CS 213 Object-Oriented Programming – Final Project

Dr. Ayorkor Korsah

Group 17 – Cohort B

Edem K. Anagbah

Stanley T. Ndlovu

Alvin Appiah

Shammah Zvikomborero Dzwairo

Project Descriptions and Design
Group 17

DeMorgan's Complaint System

Authors: Edem K. Anagbah – Office, Message and Complaint class

Stanley T. Ndlovu – DeMorganComplaint, History class and Network I/O

Alvin Appiah – Student and UserComplaints class

Shammah Zvikomborero Dzwairo – GUI design

In our time at Ashesi, we have noticed that whenever a student has a complaint concerning a problem, the student usually has to go to the administration to report their complaint. After reporting the complaint, the student did not know when their complaint would be answered unless they returned to the office. We thought this was an inefficient way to report the problem, so we devised DeMorgan's Complaint System to solve this problem.

With our system, students can log in and send their complaints to any department from their current location. When sending a complaint, the student could select the department to receive the complaint and the complaint's category. The complaint would then be sent to the department office for a response.

Our System makes use of 8 classes, each with its own function.

Below is a list of the classes and the functions.

Project Descriptions and Design
Group 17

1. Complaint: This class extends the Message class and represents a specific type of message used for making complaints. It possesses the attributes below:

- category: The category of the complaint (e.g., water, electricity).
- status: Tracks the status of the complaint (e.g., "open", "responded").

It also inherits methods from the Message class but has getters and setters for category and status. It has the markResponded() method to update the complaint's status to "responded".

2. DeMorganComplaint: This class was used for testing before connecting to the GUI. It used pseudo users and oofers to enable testing for users og in, send complaints, receive responses, view responses, and log out. It has the following attributes:

- STATE_LOGIN, STATE_SEND, STATE_RECEIVE, STATE_VIEW: Constants representing different states of the user within the system.
- state: An integer representing the current state of the user.
- user: An instance of the UserComplaints class representing the logged-in user.
- office: An instance of the Office class representing the office responsible for handling complaints.

It also has the method below:

- main(String[] args): The main method to simulate user interactions by logging in, sending complaints, receiving responses, viewing responses, and logging out.
- getState(): The getter method obtains the current state of the user.

- setState(int state): The setter method is used to set the state of the user.
- login(): It simulates the user login process. Creates a new UserComplaints instance and sets the state to STATE_VIEW.
- send(): It simulates sending a complaint by the user and ensures that the user is in the correct state (STATE_VIEW). It initializes the office object if it has not already been initialized.
- receive(): It simulates the office receiving complaints and checks if the user is in the correct state (STATE_SEND) and creates a new complaint instance.
- respond(): It simulates the office responding to a complaint and checks if the user is in the correct state (STATE_RECEIVE) and responds to the first received complaint.
- view(): It simulates the user viewing responses to their complaints, checks if the user is in the correct state (STATE_VIEW) and displays messages from the inbox.
- logout(): It simulates the user logout process. Resets the user instance to null and sets the state to STATE_LOGIN.

3. History: This class stores a user's messages (complaints or responses) in an inbox or outbox space. It has the attributes below:

- inbox: An ArrayList containing messages received by the user.
- outbox: An ArrayList containing messages sent by the user.
- ClimateChangeIssues: An array of strings representing issues related to climate change.

It also has the following methods:

History(): A constructor method to initialize an empty inbox and outbox.

- addInbox(Message msg): A method to add a message to the inbox.
- containsClimateChangeIssue(Message msg): A private method to check if a message contains issues related to climate change.
- addOutbox(Message message): A method to add a message to the outbox.
- getInbox(): A method to retrieve the inbox.
- getOutbox(): A method to retrieve the outbox.
- Remove (String ID): A method to remove a message from either the inbox or outbox based on its ID.

History also makes use of our priority algorithms to sort the complaints based on the ones that impact climate change the most.

4. Message: This class represents the messages sent and received by a user within the complaint system. It also possesses the attributes below:

- text- The textual content of the message.
- image- An optional image attachment.
- category- The category of the message (using an enum for organization).
- sender- The UserComplaints object who sent the message.
- recipient- The UserComplaints object intended to receive the message.
- ID- A unique identifier for the message.
- date- The timestamp when the message was created.

It has getter and setter methods for its attributes, and the displayMessage() method prints a message in a readable format.

5. Office: This class extends the UserComplaints class and represents the office that receives the complaint from the student. It has the following attributes:

- departmentName: A string representing the name of the department.
- Complaint: An ArrayList that contains a list of all the complaints sent to the department.

It has getter and setter methods, and methods for receiving complaints and sending responses.

6. Student: This class represents a student user and all the attributes and methods it inherits from the UserComplaints class. It has the attributes below:

- major: A string representing the major or field of study of the student.
- studentID: A string representing the unique student identifier.

It has getter and setter methods for all its attributes.

7. UserComplaints: This class creates the user sending a complaint to the chosen department office. It includes attributes:

- userName: The user's name.
- email: The user's email address.

- password: The user's password for authentication.
- History: The user's message records and storage.

It also has getter and setter methods for all its attributes. It also has methods that allow a user to send, receive and view messages.

8. Date: The eighth class, Date, is an existing class within the Java library, and it serves as a representation of time in the Message class.

The meet of the program is the ComplainSystem :
This ComplainSystem class represents a graphical user interface (GUI) application for managing complaints, and it basically links everything together.

Fields:

1. selectedImagePath: A string to store the path of the selected image.
2. button, button2: Buttons for selecting user type (student or staff).
3. offices: An ArrayList to store instances of Office, representing different departments.
4. students: An ArrayList to store instances of Student.
5.

Constructor:

1. Initializes the GUI frame with title, size, and layout.
2. Creates offices and students from CSV files.
3. Sets up UI components such as buttons and panels.
4. Defines action listeners for the buttons.

Methods:

1. displayImage: Displays an image on a panel within the GUI.
2. getSelectedImagePath: Returns the path of the selected image.
3. createOfficesFromCSV: Reads data from a CSV file and creates Office objects.
4. createStudentsFromCSV: Reads data from a CSV file and creates Student objects.
5. setFavicon: Sets a favicon for the GUI frames.

Main Method:

Creates an instance of ComplainSystem and makes the frame visible.

Functionality:

1. Users can log in as students or staff.
2. Upon selecting the staff login, a separate frame for staff login is shown, where staff can enter department name and password.
3. Upon successful staff login, a staff menu frame is displayed, showing the inbox messages. Staff can view messages and respond to them.
4. Upon selecting the student login, a separate frame for student login is shown, where students can enter username and password.
5. Upon successful student login, a student menu frame is displayed, showing options to send messages, view inbox, and view outbox. Students can send messages to offices, view received messages, and view sent messages.

The UML diagram below gives a detailed illustration of all the classes, their attributes and methods.

Project Descriptions and Design
Group 17

DeMorgan's Complaint System – UML Diagram (Link to UML)