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

SOFTWARE ENGINEERING (CO3001)

A SMART PRINTING SERVICE FOR STUDENTS AT HCMUT

GROUP L01 - SEMESTER 231

GROUP NAME: L01_10

Instructors:

Le Dinh Thuan
Bui Cong Tuan
Mai Duc Trung

Students:

Nguyen Van Khanh Nhan - 2010480
Nguyen Tan Dung - 2110098
Vo Thi Hoang Yen - 2115376
Nguyen Huu Thong - 2114917
Lam Pham Trong Phuc - 2011844

Ho Chi Minh City - 09/2023

List of Tables

List of Figures

Content

A smart printing service for students at HCMUT

The university is intent to build a Student Smart Printing Service (HCMUT_SSPS) for serving students in its campuses to print their documents.

The system consists of some printers around the campuses. Each printer has ID, brand/-manufacturer name, printer model, short description, and the location (campus name, building name, and room number).

The system allows a student to print a document by uploading a document file onto the system, choose a printer, and specifying the printing properties such as paper size, pages (of the file) to be printed, one-/double-sided, number of copies, etc. The permitted file types are limitted and configured by the Student Printing Service Officer (SPSO).

The system has to log the printing actions for all students, including student ID, printer ID, file name, printing start and end time, number of pages for each page size.

The system allows the SPSO to view the printing history (log) of all students or a student for a time period (date to date) and for all or some printers. Of course, a student can also view his/her printing log for a time period together with a summary of number of printed pages for each page size.

For each semester, the university give each student a default number of A4-size pages for printing. Students are allowed to buy some more using the feature Buy Printing Pages of the system and pay the amount through some online payment system like the BKPay system of the university. The system only allow a student to print some number of pages when it does not exeed his/her account (page) balance. Note that, one A3 page is equivalent to two A4 pages.

The SPSO has a feature to manage printers such as add/enable/disable a printer.

The SPSO also has a feature to manage other configuration of the system such as changing the default number of pages, the dates that the system will give the default number of pages to all students, the permitted file types accepted by the system.

The reports of the using of the printing system are generated automatically at the end of each month and each year and are stored in the system, and can be viewed by the SPSO anytime.

All users have to be authenticated by the HCMUT_SSO authentication service before using the system.

The system is provided through a web-based app or a mobile app.

1 Task 1: Requirement elicitation

1.1 Task 1.1

1.1.1 Domain context

The HCMUT Students Smart Printing Service (HCMUT_SSPS) is designed to cater to the printing needs of Students at HCMUT (Ho Chi Minh City University of Technology). The system aims to provide a convenient and efficient printing solution within the university campuses. By utilizing a network of printers located throughout the campuses, Students can easily upload their documents and select printing options according to their preferences.

1.1.2 Relevant stakeholders and their needs

- **Students**: The primary stakeholders of HCMUT_SSPS are the Students themselves. They rely on the system to conveniently print their academic documents, assignments, and research papers. Students seek a user-friendly interface that allows them to easily upload files, select printers, and specify printing options. They also require access to their printing history and the ability to purchase additional printing pages when needed.
- **Software Students Printing Service Officer (SSPSO)**: The SSPSO plays a crucial role in managing and maintaining the printing service. They are responsible for configuring the system settings and monitoring the overall printing process. The SSPSO needs a comprehensive overview of the printing history and usage statistics for efficient management and decision-making.
- **Hardware Students Printing Service Officer (HSPSO)**: The HSPSO also plays a crucial role in managing and maintaining the printing service. They are responsible for managing printers such as adding/enabling/disabling a printer.
- **University Administration**: The university administration holds an interest in the HCMUT_SSPS as it enhances the overall printing infrastructure and services offered to Students. They seek a system that provides reliable and accurate reporting on printing usage, facilitates cost management, and ensures a seamless printing experience for Students.

1.1.3 Benefits of HCMUT-SSPS for each stakeholder

- **Students**: HCMUT_SSPS brings convenience and efficiency to Students' printing needs. They can easily upload documents, choose printers, and customize printing options. The system provides a transparent printing history, allowing students to track their usage and manage their printing resources effectively. Additionally, the ability to purchase additional printing pages ensures flexibility and meets their specific printing requirements.
- **Software Students Printing Service Officer (SSPSO)**: HCMUT_SSPS empowers the SSPSO with comprehensive control and management capabilities. The system provides a centralized view of printing activities, enabling the SPSO to monitor usage and track printing history effectively. With access to detailed reports and usage statistics, the SSPSO can make informed decisions to optimize the printing service.
- **Hardware Students Printing Service Officer (HSPSO)**: HSPSO can base on detailed reports and usage statistics, HSPSP can make informed decisions to optimize the printing service such as adding/enabling/disabling a printer.
- **University Administration**: HCMUT_SSPS offers several benefits to the university administration. The system provides accurate usage data, allowing administrators to monitor printing trends, allocate resources efficiently, and plan for future infrastructure upgrades. The centralized management features enable streamlined administration and cost control, improving operational efficiency and enhancing Student services.

1.2 Task 1.2

1.2.1 Functional Requirements

- **Whole system**

  1. **System Configuration**
     - The system should allow administrators to configure system-wide settings, including default paper sizes, printing quotas, and permitted file types.
     - Only authenticated administrators should have access to system configuration options.
  2. **User Authentication**

- The system should integrate with the HCMUT_SSO authentication service to authenticate all users before granting access to the system.
- Users should be required to log in using their university credentials.

3. **Logging and Reporting**
   - The system should maintain detailed logs of all printing actions, including Student ID, printer ID, file name, printing start and end times, and the number of pages printed for each page size.
   - The system should generate monthly and yearly reports on printing system usage for review by administrators.

- **Students**

1. The system should allow Students to upload document files for printing.
2. The system should allow Students to cancel print requests before they are processed.
3. The system should provide a list of available printers for Students.
4. The system should allow Students to specify printing properties, including paper size, single-/double-sided printing, number of copies, etc.
5. The system should restrict permitted file types for printing, as configured by the Software Students Printing Service Officer (SSPSO).
6. The system should allow Students to view their printing logs for a specified period, along with a summary of the number of printed pages for each page size.
7. The system should only permit Students to print pages when their usage does not exceed their account (page) balance.

- **Software Students Printing Service Officer (SSPSO)**

1. The system should allow the SSPSO to view the printing history (log) of all Students or specific Students within a specified period.
2. The system should allow the SSPSO to configure various aspects, such as the default number of pages, distribution dates for default pages, and permitted file types.
3. The system should generate monthly and yearly reports on the usage of the printing system, which can be accessed by the SSPSO.

4. The system should allow the SSPSO to block Students from using the system.

- **Hardware Students Printing Service Officer (HSPSO)**

1. The system should enable the HSPSO to manage printers, including adding, enabling, or disabling printers.

- **University Admin (UA)**

1. The system should allow the UA to manage printers, including adding new printers to the system, activating printers for use, and disabling printers when necessary to prevent usage during unstable or malfunctioning conditions.
2. The UA should have access to usage reports within the printing system.

1.2.2 Non-functional Requirements

- **Cross-platform**
  - **Description:** The system should be accessible and fully functional on both web-based and mobile applications.
  - **Rationale:** To ensure that Students and administrators can access and use the system seamlessly on various platforms, enhancing accessibility and user experience.
- **Response Time**
  - **Metric:** Response time measures the time taken for the system to respond to a user's action, such as clicking a button.
  - **Measurement Unit:** Milliseconds (ms).
  - **Tool:** Use tools like Google PageSpeed Insights or WebPageTest to measure and analyze page load times and response times for web pages.
- **Scalability**
  - **Metric:** Scalability assesses the system's ability to handle a maximum number of concurrent users before performance degrades.
  - **Measurement Unit:** Number of concurrent users.
  - **Tool:** Apache JMeter or LoadRunner can be used to perform load testing and measure scalability by simulating multiple concurrent users.

- **Security**
  - **Metric:** Security measurement includes the detection and reporting of unauthorized access or security breaches.
  - **Measurement Unit:** Percentage (%) or count of security incidents.
  - **Tool:** Tools like OWASP ZAP and Nessus can be employed to perform security assessments and detect security vulnerabilities.
- **Accessibility**
  - **Metric:** Accessibility measurement assesses compliance with accessibility standards, such as WCAG.
  - **Measurement Unit:** Score (e.g., based on WCAG scoring).
  - **Tool:** Use tools like WAVE or Axe DevTools to evaluate and measure accessibility compliance in the user interface.
- **Availability**
  - **Metric:** Availability measures the uptime and downtime of the system over a specified period.
  - **Measurement Unit:** Hours (h) or Minutes (m).
  - **Tool:** Monitoring tools like Uptime Robot or Pingdom can track system uptime and report downtime incidents.
- **Resource Utilization**
  - **Metric:** Resource utilization assesses the usage of system resources, including CPU, memory, and disk space during operation.
  - **Measurement Unit:** Percentage (%) or Gigabytes (GB).
  - **Tool:** Tools like New Relic or AppDynamics can monitor resource utilization over time.
- **Storage Capacity**
  - **Metric:** Storage capacity measures the amount of storage space used and the ratio of usage to maximum capacity.
  - **Measurement Unit:** Gigabytes (GB).

  - **Tool:** Monitoring tools like Nagios or Prometheus can track storage capacity and utilization.

- **Recovery**
  - **Metric:** Recovery time measures the duration required to restore the system after an incident and verifies the success of recovery.
  - **Measurement Unit:** Minutes (m).

- **Energy Consumption**
  - **Metric:** Energy consumption quantifies the amount of energy consumed by the system during a specific period.
  - **Measurement Unit:** Kilowatt-hour (kWh).

1.3 Task 1.3

1.3.1 Use-case diagram for the whole system

Figure 1 : Use-case diagram for the whole system

ID	Use-case	Description

01	Upload New Documents	Students can upload new documents that need to be printed. This feature allows them to easily submit their print jobs for processing.
02	Preview Uploaded Documents	Before confirming a print request, Students have the option to preview their documents. This enables them to review the content and layout to ensure accuracy.
03	Specify Printing Properties	Students can specify various printing properties such as the number of copies, page size, color settings, and more. This flexibility ensures that their printouts meet their specific requirements.
04	Confirm Printing Request	After reviewing their documents, Students can confirm their printing requests. The system will perform a critical check on their account balance to ensure they have sufficient credits for the print job. It also takes into account page size conversions (e.g., A3 to A4) during this verification.
05	Cancel Printing Requests	Students can cancel their print requests at any point during the workflow **before confirming the print request.** This flexibility allows them to make changes or opt out of printing if needed. Once a print request is confirmed, it cannot be canceled.
06	View Print History and Summary	Allows Students to access their printing history and view a summary of their printed pages and their monthly usage. They can filter the data by specifying a date range, allowing them to track their printing activities, monitor their account balance, and make informed decisions about their printing needs. This use case promotes transparency and accountability in managing printing resources.

07	Purchase Pages	In situations where students anticipate exceeding their allocated printing pages, they can purchase additional printing pages to meet their needs. Their purchase is prompted to be completed through the online payment system of the university- BKPay.
08	Confirm Printing Request	When students submit print jobs, they are presented with a confirmation step to review and confirm their printing requests. After uploading a document, specifying print settings (e.g., paper size, color, number of copies), and choosing a printer, students can review the details of their request. They then confirm the request, after which the system processes the job for printing. This use case ensures that students have the opportunity to verify their printing preferences before incurring printing costs.
09	Checking Account Balance	Students can check their account balance within the system to monitor the remaining pages they can print without exceeding their allocated quota. This use case helps students stay informed about their printing resources, avoid unexpected printing interruptions, and make cost-effective choices when using printing services.
10	Upload Manual Document	This use case enables the SSPSO to manually upload documents into the system. These documents may include important notices, announcements, or user guides related to the printing service.

11	Configure System	The "Configure System" use case empowers the SSPSO to manage and customize various system settings to align with institutional requirements, including adjusting default pages and default printing properties, specifying valid files for printing.
12	View All Log (Balance)	SSPSOs can access a comprehensive log of all user activities related to account balances. This log provides a detailed record of transactions, including account balance adjustments, payments, and purchases of additional printing pages. The SSPSO can review this log for auditing, reconciliation, and resolving any balance-related issues.
13	View Reports	This use case allows the SSPSO to access various reports generated by the system weekly/monthly/annually, including usage reports, financial reports, and user-specific reports. These reports provide insights into printing activities, costs, and resource utilization. The SSPSO can use these reports for administrative purposes, budget planning, and decision-making.

14	Block/Unblock Members	The "Block/Unblock Users" use case empowers the SSPSO to manage user account statuses by either blocking or unblocking users. When issues such as policy violations, outstanding payments, or other infractions arise, the SSPSO can take appropriate action. In the "Block" mode, the SSPSO can temporarily suspend a user's access to the printing service, specifying the duration and reason for blocking. Conversely, in the "Unblock" mode, the SSPSO can reinstate a previously blocked user's access once the issue is resolved. This use case helps maintain service integrity while allowing for user account reinstatement when necessary.
15	Prioritize User	The "Prioritize User" use case allows the SSPSO to prioritize certain users or user groups for printing services such as the professor(s). This is useful in scenarios where specific users require immediate access to printing resources or need priority support. Also, this can be expanded to the scenario that users can purchase for temporary priority in printing orders for certain printing tasks.
16	Authenticate Users	Via HCMUT_SSO to authenticate users' login information.
17	View/Generate Reports	Empower the University Admin to access and generate various reports related to the HCMUT_SSPS (Student Smart Printing Service) system. University Admins have administrative privileges that allow them to access a wide range of reporting functionalities. Key aspects of this use case include: Accessing reports, generating custom reports, and downloading reports.

18	Control printers	The "Control Printers" use case grants the HSPSO and University Admin authority to manage and control the printers within the HCMUT_SSPS (Student Smart Printing Service) system. This use case encompasses several printer-related actions: Add/ Enable/ Disable printers.
19	Change Status from Pending to Processing	This use case involves the SPSO changing the status of a printing request from "pending" to "processing" once the printing process has begun. It signifies that the document is in the process of being printed.
20	Change Status from Processing to Printed	This use case involves the SPSO changing the status of a printing request from "processing" to "printed" once the printing process is successfully completed. It signifies that the document has been printed

Table 1 : Use-case description for the whole system

1.3.2 Use-case diagram for module "Printing Request"

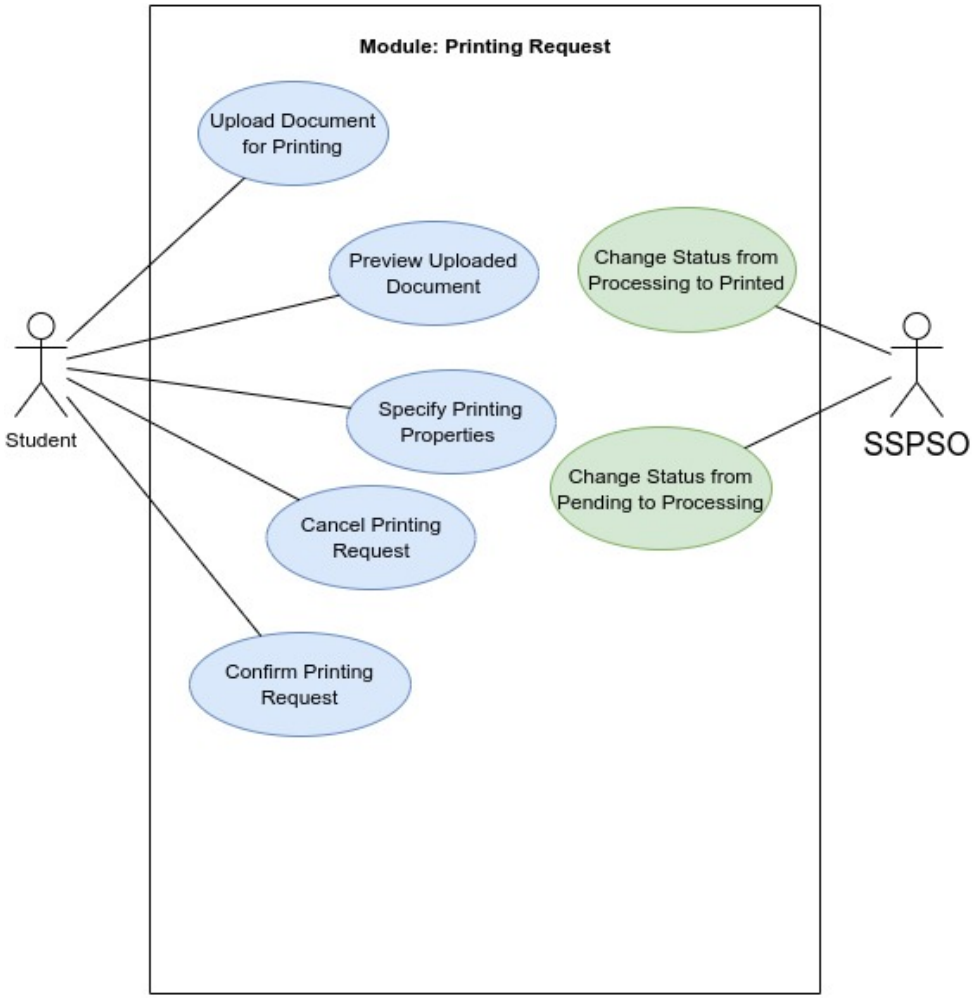

Figure 2 : Use-case diagram for module **Printing Request**

Description:

The Printing Request feature empowers Students to efficiently manage their printing requests within the system. It provides a seamless and user-friendly experience for uploading new documents, specifying printing properties, previewing documents, confirming print requests, and canceling requests when necessary.

Balance Verification: The system rigorously checks the Students' account balance when they confirm a print request. It ensures that the number of pages in the uploaded document does not exceed the available balance. The system also accounts for page size conversions, where different page sizes have specific equivalence ratios (e.g., 1 A3 page = 2 A4 pages).

File Type Acceptance: The system only accepts document files with the "pdf" ex-

tension. Students can upload and print documents in the "pdf" format.

- **Use-case description for use-case "Upload Document for Printing"**

Use-case ID	01
Use-case name	Upload Document for Printing
Actors	Student
Description	Student can upload documents to prepare for printing, one file for every upload time.
Trigger	Student clicks the "**Upload new**" button on the home page.
Pre-condition(s)	Student has logged into the system.
Post-condition(s)	The document is successfully uploaded and appears on the home page.
Basic Flow	1. After logging into the system, student clicks the "**Upload new**" button on the home page. 2. A **"Upload file"** dialog will be shown on the screen. Student clicks the "**Choose File**" button on the dialog to select a file that they want to upload. 3. After that, student clicks the "**Upload**" button to start the process of uploading the file. 4. Upload process has finished, a notification "**Upload successfully!**" appears, and the uploaded file is shown on the home page. Back to the home page. 5. Finish the use-case.
Exception Flow	4a. The format file is not supported. 4a1. The system notifies "**Format file is not supported!**" and backs to the home page. 4a2. Use-case stops.

Table 2 : Use-case description for use-case "**Upload Document for Printing**"

- **Use-case description for use-case "Preview Uploaded Document"**

Use-case ID	02		
Use-case name	Preview Uploaded Document		
Date created	22/09/2023	Last updated date	22/09/2023
Actors	Student		
Description	Student can preview documents that are uploaded before.		
Trigger	Student clicks the "**Preview**" button for each uploaded file.		
Pre-condition(s)	- Student has logged into the system. - File is shown on the home page.		
Post-condition(s)	Student previewed the uploaded documents.		
Basic Flow	1. Student clicks the "**Preview**" button which is shown for each uploaded file on the home page. 2. The system shows the file and its printing properties. 3. Student clicks the "**Back**" button to go back to the home page. 4. Finish the use-case.		
Exception Flow	None		

Table 3 : Use-case description for use-case "**Preview Uploaded Document**"

- **Use-case description for use-case "Specify Printing Properties"**

Use-case ID	03
Use-case name	Specify Printing Properties
Actors	Student
Description	Student can specify printing properties.
Trigger	Student clicks the "**Properties**" button for each uploaded file.
Pre-condition(s)	- Student has logged into the system. - File is shown on the home page.
Post-condition(s)	Student configures printing properties successfully.

Basic Flow	1. Student clicks the "**Properties**" button which is shown for each uploaded file on the home page. 2. A dialog will be shown on the screen. Student can choose options for each attribute on the dialog, which includes settings for: – Print One Sided: one-side or double-side – Orientation: portrait or landscape. – Margins: top, left, right, bottom. – Page Size: A0 - A4 – Color Settings: color or black-and-white. – Quality: normal or high quality. 3. Student clicks the "**Save**" button to save the configuration for printing and backing to the home page. 4. Finish the use-case.
Exception Flow	3a. Student click the "**Cancel**" button. 3a1. Current configuration is discarded and back to the home page. 3a2. Use-case stops.

Table 4 : Use-case description for use-case "**Specify Printing Properties**"

- **Use-case description for use-case "Confirm Printing Request"**

Use-case ID	04
Use-case name	Confirm Printing Request
Actors	Student
Description	Student can confirm to waiting for printing.
Trigger	Student clicks the "**Confirm**" button for each uploaded file.
Pre-condition(s)	- Student has logged into the system. - File is shown on the home page.

Post-condition(s)	- Student confirms to print and the file waits its turn to be printed. - Student can't change configure and cancel printing request after confirming.
Basic Flow	1. Student clicks **"Confirm"** button which is shown for each uploaded file on the home page. 2. A pop-up with the content "**Do you want to print this file?**" will be shown on the screen. If the student hasn't yet configured for printed files, default configure will be applied. 3. Student clicks the "**Yes**" button to wait for the printing and back to the home page, "**Properties**" button and "**Cancel**" button are disabled. 4. Finish the use-case.
Exception Flow	3a. The number of pages in the file exceeds the balance. 3a1. The system notifies "**The number of pages in the file exceeds the balance.**" and backs to the home page. 3a2. Use-case stops. 3b. Student clicks the "**No**" button. 3b1. Cancel confirms and backs to the home page. 3b2. Use-case stops.

Table 5 : Use-case description for use-case "**Confirm Printing Request**"

- **Use-case description for use-case "Cancel Printing Request"**

Use-case ID	05
Use-case name	Cancel Printing Request
Actors	Student
Description	Student can cancel printing request.
Trigger	Student clicks the "**Cancel**" button for each uploaded file.

Pre-condition(s)	- Student has logged into the system. - File is shown on the home page. - The request hasn't been confirmed yet
Post-condition(s)	Student cancels printing request and uploaded file is discarded from the home page.
Basic Flow	1. Student clicks the "**Cancel**" button which is shown for each uploaded file on the home page. 2. A pop-up with the content "**Do you want to discard this file?**" will be shown on the screen. 3. Student clicks the "**Yes**" button to discard this file from the home page. 4. Finish the use-case.
Exception Flow	3a. Student clicks the "**No**" button to back to the home page. 3a1. Use-case stops.

Table 6 : Use-case description for use-case "**Cancel Printing Request**"

- **Use-case description for use-case "Change Status from Pending to Processing"**

Use-case ID	19
Use-case name	Change Status from Pending to Processing
Actors	SSPSO
Description	The SPSO changes the status of a printing request from "pending" to "processing" once the printing process has begun, signifying that the document is in the process of being printed
Trigger	The printing process begins.
Pre-condition(s)	- The printing request is confirmed on the student side.
Post-condition(s)	The status of the printing request is changed to "processing."

Basic Flow	1. HSPSO selects the printing request in "pending" status. 2. HSPSO changes the status to "processing." 3. Finish the use-case.
Exception Flow	None

Table 7 : Use-case description for use-case "**Change Status from Pending to Processing**"

- **Use-case description for use-case "Change Status from Processing to Printed"**

Use-case ID	20
Use-case name	Change Status from Processing to Printed
Actors	SSPSO
Description	The SPSO changes the status of a printing request from "processing" to "printed" once the printing process is successfully completed, signifying that the document has been printed.
Trigger	The printing process is successfully completed.
Pre-condition(s)	- The printing request is in "processing" status.
Post-condition(s)	The status of the printing request is changed to "printed."
Basic Flow	1. HSPSO selects the printing request in "processing" status. 2. HSPSO changes the status to "printed." 3. Finish the use-case.
Exception Flow	None

Table 8 : Use-case description for use-case "**Change Status from Processing to Printed**"

2 Task 2: System modelling

2.1 Task 2.1

2.1.1 Preview Uploaded Document

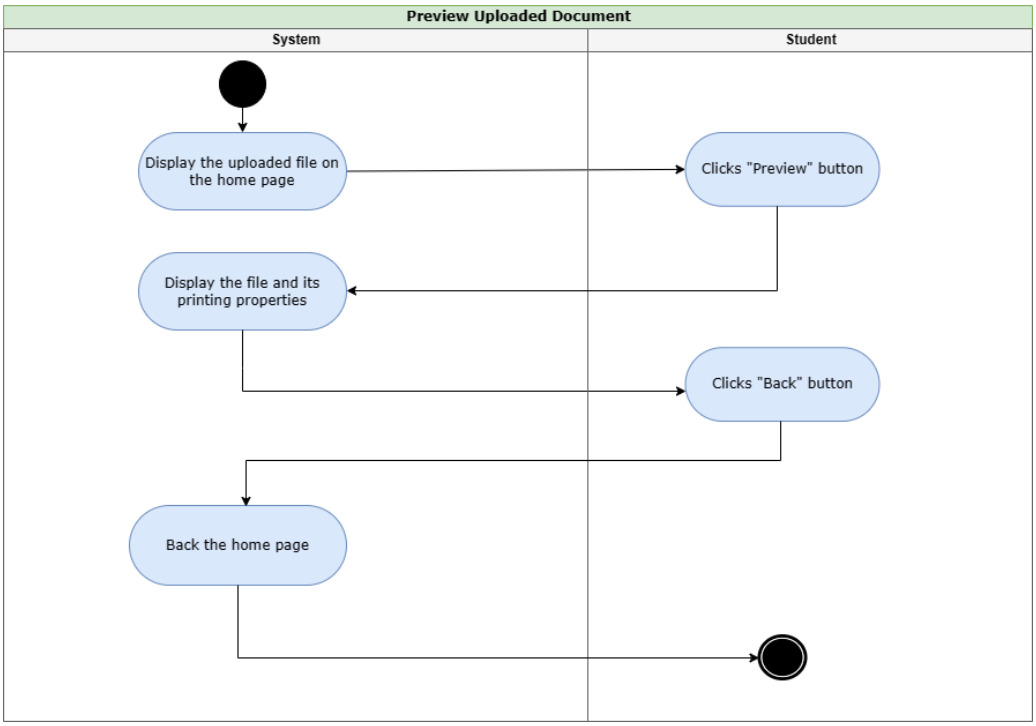

Figure 3 : Activity diagram for use-case: **Preview Uploaded Document**

In this UC, the student uploaded the file to the system. The system will then display the uploaded file on the home page. The student can then click the "Preview" button to preview the file and its printing properties. After that, student click "Back" button to back to the home page.

2.1.2 Specify Printing Properties

Link: https://drive.google.com/file/d/169NhCgHuCc2a6ptWdsy1YRMNL79sbVbi/view?usp=sharing

This activity diagram represents the process by which a logged-in student specifies printing properties for a document before initiating a print job. The student can configure various printing attributes simultaneously, including one-sided or double-sided printing, orientation (portrait or landscape), margins, page size (ranging from A0 to A4), color settings (color or black-and-white), and quality (normal or high quality). The process begins when the student clicks the "Properties" button for an uploaded file on the home page.

A dialog is displayed, allowing the student to select each attribute. After configuring the desired printing properties, the student clicks the "Save" button to save the configuration and returns to the home page. Otherwise, the student may choose to cancel the configuration using the "Cancel" button, leading to the current configuration being discarded, and the use-case stops.

2.1.3 Confirm Printing Request

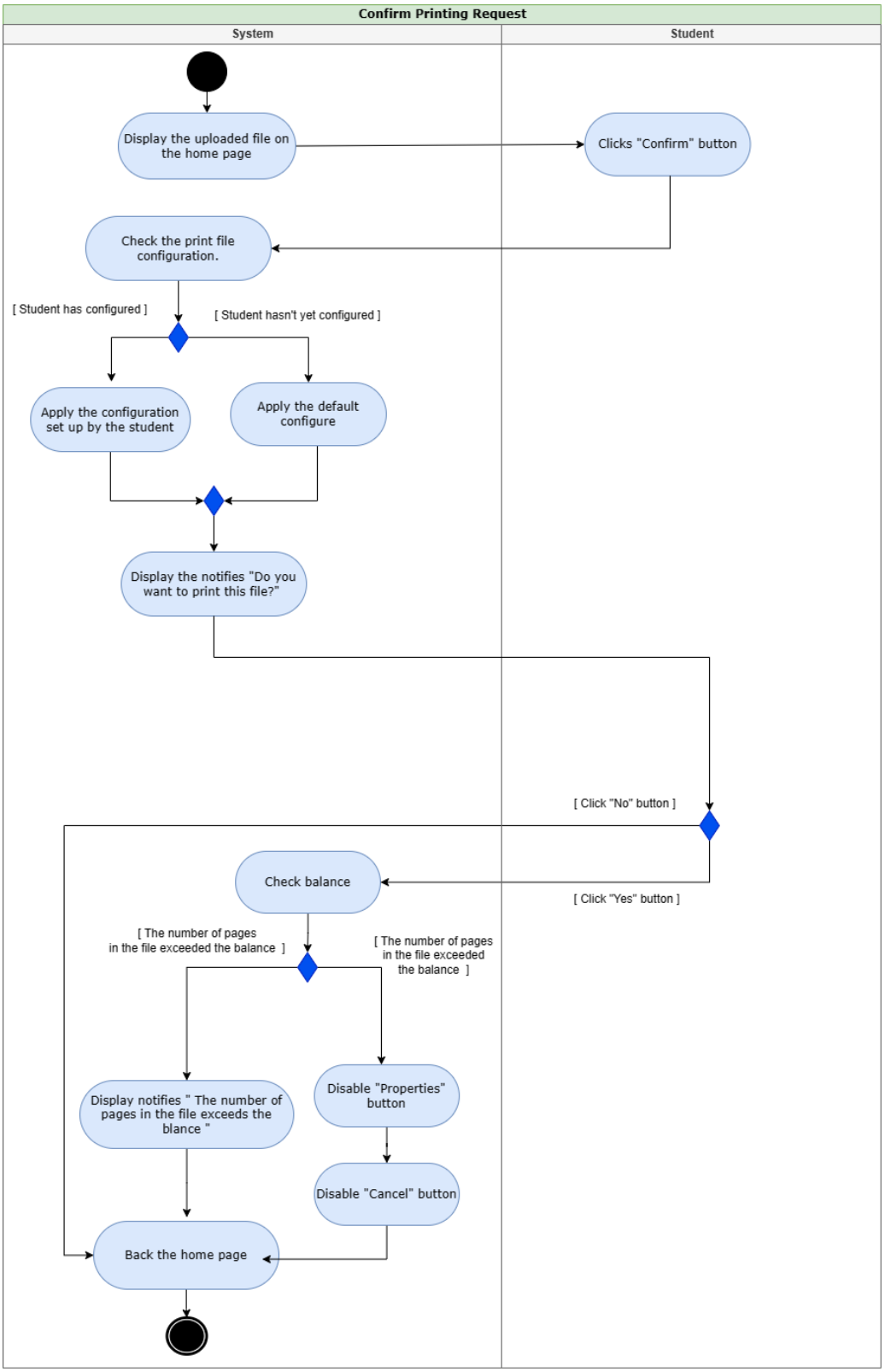

Figure 4 : Activity diagram for use-case: **Confirm Printing Request**

In this use case, after the system displays the uploaded file on the home page, the student can click the "Confirm" button. Then, the system will check the print file configuration. If the file has already been configured by the student, the system will apply the configuration set up by the student. If the file has not been configured, the system will apply the default configuration. After that, the system displays the notification message "Do you want to print this file?". Afterward, the "No" and "Yes" buttons will appear on the screen. If the student clicks the "No" button, the system will navigate back to the home page. If the student clicks the "Yes" button, the system will check the balance. If the number of pages in the file exceeds the available balance, the system will display a notification stating "The number of pages in the file exceeds the balance" and return to the home page. If the number of pages in the file does not exceed the balance, the system will disable the "Properties" button, disable the "Cancel" button, and return to the homepage. The use-case stop.

2.1.4 Upload Document for Printing

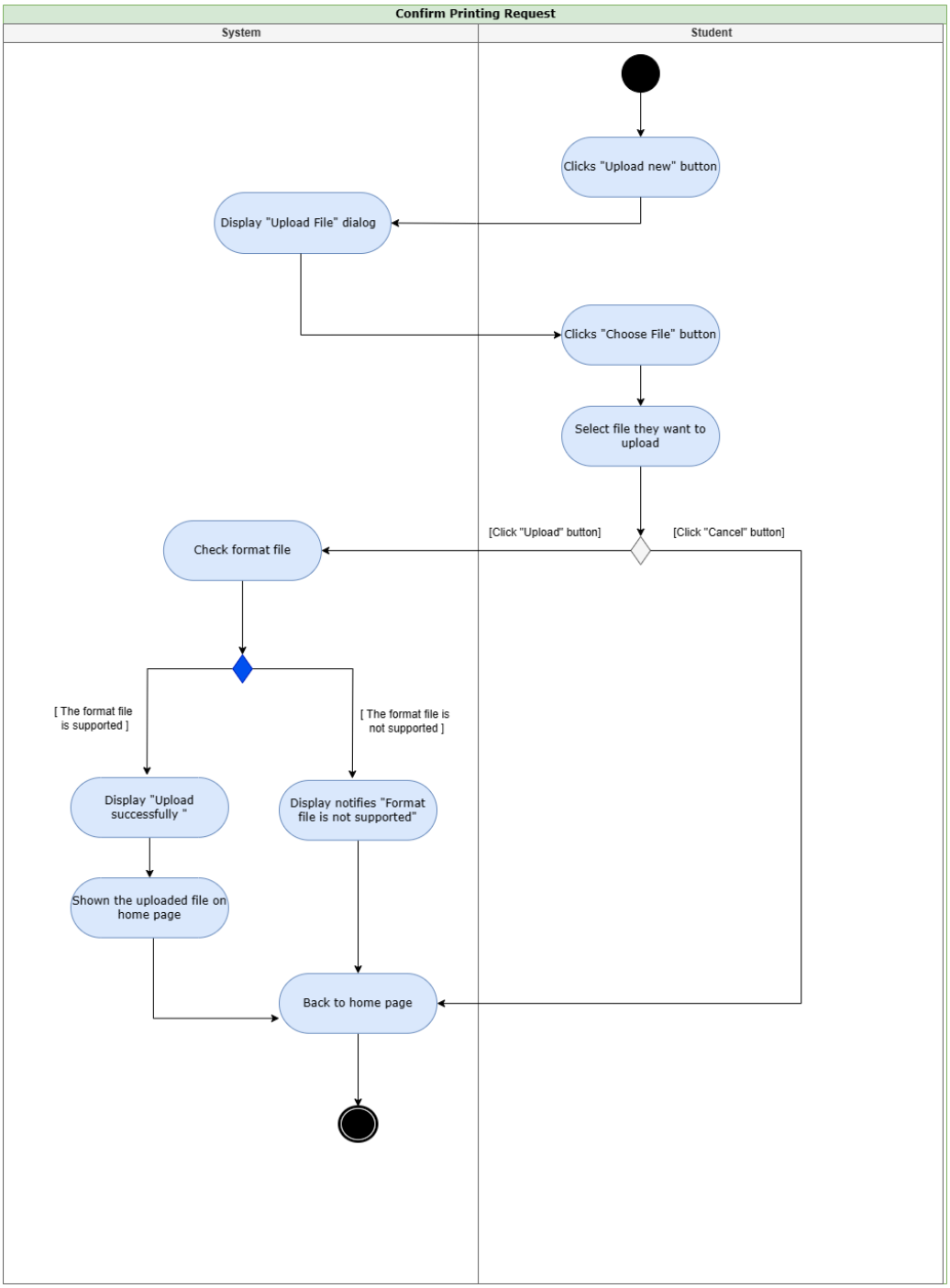

Figure 5 : Activity diagram for use-case: **Upload Document for Printing**

In this use case, the student will click the "Upload new" button to upload a new file to the system. The system will display an "Upload File" dialog. Subsequently, the student will click the "Choose File" button to select the file they want to upload to the system. If the student then chooses the "Upload" button, the system will check the file format. If the

format of the file is supported, the system will display the message "Upload successful" and show the uploaded file on the home page, then return to the home page. If the file format is not supported, the system will display a notification stating "File format is not supported" and then return to the home page. If, while selecting a file to upload to the system, the student clicks the "Cancel" button, the system will return to the home page.

2.1.5 Change Status from "Pending" to "Processing"

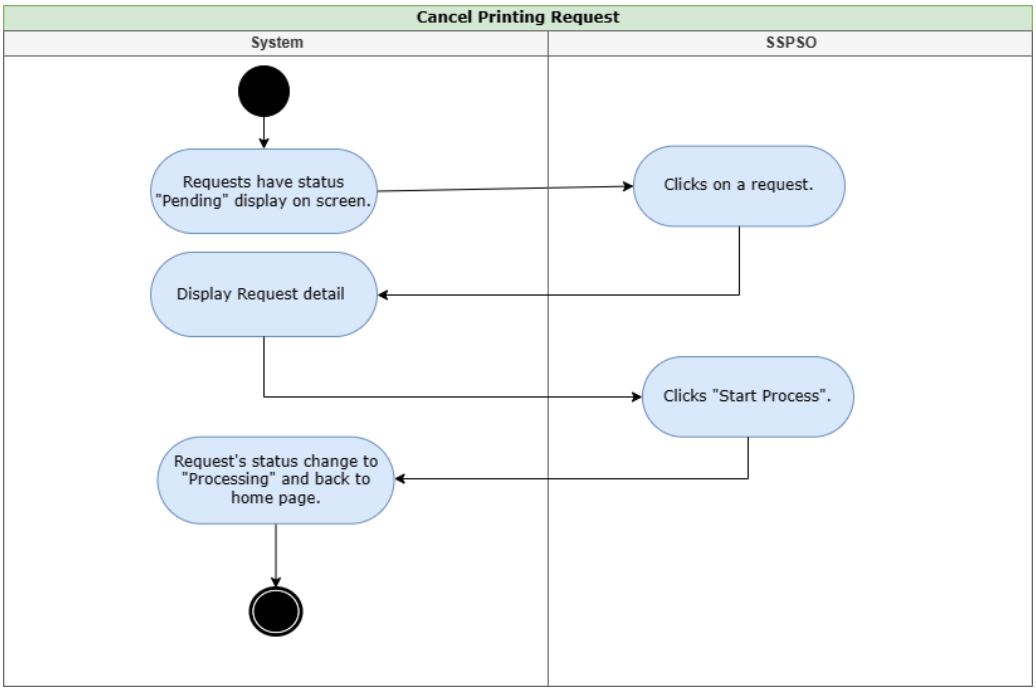

Figure 6 : Activity diagram for use-case: **Change Status from "Pending" to "Processing**

The SSPSO clicks on a request that has the status "Pending" on the screen. The system displays the request details on the screen. The SSPSO clicks on the "Start Process" button. The system changes the request's status to "Processing" and returns to the home page.

2.2 Task 2.2

2.2.1 Preview Uploaded Document

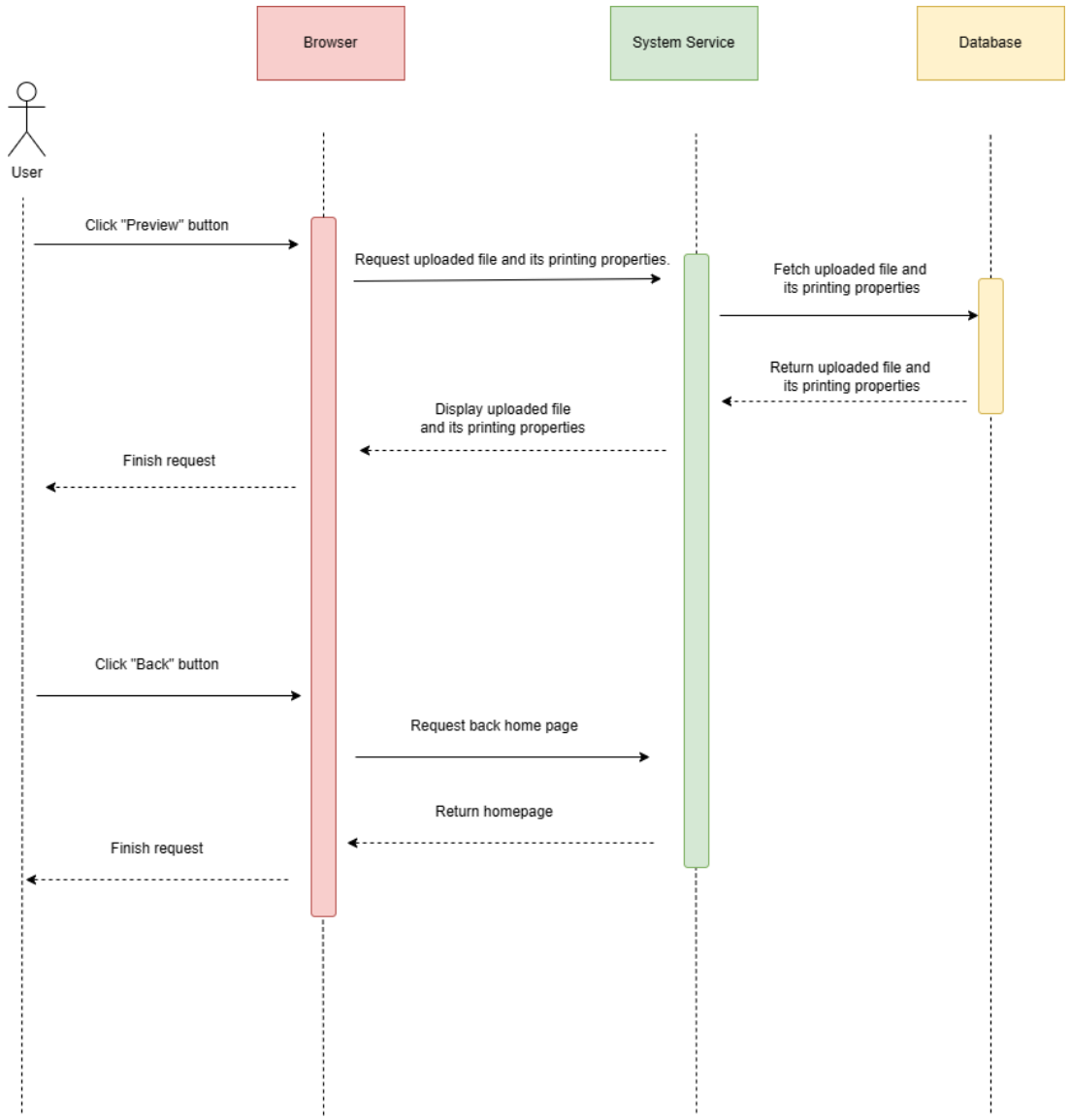

Figure 7 : Sequence diagram for use-case: **Preview Uploaded Document**

A student wants to preview uploaded document. The student opens the printing system in their browser and clicks "Preview" button for document which is uploaded. Browser will send request uploaded file and its printing properties to system service. The system service then fetches the file and its printing properties from the database. The browser displays uploaded file and its printing properties. After that, student click "Back" button, a request back homepage will send to system service and system service will return homepage.

2.2.2 Specify Printing Properties

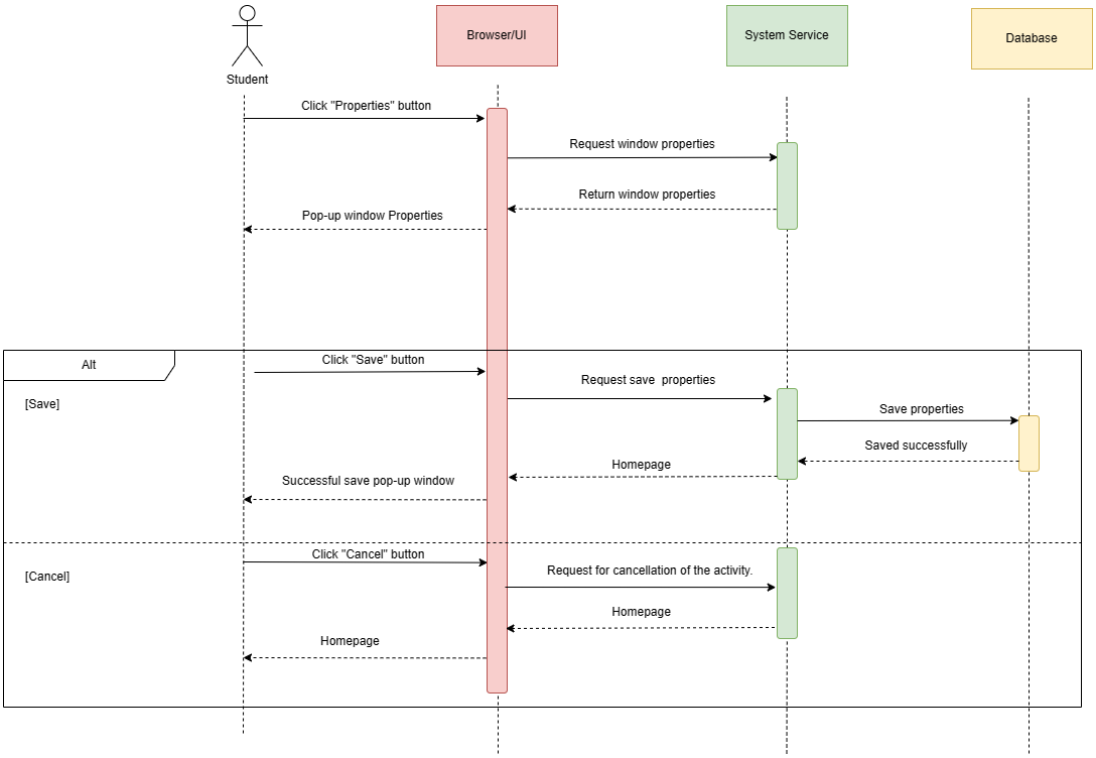

Figure 8 : Sequence diagram for use-case: **Specify Printing Properties**

A student wants to configure printing properties. The student opens the printing system within their web browser and clicks on the "Properties" button. Subsequently, the browser sends a request for the properties of a pop-up window to the System Service. The System Service then provides the browser with the properties of the pop-up window, and the browser displays the pop-up window to the student. Following this, the "Save" and "Cancel" buttons become visible. If the student clicks the "Save" button in the browser, the browser sends a request to save the properties. The System Service proceeds to save these properties in the database. It then manages the browser to return to the homepage, and the browser displays a successful save pop-up window. Conversely, if the student clicks the "Cancel" button, the browser sends a request to cancel the activity to the System Service. Subsequently, the System Service takes control the browser to return to the homepage.

2.2.3 Confirm Printing Request

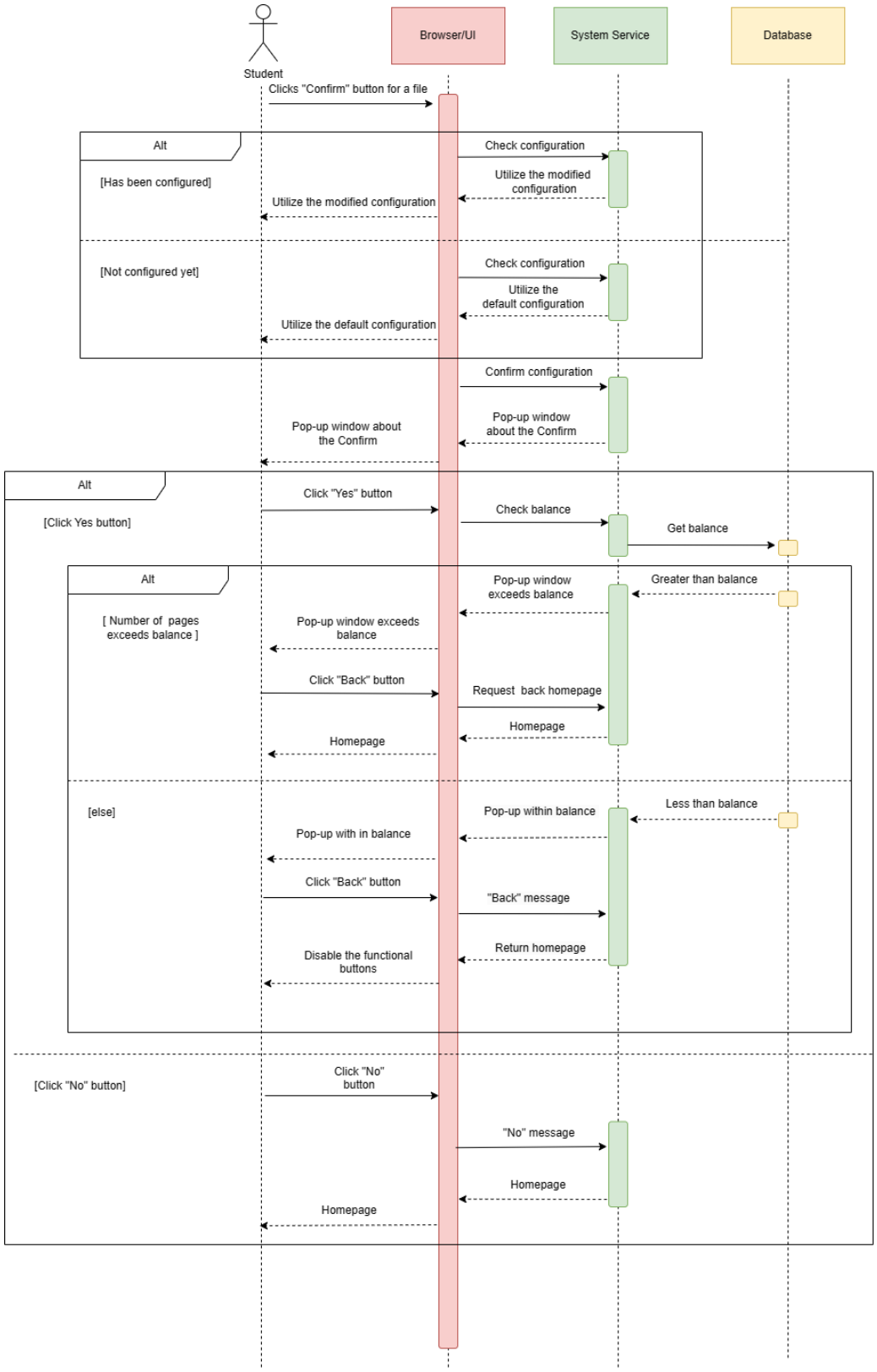

Figure 9 : Sequence diagram for use-case: **Confirm Printing Request**

A student needs to confirm a printing request. The student clicks the "Confirm" button for an uploaded file. The browser then sends a request to check the configuration to the System Service. The System Service proceeds to verify the configuration. If the file has already been configured, the entire printing system will apply the modified configuration. However, if the file hasn't been configured yet, the entire printing system will use the default configuration.

Following this, the browser requests a pop-up "Confirm" window from the System Service. The System Service provides the browser with a pop-up window containing the confirmation message, and the browser displays it. If the student clicks the "Yes" button on the confirmation pop-up window, the browser sends a request to check the balance to the System Service.

The System Service then compares the balance in the database. If the balance is greater than the required amount, the System Service returns a pop-up window indicating that the balance exceeds the required amount, and the browser displays it. If the student clicks the "Back" button, the browser sends a request to return to the homepage to the System Service, and the System Service controls the browser to navigate back to the homepage. If the balance is less than required, the System Service returns a pop-up window indicating an insufficient balance, and the browser displays it. Again, if the student clicks the "Back" button, the browser sends a request to return to the homepage to the System Service, and the System Service controls the browser to navigate back to the homepage.

In the event that the student clicks the "No" button in the confirmation pop-up window, the browser sends a request to cancel the process. The System Service takes control the browser to return to the homepage..

2.2.4 Upload Document for Printing

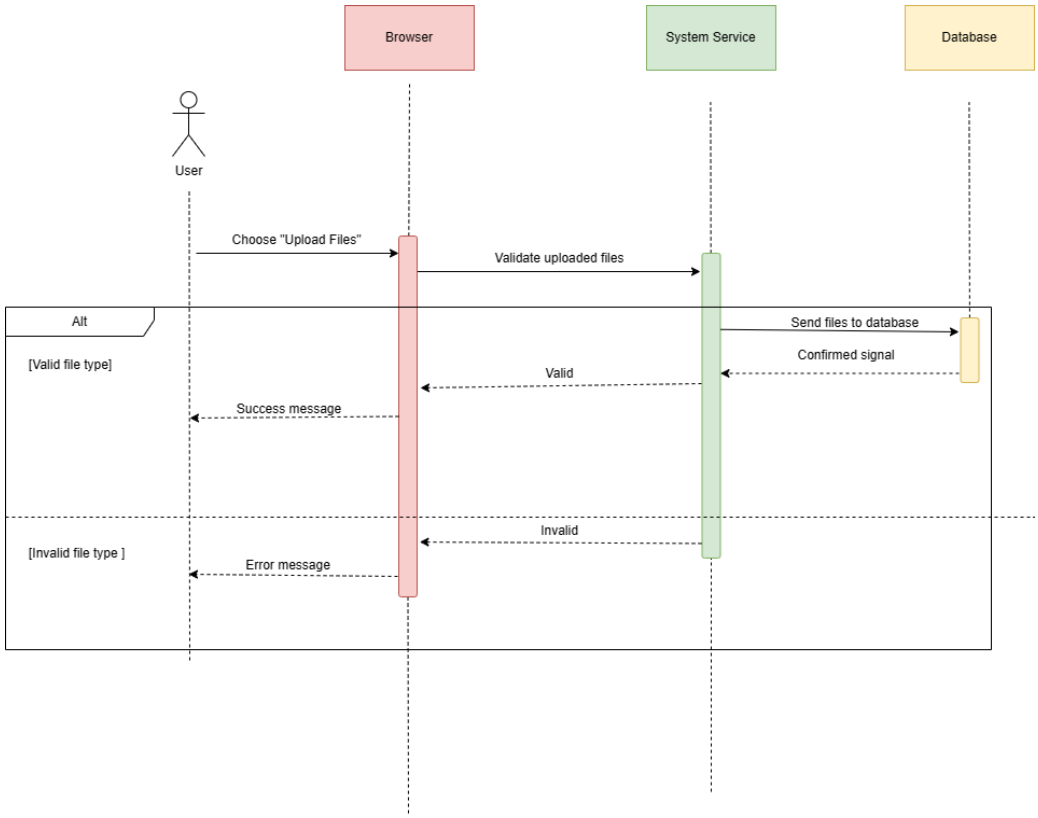

Figure 10 : Sequence diagram for use-case: **Upload Document for Printing**

A student wants to print a research paper for school. The student opens the printing system in their browser and chooses the "Upload Files" option. The student selects the research paper from their computer and the browser sends the file to the system service. The system service validates the file and finds that it is a valid PDF file. The system service then sends the file to the database and stores it. The system service then sends a confirmation signal to the browser and the browser displays a success message to the student. The student can now print the research paper.

2.2.5 Change Status from "Pending" to "Processing"

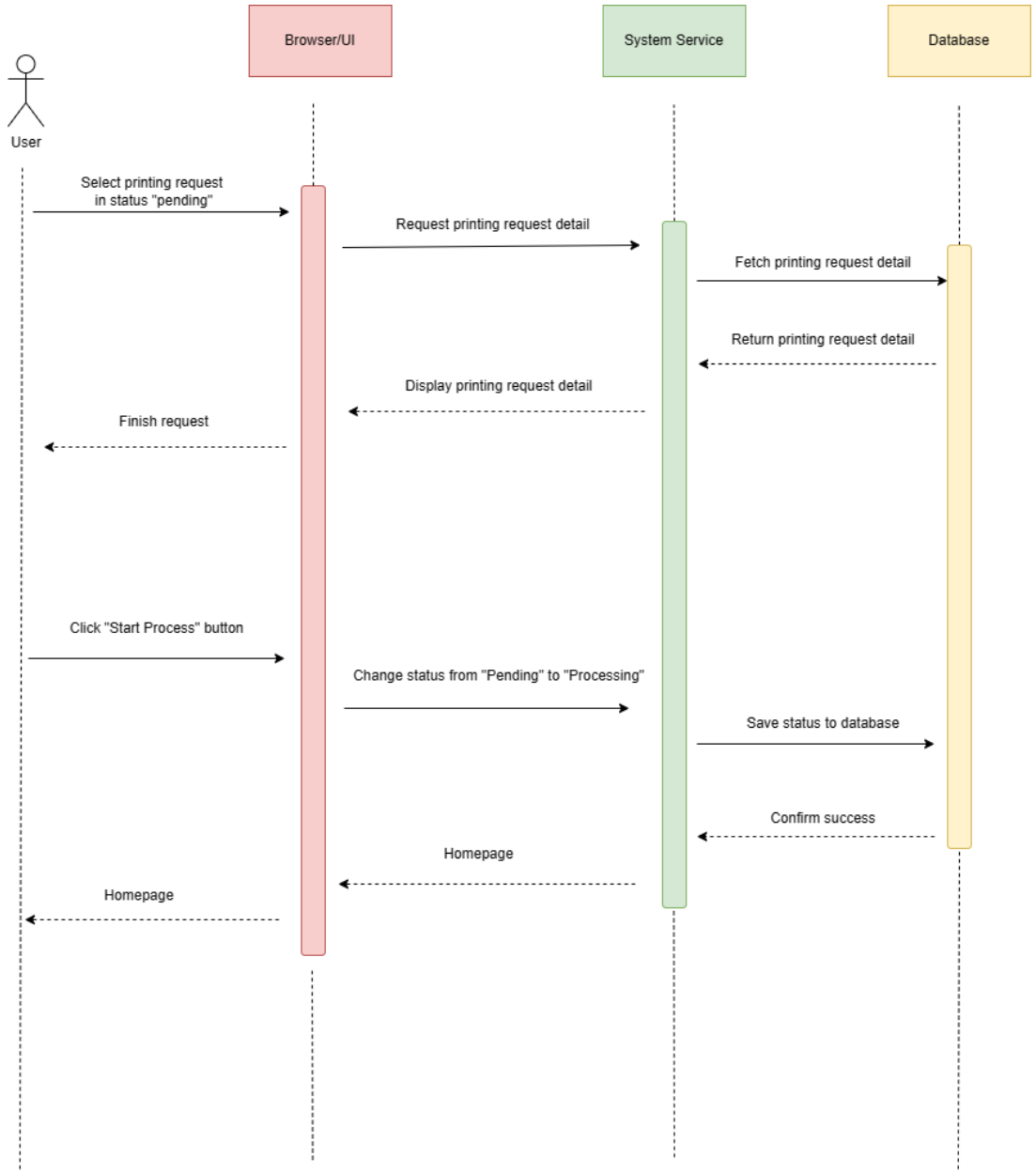

Figure 11 : Sequence diagram for use-case: **Change Status from "Pending" to "Processing**

SSPSO wants to process the printing request. SSPSO chooses one printing request which is in status "pending". Browser will send request printing request detail to system service. The system service then fetches printing request details from the database. The browser displays printing request details. After that, SSPSO clicks "Start Process" button, the browser will send the request to change the status from "pending" to "process" to system service and the system service will be saved to the database. A confirmed success signal

will be sent from the database to the system service and return the homepage.

2.3 Task 2.3

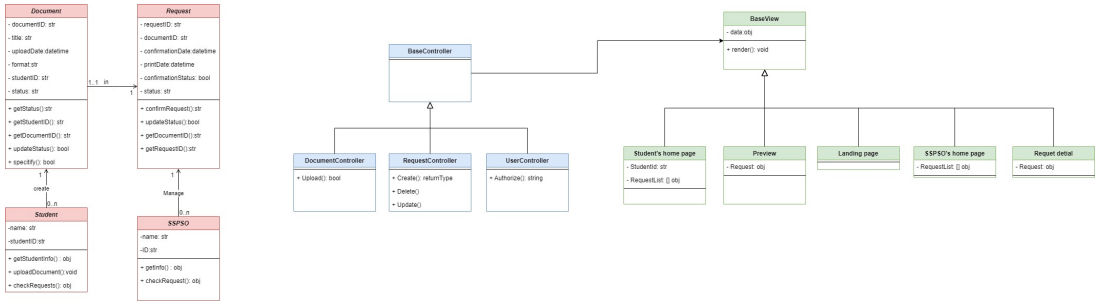

Figure 12 : Class Diagram

Class Diagram of selected module - Create Printing requests is designed based on MVC design pattern.

It divides an application into three interconnected components - Model, View, and Controller, each with its own functionality:

1. **Model**: - Represents the data and business logic of the application. - Handles data storage, retrieval, and manipulation. - Notifies the View and Controller about changes in data.

2. **View**: - Responsible for presenting the data to the user. - Displays the user interface (UI) and interacts with the user. - Receives input from the user and sends it to the Controller.

3. **Controller**:: - Acts as an intermediary between the Model and View components. - Receives user input from the View and processes it. - Updates the Model based on user actions and business logic. - Updates the View to reflect changes in the Model.

The Model, View, and Controller work together to achieve separation of concerns, making the application more modular, maintainable, and scalable.

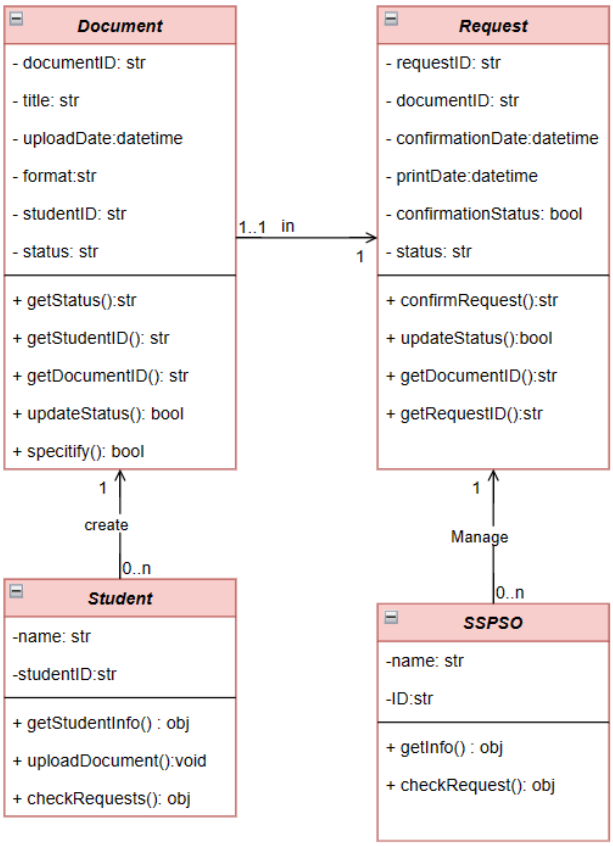

Figure 13 : Model

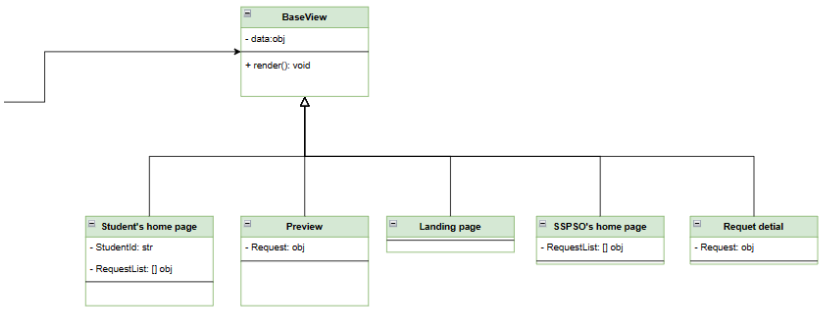

Figure 14 : View

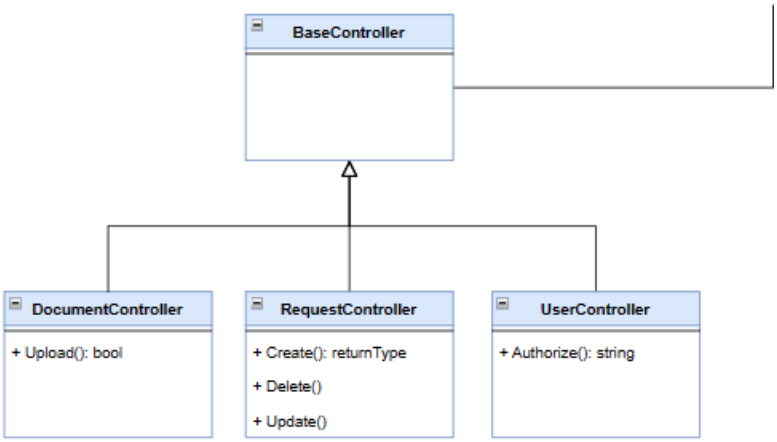

Figure 15 : Controller

Link Class Diagram: https://drive.google.com/file/d/1jrsfQwyk82B8RKnaF5zGzEw3lgf5bimY/view?usp=sharing

2.4 Task 2.4

Link Figma: https://www.figma.com/file/w2lAvEN5NqgBvjeg4Mvrm9/HK231_CNPM?type=design&node-id=0%3A1&mode=design&t=Q0NTIuOI3ihJqSMK-1

3 Task 3: Architecture design

3.1 Task 3.1

3.1.1 Architectural diagram

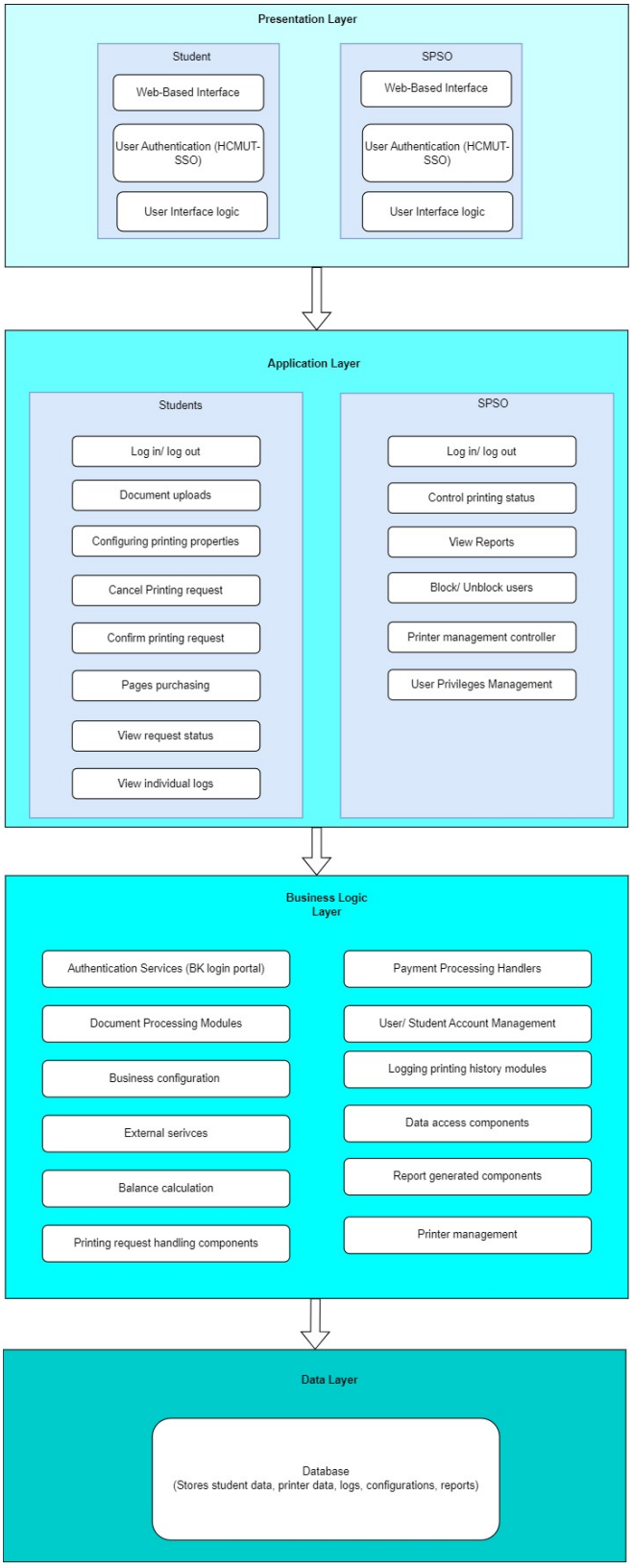

Figure 16 : diagram

Component	Function
Web-Based Interface (Presentation Layer)	Web-Based Interface
User Authentication (HCMUT-SSO) (Presentation Layer)	Processing authentication for Student/SSPSO accounts
User Interface logic (Presentation Layer)	Processing operations, interacting with users on the web interface
Log in/ log out (Application Layer)	Log in/ log out
Document uploads (Application Layer)	Student upload file
Configuring printing properties (Application Layer)	Student configures print properties.
Cancel Printing request (Application Layer)	Student cancel printing request
Confirm printing request (Application Layer)	Student confirm printing request
Pages purchasing (Application Layer)	Student buys printing supplies.
View request status (Application Layer)	Student view request status
View request status (Application Layer)	Student view request status
View individual logs (Application Layer)	Student view individual logs
Control printing status (Application Layer)	SSPSO control printing status
View Reports (Application Layer)	SSPSO views the printing log
Block/ Unblock users (Application Layer)	SSPSO Block/ Unblock users to manage system
Printer management controller (Application Layer)	SSPSO controls the printer.
User Privileges Management (Application Layer)	SSPSO manages user privileges.
Printer management controller (Application Layer)	SSPSO controls the printer.

Component	Function
Authentication Services (BK login portal) (Business Logic Layer)	Processing authentication for Student/SSPSO accounts
Document Processing Modules (Business Logic Layer)	Handling printing properties, which may include file format conversion (from docs to PDF)
Business configuration (Business Logic Layer)	Components to define valid file types (docs, pdf), set default page numbers, and establish a currency conversion method based on the number of pages (2 sheets of paper = 1k).
External services (Business Logic Layer)	Integrating external services such as BKPay, Momo to deposit money into the account.
Balance calculation (Business Logic Layer)	Handling logic such as adding balance after deposit, deducting for printing.
Printing request handling components (Business Logic Layer)	After uploading a file and confirming printing, display the status of requests on the UI (processing, done, canceled) Additionally, include a feature to assign priority permissions for faster printing.
Payment Processing Handlers (Business Logic Layer)	Processing payment after purchasing printing pages, when placing a print order.
User/ Student Account Management (Business Logic Layer)	Viewing printing history, account information, blocking/unblocking accounts, and managing priority privileges for students.
Logging printing history modules (Business Logic Layer)	Viewing history printing
Data access components (Business Logic Layer)	SContaining features to interact with the database.

Component	Function
Report generated components (Business Logic Layer)	Handle Report generated components
Printer management (Business Logic Layer)	Handling operations with the printer, for example, displaying the current status information (such as the number of current requests), indicating color printing availability, and notifying about maintenance schedules.
Database (Stores student data, printer data, logs, configurations, reports) (Data Layer)	Handling the storage of necessary information.

3.1.1.a Interface Presentation strategy

For the presentation strategy of the HCMUT Student Smart Printing Service (HCMUT_SSPS), we propose a responsive and user-friendly web-based application to ensure accessibility for students across various devices. The user interface will be designed with a focus on simplicity and ease of use, allowing students to quickly upload documents, select printers, and specify printing properties. We'll utilize modern front-end technologies such as ReactJs for the web application, ensuring a consistent and intuitive user experience across platforms. Additionally, a well-structured user manual and interactive tutorials will be provided within the application to guide users in using the service effectively. To enhance the user experience, we will also incorporate a notification system to keep students informed about their printing balance and essential updates.

3.1.1.b API Management

API management for the HCMUT-SSPS system is essential to ensure secure and efficient external access to specific functionalities. An API gateway, such as AWS API Gateway, will be implemented to handle requests, enforce security policies using OAuth 2.0, and manage rate limiting. It will also provide detailed monitoring and analytics to track usage patterns and identify issues. A well-documented developer portal will be set up to assist external developers in using the system's APIs. Versioning of APIs will be implemented

to maintain backward compatibility. This approach is justified by the need to provide a secure and user-friendly environment for external developers.

3.1.1.c Data storage approach

To efficiently manage and store data for the HCMUT_SSPS, we will adopt a robust and scalable database system. A relational database, namely MySQL, will be used to store critical information like student details, printer information, printing logs, and configuration settings. Additionally, for document file storage, a cloud-based file storage service like Amazon S3 or Azure Blob Storage will be utilized to ensure reliable and secure document uploads. This approach enables easy scalability as the system expands and ensures data integrity and availability. Regular backups and redundancy measures will be in place to prevent data loss, and data encryption will be implemented to safeguard sensitive information.

3.1.2 Deployment diagram

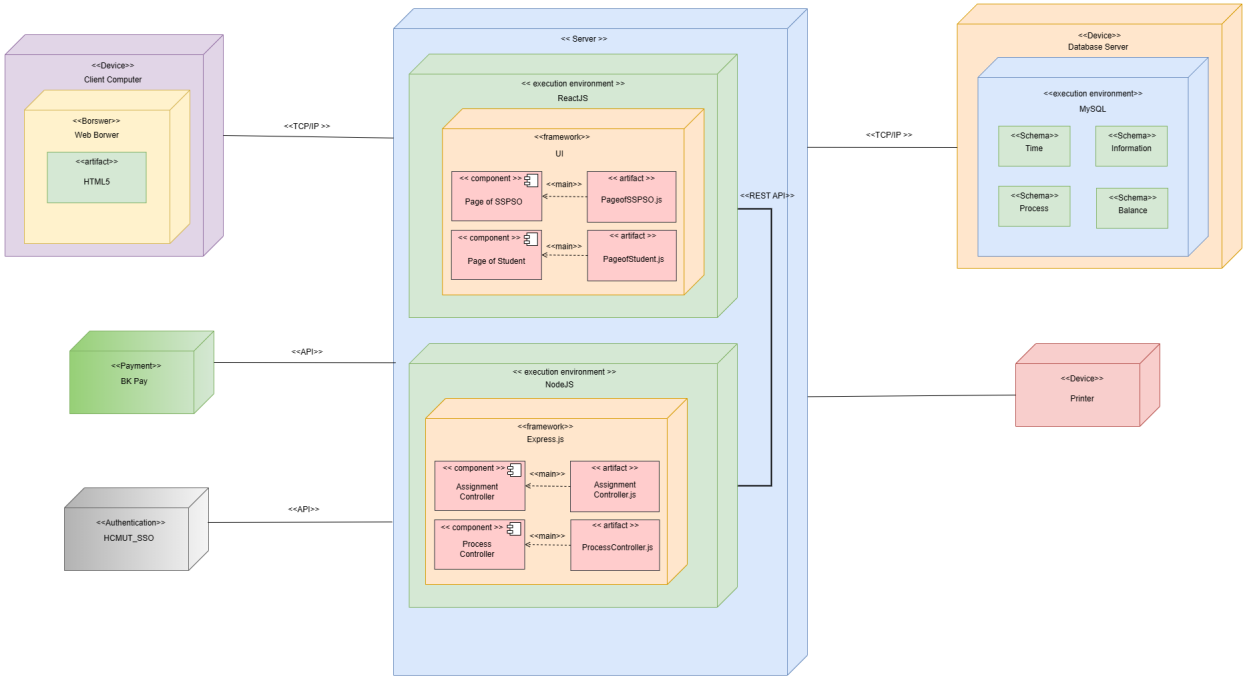

Figure 17 : Deployment

The system is divided into several key components, including "Server", "Database Server", "Client Computer", "Paymet", "Authentication".

- The Client server will connect to the Server using the TCP/IP protocol. The data received from the Server will consist of interface implementation files and will be

displayed on a website in the user's computer (Client) through an HTML5 reader embedded in the web browser.

- The Server consists of the execution environment and the main modules of the system. The back-end execution environment is Node.js, utilizing the Express.js framework, which includes two primary components: AssignmentController and ProcessController. These components handle core processing tasks and control the execution of processes related to printing operations and interactions between the system and users.On the front-end, the execution environment is React.js, which includes two primary components: PageofSSPSO and PageofStudent. These components are responsible for implementing user interfaces related to task assignments. These components will be implemented from corresponding JavaScript source files. Through REST APIs, the front-end and back-end can communicate with each other to update, display interfaces, and receive user task responses.
- The Server will connect to the Database Server via the TCP/IP protocol to retrieve and store data as needed.
- The Database Server comprises the execution environment MySQL and stores information including: Time, Information user, Process, Balance.
- The printer will receive processing instructions from the server and execute them based on the requests sent from the server.
- The Payment Server, represented by BK Pay, connects to the main server via an API to facilitate payment processing.
- The Authentication Server, represented by HCMUT_SSO, connects to the main server via an API to facilitate login and authentication processes.

3.2 Task 3.2

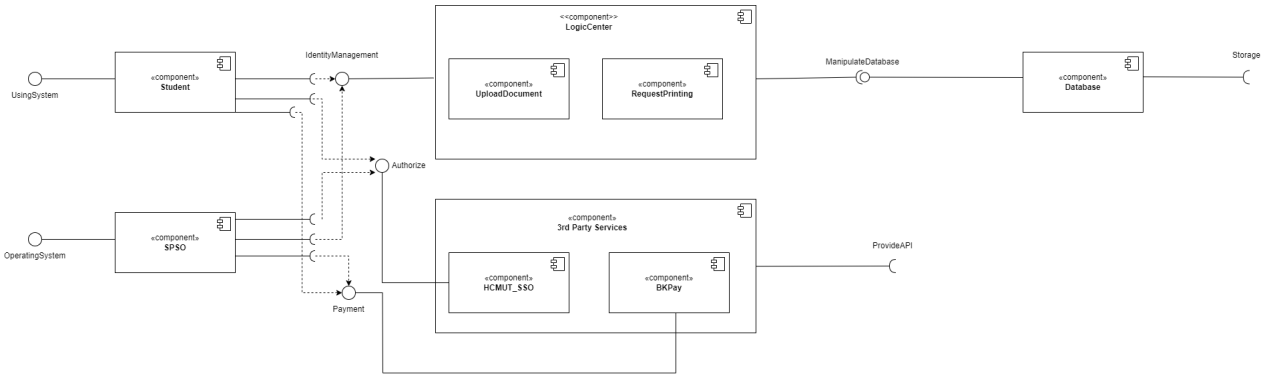

Figure 18 : Component diagram

The system is divided into several key components:

- "Student" represents the user component. This component directly interacts with the system and can perform activities such as logging in, uploading documents, requesting printing, etc.
- "SPSO" is a specialized component designed to support students in addressing printing request and managing system.
- LogicCenter
  - "UploadDocument": Represents the "UploadDocument" component in the system. This component is responsible for allowing users to upload documents.
  - "RequestPrinting": Represents the "RequestPrinting" component in the system. This component handles the functionality related to requesting printing services.
- The "3rd Party Services" component represents external services that the system interacts with and utilizes such as BKPay.
  - "HCMUT_SSO": Represents the "HCMUT_SSO" component in the system. It provides single sign-on (SSO) functionality for authentication and authorization of users from the HCMUT system.
  - "BKPay": Represents the "BKPay" component in the system. This component handles the payment processing and integration with banking services.
- The "Database" component serves as the storage for the system's data. It includes tables and data relationships necessary for storing user information and other relevant information within the system.

There are some API for system:

- `Identify API`: Used for verifying the identity of a user. Typically involves a POST request with user credentials, responding with success/failure messages or additional user identity information.
- `Authority API`: Validates if a user has the necessary permissions for a specific action. Generally initiated through a GET request with user credentials and desired action, responding with success/failure messages or additional user permission details.
- `Upload Document API`: Facilitates document uploads to the system. Involves a POST request with the document file as the payload, responding with success/failure messages or additional information about the uploaded document.
- `Payment API`: Processes payments from users, typically triggered by a POST request with payment details. Responds with success/failure messages or additional information about the payment transaction.

4 Task 4: Implementation - Sprint 1

4.1 Task 4.1

Link: https://github.com/khanhnhanbk/HK231_CNPM

4.2 Task 4.2

Report the change log in the selected version control system:
Using: **git log > log.txt**

```
commit f993af11df178cbdb6d55c39cfda8f0835378862
Merge: 9cab41b 4b0647e
Author: TDung <91122031+dungnguyen73@users.noreply.github.com>
Date:   Sat Dec 9 09:13:37 2023 +0700

    Merge branch 'main' of https://github.com/khanhnhanbk/HK231_CNPM into DuqBE

commit 9cab41ba4e0b817198dc9eafcb52ccfb1530f3bd
Merge: 8d36deb 40f3290
Author: TDung <91122031+dungnguyen73@users.noreply.github.com>
Date:   Sat Dec 9 09:10:24 2023 +0700

    Merge branch 'DuqBE' of https://github.com/khanhnhanbk/HK231_CNPM into DuqBE

commit 4b0647ec04ea2d1d9e717de809a399885dfac55a
Merge: b63981f 40f3290
Author: Nguyễn Văn Khánh Nhân <75458182+khanhnhanbk@users.noreply.github.com>
Date:   Sat Dec 9 09:09:37 2023 +0700

    Merge pull request #3 from khanhnhanbk/DuqBE

    Duq be

commit 40f32901f53ba9c42c8b67fe02a71ddb6d5781ed
Merge: 763016b 243dd63
Author: khanhnhanbk <nhanhoa21012002@gmail.com>
```

```
Date:   Sat Dec 9 09:07:06 2023 +0700

    Merge branch 'DuqBE' of github.com:khanhnhanbk/HK231_CNPM into DuqBE

commit 763016b735801db6adc33bed77e2d59324af68ae
Author: TDung <91122031+dungnguyen73@users.noreply.github.com>
Date:   Fri Dec 8 11:06:32 2023 +0700

    Add routes before login + adjust page before login

commit 4d9039483f759c013f973c589ab7dda19dc78197
Author: TDung <91122031+dungnguyen73@users.noreply.github.com>
Date:   Mon Dec 4 16:46:21 2023 +0700

    add react- drag file

commit e5bf8a5b448804b489f7be5f127db143e24a7026
Author: khanhnhanbk <nhanhoa21012002@gmail.com>
Date:   Sat Dec 9 08:46:55 2023 +0700

    Add PrintingSPSORequest component and update
    status in printRequest.controller

commit e48894375e57ec4b9c3d935f1bc2eb053dd269b8
Author: khanhnhanbk <nhanhoa21012002@gmail.com>
Date:   Wed Dec 6 16:51:35 2023 +0700

    edit and detail page

commit d6f115cc2ecab6ef3399e1a646b9ac0962923070
Author: khanhnhanbk <nhanhoa21012002@gmail.com>
Date:   Mon Dec 4 16:36:23 2023 +0700

    Add SPSO menu item and routes
```

```
commit 5cdd875fbf6559d7ceb84eff52f74ccad8211faf
Author: khanhnhanbk <nhanhoa21012002@gmail.com>
Date:   Mon Dec 4 15:42:26 2023 +0700

    upload file

commit 5bcf677a63fcc12e72bab03891da8c090189e0e0
Author: TDung <91122031+dungnguyen73@users.noreply.github.com>
Date:   Sun Dec 3 22:23:14 2023 +0700

    page for requests

commit 2856d68075ffe97c293a3009b0ff103a2ce36d3b
Author: TDung <91122031+dungnguyen73@users.noreply.github.com>
Date:   Wed Nov 29 09:00:53 2023 +0700

    add balance

commit 8d36deba5cf0d422c4712f7799b3b0610722b687
Author: TDung <91122031+dungnguyen73@users.noreply.github.com>
Date:   Sat Dec 9 08:56:31 2023 +0700

    file types

commit 243dd63587881835dd93d151c12de2227afba6dd
Merge: 2f0e1c0 c7cfe85
Author: khanhnhanbk <nhanhoa21012002@gmail.com>
Date:   Sat Dec 9 08:47:00 2023 +0700

    Merge branch 'DuqBE' of github.com:khanhnhanbk/HK231_CNPM into DuqBE

commit 2f0e1c0e4e471c631b613155ee0b424567839e9f
Author: khanhnhanbk <nhanhoa21012002@gmail.com>
```

```
Date:   Sat Dec 9 08:46:55 2023 +0700

    Add PrintingSPSORequest component and update
    status in printRequest.controller

commit 1b7c09525c439c171938374ca46b11e40107915d
Author: TDung <91122031+dungnguyen73@users.noreply.github.com>
Date:   Fri Dec 8 21:57:45 2023 +0700

    add some font-end in the detail and config req

commit c7cfe858ef2ec605445035ce8dcf5a664dd7be74
Author: TDung <91122031+dungnguyen73@users.noreply.github.com>
Date:   Fri Dec 8 11:06:32 2023 +0700

    Add routes before login + adjust page before login

commit 0b62f8712ba8b44cd4b63d522fdba8a6f8170432
Merge: 6f877d0 8daaa7c
Author: TDung <91122031+dungnguyen73@users.noreply.github.com>
Date:   Wed Dec 6 21:12:59 2023 +0700

    Merge branch 'DuqBE' of https://github.com/khanhnhanbk/HK231_CNPM into DuqBE

commit 8daaa7cf08589c6b4d785c3aeab878705e732c96
Author: khanhnhanbk <nhanhoa21012002@gmail.com>
Date:   Wed Dec 6 16:51:35 2023 +0700

    edit and detail page

commit 6f877d03c51cca6e74bcd59b54a00c04743285fd
Merge: f15b9cd ab42ebc
Author: TDung <91122031+dungnguyen73@users.noreply.github.com>
Date:   Mon Dec 4 16:46:27 2023 +0700
```

```
    Merge branch 'DuqBE' of https://github.com/khanhnhanbk/HK231_CNPM into DuqBE

commit f15b9cdc369f6598259c5b23b41dc4478eecd574
Author: TDung <91122031+dungnguyen73@users.noreply.github.com>
Date:   Mon Dec 4 16:46:21 2023 +0700

    add react- drag file

commit ab42ebc63c79970370e03d439a7eb5a0add0a2f4
Author: khanhnhanbk <nhanhoa21012002@gmail.com>
Date:   Mon Dec 4 16:36:23 2023 +0700

    Add SPSO menu item and routes

commit 35963a7a36b8dabda2b92da3ca5c9273c97f52fe
Author: khanhnhanbk <nhanhoa21012002@gmail.com>
Date:   Mon Dec 4 15:42:26 2023 +0700

    upload file

commit 451183f6a8591ddfad37f612d078dfae7e22b989
Author: TDung <91122031+dungnguyen73@users.noreply.github.com>
Date:   Sun Dec 3 22:23:14 2023 +0700

    page for requests

commit 864614fb955046bfffd6645d209e22bbb0556f91
Author: TDung <91122031+dungnguyen73@users.noreply.github.com>
Date:   Wed Nov 29 09:00:53 2023 +0700

    add balance

commit b63981fde16eb7f6cfc7763ea309848c505cff60
```

```
Merge: b28f0c8 9957043
Author: Nguyễn Văn Khánh Nhân <75458182+khanhnhanbk@users.noreply.github.com>
Date:   Sat Nov 25 14:58:19 2023 +0700

    Merge pull request #1 from khanhnhanbk/landingPage

    Landing page

commit 995704370f64b708fbc2c61c3af5da25705b62c2
Author: TDung <91122031+dungnguyen73@users.noreply.github.com>
Date:   Sat Nov 25 10:56:49 2023 +0700

    adjust header in notifications

commit 793fca164dd236d396c67e5c86f4245a7c97d281
Author: TDung <91122031+dungnguyen73@users.noreply.github.com>
Date:   Sat Nov 25 10:08:10 2023 +0700

    adjust the drawer aka sider

commit 0207db8f596c4702385784b6055880e66e5ee03a
Author: TDung <91122031+dungnguyen73@users.noreply.github.com>
Date:   Fri Nov 24 16:39:30 2023 +0700

    create landing page UI

commit b28f0c847e9a205939096b46097cecabf0ea5731
Author: khanhnhanbk <nhanhoa21012002@gmail.com>
Date:   Fri Nov 17 21:30:33 2023 +0700

    init commit
```

4.3 Task 4.3

4.3.1 Summary

4.3.1.a Summary of Work Performed

- This report presents the results of the evaluation test on user satisfaction and the optimization and logic of the SPSO website through its functionalities and interface.
- The purpose of the test is to determine usability and list essential questions that can be answered through the testing process.

4.3.1.b Summary of Findings

- The usability test brought to our attention some very important findings.
- Identified the level of effectiveness and also pinpointed the areas that need improvement.
- The evaluation results, along with recommendations for improvement, are outlined in this report.

4.3.2 Evaluation Procedure

4.3.2.a Test Objective

The objective of this experiment is to assess the user-friendliness, satisfaction level, and logic of the SPSO website.

4.3.2.b Participants/Testers

This usability test was conducted with a number of individuals (specifically 5 participants) who met one or more of the following criteria:

- Are regular to print documents.
- Are familiar with the university environment.
- Have a basic understanding of this print system.

4.3.2.c Test method

The strategy adopted for this test is **qualitative usability** and **unmoderated remote testing**.

4.3.2.d Procedure

Describe the overall process of testing for participants, which includes the following tasks:

- Upload files to the system.
- Preview before printing.
- Select print configuration.
- Delete printing materials.
- Confirm printing requests as users.
- View printing request as SPSO.
- Cancel printing request as SPSO.

4.3.2.e Evaluation Measures

During the test, any problems incurred by the participants received a severity rating:

Severity Ranking	Severity Description	Severity Definition
4	Critical	The identified issue is so severe, the user will not be able to complete the task, and may not want to continue using the website
3	Major	Users can accomplish the task but only with considerable frustration and/or performance of unnecessary steps. The user will have great difficulty in circumventing the problem; users can overcome the issue after they have been shown how.
2	Moderate	The user will be able to complete the task in most cases but will have to undertake some moderate effort in getting around the problem. They may need to investigate several links or pathways through the system to determine which option will allow them to accomplish the intended task. Users will most likely remember how to perform the task on subsequent encounters with the system
1	Minor	The problem occurs only intermittently, can be circumvented easily but is irritating. Could also be a cosmetic problem.
0	None	

Comment: The higher the rating of severity, the more critical the problem is to the user's experience or ability to accomplish the task.

4.3.3 Testing results evaluation and feedback

4.3.3.a Task success rate and completion time.

Task 1 - Upload files

- Success rate: 100 %

- Average time of completion: below 1 minute.
- Average severity ranking (from 0 - 4): 0.5

Task 2 - Preview Select print configuration

- Success rate: 100 %
- Average time of completion: below 1 minute.
- Average severity ranking (from 0 - 4): .25

Task 3 - Select print configuration

- Success rate: 100 %
- Average time of completion: below 1 minute.
- Average severity ranking (from 0 - 4): 0.8

Task 4 - Delete printing materials

- Success rate: 100 %
- Average time of completion: below 1 minute.
- Average severity ranking (from 0 - 4): 0.75

Task 5 - Confirm printing requests as users

- Success rate: 100 %
- Average time of completion: below 1 minute.
- Average severity ranking (from 0 - 4): 0

Task 6 - View printing request as SPSO

- Success rate: 100 %
- Average time of completion: below 1 minute.
- Average severity ranking (from 0 - 4): 0

Task 7 - Cancel printing request as SPSO

- Success rate: 100 %
- Average time of completion: below 1 minute.
- Average severity ranking (from 0 - 4): 0.5

4.3.3.b Feedback from testers

- Critical usability issue: None
  Recommendation: None
- Major usability issue: None
  Recommendation: None
- Minor usability issue: None
  Recommendation: None

4.3.4 Post-testing questionnaire results

Below are the feedback received after conducting the tasks.

- I was able to complete the tasks.
- The information was easy to find.
- The organization of information was intuitive.
- The site was easy to navigate.
- The language of the website was understandable and appropriate.
- Overall, I am satisfied with this website.
- I would use the website regularly for most of my printing needs.

5 Task 5: Implementation - Sprint 2

In this section, the team will present images of the interface implemented from the selected modules.

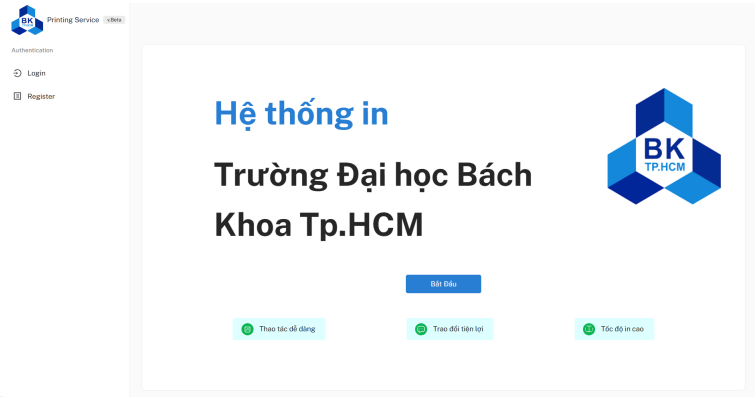

Figure 19 : Landing page

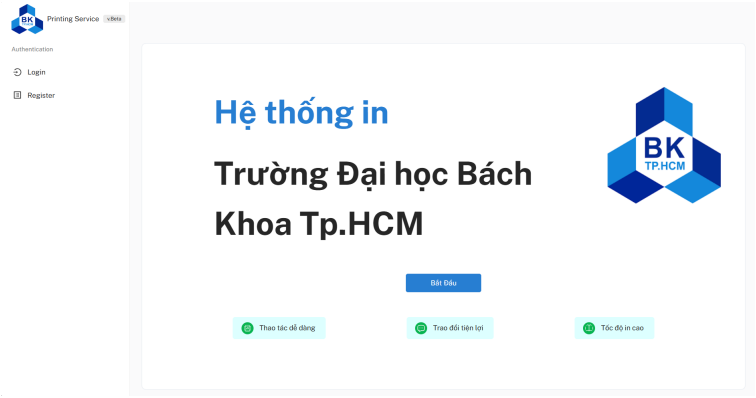

Figure 20 : Implementation of the login interface

The login interface is shared for both Students and SPSO. If the login information matches the system's data, the system will navigate to the page associated with the role, which is stored in the database.

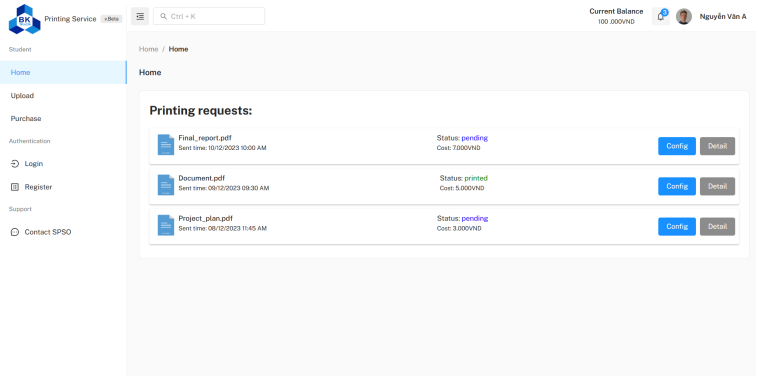

Figure 21 : Implementation of the Students' Home interface

After logging in, this will be the display page for Students. Students can view the log of their printing requests on the Home page.

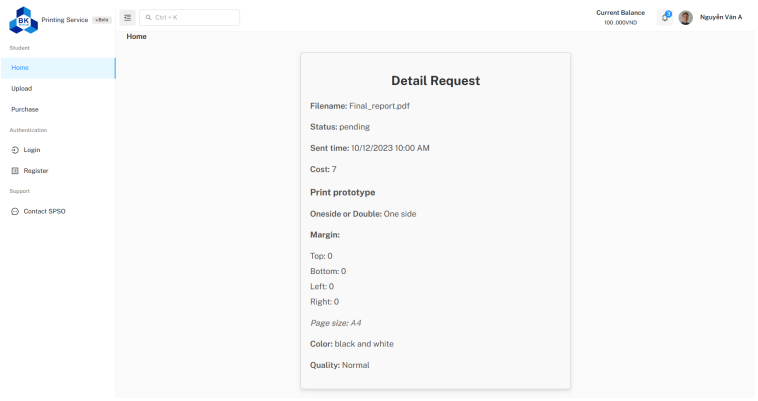

Figure 22 : Implementation of details in the printing request

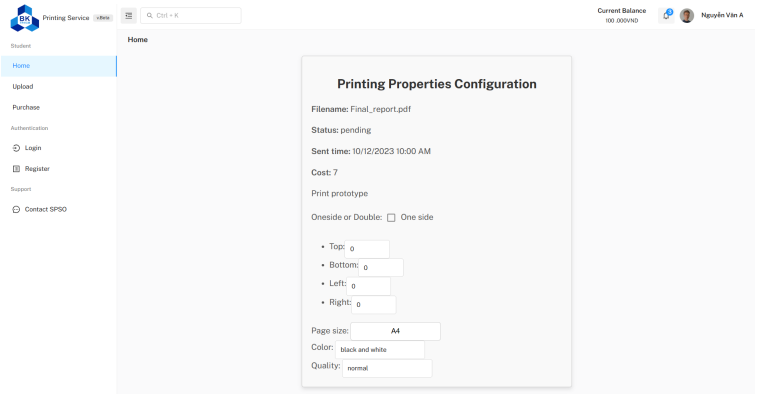

Figure 23 : Implementation of printing properties configuration in the printing request

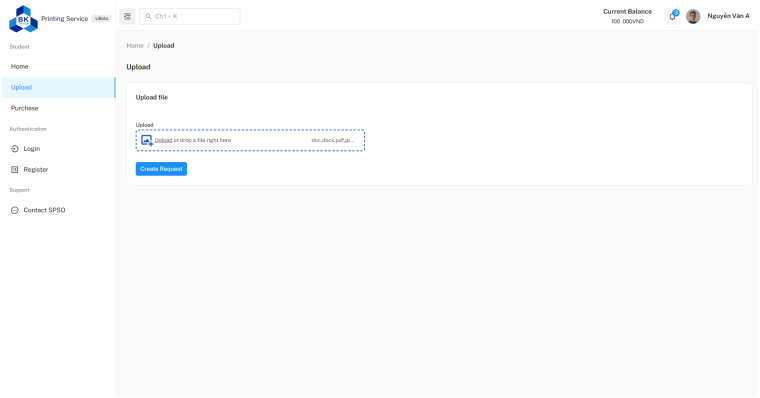

Figure 24 : Implementation of Uploading page

This page allows Students to drag or choose printing materials and then choose "Create request". The system will retrieve the information from those printing materials to generate new Printing Requests.

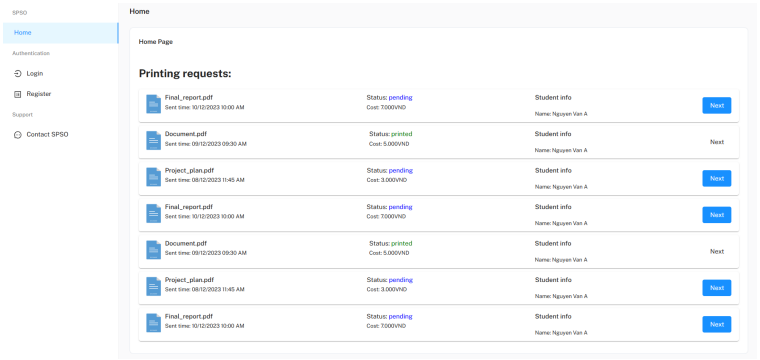

Figure 25 : Implementation of Requests Log in SPSO UI

This page allows SPSO to view the log of all printing requests as well as marked it from "pending" status to "printed" status.

6 Tasks division

Student ID	Full Name	Task	Rate
2010480	Nguyen Van Khanh Nhan	- Task 1: Functional requirement, Complete use-case Printing Request - Task 2: Activity Diagram for UC Upload Document for Change Status from "Pending" to "Processing". Figma Landing page - Task 3: Component diagram. - Task 5: Implement SPSO home page, upload page.	100%
2110098	Nguyen Tan Dung	- Task 1:Non-functional requirement, complete Whole-system Use case Diagram - Task 2: Activity Diagram for UC specify Printing Properties, Sequence Diagram for Upload, Figma Preview Page. - Task 3: Architectural Diagram, strategy for Presentation, Data storage, and API management. - Task 4: Define test strategy, and conduct the test. - Task 5: Implement the Student's Homepage, and Landing Page for the system, and add routes for the Student's pages. Adjust the Drawer and Header in MainLayout.	100%

Student ID	Full Name	Task	Rate
2115376	Vo Thi Hoang Yen	- Task 1: Non-functional requirement, Use Case Cancel Printing Request - Task 2: Activity Diagram for UC Upload file, Sequence Diagram for Specify, Figma Preview Page, Homepage SPSO - Task 3: Deployment diagram. - Task 4: Created from survey. - Task 5: Implement SPSO student.	100%
2114917	Nguyen Huu Thong	- Task 1: Functional requirement. Use Case Confirm Printing request. - Task 2: Activity Diagram for UC Confirm Printing Request, Sequence Diagram for Confirm Printing Request, Figma pop-up, HomePage Student. - Task 3: Deployment diagram, architectural diagram. - Task 4: Implement survey reports. - Task 5: Implement Homepage Student, function Sign In, Sign In Page.	100%
2011844	Lam Pham Trong Phuc	- Task 1: Stakeholders and their needs, benefits for each stakeholder and functional requirements. - Task 2: Activity Diagram for UC Preview Upload Document, Sequence Diagram for Change Status and Preview, figma Homepage. - Task 3: Component Diagram.	100%

Our group also uses Notion to manage tasks among members, as well as divide tasks into small tasks for each member to take.

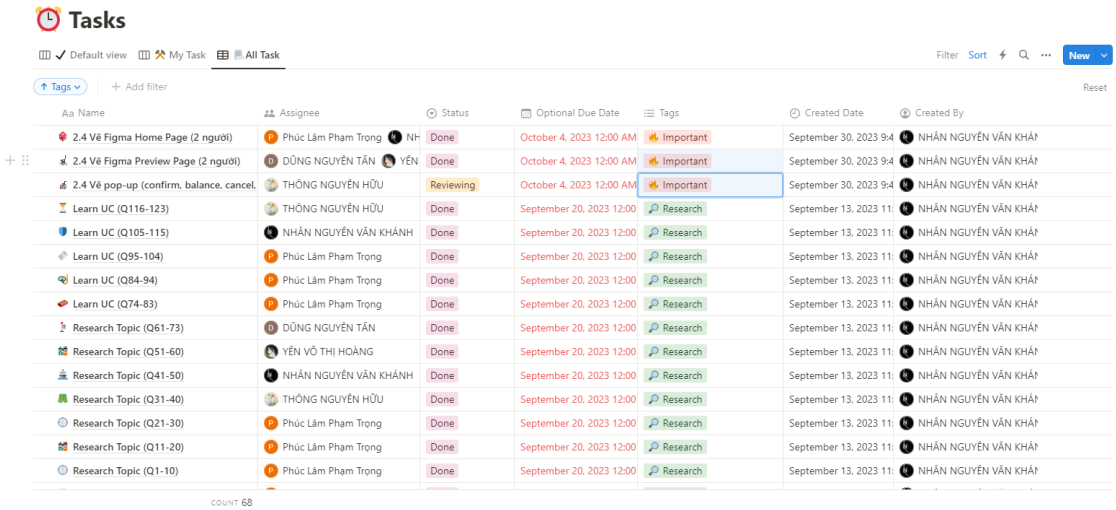

Figure 26 : Task control on Notion

7 Summary

During the implementation of this assignment, the group carried out stages in the process of developing software, from describing and defining the project requirements to implementing the interface and main functions of a system that supports waste processing in the city. Specifically, the team identified the context, business process, scope of the project, and requirements from users, system, functional and non-functional requirements, then continued to model the system and design the architecture. To clarify the changes in the system during operation and to describe the structure as well as understand the process of handling functions, the group has drawn UML Diagrams such as Use-case Diagram, Activity Diagram, Sequence Diagram, Class Diagram, Component Diagram, and Deployment Diagram.

Up to the report-submitted time, our group implementation of MVC2 has essentially met the necessary function of uploading and creating printing Requests, as initially listed. Even though the implementation has not been completed in terms of Database and BackEnd, the MVC2 can demonstrate properly how the system will work in the selected component as well as illustrate the Interface for both Users (Students) and SPSOs.

In addition to the achievements, the group also recognizes that there are many weaknesses that all members need to improve, mainly in teamwork. Because the workload is large and new, the task assignment and implementation have not been highly effective, causing the team to face many difficulties and challenges.

Looking on the bright side, the group members have gained a lot of experience in the process of implementing software through this exercise. This includes experience in teamwork, problem-solving, and technical aspects such as learning and applying new technologies to implement the exercise. It can be said that this topic has provided the team with a lot of important specialized knowledge about the process of developing software, from using UML Diagrams, GitHub, and leveraging technologies like ReactJs, and Material UI to complete the project.

Finally, our group would like to express our sincere thanks to the teachers who have enthusiastically guided us throughout the process of completing this assignment. Thanks to the feedback, answers and corrections from the teachers after each part, our work has been improved and become more complete.

References

[1] *Swimlane*. Link: https://circle.visual-paradigm.com/activity-diagram-example-swimlane/

[2] *What is Activity Diagram?*. Link: https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-activity-diagram/

[3] *What is Sequence Diagram?*. Link: https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-sequence-diagram/

[4] *What is Class Diagram*. Link: https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-class-diagram/

[5] *Component Diagram Tutorial*. Link: https://online.visual-paradigm.com/diagrams/tutorials/component-diagram-tutorial/

[6] *Usability Testing 101*. Link: https://www.nngroup.com/articles/usability-testing-101/