Coding Style:

As similar to provided code as possible - keeping indents, space length, variable naming, etc. as close as possible (example below). Maintains cleanliness, keeps things organized, comment processes' function, and separates methods into individual, clearly-identifiable blocks. Each method should include a brief Javadoc header noting its purpose, inputs, and outputs.

```
175    /*********
176     * <p> Method: performAddRole() </p>
177     *
178     * <p> Description: This method adds a new role to the list of role in the ComboBox select
179     * list. </p>
180     *
181     */
182    protected static void performAddRole() {
183
184        // Determine which item in the ComboBox list was selected
185        ViewAddRemoveRoles.theAddRole =
186                (String) ViewAddRemoveRoles.combobox_SelectRoleToAdd.getValue();
187
188        // If the selection is the list header (e.g., "<Select a role>") don't do anything
189        if (ViewAddRemoveRoles.theAddRole.compareTo("<Select a role>") != 0) {
190
191            // If an actual role was selected, update the database entry for that user for the role
192            if (theDatabase.updateUserRole(ViewAddRemoveRoles.theSelectedUser,
193                    ViewAddRemoveRoles.theAddRole, "true") ) {
194                ViewAddRemoveRoles.combobox_SelectRoleToAdd = new ComboBox <String>();
195                ViewAddRemoveRoles.combobox_SelectRoleToAdd.setItems(FXCollections.
196                    observableArrayList(ViewAddRemoveRoles.addList));
197                ViewAddRemoveRoles.combobox_SelectRoleToAdd.getSelectionModel().clearAndSelect(0);
198                setupSelectedUser();
199            }
200        }
201    }
202
203
```

Meetings, Communication, and Sharing:

We want to meet at least twice per week; though understandably, as all are engineering students, most with jobs, this is only ideal and most likely unrealistic - thus, frequent communication via a text-based server (Discord) is going to be key. Updates made to the project will be held inside a Github repository, updated after any changes are made. Task assignments and progress will be tracked using GitHub Issues or equivalent, so each contribution is linked to a story or feature.

Documentation and Commenting (Exemplar):

Each source file must begin with a short description of its purpose. Major functions and classes should include inline comments where necessary, in addition to Javadoc headers. This ensures clarity for future team members and consistency in grading.

UI/UX Rules:

- Maintain overall similarity to the provided application to preserve familiarity and usability.
- Keep font style and size consistent across all screens.
- Error messages should be clear, concise, and displayed in a uniform style (e.g., red text beneath the input field).
- Buttons should have consistent sizing and placement; forms should be aligned left with even spacing between elements.
- Color scheme should remain simple and cohesive; avoid mixing multiple styles unless agreed upon.
- Where possible, maintain accessibility by using legible font sizes and sufficient contrast.

Signed by:

Ahmed Meftah

Lucas Conklin

Sutton Harr

Bella Rayner

Nathaniel Taylor