Database and Information System Assignment Report

PigeonBroadcast: A messaging system for colleges

Hanshu Rao
Sichen Li
Ziqin Ma
Shibo Wang

Group 2

May 26, 2023

1 System Description

1.1 Introduction

PigeonBroadcast is an information system for posting messages inside a college. It allows the faculty, administrative staff and students to send and check messages so that they do not have to joining in numerous Wechat group or receiving emails that could be missed. The system consists of two modules in the view of a user, which are:

1. Epistles: for the faculty and other staff to post messages,
2. Feathers: for individual students and student organization managers to post messages.

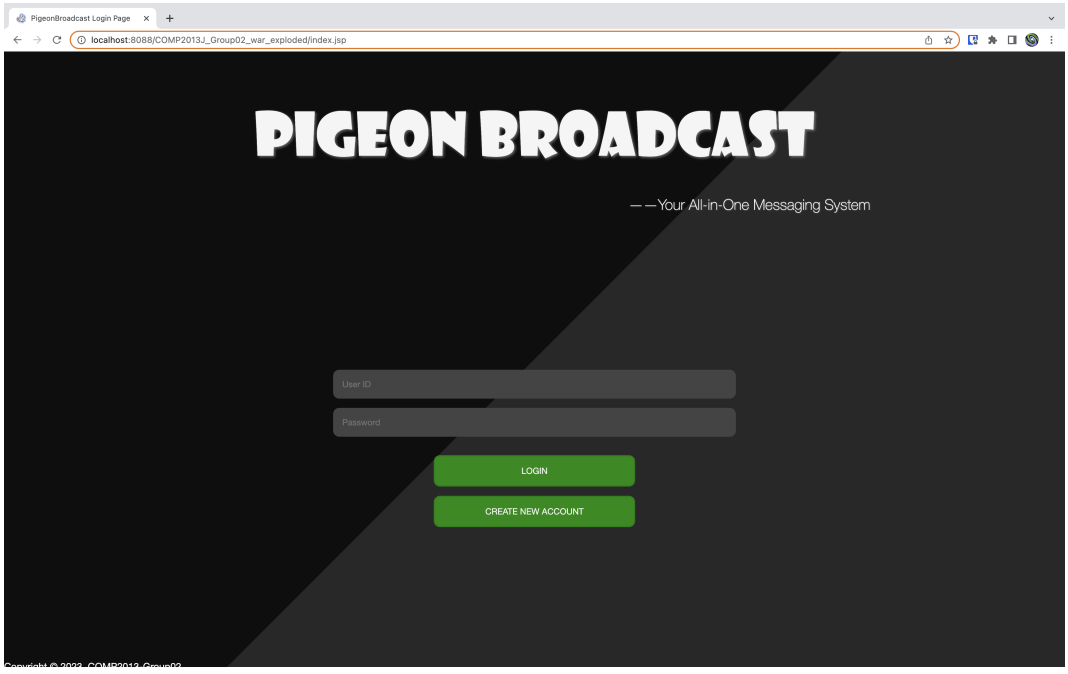

Figure 1: The login page of PigeonBroadcast

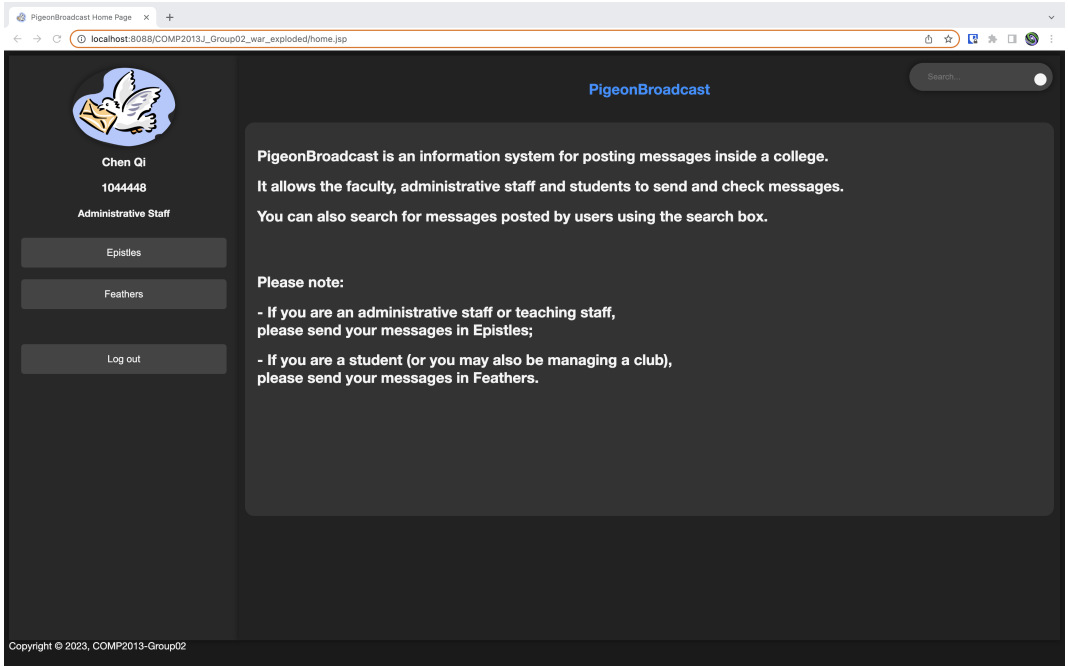

Figure 2: The home page of PigeonBroadcast

1.2 Database Summary

The database includes five tables, which are `notification` for storing messages sent by users, `organization` and `organizationmembers` for storing information about student organizations and its managers, `schoolmembers` for maintaining the information of all member within the college, and `user` for keeping users information. The details of the construction and design of this database will be presented in the "Tasks" part in the report. The database follows the first, second and third normal form.

1.3 Features

1.3.1 Automatic Database Initialization

Before running the project, the user is required to add his or her MySQL account into the `database\_info.txt` so that the project can access the database management system. When the project run at the first time, it will check if there is a database called "pigeon_broadcast" in the database management system and create one with all the table it needs if there is not. The tool methods that help to implement that feature are in `JDBCTool.java` and `TextTool.java` in `pb.tool`. The SQL code for creating the database and its tables can be seen in `database\_init.sql`.

For the college that intend to apply this system, it is required to provide a txt file with all its members in it, which format is expected to be `1397751 Zhang San`. The project already contains a default txt file called `school\_member.txt` for demonstration use. When creating the database, this txt file will be read into the `schoolmembers` table in the database.

1.3.2 Account Creating

PigeonBroadcast enables users to sign up in the login page. When a user trying to create an account, it will check if the member ID and name can be found in the `schoolmember` table. In order to sign up successfully, the member ID and name must match one certain row in the table, otherwise the registration process will fail.

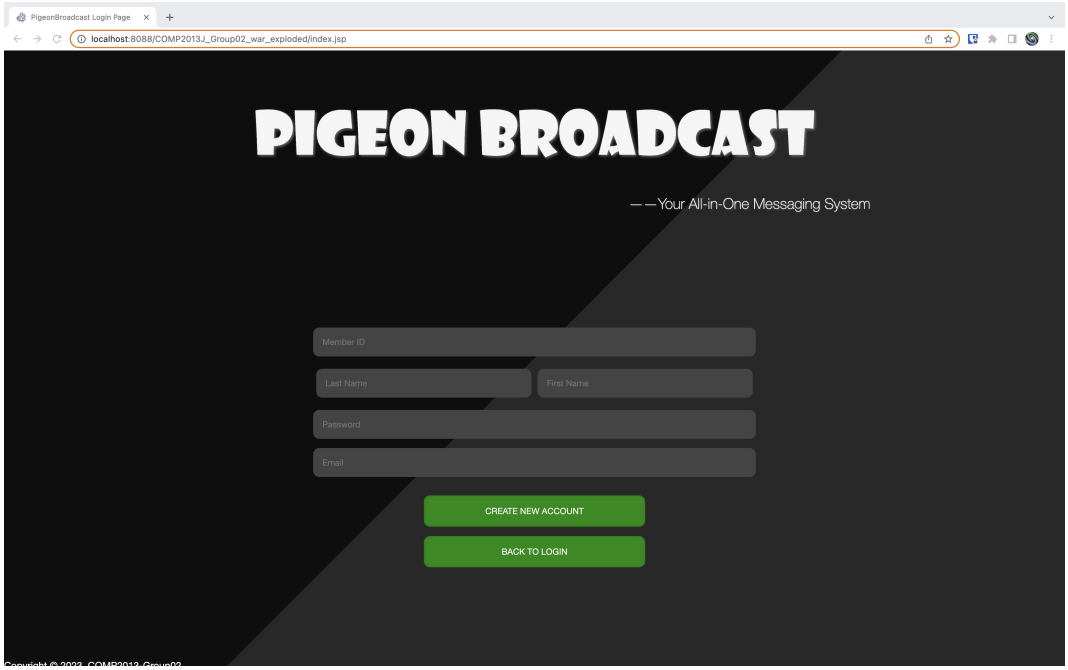

Figure 3: The sign-up page of PigeonBroadcast

1.3.3 Permission Checking

PigeonBroadcast includes a permission mechanism, which reinforces the functionality of the two parts. For instance, if a user is registered as a "administrative staff" or "teaching staff", he or she will be able to send messages in Feathers and vice versa. This feature is thanks to an attribute in the table `user` named `identity`. For example, in `epistlePage.jsp` where the Epistles part is implemented, there is some code to perform the permission checking. When a user try to send a message in a module where

```
String type = null;
if (Objects.equals(user.getIdentity(), "admin")) type = "school";
else if (Objects.equals(user.getIdentity(), "teacher")) type = "course";

if (type != null) {
    Notification notif = new Notification(title, message, type, time, user.getUserID());
    NotificationDAO.insertNotification(notif);
} else {
    out.print("Unable to send message. Perhaps you do not have the permission.");
}
```

Listing 1: Permission Checking Code

he or she does not have the permission, the web page will report as follow.

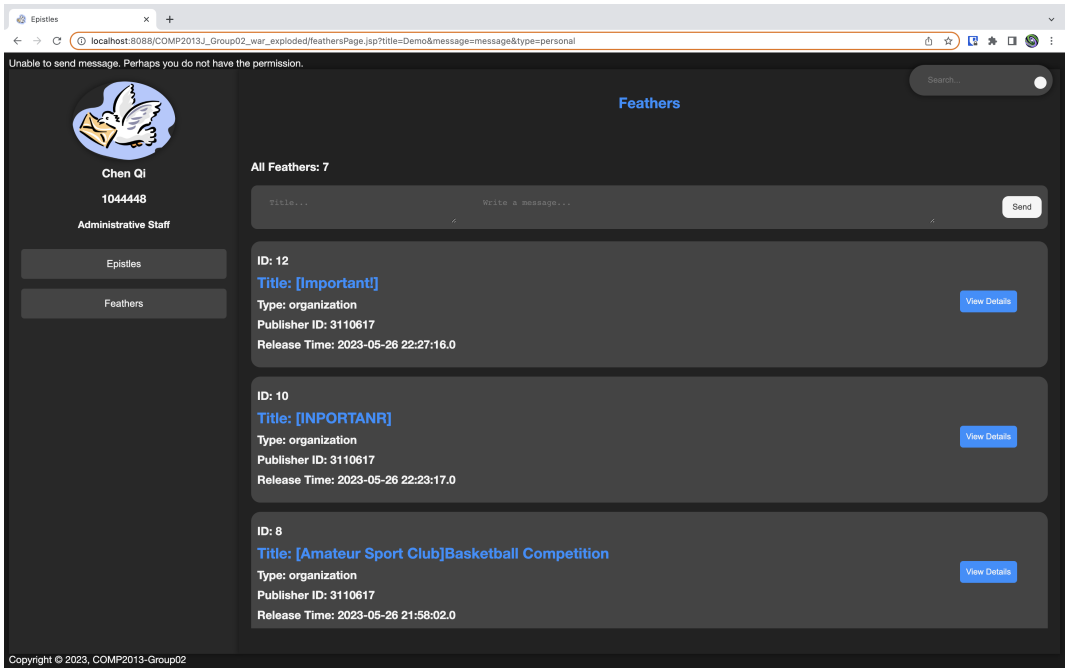

Figure 4: Fail to send a message in due to insufficient permission

Additionally, users cannot edit others' messages. The "modify" button and "delete" button will not be presented when one is checking messages send by other.

1.3.4 Message Searching

Users can search messages freely using the search box. They can either use global search to get any messages that contain the target string or use modifiers to clarify what part of the message they want to search by. For example, if they want to search messages by title, they can input *#TI + (what they want to search)* to find the messages they desire.

There are six modifiers that can be used to help the user search easily. They are:
#ID: to search messages by their IDs,
#TI: to search messages by their titles,
#CO: to search messages by their contents,
#TY: to search messages by their types,
#RD: to search messages by their release date,
#PI: to search messages by the ID of their publishers.

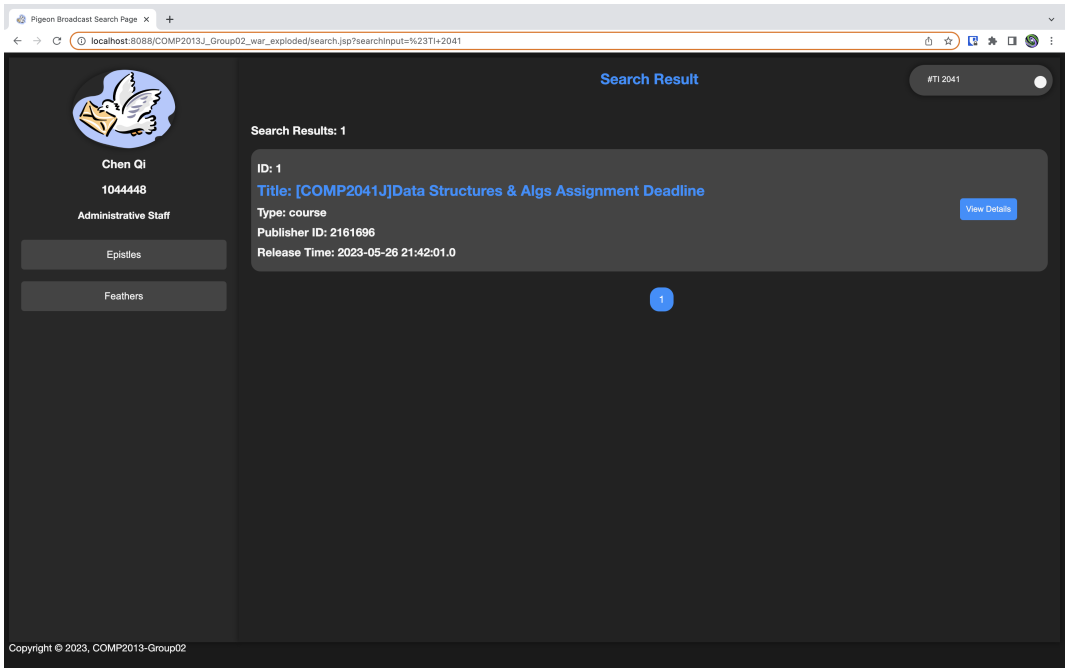

Figure 5: Searching messages with the title "2041"

1.3.5 Message Modification and Deletion

Users are allowed to modify or delete the message they send. When processing the logic, Pigeon-Broadcast will firstly check if the title and message are not empty, then create a POJO to store the notification data and call the method 'NotificationDAO.updateNotification' to process the update operation in SQL. After that, it will redirect to Epistles or Feathers according to the identity of the user. In the details page, the user can modify and delete his or her message.

```
// Check if the title and message are not empty
if (title != null && !title.trim().isEmpty() && message != null && !message.trim().isEmpty()) {
    LocalDateTime now = LocalDateTime.now();
    Timestamp time = Timestamp.valueOf(now);

    String type = (String) session.getAttribute("notif-type");

    if (type != null) {
        Notification notif = new Notification(title, message, type, time, user.getUserID());
        NotificationDAO.updateNotification(notif, noteID);
        if (Objects.equals(user.getIdentity(), "admin") ||
            Objects.equals(user.getIdentity(), "teacher")) {
            response.sendRedirect("epistlesPage.jsp");
        } else {
            response.sendRedirect("feathersPage.jsp");
        }
    } else {
        response.sendRedirect("errorPage.jsp");
    }
}
```

Listing 2: Code for updating message

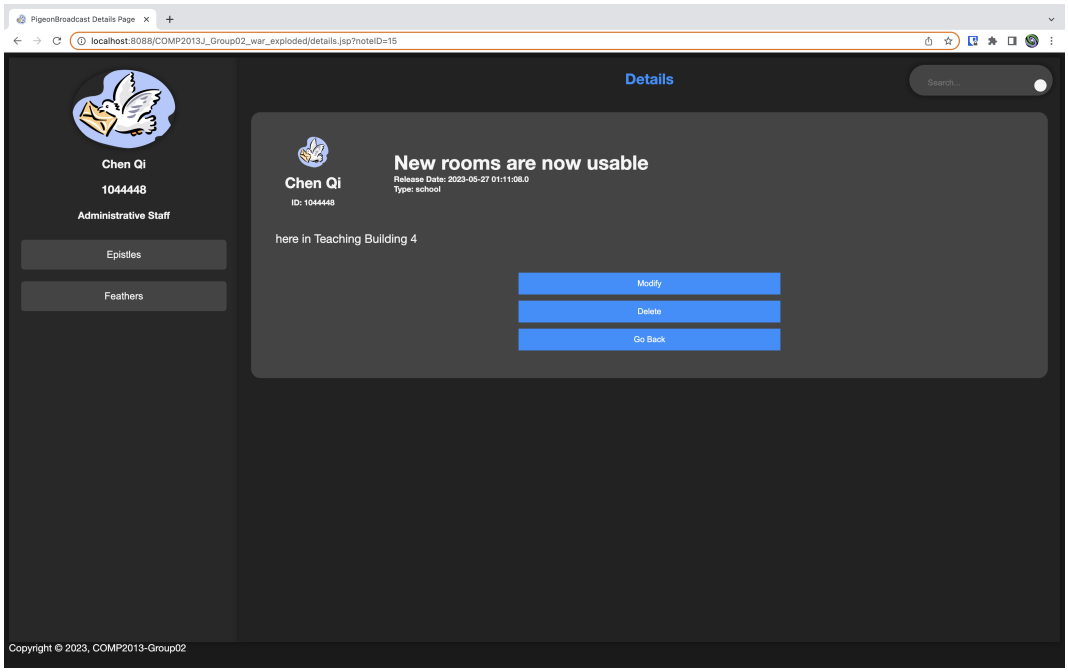

Figure 6: Details page with modify and delete button

2 Tasks

Based on the basic idea of this system, possible tables and corresponding attributes were considered and an E-R diagram describing the relationship between tables and corresponding functions was drawn which later was mapped to a relational model.

The table `schoolmembers` which table stores all personnel information in the school, including the member ID and name of each person. `memberID` of the three type of personnel start with 3, 2, and 1 respectively. It also includes an attribute called `indentity`, an enumeration to distinguish administrative staff, teaching staff and students. Since this is a information system for people within the school, only people with `memberID` information can register an account. The table `organization` allows people to send information in the name of a organization, which makes the system more adaptive.

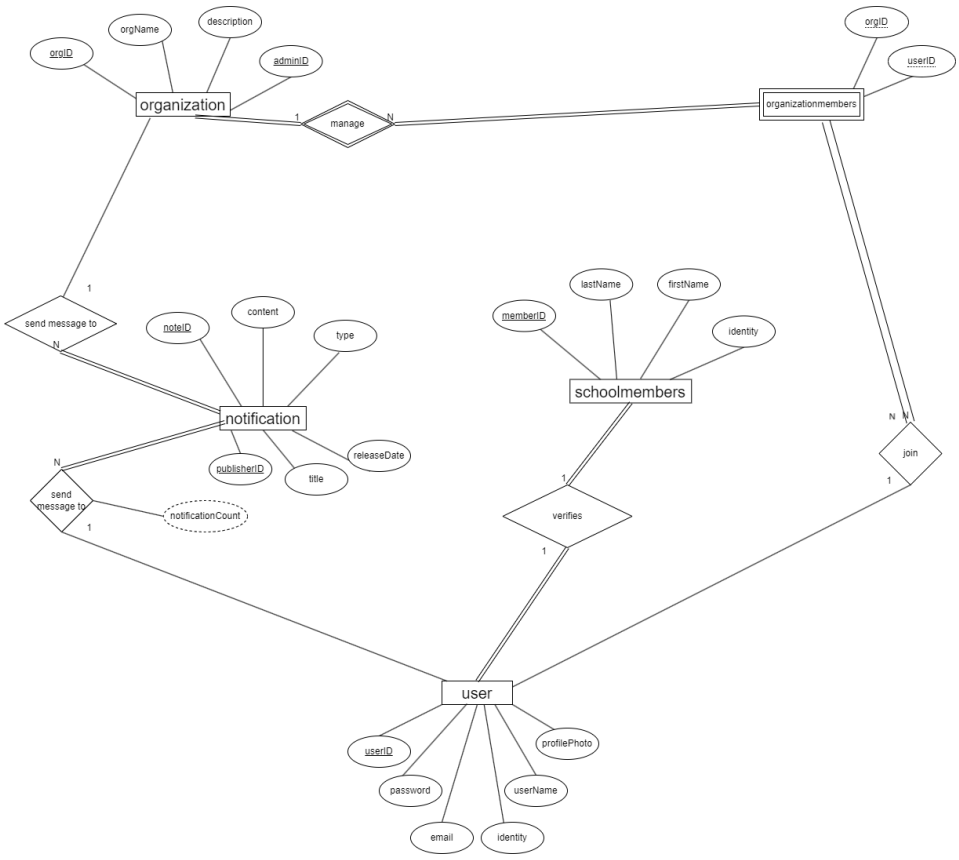

Figure 7: The ER diagram of the database

3 Team Member Contribution

- Hanshu Rao: Maintainer of the architecture, and author of main back-end JSP code.
- Sichen Li: Proposer the initial architecture and author of the DAO and POJO code and the related JSP code.
- Ziqin Ma: Designer of the ER diagram of the database and implementer of the related SQL code.
- Shibo Wang: Designer of the web page style and author of the related front-end JSP code.

4 System Architecture Diagram

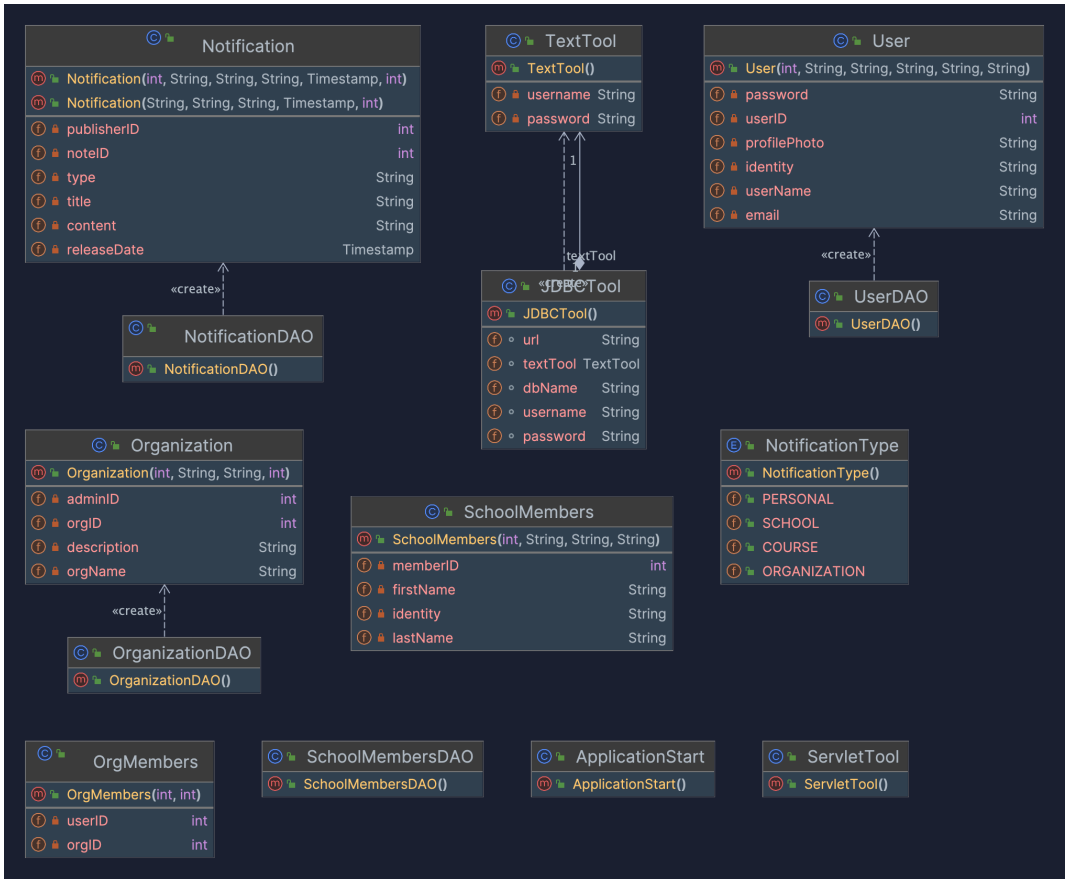

Figure 8: The UML Diagram of the System backend

5 Self-assessing

This information system demonstrates the combination of a fine-designed database which satisfies the first, second and third normal form and a fair utilization of the database. It contains a modern and user-friendly interface with several functions that apply the CRUD process to the database. The permission system serves as a robust security measure, preventing users from sending messages to areas where they do not have access or altering or removing other users' messages. This feature upholds the system's integrity, thereby promoting a respectful and secure communication environment. The searching system is effective and an in-depth use of the operation of the database. However, there is still some room for improvement. For example, the design of the database did not cover every part of the SQL features that has been discussed in the lectures.

Overall, the PigeonBroadcast system is a rather good implementation of a kind of information system. Its thoughtful design, user-friendly interface, robust permission system, and effective functionality combine to offer an efficient and secure information system and shows a reasonable understanding of database.