Minutes: The group met for about an hour and a half.

Events:

Wellness check

We worked on Deliverable 3, we finished our informal scenarios and use cases.

* Updated trello.

INFORMAL SCENARIO #1: RIC Student tries to add a class that is full.

Current State: Diego logs in and is looking for a class to add. The list of classes are displayed to him.

Informal Scenario: Diego selects the class and hits the button to add the class. Diego didn’t notice that the class is already full.

Next Scenario: A message pops up notifying Diego that the class is full. He will be able to choose a different class to add.

INFORMAL SCENARIO #2: Time conflicts

Informal Scenario: Student wants to add a class to their schedule.

Current state: Student has selected a class to add to their schedule.

Informal scenario: The student attempts to add the class to their schedule, however there is a scheduling conflict. with a class already added to their schedule, they cannot attend two classes at the same time.

Next scenario: An error message is displayed, indicating that there is a scheduling conflict between the class they are adding and the other conflicting class. For the student to add this class, they will have to make sure there are no scheduling conflicts with other classes.

# **Use Case: Logging in**

Precondition: User starts the program

Main flow of events: Use case starts when a student attempts to log in, they input a username and password. The system checks if the username and password are both valid. If valid, they are allowed into the application, thus ending this use case.

Exceptional flow of events: If the user enters a invalid username and a valid password, the system prompts: invalid username and doesn’t allow login, and thus this use case ends.

Exceptional flow of events: If the user enters a valid username and a invalid password, the system prompts: invalid password and doesn’t allow login, and thus this use case ends.

# **Use Case: Drop Class**

Pre condition: User has logged in with a valid username and password.

Main flow of events: User wants to remove a class from their schedule, they navigate to the Drop class section and selects the class they want to drop, the system confirms that they would like to drop a class. The student confirms and the class is removed from the schedule, thus ending the use case.

Exceptional flow of events: Student tries to remove a class, and there is no class in the schedule, thus ending the use case.

Exceptional flow of events: Student cancels the request to drop the class, thus ending this use case.

# **Use Case: Add class**

Precondition: User starts application. User successfully validates their identity with a valid username and password. User applies desired filters to available classes

Main Flow of Events: User selects a class from the filtered list. User attempts to add class by clicking add button.

Exceptional of Events: The class is successfully added to the user's schedule

Exceptional Flow of Events: An error message displays telling the user that the class is already at full capacity.

Exceptional Flow of Events: An error message displays telling the user that the class that there is a time conflict with another class they has already added

Exceptional Flow of Events: An error message displays telling the user that they do not meet the requirements to take that class