VIETNAM NATIONAL UNIVERSITY, HO CHI MINH CITY
HO CHI MINH UNIVERSITY OF TECHNOLOGY
COMPUTER SCIENCE & ENGINEERING DEPARTMENT

ASSIGMENT REPORT

Subject: Software Engineering

TASK 3: ARCHITECTURE DESIGN

Supervisor:	PhD. Truong Tuan Anh
Students:	
Nguyen Phuc Gia Khiem	2211573
Nguyen Quang Huy	2211235
Nguyen Quang Minh	2212063
Nguyen Tien Khoa	2211632
Nguyen Tran Dang Khoa	2211635

Ho Chi Minh City, November 9, 2024

CONTENTS

LIST OF FIGURES

Chapter 1. LAYERED ARCHITECTURE

1.1 Layered Architecture

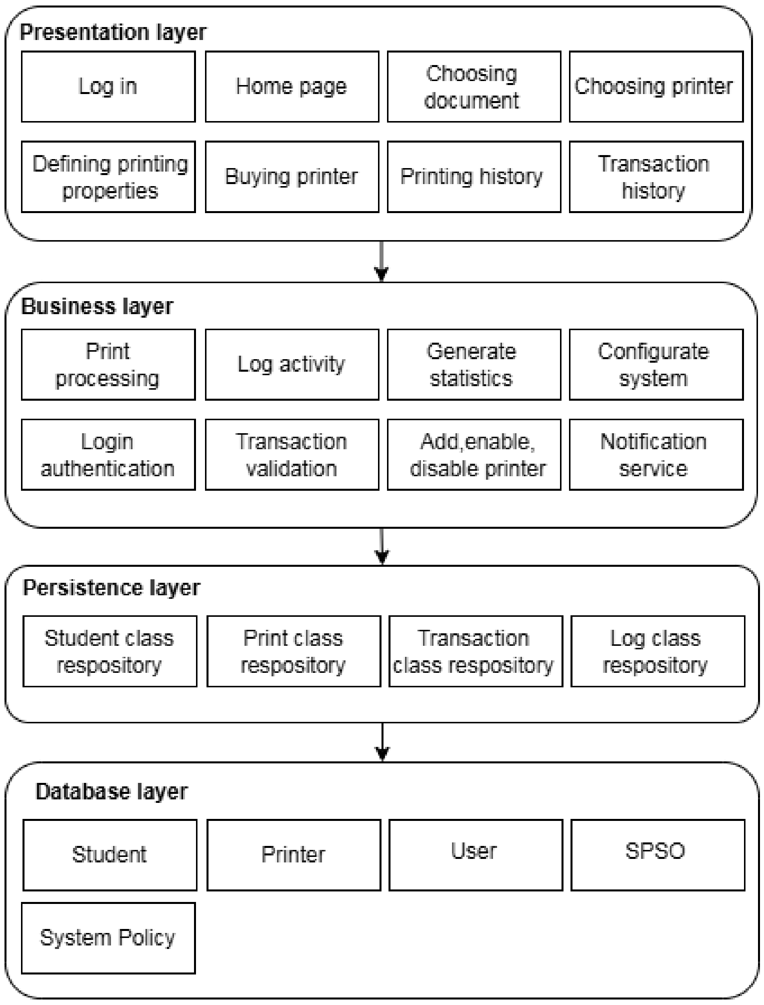

Figure 1.1: Layered Architecture

1.2 Presentation strategy

The **Presentation Layer** in the HCMUT-SSPS system is responsible for providing an intuitive, responsive, and user-friendly interface that bridges users and the underlying services. This layer facilitates user interaction with the system's functionalities and ensures a streamlined experience for both students and administrators. The design strategy prioritizes ease of navigation, responsiveness across devices, and role-based customization, ensuring that each type of user has a tailored experience.

The system is accessible via both **web-based**, with designs that adapt seamlessly to different screen sizes, providing a consistent user experience whether accessed from a laptop, tablet, or smartphone. It includes the following pages:

1.2.1 Log in Page

Purpose: The **Log in Page** serves as the entry point for users to access the HCMUT-SSPS system securely. It validates users through the HCMUT_SSO authentication service, allowing only authorized students and administrators to proceed.

User interface design

- **Minimalist Layout**: A simple and clean design with input fields for username (student ID or admin ID) and password.
- **Button Design**: A prominent "Log In" button is positioned below the input fields to make it clear and easily accessible.
- **Error Messaging**: In case of login failures (e.g., incorrect credentials), the page will display clear error messages, helping users understand the issue and guiding them to retry.
- **Forgot Password**: A link to assist users with forgotten passwords, which directs them to the HCMUT_SSO password recovery feature.

User Interaction Flow

- Upon entering credentials, users will click the "Log In" button.
- If the credentials are correct, they will be directed to either the **Student Interface** or **Admin Dashboard based** on their user type.
- In the case of login errors, error messages will appear dynamically, keeping users on the same page to correct their input without refreshing or reloading the page.

Integration with Authentication Service

- The **Log in Page** will interact with the HCMUT_SSO authentication service through a secure API request.
- All user credentials are securely transmitted to the HCMUT_SSO service for validation. Upon successful authentication, a session token will be generated to maintain the user session across pages, ensuring both security and seamless navigation within the application.

1.2.2 Home Page

Purpose: The **Home Page** serves as the primary entry point for students after logging into the system. It provides centralized access to essential functions, such as locating available printers, viewing printing history, and topping up page balances, ensuring students can navigate the system efficiently.

User Interface Design

- **Dashboard Layout**: The Home Page is organized with a dashboard-style layout, displaying key sections with large, easily recognizable icons or buttons for each feature (e.g., "Locate Printers," "View Printing History," "Top-Up Pages").
- **Quick Links**: Core functions are prominently positioned at the top of the page to facilitate rapid access, reducing the need for scrolling.
- **Notifications/Updates Panel**: An area displays any system notifications (e.g., printer maintenance alerts, balance low warnings) to keep students informed without navigating to separate sections.
- **User-Friendly Navigation**: Icons and labels are clear and straightforward, making it easy for users to identify and select desired options without confusion.

User Interaction Flow

- **Quick Access**: Upon logging in, students are immediately taken to the Home Page, where they can select from the main features with one click, ensuring a smooth start to their tasks.
- **Information Panels**: Relevant information, such as current page balance or recently used printers, is presented in concise panels, giving students at-a-glance updates on their status.

- **Responsive Design**: The page layout is responsive, adapting seamlessly to both desktop and mobile devices, ensuring a consistent experience across platforms.

Integration with Other Layers

- **Real-Time Data Access**: The Home Page integrates with the Business Logic Layer to retrieve live data, such as printer status and balance information, allowing students to see up-to-date details immediately.
- **API Access to Persistence Layer**: Through APIs provided by the Persistence Layer, the system can quickly fetch recent user activity and account details, ensuring that all displayed information, like page balances and printer locations, is accurate.
- **Notification System**: Notifications about system updates, maintenance, or low balances are dynamically updated via the Business Logic Layer, enabling the Home Page to reflect real-time changes without requiring manual refreshes.

1.2.3 Choosing printer Page

Purpose: The **Choosing printer Page** enables students to browse and select from a list of available printers on campus, helping them quickly identify and choose an operational printer for their printing tasks.

User Interface Design

- **Simple List Layout**: The page displays a straightforward, scrollable list of available printers, each entry showing essential details like printer ID, location (campus, building, room), and current status (e.g., "Available," "In Use," or "Out of Service").
- **Status Indicators**: Each printer entry features a status indicator (e.g., green for available, red for out of service) to give students a clear view of each printer's availability at a glance.
- **Sorting and Filtering**: Basic sorting options allow students to organize printers by criteria such as location or status, while filters help them narrow down results to specific buildings or campuses.

User Interaction Flow

- **Quick Selection**: Students can click on a printer entry to proceed to document upload and print settings, allowing a streamlined transition to the next steps.

- **Real-Time Status**: Printer status updates in real-time, so students are always working with the most current availability information without needing to refresh manually.
- **Minimal Navigation**: The simple list format minimizes navigation requirements, enabling students to quickly locate and select a printer without unnecessary steps.

Integration with Business Logic and Persistence Layer

- **Authentication**: Access to this page requires secure login via the HCMUT_SSO authentication service, ensuring that only authorized users can view and select printers.
- **Real-Time Data Access**: The Business Logic Layer retrieves printer data from the Persistence Layer to display an accurate list of printer statuses. This includes real-time availability updates to reflect any changes in printer usage.
- **Printer API**: The Printer API in the Persistence Layer is used to fetch a current list of printers and their statuses, allowing students to make well-informed selections based on updated information.

1.2.4 Printing properties Page

Purpose: The **Printing properties Pag**e enables students to set up document printing with tailored options, ensuring they have full control over their printing preferences and can adjust settings to meet specific requirements.

User Interface Design

- **Printing Options Panel**
  - **Paper Size**: Select between A4, A3, A2,... with size descriptions.
  - **Page Range**: Enter a custom range or choose "All Pages." Invalid ranges trigger an error message.
  - **Print Mode**: Toggle between Single-sided and Double-sided.
  - **Number of Copies**: Input the number of copies, with a limit to prevent excess.
- **Settings Review Panel**: Displays a summary of selected settings (paper size, page range, print mode, copies) before submission.
- **Real-Time Feedback**: Updates estimated page count as options are adjusted, helping manage page balance.

User Interaction Flow

- **Step-by-Step Process**: The layout guides students through each setting in a sequential, easy-to-follow format, from uploading a document to confirming their preferences.
- **Confirmation Step**: After setting all properties, students see a summary of their selections to confirm accuracy before submitting the print job.
- **Real-Time Balance Check**: An indicator displays the student's remaining page balance, updating in real-time if selections impact their balance, helping students manage their printing limits.

Integration with Business Logic and Persistence Layer

- **Authentication**: Secure login through the HCMUT_SSO authentication service ensures that only authenticated students access the printing setup.
- **File Type Validation**: Upon upload, the Business Logic Layer checks the document type against allowed formats through the System Policy API in the Persistence Layer, accepting or rejecting the file as needed.
- **Page Balance Check**: Before proceeding, the system verifies that the student's page balance is sufficient to cover the print job, using the Student API.
- **Printer API**: The page pulls printer details from the Printer API to display the current list of available printers and their capabilities, such as double-sided printing support.

1.2.5 Buying Page

Purpose: The **Buying Page** allows students to purchase additional pages when their balance is insufficient, ensuring they can complete their print jobs without interruption. It integrates with the BKPay payment system for a seamless, user-friendly transaction experience.

User Interface Design

- **Page Balance Display**: o Shows the current page balance and highlights if it is insufficient to complete the print job.
- **Top-Up Options**:
  - **Amount Selection**: Students can choose a predefined amount or enter a custom value to top up their balance.
  - **Payment Integration**: The page links directly to the BKPay payment system for easy and secure top-up processing.

- **Transaction Flow**:
  - **Instructions**: Step-by-step instructions guide students through the payment process, ensuring clarity.
  - **Confirmation**: A confirmation message is displayed after the payment, and the updated page balance is shown.

User Interaction Flow

- **Step-by-Step Process**: The page guides students sequentially, from checking their balance to completing the payment, ensuring an intuitive flow.
- **Confirmation Step**: After entering the payment amount and completing the transaction, a summary of the new balance is shown for confirmation.
- **Real-Time Balance Update**: Once the payment is successful, the page balance updates instantly, allowing students to see the new balance in real time.

Integration with Business Logic and Persistence Layer

- **Authentication**: Secure student authentication via the HCMUT_SSO system ensures only authorized users can access the page and perform transactions.
- **Transaction Validation**: The Business Logic Layer validates the payment amount and ensures the transaction is processed correctly via the BKPay integration.
- **Balance Update**: Upon successful payment, the updated page balance is saved through the Student API and reflects immediately on the user interface.

1.2.6 Printing History Page

Purpose: The **Printing History Page** allows students to view and manage their past print jobs. It provides detailed insights into each print job, including printer usage, page count, and cost, helping students track their printing activity and manage their resources effectively.

User Interface Design

- **History Display**:
  - **Print Job Details**: Shows a list of past print jobs, including details such as Printer used, Number of pages printed, and Total cost.
  - **Date Filter**: A filter option allows students to sort or search for print jobs by date.

- **Usage Summary**:
  - **Total Pages Used**: A summary section displays the total number of pages printed across all jobs.
  - **Category Breakdown**: Pages are categorized by paper size (e.g., A4, A3), providing an overview of printing usage.

User Interaction Flow

- **Step-by-Step Navigation**: The page is organized for easy navigation, guiding students from viewing their history to filtering and analyzing print job details.
- **Detailed View**: Students can click on any print job for more detailed information, such as printer specifications, the exact page range printed, and the cost.
- **Real-Time Updates**: The page automatically updates with new print job history as students complete print jobs, ensuring it stays current.

Integration with Business Logic and Persistence Layer

- **Authentication**: The HCMUT_SSO system ensures that only authenticated students can access their print history.
- **History Retrieval**: The system queries the Printing History API to pull detailed data, including printer usage and cost, for each print job.
- **Data Categorization**: The Business Logic Layer categorizes the print job data by paper size and calculates the total page count and associated costs, reflecting these in the summary section.

This interface is designed to be intuitive and responsive, helping students navigate seamlessly on both web and mobile platforms. The color scheme and typography are chosen to be visually appealing yet unobtrusive, supporting an efficient and pleasant user experience.

1.2.7 User Experience (UX) and Design Elements

- **Responsive Design**: Both the Student and Administration interfaces are optimized for different screen sizes. Buttons, text fields, and other interactive elements adjust automatically to ensure a smooth experience on mobile devices, tablets, and desktops.
- **Consistency and Branding**: The interface design is aligned with the HCMUT brand, using colors, logos, and fonts that are consistent with the university's visual identity. This helps reinforce brand recognition and creates a cohesive experience across all user touchpoints.

- **Error Handling and User Guidance**: Error messages are displayed in a clear, friendly tone, providing actionable steps to resolve issues. Tooltips and inline prompts are also used to guide users, particularly when configuring print settings or managing account balances.

1.3 Data storage approach

The application uses **PostgreSQL** to store data, there are some entities and relationships as well as constraints.

1.3.1 Entity types and Attributes

1. **USER**: People who interact directly with the application through their **HCMUT_SSO** account so that there are some main attributes of the users:

Attribute	Type	Description
Name	String	The full name of the user.
ID	String	This attribute represents a unique identifier to specify a user as well as distinguish that a user is **STUDENT** or **SPSO**.
Email	String	The email of the user, it should end with "@hcmut.edu.vn".

2. **STUDENT**: The entity type that specializes from **USER**, they have some own attributes.

Attribute	Type	Description
RemainingPages	Integer	A non-negative integer represents the number of available pages for printing jobs that the student can use.
TransactionHistory	Composite attribute	This attribute stores the information about buying more pages transactions, it is a combination of **Time** (Date), **PricePerPage** (Float), **No.ofPages** (Integer).

3. **SPSO**: The manager of the system.
4. **PRINTER**: The entity type represents the printers that are used to process print jobs.

Attribute	**Type**	**Description**
PrinterID	String	The unique identifier to distinguish the printers.
Location	String	The attribute refers to the location of the printer, it should be a string structured from the campus name, building name, and room number.
ManufacturerName	String	The name of the printer manufacturer.
PrinterNodel	String	The model of the printer.
ShortDescription	String	The paragraph mentions the information of the printer such as manufacture date, etc.

5. **SYSTEM POLICY**: The entity type is the configuration of the principles of the system about the initial pages for students, permitted file types, etc.

Attribute	**Type**	**Description**
ID	Integer	An identifier for the system to meet with the principles of SQL. In practice, there is only one instance for this entity type.
DefaultPages	Integer	The number of printing pages that a student can receive every semester.
AllocatedDate	Date	The date that the system provides default pages.
PermittedFileType	List	The list of file types that the system can handle such as "DOC, DOCX, PDF, etc".
MaxFileSize	Integer	The largest size of a file that the application can print.

1.3.2 Relationships and Attributes

1. **CHANGES (SPSO - SYSTEM POLICY)**: **SPSO** can config the system via **SYSTEM POLICTY**, this relationship allows the SPSOs change the attribute of the system.

Attribute	Type	Description
DefaultPages	Integer	The number of printing pages that a student can receive every semester.
AllocatedDate	Date	The date that the system provides default pages.
PermittedFileType	List	The list of file types that the system can handle such as "DOC, DOCX, PDF, etc".
MaxFileSize	Integer	The largest size of a file that the application can print.

2. **ENABLES (SPSO - PRINTER)**: The SPSOs can enable a printer to allow students to use it.

Attribute	Type	Description
Time	Date	The time that the printer is enabled.

3. **DISABLES (SPSO - PRINTER)**: The SPSOs can disable a printer to forbid students from using it.

Attribute	Type	Description
Time	Date	The time that the printer is disabled.

4. **PRINTS (STUDENT - PRINTER)**: A student can print document via a particular printer.

Attribute	**Type**	**Description**
FileName	String	The name of the file that is printed.
No.ofCopies	Integer	A positive integer refers to the number of copies that the student wants to print.
No.ofPages	Integer	Total pages that the printer needs to print the document corresponding to the page type.
PageType	String	The type of page used to print.
StartTime	Date	The date that the print job starts.
EndTime	Date	The date that the print job ends.
DoubleSided	Boolean	It is TRUE if the document is printed in double-sided principle and is FALSE if the principle is single-sided.

1.3.3 Sematic Constraints

1. **The User ID**: The student ID should start with two digits referring to their start year while the ID of SPSO should start with `00`.
2. **Enable and Disable**: A printer can only be enabled if it is in a disable state and reverse. In detail, if we call X, Y is the times that an SPSO A enables and disables printer B. We will have
$$X = Y$$
or
$$X = Y + 1$$
.

1.3.4 Enhanced Entity-Relationship Diagrams (EERD)

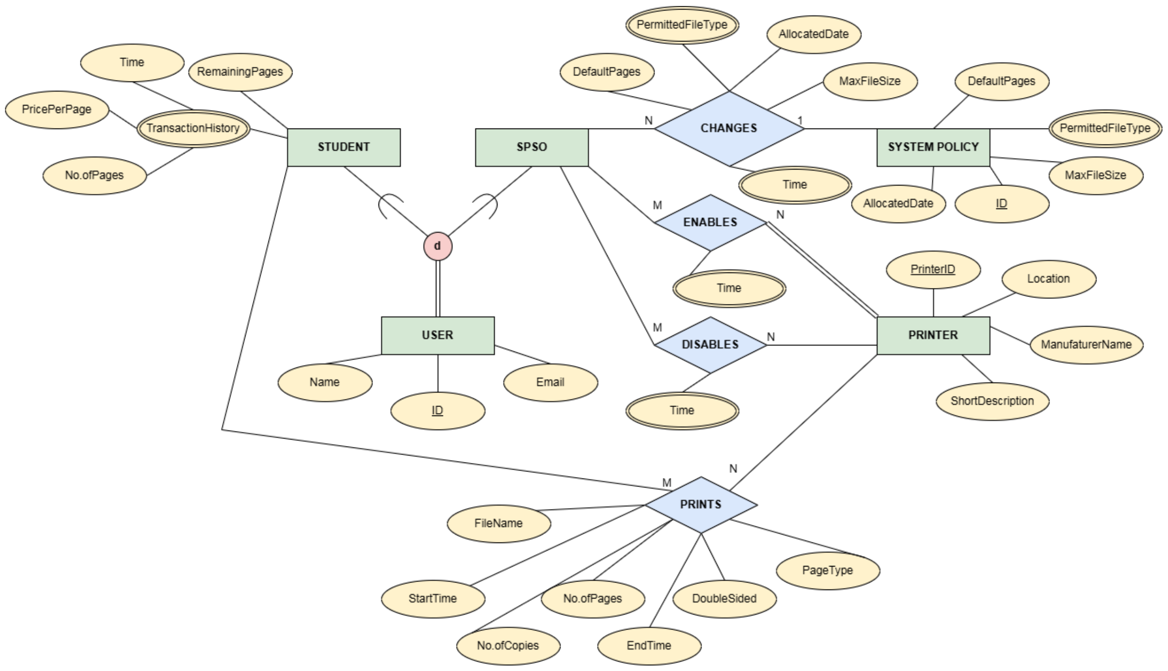

Figure 1.2: EERD for the system

1.4 API management

An API, or application programming interface, is a set of protocols and definitions that enable the creation and integration of software applications. APIs allow the product or service to interact with others without needing to understand their underlying implementations. For the HCMUT_SPSS system, three main API services are used:

- An API linking the Authentication Service and System Controller to validate user accounts.
- An API linking the Online Payment Service and System Controller to facilitate student purchases.
- An API connecting system components to transfer data between the controller and the database bidirectionally.

In this section, our group focuses on the third API service. There are four main object APIs, each corresponding to an object in the database, including Printer API, and Student API. Printing history API, System policy API.

1.4.1 Printer API

This object API includes the following methods

Method	**Parameters**	**Description**
getPrinterName	String (ID)	Retrieves the name information of a printer from the database using its ID. Return string value.
getPrinterBrand	String (ID)	Retrieves the brand information of a printer from the database using its ID. Return string value.
getPrinterStatus	String (ID)	Retrieves the status of a printer from the database using its ID. Return string value.
getPrinterBuilding	String (ID)	Retrieves the building location of a printer from the database using its ID. Return string value.
getPrinterRoomNumber	String (ID)	Retrieves the room number location of a printer from the database using its ID. Return string value.
getPagePrinted	String (ID)	Retrieves the total number of pages printed by a printer from the database using its ID. Return int value.
addNewPrinter	Object (Printer Information)	Creates a new printer entry and adds it to the database.
changePrinterStatus	String (ID)	Changes the status of a printer and updates this status in the database using its ID.

The API will be used in the following scenarios:

- **Printing Document**: When a student initiates the Printing Document function, it calls the PrinterAPI to update the number of pages printed by the printer.
- **View Printer's Log**: When an SPSO (admin) accesses the View Printer's Log, it calls the PrinterAPI to retrieve printer details.

- **Manage Printer's Configuration**: When an SPSO (admin) accesses the Manage Printer's Configuration, it calls the PrinterAPI to update a printer's status or add a new printer to the database.

These use cases ensure efficient management and tracking of printer operations within the system.

1.4.2 Student API

This object API includes the following methods.

Method	Parameter	Description
getStudentName	String (ID)	Retrieves the student's name from the database using their ID. Returns a string value.
getStudentEmail	String (ID)	Retrieves the student's email from the database using their ID. Returns a string value.
getStudentFaculty	String (ID)	Retrieves the student's faculty information from the database using their ID. Returns a string value.
addPrintingActivity	Object (Printing Activity)	Adds a new printing activity to the printing history list.
addTransactionActivity	Object (Transaction Activity)	Adds a new transaction activity to the transaction history list.
getPrintingTime	String (ID), Integer (Index)	Retrieves the time of a specified printing activity for the student, based on activity index. Returns a time value.
getPrintingFileName	String (ID), Integer (Index)	Retrieves the filename of a specified printing activity for the student, based on activity index. Returns a string value.

Method	Parameter	Description
getPrintingPaperNumber	String (ID), Integer (Index)	Retrieves the number of pages printed in a specified printing activity for the student, based on activity index. Returns an integer value.
getPrintingPaperType	String (ID), Integer (Index)	Retrieves the type of paper used (A3, A4, A5) in a specified printing activity for the student. Returns a string value.
getPrintingLocationBuilding	String (ID), Integer (Index)	Retrieves the building location of the printer used in a specified printing activity for the student. Returns a string value.
getTransactionTime	String (ID), Integer (Index)	Retrieves the time of a specified transaction activity for the student, based on transaction index. Returns a time value.
getTransactionCost	String (ID), Integer (Index)	Retrieves the cost of a specified transaction activity for the student. Returns an integer value.
getTransactionPage	String (ID), Integer (Index)	Retrieves the number of pages purchased in a specified transaction activity for the student; by default, the paper type is A4. Returns an integer value.

When a student navigates to their personal account page, the controller uses this API to retrieve and display all necessary information. For example, in the User Interface (UI) designed by our group, the account page includes the following details:

- **Student Name**: (e.g., Nguyen Tien Khoa)
- **Student ID**: (e.g., 2211632)
- **Student Email**: (e.g., khoa.nguyentien@hcmut.edu.vn)
- **Student Faculty**: (e.g., CSE)

- **List of All Printing Activities**: Displays the history of all printing activities.
- **List of All Transactions**: Shows a history of all transactions.

Additionally, each time a student completes a printing task, this API is called to log both the printing activity and corresponding transactions in the database, ensuring records are updated in real time.

1.4.3 Printing History API

This API is specifically designed for admin use to review the printing history of all students, and for users to review their own printing history, includes only two methods:

- GetAllPrintingHistory: Called by the admin to retrieve a complete list of all printing activities, ordered chronologically from the most recent activity to the earliest.
- AddNewPrintingHistory: Automatically called whenever a student completes a printing task, adding that printing activity to the database.

These methods enable the admin and user to monitor and maintain a detailed log of student printing activities effectively. Each student can only review their own history.

1.4.4 System Policy API

This object API includes the following methods.

Method	**Parameter**	**Description**
getDefaultPageNumber	None	Retrieves the default number of pages allocated to each student. Return int value.
setDefaultPageNumber	None	Updates the default number of pages allocated to each student.
getAllocDate	None	Retrieves the date when the system allocates the default number of pages to all students. Return time value.

Method	**Parameter**	**Description**
setAllocDate	None	Updates the date when the system allocates the default number of pages to all students. Return time value.
getMaximumPageSize	None	Retrieves the maximum allowed file size for printing. Return int value.
setMaximumPageSize	None	Updates the maximum allowed file size for printing.
getPermittedFile	None	Retrieves a list of all file types permitted for printing. Return list of string values.
setPermittedFile	None	Updates the list of file types permitted for printing.

This API is utilized when the admin needs to modify system configurations, including:

- Changing the Default Number of Pages: Adjusting the number of pages allocated to each student by default.
- Setting the Allocation Date: Specifying the date on which the system will assign the default number of pages to all students.
- Updating Permitted File Types: Modifying the list of file types that the system accepts for printing.

These configuration adjustments help the admin customize and maintain control over the system's printing policies.

Chapter 2. COMPONENT DIAGRAM FOR PRINT DOCUMENT MODULE

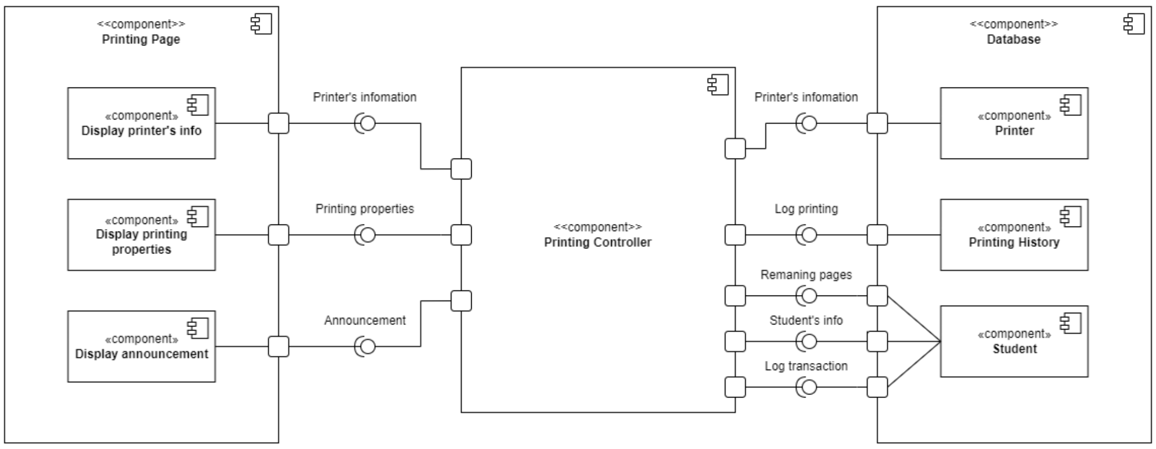

Figure 2.1: Component Diagram for Print document module

The component diagram above shows the components of the system in the Print Document module. The **Printing Page** component is the user interface when a student prints a document. **Printing Page** consists of 3 smaller components:

- **Display printers' info** component: the section that displays a list of printers and their information such as printer model and location for students to choose. This requires `Printer's information` interface from the component **Printing controller**.
- **Display printing properties** component: the section that allows students to specify printing properties such as paper size, the number of pages to be printed, one/double-sided, and the number of copies. This requires `Printing properties` interface from the component **Printing controller**.
- **Display announcement** component: requires `Announcement` interface from the component **Printing controller** and consists of
  - The page that notifies students that their document is printing and reminds students to go to the printer to take the document.

  - The page that notifies students that his/her page balance is not enough and requires he/she to buy more pages.

The **Printing controller** component, which provides interfaces for **Printing Page** component and requires interfaces from **Database** component.

- **Printing controller** component requires `Printer's information` interface from **Printer** component in **Database** to provide for **Display printer's info** component.
- **Printing controller** component requires `Log printing` interface from **Printing History** component in **Database** to log the printing action of students as well as printers.
- **Printing controller** component requires `Remaining page` interface from **Student** component in **Database** to check whether the page balance of the student is enough or not.
- **Printing controller** component requires `Student's info` interface from **Student** component in **Database** to log the printing actions, which requires student's information.
- **Printing controller** component requires `Log transaction` interface from **Student** component in **Database** to log the number of pages that a student buys and the amount of money he/she pays.

The **Database** component consists of 3 smaller components: **Printer component**, **Printing History** component, and **Student** component. The database component delegates the interfaces provided to its internal components through ports.