Contents

I. Document Change Log

Date	Version	Description	Author
10/19/2025	1.0	Created document	Truong Do Vuong
10/21/2025	1.1	Added Functional/NonFunctional Requirements	Truong Do Vuong
10/24/2025	1.2	Rewrite Requirements, added FDD	Truong Do Vuong
10/26/2025	1.3	Added Class diagram	Hoang Van Hung
10/27/2025	1.4	Added UseCase diagram	Nguyen Tran Viet Nhat
10/28/2025	1.5	Added UseCases Detail descriptions	Nguyen Tran Viet Nhat
10/28/2025	1.6	Added UI-Design	Truong Do Vuong
10/28/2025	1.7	Added DataFlow diagram level 0,1	Le Huynh Anh Khoi
10/29/2025	1.8	Added Database Design	Hoang Van Hung
10/30/2025	1.9	Added DataFlow diagrams level 2	Le Huynh Anh Khoi

Specification of Student Management System

Group 1

Version 1.9

October 30, 2025

II. Student Management system

The **Student Management system (SMS)** is designed to administrative activities within a school or university. The **system** provides essential functions such as **managing students, grades, and user accounts for students and Administrators.**

1 Main Functional Requirements

1.1 Student Registration and Profile Management

- **Administrators** can register new **students** by entering information such as full name and citizen ID.
- During registration the student's **StudentID** is set exactly to their citizen ID (i.e., **StudentID = CitizenID**).
- The system issues a default temporary password equal to the student's date of birth (stored and applied in MMDDYYYY format). The student is required to change this default password at first login.
- Newly registered students can log in using their StudentID (citizen ID) and the default password, then change their password and update additional profile information including **date of birth, gender, enrollment year, address, contact number, and email, portrait**.
- **The system** allows viewing and (where permitted by role) updating of student information.
- Each student has a unique ID for identification within the **system** (the citizen ID).

1.2 Fee and Financial Management

- Financials page
  - **Quick access from profile:** From the student's main profile page a single link ("Financials") opens a consolidated financial view for that student.

  - **Profile-level summary:** The top of the financial view shows: **StudentID** (which equals CitizenID), Date of Birth (for verification), Full Name, Major/Program, Enrollment year, total outstanding balance, and next due date.
  - **Line-item fee list:** A compact table lists every charge related to the student (course tuition, lab fees, insurance, hostel, fines, etc.) with columns: Fee Code, Description, Period, Amount, Due Date, and Status (paid/partial/overdue).
  - **Transaction summary:** Under the line-items show: Subtotal, Discounts/Scholarships, Taxes (if any), Amount Paid, and **Total Due** (final balance to pay).

- **Payment page:**
  - Displays StudentID (CitizenID), DOB, Name at the top for confirmation.
  - Shows the same line-item list with checkboxes allowing full or partial selection for payment.

1.3 User Authentication and Role Management

- **The system** supports two types of users: Administrator and Student.
- Each user logs in with a username and password.
- Password recovery is available via registered email.
- Permissions and access levels differ by role:
  - **Administrator:** Full system control and data management
  - **Student:** View profile

1.4 ADMIN Management

- **Administrator landing page:** When an **Administrator** logs in the system displays an **Admin Dashboard** instead of the student home page. The dashboard's primary purpose is administrative data management and quick navigation to operational tools.
- **Table selector (primary control):** A prominent control (dropdown or left-side list) lets the admin choose which database table to open. table choices include: `Students`, `Payments`.
- **Data grid / table viewer:**
  - Displays the selected table in a sortable and filterable grid with column.
  - Every entries can be **Edit**,

– Every change are "soft" meaning that **Admin** have to "Save" **comfirm change** button to save.

1.5 Notifications and Announcements

- **Administrators** have the option to post announcements in **Admin Dashboard**.
- **students** receive notifications for new announcements in the login page and after login.

2 Functional Decomposition Diagram

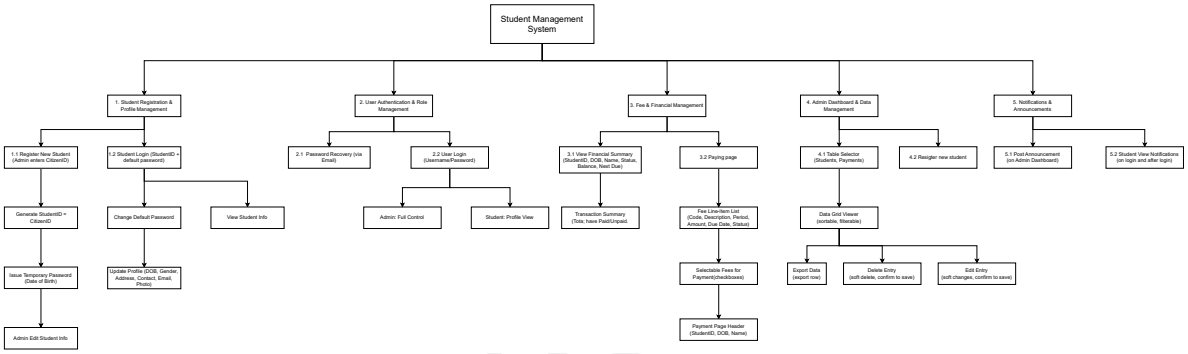


Figure II.1: FDD for Student Management System

3 Non-Functional Requirements

3.1 Usability

- Simple and intuitive user interface for basic testing and demonstrations.
- Layout is partially responsive, suitable for desktop browsers.
- Minimal visual design focused on clarity and ease of understanding rather than aesthetics.

3.2 Reliability

- Intended for local testing; continuous 24/7 uptime is not required.
- Occasional restarts or maintenance are acceptable in a learning environment.
- Data loss is not critical; database resets may occur during experiments.

3.3 Performance

- Optimized for small-scale testing (up to 10 simultaneous users).
- Response time within 1–3 seconds under normal test load.
- Performance tuning (e.g., caching, load balancing) is not a priority for this learning project.

3.4 Supportability

- Codebase is modular and commented for easy understanding by students.
- Documentation provided for setup, execution, and testing steps.

3.5 Security

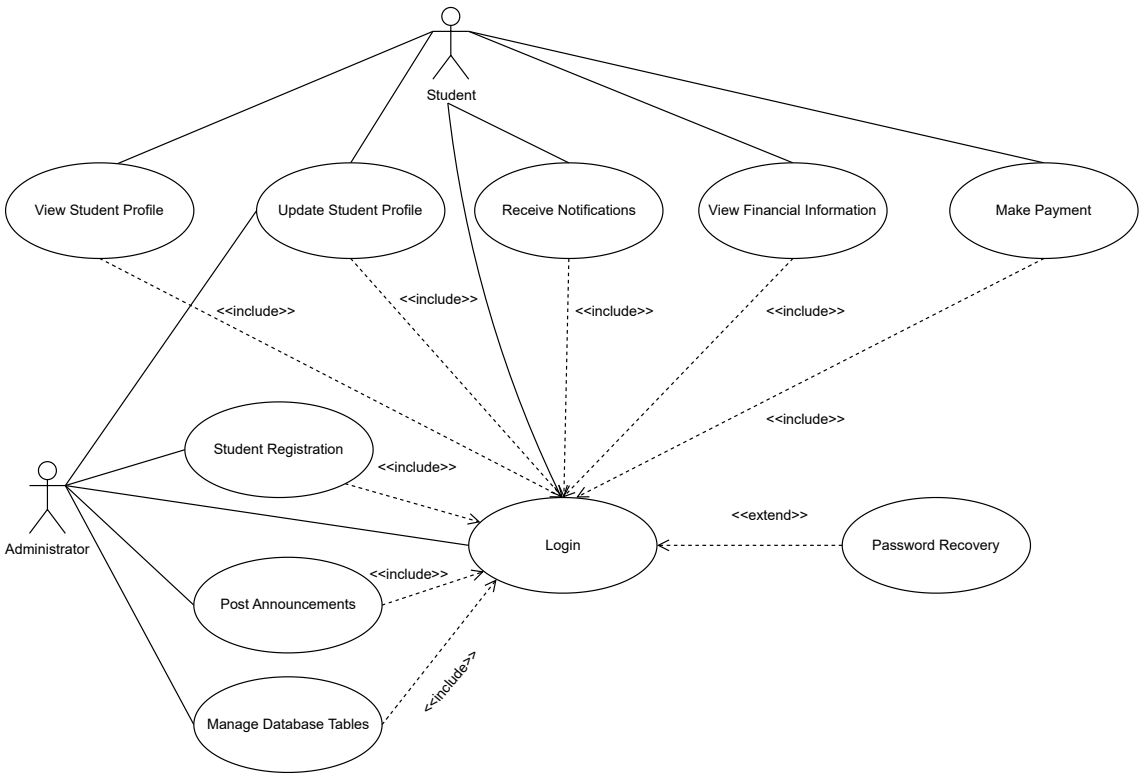
- Authentication for users (student, admin roles).
- Input validation to prevent common mistakes, simple injection attempts, DOS/DDOS.

3.6 Design Constraints

- Platform: Web-based (HTML5, CSS3, Python, MongoDB).
- Operating System: Works on Windows, Linux, or macOS.
- Framework: May use Flask or Django for backend logic.
- Deployment: Localhost, free-tier hosting (e.g., Render, Replit,Vercel) or Tunneling (VScode port forwarding or Ngrok)

III. Use-Case Diagram

1 Diagram Overview



1.1 List of Actors

No.	Actor	Meaning
1	Student	Student using the system.
2	Administrator	System administrator.

1.2 List of Use Cases

No.	Use Case	Notes (Group)
1	Login	2
2	View Student Profile	1
3	Update Student Profile	1
4	Receive Notifications	5
5	View Financial Information	5
6	Make Payment	3
7	Student Registration	1
8	Post Announcements	5
9	Manage Database Tables	4
10	Password Recovery	2

2 Use Case: Login

2.1 Summary

This use case allows a user (student or administrator) to log in to the system using a valid username and password to access role-specific features. The system also supports password recovery via the registered email.

2.2 Flow of Events

2.2.1 Main Flow

1. The use case starts when the user selects the Login function on the system home page.
2. The system displays the login form, including: username and password fields.
3. The user enters login credentials and presses the `Continue` button.
4. The system determines the account type:
   - If the account is a student, the student home page is displayed.
   - If the account is an administrator, the administrator home page is displayed.

2.2.2 Alternate Flows

2.2.2.1 Invalid Username/Password If, in the main flow, the user enters an incorrect username or password, the system displays an error message. The user may choose to return to the beginning of the main flow or cancel the login; in that case the use case ends.

2.3 Special Requirements

None.

2.4 Preconditions (System state before the use case)

The user is not logged into the system.

2.5 Postconditions (System state after the use case)

If the use case is successful, the user is logged in and may use functions appropriate to their role. Otherwise, the system state remains unchanged.

2.6 Extension Points

- Password Recovery: The user may switch to the password recovery function if they forgot their password.

3 Use Case: View Student Profile

3.1 Summary

This use case allows a student to view their personal information in the Student Management System. Displayed information includes: student ID, name, avatar (profile photo), date of birth, gender, address, phone number, email, and major.

3.2 Flow of Events

3.2.1 Main Flow

1. The use case starts when the student wants to view personal information.
2. The student has successfully logged into the system.
3. The system displays the student home page.

4. The student selects `More Information` (to view detailed information).
5. The system retrieves the information from the database.
6. The system displays the student's information including:
    - Full name
    - Student ID (StudentID = CitizenID)
    - Avatar (profile photo)
    - Date of birth, gender
    - Address, phone number, email
    - Major
7. The use case ends when the system displays the full student profile.

3.2.2 Alternate Flows

None.

3.3 Special Requirements

None.

3.4 Preconditions

- The student is logged into the system.
- The system is operating normally.

3.5 Postconditions

- The student's personal information is displayed on the interface.
- No changes are made to the database data.

3.6 Extension Points

None.

4 Use Case: Update Student Profile

4.1 Summary

This use case allows a logged-in student to update their personal profile information in the Student Management System. A student may edit fields such as date of birth, gender, address, phone number, email, or avatar. Some fields are fixed and cannot be changed (student ID, full name, major).

4.2 Flow of Events

4.2.1 Main Flow

1. The use case begins when the student wants to edit personal information.
2. The student has successfully logged into the system.
3. The student selects `Update Info`.
4. The system navigates to the Update Info page.
5. The student enters new information.
6. The student clicks the `Update` button.
7. The system validates the new information.
8. If valid, the system updates the database and displays the message: "Profile updated successfully."
9. The use case ends when the profile information is successfully updated.

4.2.2 Alternate Flows

4.2.2.1 Invalid Information If the entered information is invalid (e.g., phone number in incorrect format or left empty):

- When the user clicks `Update`, the system will not save the entered information and will display an error message: "Invalid information. Please check again. e.g.: Date of birth must not be empty."
- The user reviews and corrects the information and then clicks `Update` again.

4.3 Special Requirements

Fixed fields (Student ID, full name) cannot be changed.

4.4 Preconditions

- The student is logged in and on the student home page.
- The system is operating normally.

4.5 Postconditions

- If the update is successful, the student's personal information in the database is changed and the updated record is displayed.
- If the update fails, the database remains unchanged.

4.6 Extension Points

None.

5 Use Case: Receive Notifications

5.1 Summary

This use case allows students to receive notifications and news from the administrator within the Student Management System. Notifications may include academic news, tuition deadlines, policy changes, or urgent alerts. Students can view these notifications at the login page or after logging into the system.

5.2 Flow of Events

5.2.1 Main Flow

1. The use case starts when a student wants to view notifications sent by the administrator.
2. The administrator uses the admin home page to post an announcement.
3. The announcement is saved into the database.
4. When a student is on the login page, a panel displays the announcements posted by the administrator; or after logging in, the student selects `View notifications` to see announcements.
5. The student selects a specific notification to view detailed content.
6. The use case ends when the student finishes reading the notification.

5.2.2 Alternate Flows

None.

5.3 Special Requirements

None.

5.4 Preconditions

The student is either logged into the system or is on the login page.

5.5 Postconditions

None.

5.6 Extension Points

None.

6 Use Case: View Financial Information

6.1 Summary

This use case allows a student to view their outstanding financial information in the Student Management System. Information includes tuition fees, other fees, payment status, total outstanding balance, and due dates.

6.2 Flow of Events

6.2.1 Main Flow

1. The use case starts when a student wants to check tuition and related fees.
2. The student has successfully logged into the system.
3. From the home page, the student selects `Financial Summary`.
4. The system retrieves the student's financial information from the database.
5. The system displays the student's financial obligations.
6. The use case ends when the financial information is successfully displayed.

6.2.2 Alternate Flows

None.

6.3 Special Requirements

None.

6.4 Preconditions

The student is logged into the system and is on the student home page.

6.5 Postconditions

The system displays the student's financial information.

6.6 Extension Points

None.

7 Use Case: Make Payment

7.1 Summary

This use case allows a student to pay their outstanding balances in the Student Management System. The student may choose to pay partially or fully. After selecting `Pay`, the system saves the transaction to the database and updates the payment page.

7.2 Flow of Events

7.2.1 Main Flow

1. The use case starts when a student wants to pay tuition or other fees.
2. The student has successfully logged into the system and accesses the payment function.
3. The Payment page displays unpaid charges.
4. The student selects the charges to pay.
5. The student clicks `Pay`.
6. The system saves the payment in the database.

7. The system displays a confirmation message indicating successful payment.
8. The use case ends when the system confirms the payment and updates data.

7.2.2 Alternate Flows

None.

7.3 Special Requirements

None.

7.4 Preconditions

The student is logged into the system and on the student home page.

7.5 Postconditions

A confirmation message of successful payment is displayed.

7.6 Extension Points

None.

8 Use Case: Post Announcements

8.1 Summary

This use case allows an administrator to post announcements, news, or instructions on the Student Management System's home page.

8.2 Flow of Events

8.2.1 Main Flow

1. The use case starts when an administrator wants to create and post a new announcement to students.
2. The administrator logs into the system successfully.
3. From the administrator home page, the admin selects `Post Announcements`.
4. The system displays the announcement creation interface, including:

- Title
- Content

5. The admin enters the announcement content and clicks `Post`.
6. The system confirms and displays the message: "Announcement posted successfully."
7. The use case ends when the announcement is posted successfully.

8.2.2 Alternate Flows

None.

8.3 Special Requirements

None.

8.4 Preconditions

The administrator is logged in and on the administrator home page.

8.5 Postconditions

The new announcement is saved in the system and displayed to students.

8.6 Extension Points

None.

9 Use Case: Manage Database Tables

9.1 Summary

This use case allows the administrator to directly manage and manipulate database tables in the Student Management System (CRUD operations).

9.2 Flow of Events

9.2.1 Main Flow

1. The use case starts when the administrator wants to perform CRUD tasks on system data.

2. The administrator logs into the system successfully.
3. The system displays the Admin panel.
4. The admin selects one of the tables to manage.
5. The system displays the corresponding data records.
6. The admin can perform CRUD operations.
7. The admin clicks `Save` to persist changes.
8. The system displays the message: "Changes saved."
9. The use case ends when the system updates the data successfully.

9.2.2 Alternate Flows

9.2.2.1 Quick Student Registration In the Student table there is an option `Register new student` for quick account creation: only Username and Password need to be entered. Other fields remain empty/default until the student logs in and updates them.

9.3 9.3 Special Requirements

None.

9.4 Preconditions

The administrator has logged in successfully and is on the administrator home page.

9.5 Postconditions

Modified data is updated and saved in the system.

9.6 Extension Points

None.

10 Use Case: Password Recovery

10.1 Summary

This use case allows a user to create a new password when they forget their password on the Student Management System.

10.2 Flow of Events

10.2.1 Main Flow

The use case begins when the user selects `Forgot password` on the login page.

1. The system requests that the user enter their email address.
2. The system checks whether the email exists in the system. If the email exists, the system sends a new password to the user's email and notifies the user of the recovery result.
3. The use case ends when the user successfully logs in with the new password.

10.2.2 Alternate Flows

- Email does not exist in the system: The system informs the user, who may retry this function or exit.

10.3 Special Requirements

The new password must be generated randomly.

10.4 Preconditions

None.

10.5 Postconditions

If the use case succeeds, the password is changed. Otherwise, the system state remains unchanged.

10.6 Extension Points

None.

IV. Data Flow Diagrams

1 Diagram Overview

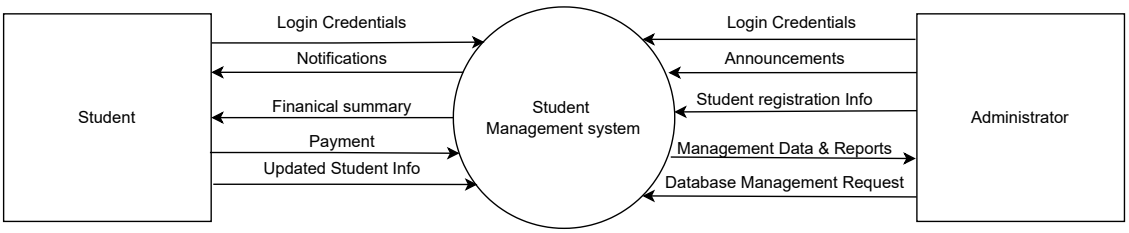


Figure IV.1: Data Flow Diagrams Level 0

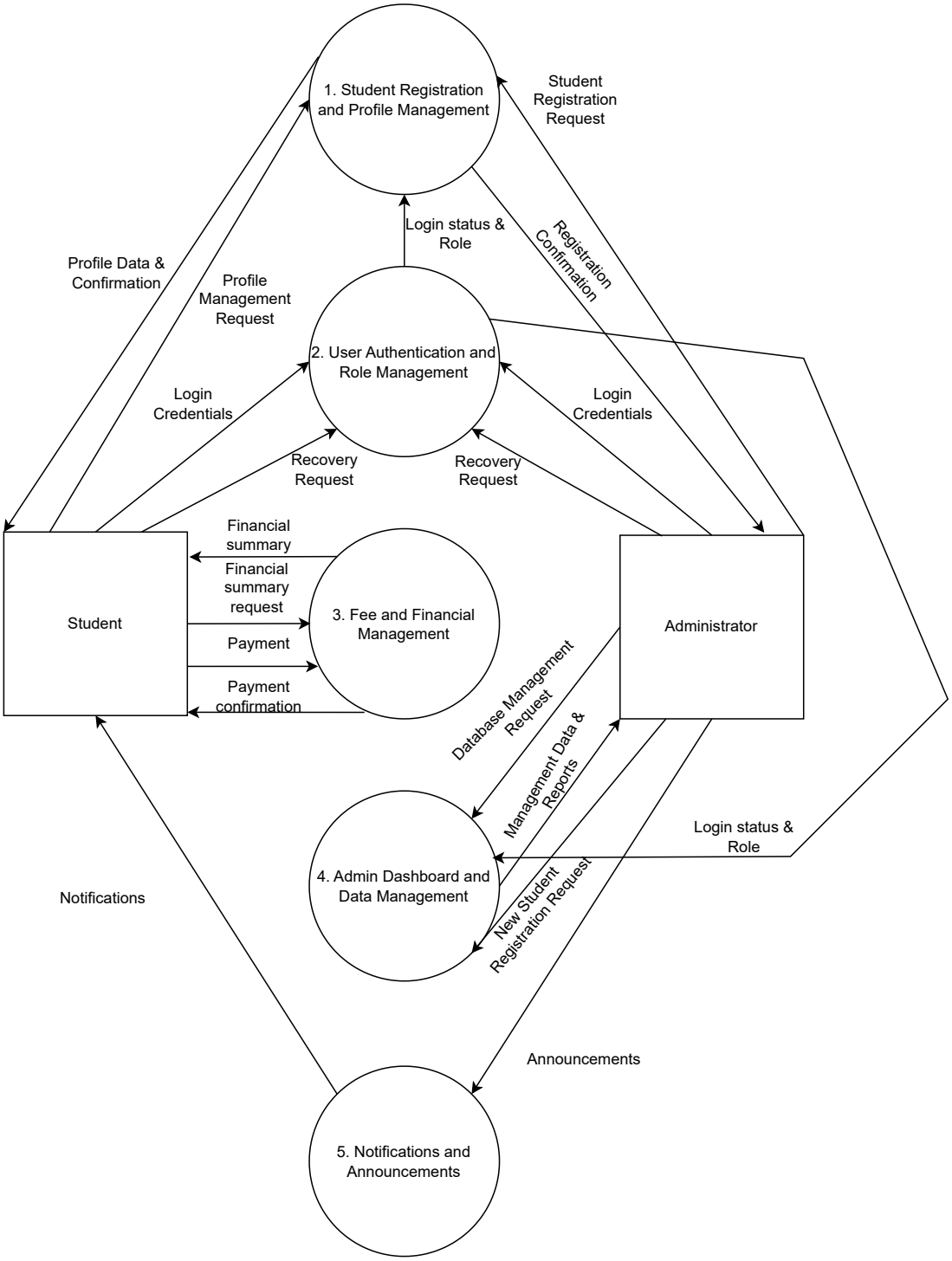


Figure IV.2: Data Flow Diagrams Level 1

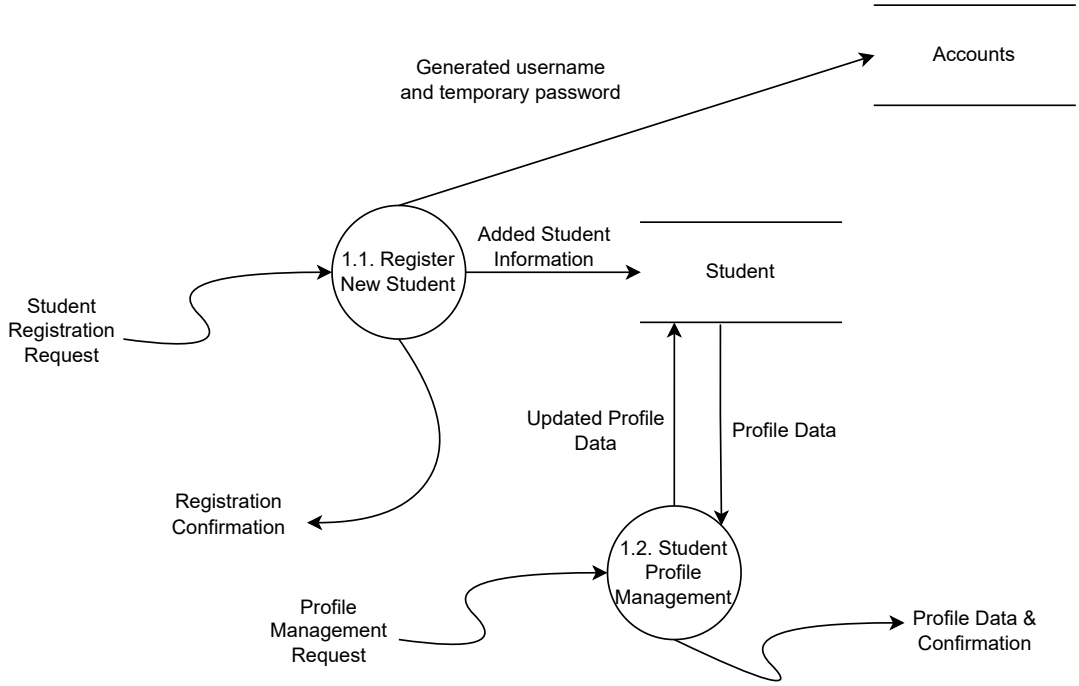


Figure IV.3: Data Flow Diagrams Level 2.1

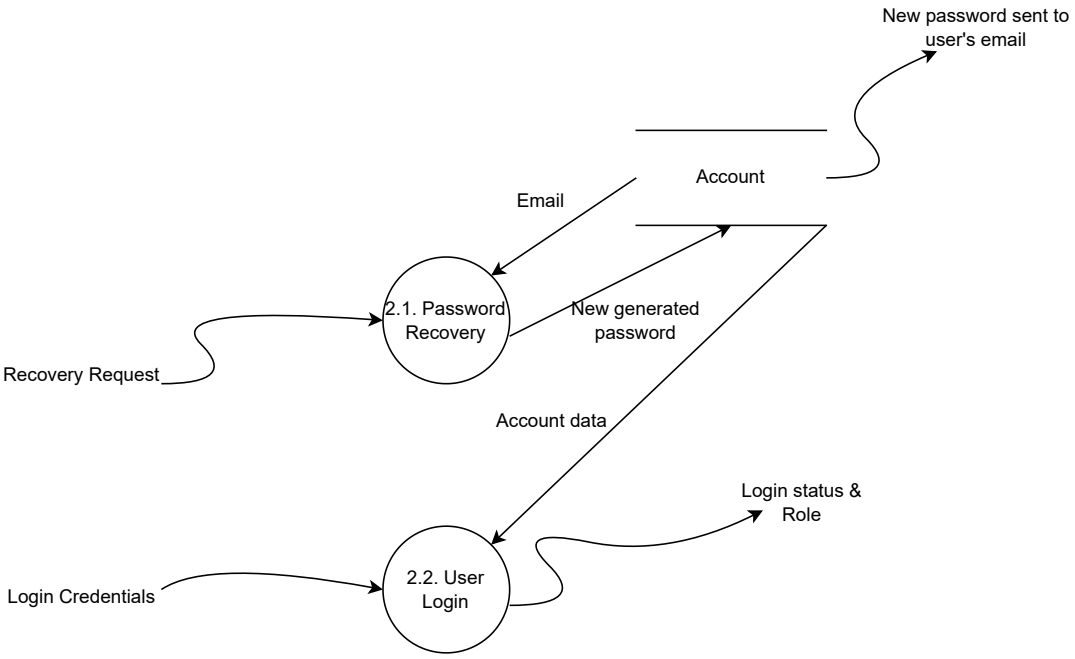


Figure IV.4: Data Flow Diagrams Level 2.2

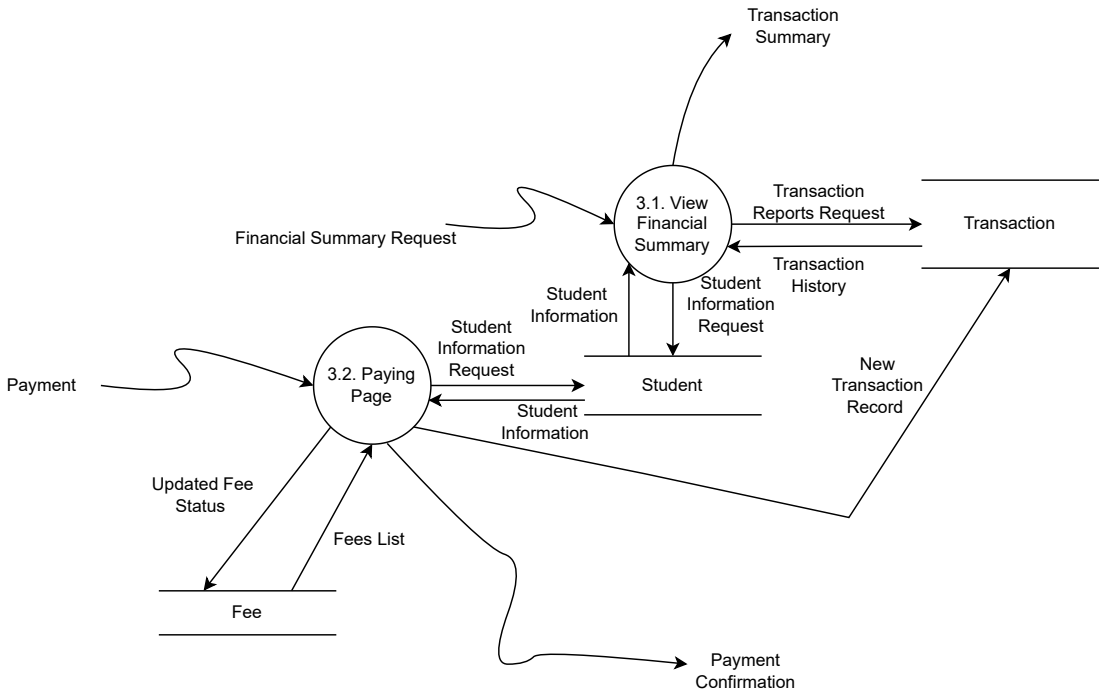


Figure IV.5: Data Flow Diagrams Level 2.3

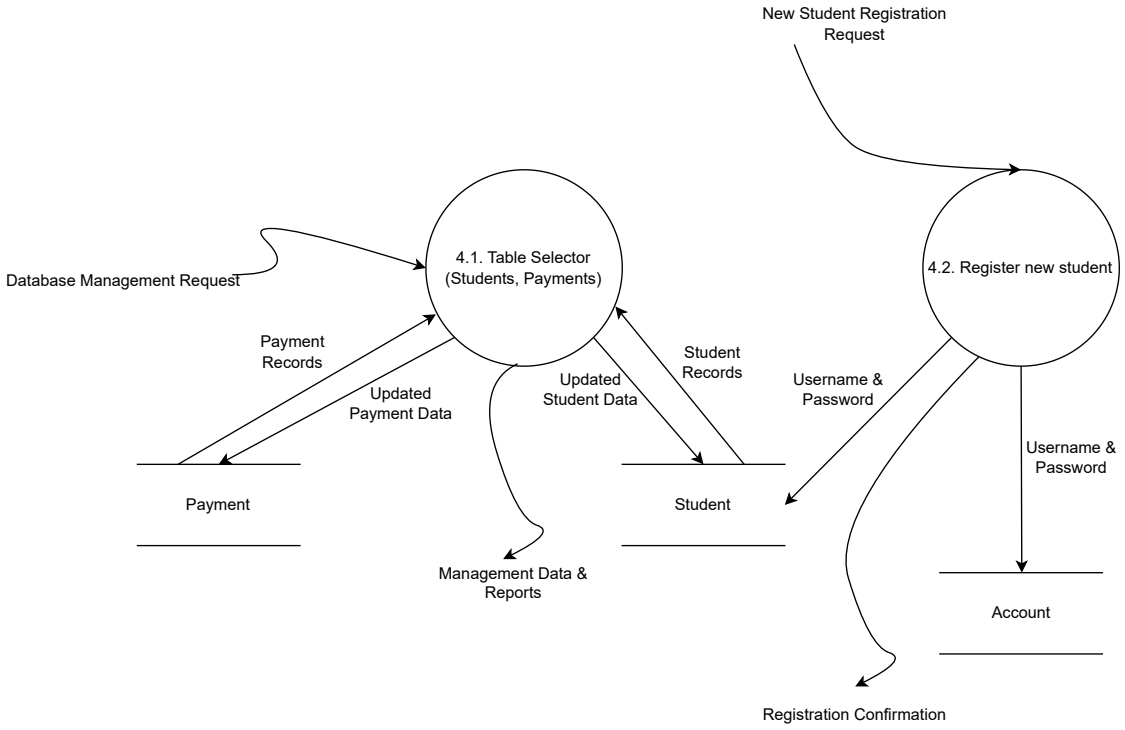


Figure IV.6: Data Flow Diagrams Level 2.4

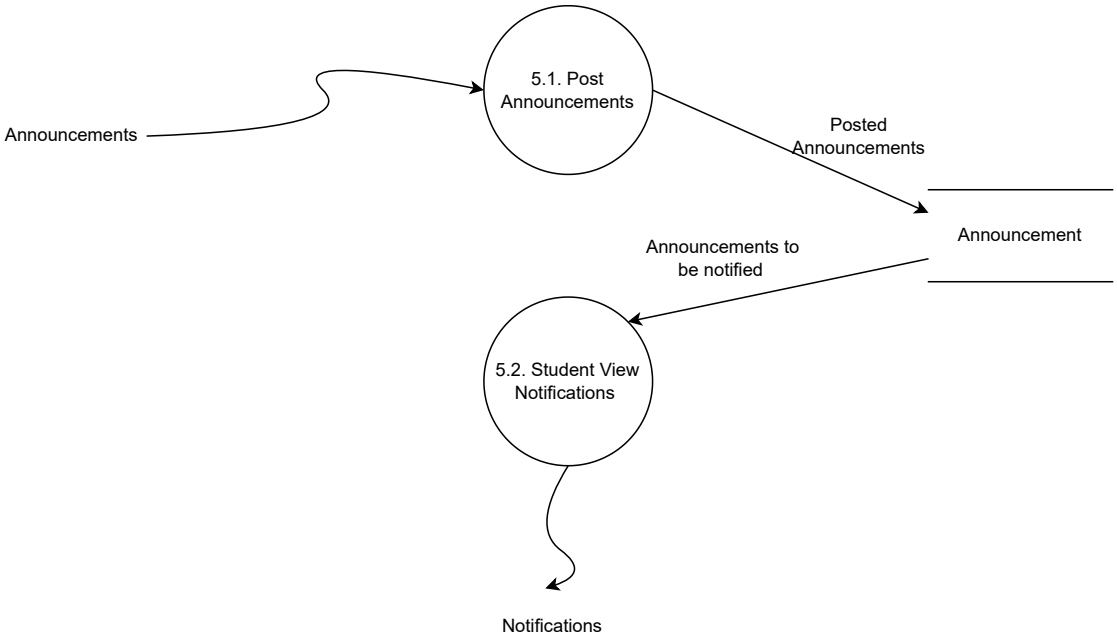


Figure IV.7: Data Flow Diagrams Level 2.5

V. Class Diagram

1 Diagram Overview

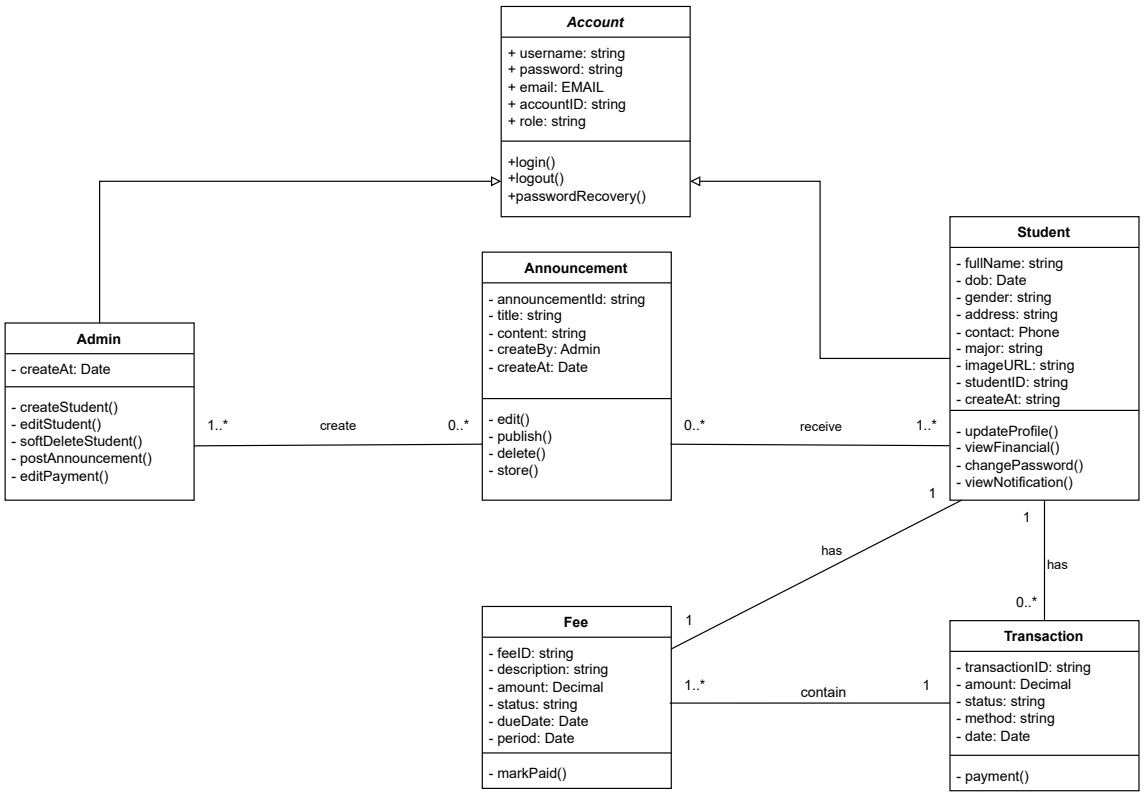


Figure V.1: Class Diagram