InCollege Project: Week 9 Deliverable - Basic Messaging System - Part 2 (View Messages)

Objective: This week, your team will enhance the messaging system to allow logged-in users to retrieve and display the messages they have received. This completes the fundamental two-way communication functionality, with all input and output handled via files as per our testing protocol.

Focus Areas:

1. **Viewing Received Messages:**
    - The "View My Messages" option within the "Messages" sub-menu (which was "under construction" last week) will now become fully functional.
    - When a user selects this option, the system should display all messages that have been sent *to* their username.
    - Each message should clearly indicate:
        - The sender's username.
        - The message content.
        - (Optional but recommended) The timestamp of the message (if you implemented this in Week 8).
    - Messages should be displayed in a clear, readable format, possibly in chronological order (newest first or oldest first, your team's choice).
2. **Handling No Messages:**
    - If a user has no new or existing messages, the system should inform them with an appropriate message (e.g., "You have no messages at this time.").
3. **Message Status (Optional but good to consider):**
    - For a more complete system, you might consider marking messages as "read" once viewed. However, for this alpha version, simply displaying them is sufficient. This can be a future enhancement.
4. **I/O Requirements:**
    - **Input:** All user input (e.g., menu selections) will be read from a predefined input file.
    - **Output Display:** All program output (e.g., prompts, confirmation messages, displayed messages, "no messages" alerts) must be displayed on the screen (standard output).
    - **Output Preservation:** The exact same output displayed on the screen must also be written to a separate output file for testing and record-keeping purposes.

COBOL Implementation Details (For Programmers):

- **Message Retrieval Logic:** Develop COBOL modules to efficiently read through your persistent message storage (created in Week 8) and identify all messages where the logged-in user is the recipient.

- **Message Display Formatting:** Implement COBOL routines to format and display each retrieved message clearly on the console, showing the sender and the message content.
- **No Messages Scenario:** Include logic to check if any messages are found for the user and display a "no messages" prompt if applicable.
- **Input File Handling:** Continue to implement COBOL READ statements to read user input from the designated input file for all menu selections.
- **Output File Handling:** Ensure all program output, including prompts and displayed messages, is written to your dedicated output file *identically* to what is displayed on the screen.
- **Menu Integration:** Ensure the "View My Messages" option correctly navigates to the message display functionality.

Testing Responsibilities (For Testers):

- **Test Case Development:** Create comprehensive test cases in Jira for all Week 9 functionalities, including:
    - **Positive Test Cases:** Scenarios where a user successfully views one message, multiple messages, and messages from different senders.
    - **Negative Test Cases:** Scenarios where a user attempts to view messages but has none.
    - **Persistence:** Verify that messages that were sent in Week 8 are correctly retrieved and displayed after program restarts.
- **Test Execution:** Execute all developed test cases using the specified input file.
- **Bug Reporting:** For every issue or discrepancy found, create a detailed bug ticket in Jira. Include steps to reproduce, actual results, and expected results.
- **Output Verification:** Meticulously compare the program's console output against the generated output file to ensure they are absolutely identical for all message viewing scenarios.
- **Collaboration:** Work closely with the programmers to help them understand and reproduce bugs.

Jira Requirements (For Scrum Master, Programmers, & Testers):

Your team's Jira board for Week 9 should continue building on **Epic #5: Messaging System**.

- **User Stories for Viewing Messages:**
    - "As a logged-in user, I want to view all messages I have received."
    - "As a user viewing messages, I want to clearly see who sent each message and its content."
    - "As a user, I want to be informed if I have no messages to display."
    - "As a user, I want the 'Messages' menu to allow me to 'View My Messages'."
- **New User Story (for Testing):** "As a tester, I want the program to read all user inputs for viewing messages from a file so I can automate testing."

- **New User Story (for Testing):** "As a tester, I want the program to write all screen output related to viewing messages to a file so I can easily verify results."
- **Tasks:** Break down each User Story into granular tasks that individual team members can work on. (e.g., *Programmers:* "Develop COBOL routine to read messages for current user," "Implement COBOL display logic for individual messages," "Add logic for 'no messages' scenario," "Integrate message viewing into the 'Messages' sub-menu," "Update I/O for message viewing to use file input/output consistently." *Testers:* "Develop test cases for message viewing," "Execute message viewing tests," "Log bugs related to message display," "Verify I/O consistency for message viewing features").
- **Bug Tickets:** Log any issues found during development and testing.

GitHub Requirements:

- **New Modules/Files:** Commit any new COBOL modules or updated existing files for handling message viewing.
- **Branching:** Continue to follow your team's established branching strategy for new features.
- **Regular Commits:** Ensure consistent, descriptive commits throughout the week. Testers should commit their test files.
- **README.md Update:** Update your README.md to reflect the new functionality for viewing messages, explicitly detailing how to prepare input for these features and where to find the corresponding output.

Deliverables for End of Week 9:

1. **Roles.txt**: List of team members and the roles that they played this week.
2. **InCollege.cob: Working COBOL Program:** A console-based COBOL application that allows logged-in users to:
    a. Navigate to the "Messages" menu.
    b. Select "View My Messages."
    c. Display all messages received by the user, or a "no messages" prompt if applicable.
    d. This must seamlessly integrate with the message sending functionality from Week 8, and all previous weeks' functionality (login, profile, connections, job board). All inputs must be read from a file, all outputs displayed on the screen, and the exact same outputs written to a separate file.
3. **InCollege-Input.txt: Sample Input File:** A sample text file demonstrating the format of input your program expects for Week 9's functionality (e.g., menu choices leading to message viewing).
4. **InCollege-Output.txt:** A sample text file showing the expected output for a typical run of your program, demonstrating a user viewing their messages.

```
--- SAMPLE_OUTPUT_WEEK9.TXT ---
Welcome to InCollege!
1. Log In
2. Create New Account
Enter your choice:
Please enter your username:
Please enter your password:
You have successfully logged in.
Welcome, ReceivingUser!
1. View My Profile
2. Search for User
3. Learn a New Skill
4. View My Pending Connection Requests
5. View My Network
6. Messages
Enter your choice:
--- Messages Menu ---
1. Send a New Message
2. View My Messages
3. Back to Main Menu
Enter your choice:
--- Your Messages ---
From: SendingUser
Message: Hi there! Glad we connected on InCollege.
(Optional: Sent: 2025-07-28 10:30)
---
From: AnotherConnection
Message: Check out this interesting job posting I found!
(Optional: Sent: 2025-07-29 09:15)
---------------------
1. Send a New Message
2. View My Messages
3. Back to Main Menu
Enter your choice:
--- END_OF_PROGRAM_EXECUTION ---
```

5. **Epic9-Storyx-Test-Input.zip: Test Input Files:** A set of test input files used by the testers, covering positive, negative, and edge cases for each of this week's stories.
6. **Epic9-Storyx-Test-Output.zip: Actual Test Output Files:** The exact output generated by running your program with the Epic9-Storyx-Input input Files, submitted for review.
7. **Jira.jpg: Updated Jira Board:** All relevant User Stories, tasks, and bugs (with their status) for Week 3 should be updated in Jira.
8. **Jira:** Two Burndown charts. The first created on Monday and the second created when the Sprint is complete.
9. **GitHub.jpg:** Go to the repository's main page. Click the "Commits" link (next to the green "Code" button). Show a chronological list of all commits with messages, authors, and timestamps.

Your testers will be vital this week, ensuring that all messages sent in Week 8 are correctly received and displayed by the intended recipients. They should also test scenarios where a user has many messages, a few messages, and no messages, all while meticulously comparing console output with the generated output file for consistency. The scrum master will continue to facilitate the team's progress and ensure any impediments are resolved.