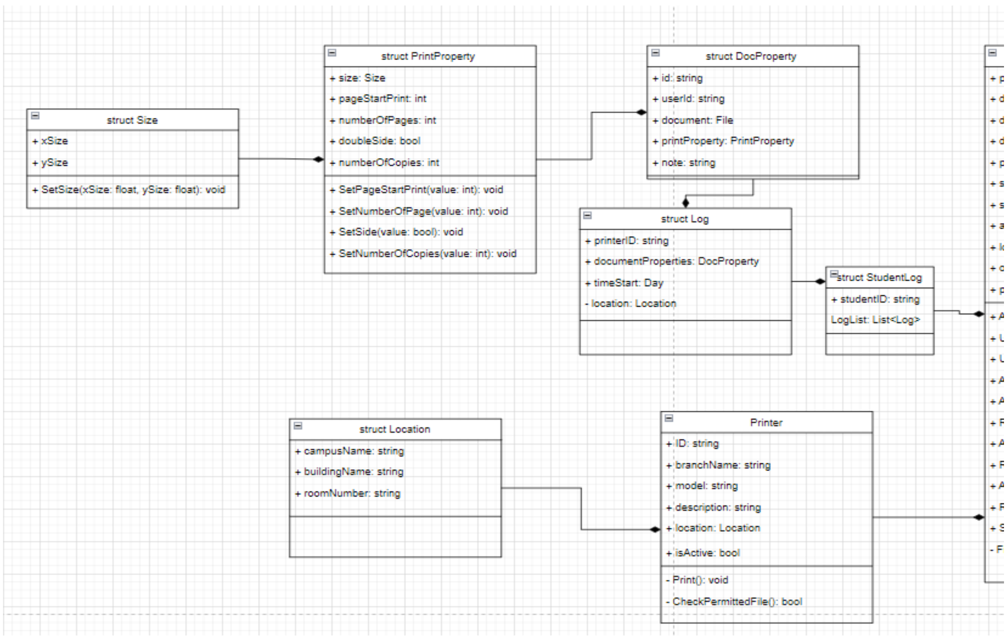


Figure 7.6: Database controller class in module 2

The image above represents the "Model" of class diagram. This component contain the data of app and have some function for "Controller" accession to query the database.

8 Develop MVP 1

Figma link: https://www.figma.com/proto/ayLtoq8uyz7qlRQuna5Ex8/SPSS?type=design&node-id=387-3381&t=NcQWAH9SiJOYAXv7-1&scaling=scale-down&page-id=0%3A1&starting-point-node-id=65%3A647&mode=design
Firstly, the user will start with choosing their role.

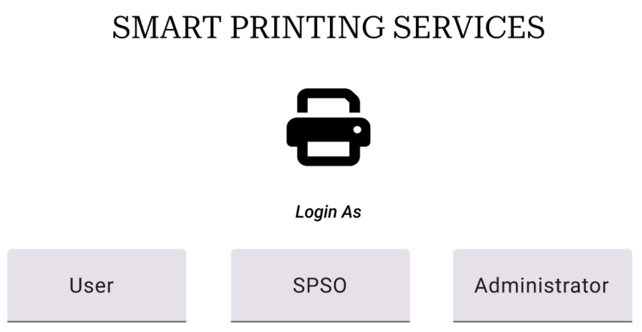


Figure 8.1: Choose role

After choosing their role, the user need to provide their username and password.

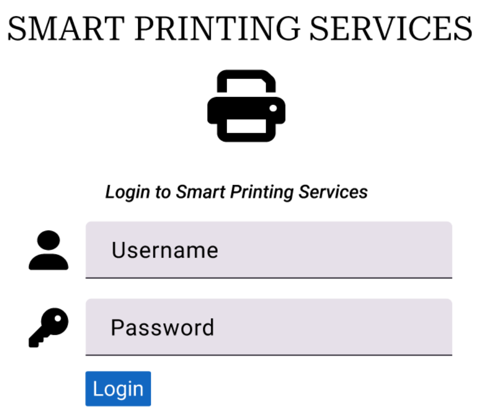
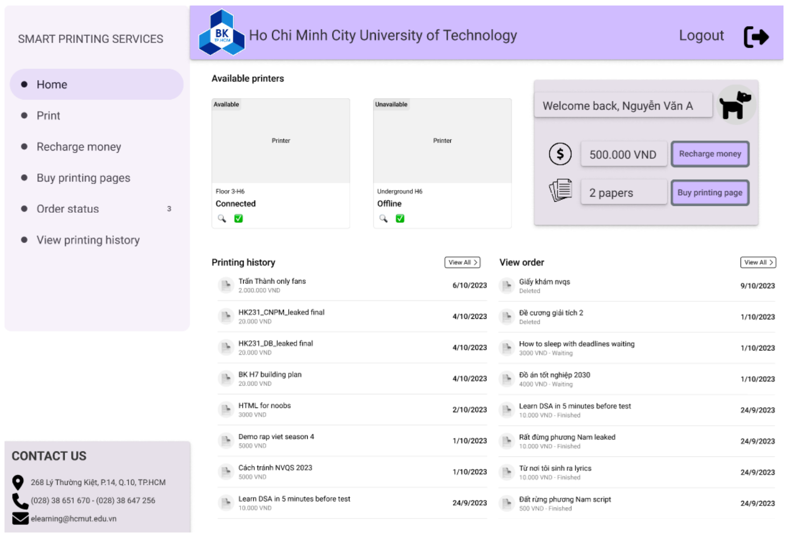


Figure 8.2: Login

8.1 Login as user

8.1.1 Homepage

After logging in successfully, user reaches the homepage.



Available printers

Floor 3-H6
Connected

Underground H6
Offline

Welcome back, Nguyễn Văn A

500.000 VND

2 papers

Printing history

Trấn Thành only fans	2.000.000 VND	6/10/2023
HK231_CNPM_leaked final	20.000 VND	4/10/2023
HK231_DB_leaked final	20.000 VND	4/10/2023
BK H7 building plan	20.000 VND	4/10/2023
HTML for noobs	3000 VND	2/10/2023
Demo rap viet season 4	5000 VND	1/10/2023
Cách tránh NVQS 2023	5000 VND	1/10/2023
Learn DSA in 5 minutes before test	10.000 VND	24/9/2023

View order

Giấy khám nvqs	Deleted	9/10/2023
Đề cương giải tích 2	Deleted	1/10/2023
How to sleep with deadlines waiting	3000 VND - Waiting	1/10/2023
Đồ án tốt nghiệp 2030	4000 VND - Waiting	1/10/2023
Learn DSA in 5 minutes before test	10.000 VND - Finished	24/9/2023
Rất đừng phương Nam leaked	10.000 VND - Finished	24/9/2023
Từ nơi tôi sinh ra lyrics	10.000 VND - Finished	24/9/2023
Đất rừng phương Nam script	500 VND - Finished	24/9/2023

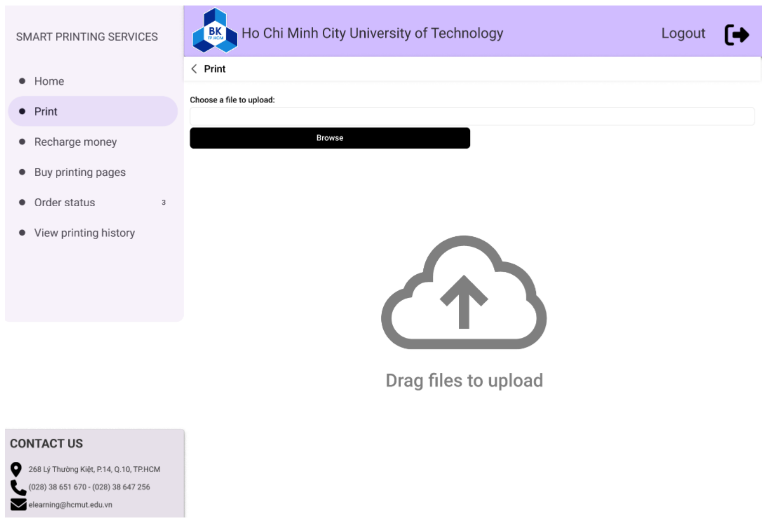
CONTACT US
268 Lý Thường Kiệt, P.14, Q.10, TP.HCM
(028) 38 651 670 - (028) 38 647 256
elearning@hcmut.edu.vn

Figure 8.3: User homepage

In this part, we will show how users can perform some activites from the homepage.

8.1.2 Print

The user can print some documents by clicking on the "Print" button on the left.

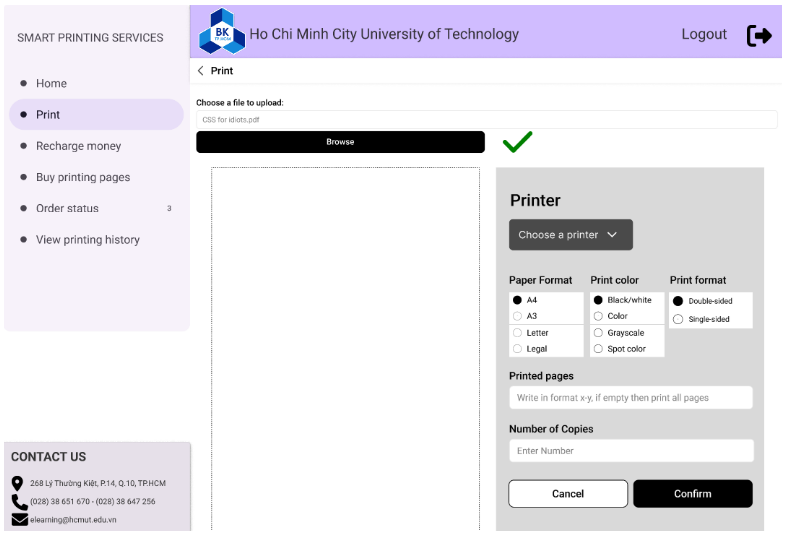


Choose a file to upload:

CONTACT US
268 Lý Thường Kiệt, P.14, Q.10, TP.HCM
(028) 38 651 670 - (028) 38 647 256
elearning@hcmut.edu.vn

Figure 8.4: Print

Next, the user can upload their document and wait for the system to validate. After the system validate the document, user can choose the printing properties, select the printed pages, number of copies and see the printing preview of the document. Finally, the user click on "Confirm" to confirm or "Cancel" if they do not want to continue printing.



Choose a file to upload: CSS for idiots.pdf

Paper Format	Print color	Print format
● A4	● Black/white	● Double-sided
○ A3	○ Color	○ Single-sided
○ Letter	○ Grayscale	
○ Legal	○ Spot color	

Printed pages: Write in format x-y, if empty then print all pages

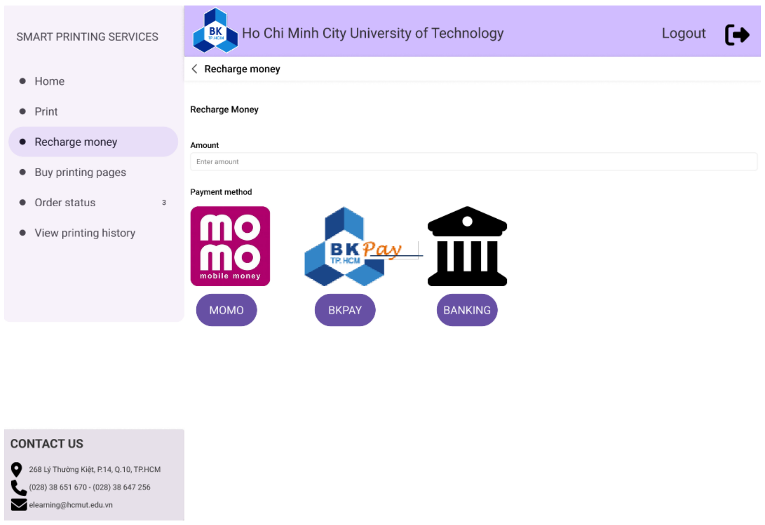
Number of Copies: Enter Number

268 Lý Thường Kiệt, P.14, Q.10, TP.HCM
(028) 38 651 670 - (028) 38 647 256
elearning@hcmut.edu.vn

Figure 8.5: Choose printing property

8.1.3 Recharge money

The user can recharge money by clicking on "Recharge money" button on the left.



Recharge Money

Amount: Enter amount

Payment method

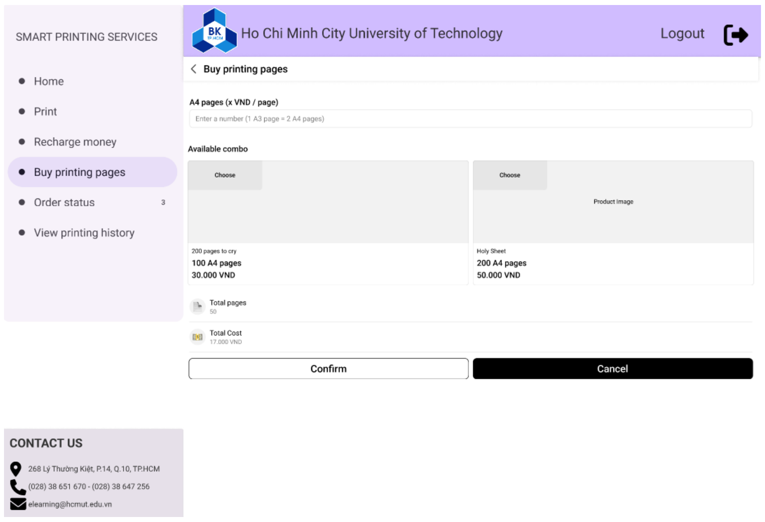
268 Lý Thường Kiệt, P.14, Q.10, TP.HCM
(028) 38 651 670 - (028) 38 647 256
elearning@hcmut.edu.vn

Figure 8.6: Recharge money

Next, the user can enter the amount they want to recharge and choose the payment method.

8.1.4 Buy printing pages

The user can buy some printing pages by clicking on "Buy printing pages" button on the left.



200 pages to cry
100 A4 pages
30.000 VND

Holy Sheet
200 A4 pages
50.000 VND

Total pages
50

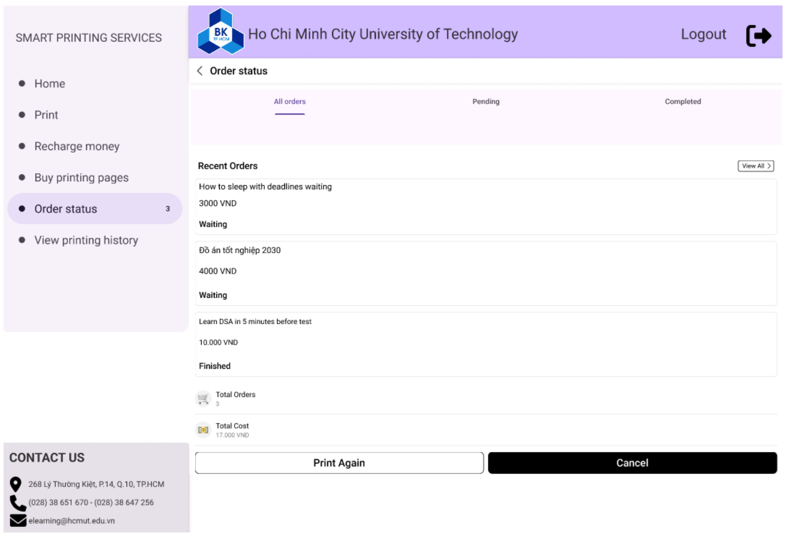
Total Cost
17.000 VND

Figure 8.7: Buy printing pages

Next, the user can enter the number of pages they want to buy and confirm or cancel it.

8.1.5 Order status

The user can view the order status by clicking on "Order status" button on the left.



Recent Orders

How to sleep with deadlines waiting
3000 VND
Waiting

Đồ án tốt nghiệp 2030
4000 VND
Waiting

Learn DSA in 5 minutes before test
10.000 VND
Finished

Total Orders
3

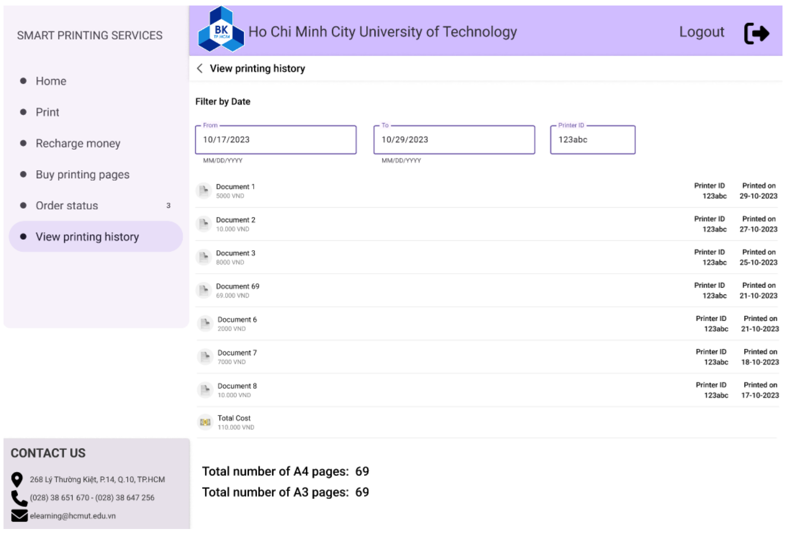
Total Cost
17.000 VND

Figure 8.8: View order status

Here, the user can see their order queue with its quantity, price and status.

8.1.6 View printing history

The user can view the printing history by clicking on "View printing history" button on the left.



View printing history

Filter by Date

From: 10/17/2023 (MM/DD/YYYY)
To: 10/29/2023 (MM/DD/YYYY)
Printer ID: 123abc

		Printer ID	Printed on
Document 1	5000 VND	123abc	29-10-2023
Document 2	10.000 VND	123abc	27-10-2023
Document 3	8000 VND	123abc	25-10-2023
Document 69	69.000 VND	123abc	21-10-2023
Document 6	2000 VND	123abc	21-10-2023
Document 7	7000 VND	123abc	18-10-2023
Document 8	10.000 VND	123abc	17-10-2023
Total Cost	110.000 VND		

Total number of A4 pages: 69
Total number of A3 pages: 69

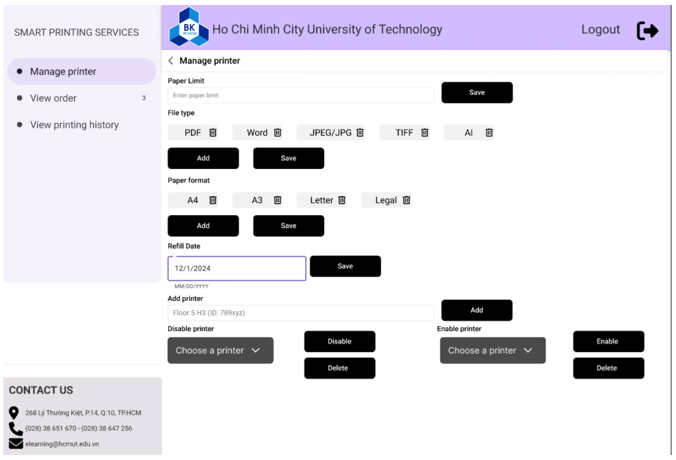
Figure 8.9: View printing history

Here, the user can see their printing history within a time period for all or chosen printers and a summary of number of printed pages for each page size.

8.2 Login as SPSO

8.2.1 Manage printer

The SPSO can manage the printers by clicking on "Manage printer" button on the left.



Manage printer

Paper Limit: Enter paper limit

File type: PDF, Word, JPEG/JPG, TIFF, AI

Paper format: A4, A3, Letter, Legal

Refill Date: 12/1/2024 (MM/DD/YYYY)

Add printer: Floor 5 H3 (ID: 789xyz)

Disable printer: Choose a printer

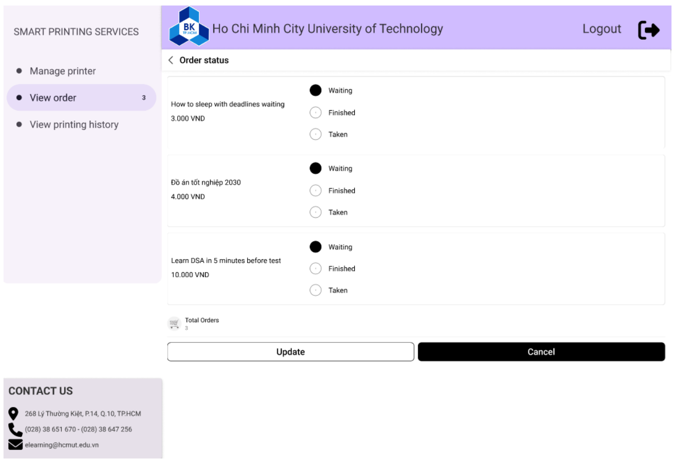
Enable printer: Choose a printer

Figure 8.10: Manage printer

Here, the SPSO can change the default number of pages, the permitted file types and page format accepted by the system, the refill date that the system will give the default number of pages to all students and add or delete or disable some chosen printers.

8.2.2 View order

The SPSO can view the order by clicking on "View order" button on the left.



How to sleep with deadlines waiting 3.000 VND	[✓ Waiting] Finished Taken
Đồ án tốt nghiệp 2030 4.000 VND	[✓ Waiting] Finished Taken
Learn DSA in 5 minutes before test 10.000 VND	[✓ Waiting] Finished Taken

Total Orders
3

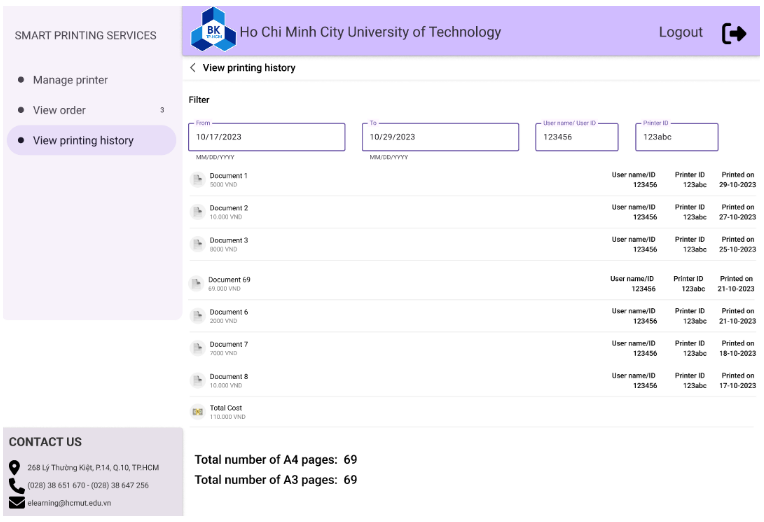
268 Lý Thường Kiệt, P.14, Q.10, TP.HCM
(028) 38 651 670 - (028) 38 647 256
elearning@hcmut.edu.vn

Figure 8.11: View order

Here, the SPSO can see the order queue and can update the status of each order.

8.2.3 View printing history

The SPSO can view the printing history by clicking on "View printing history" button on the left.



From: 10/17/2023 (MM/DD/YYYY)
To: 10/29/2023 (MM/DD/YYYY)
User name/ User ID: 123456
Printer ID: 123abc

	User name/ID	Printer ID	Printed on
Document 1 5000 VND	123456	123abc	29-10-2023
Document 2 10.000 VND	123456	123abc	27-10-2023
Document 3 8000 VND	123456	123abc	25-10-2023
Document 69 69.000 VND	123456	123abc	21-10-2023
Document 6 2000 VND	123456	123abc	21-10-2023
Document 7 7000 VND	123456	123abc	18-10-2023
Document 8 10.000 VND	123456	123abc	17-10-2023

Total Cost
110.000 VND

268 Lý Thường Kiệt, P.14, Q.10, TP.HCM
(028) 38 651 670 - (028) 38 647 256
elearning@hcmut.edu.vn

Total number of A4 pages: 69
Total number of A3 pages: 69

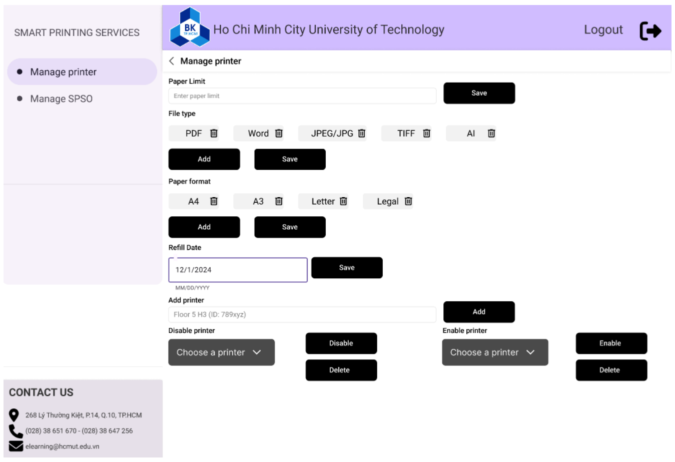
Figure 8.12: View printing history

Here, the user can see their printing history of a student within a time period for all or chosen printers and a summary of number of printed pages for each page size.

8.3 Login as administrator

8.3.1 Manage printer

Like SPSO, the administrator can manage the printers by clicking on "Manage printer" button on the left.



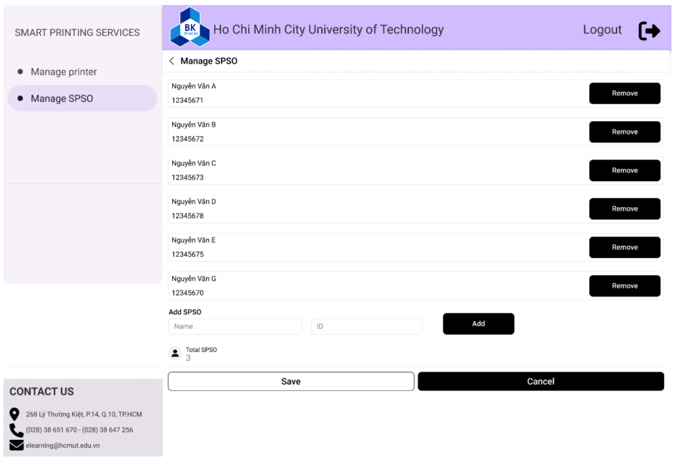
Refill Date: 12/1/2024

CONTACT US
268 Lý Thường Kiệt, P.14, Q.10, TP.HCM
(028) 38 651 670 - (028) 38 647 256
elearning@hcmut.edu.vn

Figure 8.13: Manage printer

8.3.2 Manage SPSO

The administrator can manage the SPSOs by clicking on "Manage SPSO" button on the left.



Nguyễn Văn A	12345671
Nguyễn Văn B	12345672
Nguyễn Văn C	12345673
Nguyễn Văn D	12345678
Nguyễn Văn E	12345675
Nguyễn Văn G	12345670

Total SPSO: 3

CONTACT US
268 Lý Thường Kiệt, P.14, Q.10, TP.HCM
(028) 38 651 670 - (028) 38 647 256
elearning@hcmut.edu.vn

Figure 8.14: Manage SPSO

Here, the administrator can view the list of SPSOs and can remove some existing SPSOs or add some new SPSOs.

9 Architecture Design

Our system is designed and implemented using the Model-View-Controller (MVC) architectural pattern. This approach provides a clear separation of responsibilities, with the Model handling data and business logic, the View focusing on the user interface, and the Controller managing the interaction between the two. By using the MVC model, our system benefits from modularity, code reusability, and easier maintenance, allowing for efficient development and scalability.
In the MVC model, the system is divided into three main components: the Model, the View, and the Controller.

- The Model represents the data and the business logic of the application. It encapsulates the data structures, algorithms, and database interactions.
- The View is responsible for presenting the data to the user and providing the user interface. It handles the visual representation and rendering of the information.
- The Controller acts as an intermediary between the Model and the View. It receives user input, processes it, interacts with the Model to update the data, and then updates the View accordingly.

One of the key advantages of using the MVC model is its ability to promote modularity and maintainability. By separating the concerns, each component can be developed and tested independently, allowing for easier maintenance and code reuse. Changes to one component do not necessarily impact the others, making it easier to modify and extend the system. This separation also enables different teams or developers to work on different components simultaneously, improving development efficiency.
Because the Model and the Controller are seperated from the specific presentation layer, different views can be implemented to cater to different user interfaces or platforms. This flexibility allows for the creation of web-based interfaces, mobile applications, or APIs, all utilizing the same underlying business logic. The separation of concerns in the MVC model also enables better testability. Each component can be tested independently, making it easier to write unit tests and perform integration testing.

9.1 Presentation strategy

For our User Interface (UI), we will utilize a combination of JavaScript, HTML, and CSS to create an interactive and visually appealing frontend. To ensure a responsive and consistent design across different devices and browsers, we will leverage the Bootstrap framework. Bootstrap provides a set of pre-built UI components and responsive grid system, allowing us to quickly develop a professional-looking interface. We will focus on delivering a user-friendly and intuitive experience, incorporating modern design principles and best practices. For more details on implementing the UI using JavaScript, HTML, CSS, and Bootstrap, you can refer to the official documentation and tutorials available at:

- JavaScript
- HTML
- CSS
- Bootstrap

9.2 Data storage approach

To store our data, we have chosen Microsoft SQL Server as our relational database management system (RDBMS). SQL Server offers advanced features like transaction support, data integrity constraints, and powerful querying capabilities. We can take advantage of SQL Server's performance optimizations and indexing techniques to ensure efficient data retrieval and manipulation. Additionally, SQL Server integrates well with the .NET ecosystem, making it a suitable choice for our ASP.NET application. For more information and guidance on working with SQL Server, you can refer to the official documentation provided by Microsoft:

- Microsoft SQL Server Documentation

9.3 External services and API management

We will implement a RESTful API using ASP.NET. ASP.NET provides a strong framework for building web APIs. We will design our API endpoints to follow REST principles and use standard HTTP methods and status codes. ASP.NET offers features like routing and authentication/authorization mechanisms to simplify API development. For detailed guidance on building RESTful APIs with ASP.NET, you can refer to the official Microsoft documentation:

- ASP.NET Core Web API Documentation

9.4 Deployment Diagram

Follow this link for a better view of the deployment diagram: Deployment diagram

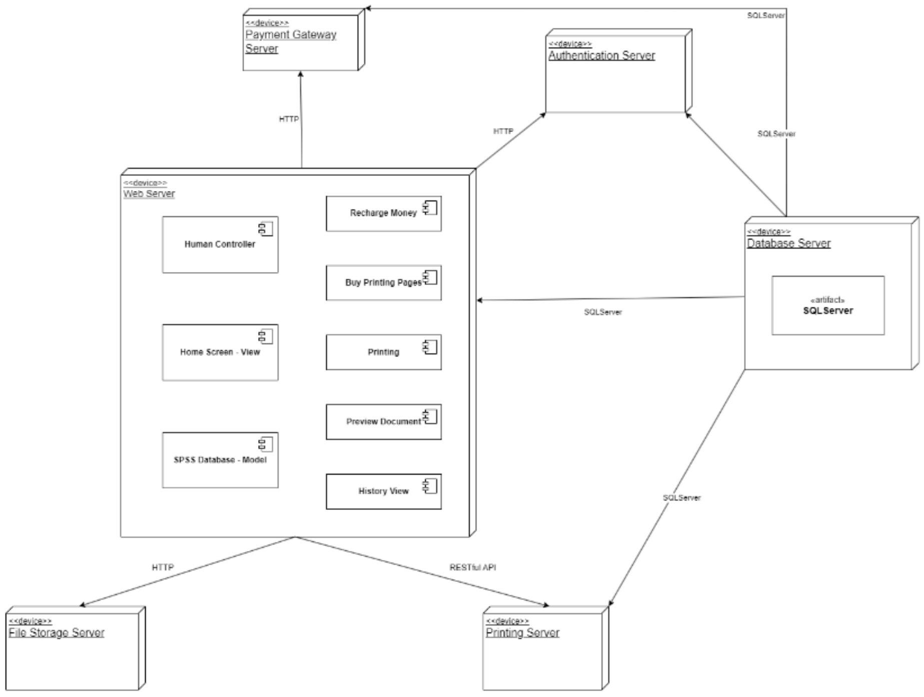


Figure 9.1: Deployment Diagram

- Web Server: Hosts the core application logic, including the Model, View, and Controller components, responsible for handling user requests, data processing, and rendering the user interface.
- File Storage Server: Provides a dedicated space for storing and managing uploaded documents and files, which can be accessed by the web application and printers as needed.
- Printing Server: Handles print requests received from the web application, processing and generating printed output based on the provided documents and configurations.
- Authentication Server: Manages user authentication and authorization, ensuring secure access to the web application by validating user credentials and providing appropriate access permissions.
- Payment Gateway Server: Handles payment processing, securely handling financial transactions between the web application and external payment systems or services (BKPay).
- Database Server: Stores and manages the application's data, providing a reliable and scalable storage solution for the web application and other associated servers.

9.5 Box-line Diagram

Follow this link for a better view of the box-line diagram: Box-line diagram

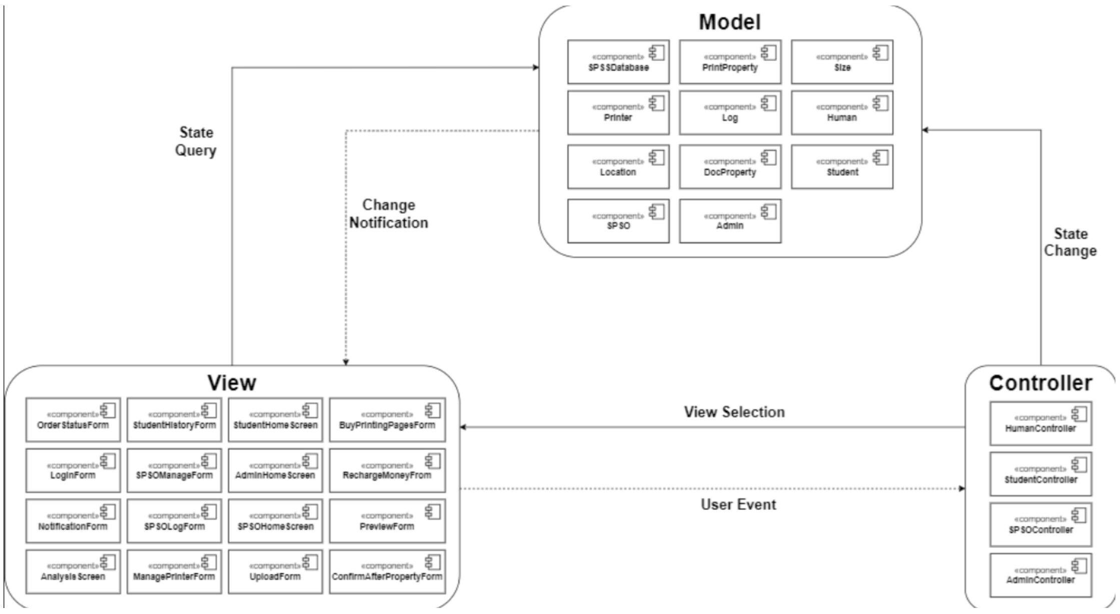

Model
«component» SPSSDatabase
«component» PrintProperty
«component» Size
«component» Printer
«component» Log
«component» Human
«component» Location
«component» DocProperty
«component» Student
«component» SPSO
«component» Admin
State Query
Change Notification
State Change
View
«component» OrderStatusForm
«component» StudentHistoryForm
«component» StudentHomeScreen
«component» BuyPrintingPagesForm
«component» LoginForm
«component» SPSOManageForm
«component» AdminHomeScreen
«component» RechargeMoneyFrom
«component» NotificationForm
«component» SPSOLogForm
«component» SPSOHomeScreen
«component» PreviewForm
«component» AnalysisScreen
«component» ManagePrinterForm
«component» UploadForm
«component» ConfirmAfterPropertyForm
View Selection
User Event
Controller
«component» HumanController
«component» StudentController
«component» SPSOController
«component» AdminController

Figure 9.2: Box-line Diagram

10 Component diagram for module

10.1 Draw component Diagram for chosen module

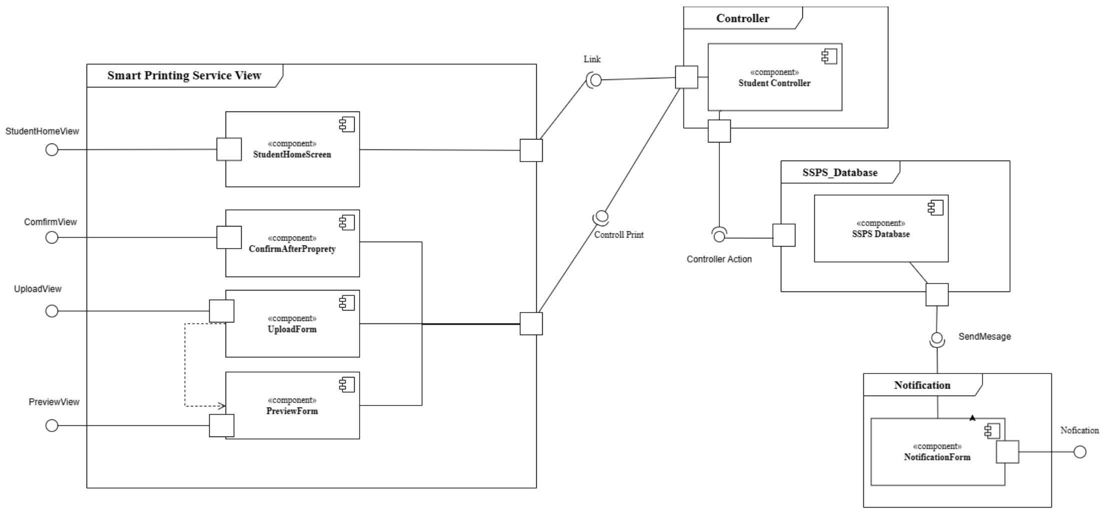


Figure 10.1: Component Diagram

10.2 Description

The diagram can be separated into 4 different layers:

10.2.1 Smart Printing View Layer

- **SmartHomeScreen:** Use the Interface "Link" provided by the Student Controller, including the following functions:
  - **Showlog():** Switch to the print history view screen.
  - **OpenUploadForm():** Switch to the screen to upload files to the web for printing.
  - **BuyPrintingPage():** Switch to the screen to buy more printing paper.
  - **RechargeMoney():** Switch to the account deposit screen.
- **ConfirmAfterProperty:** Use the "Controller Print" Interface provided by the Student Controller, including the following functions:
  - **ConfirmPrint():** User confirms print request. The order will be sent to the corresponding printer.
- **UpLoadForm:** Use the "Controller Print" Interface provided by the Student Controller, including the following functions:
  - **Upload():** upload files to the system.
  - **CheckValidFile():** Checks to see if the uploaded file type is valid with the rules available in the system. If it is valid, pass it on.

- **Preview:** Use the "Controller Print" Interface provided by the Student Controller, including the following functions:
  - **ShowPrinterAvailble():** List of available printers in the system.
  - **ChoosePrinter():** User selects printer to print.
  - **CheckValidProperty():** Check the properties the user has selected.
  - **ConfirmPrint():** the user confirms, moving to the screen provided from the ConfirmAfterProperty component.

10.2.2 Controller Layer

- **Student Controller:** Use Interface Controller Action provided by SSPS_Database (Layer will be presented next) including functions:
  - **AddLog():** for completed orders, this function will be called to add that order to the transaction history.
  - **UpdateMoney(student: Student)**When a student deposits money into the system or uses money to buy paper, this function will be called to update the amount in the student's account.
  - **UpdatePageNumber(student: Student):** When a student uses money to buy more paper or when using paper to pay for a printing order, this function will be called to update the number of remaining pages in the student's account.
  - **AssDocProperty(docProperty: Docproperty):**
  - **AddStudent(student: Student):** Add a new student who can use the service to the system.
  - **RemoveStudent(id: string):** Delete an existing student in the .system
  - **AddSPSO(spso: SPSO):** Add a SPSO to the system.
  - **RemoveSPSO(id: string):** Remove a SPSO from the system.
  - **AddOrderQueue(id: string, isPriority: bool):** Add an order to the queue corresponding to fast or normal printing mode.
  - **RemoveOrderQueue(id: string, isPriority: bool):** Remove a print order from the corresponding queue.
  - **findHumanById(id: string):** Find a person in the system by ID.

10.2.3 SSPS_Database Layer

- **SSPS_Database:**
  - **SendNotification(string: message):** This function will also provide a string message that SPSS wants to send to the screen.

10.2.4 Interfaces provided by Smart Printing Service views

- **StudentHomeView**(provided from StudentHomScreen): This is an interactive user interface, used for navigation by calling corresponding functions in the student controller when users click on buttons.

- **ConfirmView**(provided from ConfirmAfterPrint): This is the interface that appears after the user has selected the attributes and pressed the confirm button, helping the user check the selected attributes again.
- **uploadView**(provided from UploadForm): This is the interface that appears when the user clicks the upload button on studentHomeView, this screen provides a file upload button.
- **previewView**(provided from PreviewForm): This is the screen that appears after successfully uploading a document. Users can select properties for their print order here such as: Printer, paper size, number of copies, number printed pages, regular printing or fast printing... . Then there is a confirm button to confirm your choices.

Sumary: All of the above views use interfaces provided by the controller to be able to call the corresponding function when there is any event for the user to trigger.