Project 2 (Final project) (Due on 20th December at 11:59pm in ICON Dropbox)

In this project you will develop a private messaging system where users (clients) communicate with a central database (Oracle 11g XE) by using a Java desktop application. The database is responsible for storing messages and user information. When a user sends a message to one or more recipients, the application simply stores the message in the database. Users retrieve messages sent to them from database periodically by using a timer. Your application should support both individual and group messages. Below are the descriptions of different components of this project:

Back-end Database:
Create a database (workspace) in Oracle 11g XE and create following tables:

- **USER_TABLE**: This table stores all user information. The table should have fields such as username (id), password, name, designation, phone, email etc. The field username should be defined as a primary key.
- **MESSAGE_TABLE**: This table stores all messages. The table should have fields such as message_id (unique id for a message), sender (username of the sender), recipient (username of the recipient or the first recipient), time, subject, is_group (indicates if the message is sent to multiple users), content etc. The field sender and recipient should be foreign keys referring to username in the USER_TABLE. The message_id should be generated by using a sequence.
  
  Note: Check the URL posted in course page that describes foreign keys. This will also be discussed in class after the thanksgiving break.
- **MULT_RECIPIENT_TABLE**: The purpose of this table is to store recipients of a group message. The table should have only two columns: message_id and recipient. A row in this table is a pair consisting of a message_id and a recipient, therefore a row should be inserted for each recipients of a message. The field message_id should be a foreign key referring to message_id in MESSAGE_TABLE and the field recipient should be a foreign key referring to username in USER_TABLE.

Once you have created these tables, create at least 10 dummy users by inserting rows in the USER_TABLE. Write a SQL script containing statements that do everything above. You need to submit this script so that I can create your database while grading your project.

Front-end Java Application:
The Java application enables user to see messages that are sent to them, compose new messages, send individual or group messages, search messages and search users. Here are descriptions for the major features of the application:

- **Sign up**: Your application should have a user interface to enable new users to sign up. Let users enter required information and check the availability of the username. It is up to you to decide what fields are required. The required fields should be marked with a red asterisk (*) in the UI. Verify that all required fields are entered and also verify other user inputs. For example, a phone number may contain only digits and parentheses, and you should notify the user an error if any other character is present in the phone number.
- **Log in**: Log in UI allows user to enter username and password. Check if the username-password combination is valid by running a query to the database. Make users see the inbox UI if login is successful.
- **Inbox and Sent messages**: Design user interface to show inbox and ‘sent messages’. You may use a JTable to show the list of messages. A timer should be implemented that updates the inbox in every 500 milliseconds. Allow users to open a message by clicking on it (e.g. clicking on the row in the table). You may use a JTextArea to show the message content. You may either use a JDialog, or a new tab by using
JTabbedPane, to show an opened message. Add “Reply”, “Reply All”, and “Forward” buttons with the UI that shows the message. When any of these three buttons are pressed, allow users to compose message in a JTextArea loaded with the content of current message (like standard email systems). Show related information such as sender’s and recipient’s name (from and to), subject, time and date. If is_group field for this message in MESSAGE_TABLE is set to 1 (i.e. message is sent to multiple users), show all recipients of this message by running a query at the MULT_RECIPIENT_TABLE.

- **Compose and send message:** There should be a "Compose" button that allows user to write a new message. A JTextArea should show up to allow users compose their message when the "Compose" button is pressed. You may use the same UI that you use for Reply/Reply All/Forward options. Allow users to add one or more recipients (recipient must be a user in the system). Allow users to search for other users (recipients). You also need to add “Send” and “Cancel” buttons. The "Send" button should insert a message in the MESSAGE_TABLE. If the message is sent to multiple users, set is_group field in MESSAGE_TABLE to 1 (zero otherwise) for that message and insert a row for each recipient in the MULT_RECIPIENT_TABLE.

- **Search user and messages:** The application should allow users to search other users or search messages (both in inbox and ‘sent messages’). When searching for a message, you should match the search key with sender’s name, recipients’ names, subject, message content, date etc. You may check the advanced search option of the AGM project posted in ICON to get some idea and code.

- **Log out:** User should be able to log out from the system. Logging out should clear all user-specific data from memory (i.e. should clear all variables or object references that may contain information specific for the user).

**Java Classes:**
Here are descriptions of some Java classes that you may use:

- **User:** This class can be static. Possible fields are: username, name, address etc. Possible methods are: login, log out etc.

- **MessageList:** You may use an ArrayList as a member of this class to hold a list of messages. An object of MessageList can be used both for inbox and ‘sent messages’. Possible methods are loadMessageList, loadMessage (int index) etc.

- **Message:** This class should have member variables that hold details of a message (e.g. recipient, subject, time, content etc.).

**Grading:**
This project covers 18% of your total grade. Here are point distributions:

- Database design: 40 points
- UI design and user friendliness: 30 points
- Features: Signup, login and logout: 30 points
- Features: Compose and send messages, inbox and sent: 50 point
- Features: Search users and messages: 30 points

**Submission:**
Submit the project folder and database script as a zipped file. Add a readme file if you have any instructions for me about how to run your project.