TP3 SCREENCAST SCRIPTS

Screencast 1: Code Walkthrough

Joshua Wright

- The first order of business was figuring out the architecture of the program. We decided to expand upon the architecture we used in the previous phase.
- Jad was the one who wrote `Review.java` and `ReviewManager.java`, which were structurally similar to other classes in our `application.obj` package.
- I continued the trend as I implemented the backend code for ReviewerProfiles.
- Later on, I implemented the first pass for `ReviewerProfilePage.java` and `ReviewManagementPage.java`.
- I also implemented `MessagingPage.java`.

Kyle Ferolito

- I did a lot of work on the UI for this phase in general:
    - `InstructorHomePage.java`: for allowing Instructors to promote Users to Reviewers
    - `UserReviewsPage.java`: for allowing users to review... Reviews
    - `UserHomePage.java`: added navigation buttons

Steven Grisham

- Before we started writing code for TP3, I made sure to merge the old `phase2` branch from TP2 into the `main` branch, and otherwise cleaned up the repository a bit.
- As the team worked on different features concurrently, I wrote documentation for them. This way, they can focus on the code, and I can focus on the docs.
- Before submitting TP3, I generated the Javadoc for this project.

Jarod Wagner

- I helped with odd jobs and tasks around the codebase, acting as a flexible developer.
- The code architecture is visualized in the UML diagrams I provided.

Jad Khayyati

- After providing the backend code for `Review.java` and `ReviewManager.java`, I went to write tests for the new review system.
- Josh helped out by adding tests for interactivity between UserRoles and the database.

TP3 SCREENCAST SCRIPTS

Screencast 2: Program Demonstration

Note: You'll want to create actual user accounts and use them. Since the dev login method bypasses the database, it can cause desync issues that eventually break the program. Create a StudentUser, ReviewerUser, and InstructorUser account. You can use the dev login method for doing stuff as the Admin role.

Joshua Wright

- To speed up the testing process, I implemented this "dev login" mode where we can skip the account creation and login process. This mode is used purely to save time — we made sure the account creation and login systems still worked and their relevant tests still passed.

Kyle Ferolito

- First, I'll login as *StudentUser* and navigate to the questions page.
    - In the bottom-right corner, there is a "Request Reviewer Role" button. I'll click this for the time being — we'll get back to that later.
    - If we go to the questions page and select a question that has an answer, we can see a "Reviews:" header beneath the answer.

Steven Grisham

- Now, let's restart the program and log in as *ReviewerUser*. The home page for a reviewer looks similar to the one for a student, but with some extra buttons in the top-left corner.
    - Manage My Reviews: This lets me view the reviews I've posted. I can also delete reviews from here.
    - Edit My Profile: This lets me edit my profile as a reviewer. I can make some changes, return to the previous screen, then move back to this screen and see those changes have taken effect.
    - Messaging: This lets me send messages to users and read messages I've received.

Jarod Wagner

- To show off the instructor page, I need an instructor user. Here, I'll login as an admin using the dev login method and change the role of *InstructorUser* to Instructor before clicking the save button.
- Let's restart the program and log in as *InstructorUser*. The home page for an instructor is simple, consisting of just a table containing pending reviewers — that is, students who have requested to become a reviewer.
- Here, we can see a pending reviewer request. Let's go ahead and approve it.

Jad Khayyati

- Finally, for the messaging system. First, I'll log in as *ReviewerUser* and send a message to another *ReviewerUser*. Then, I'll restart the program and log and open the message.