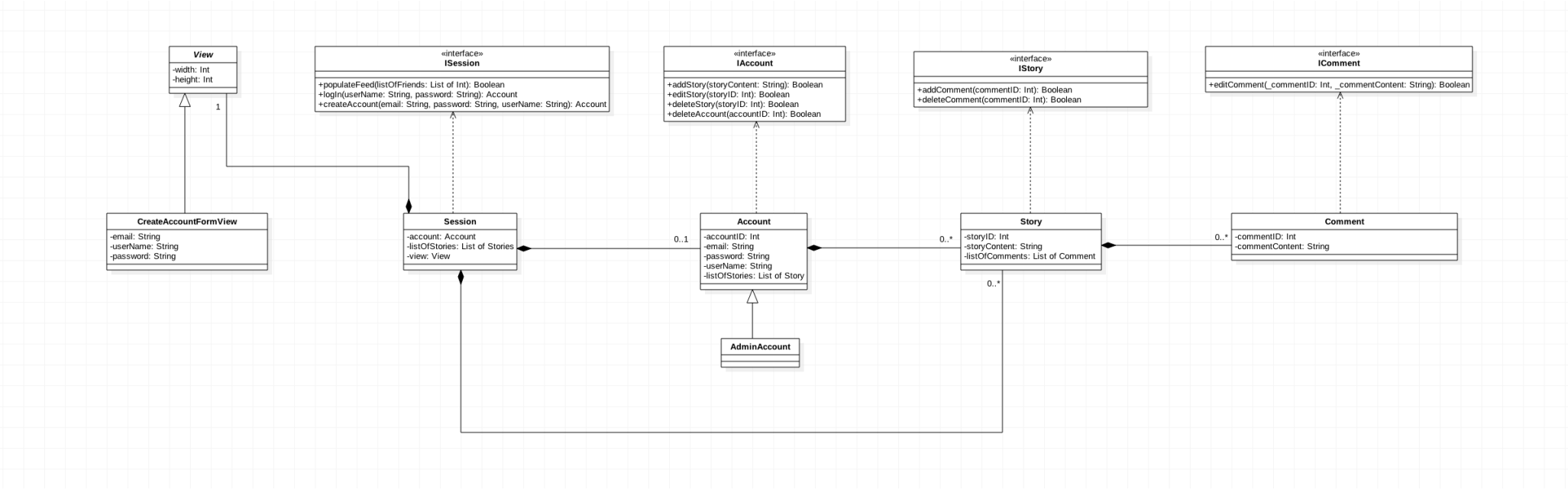
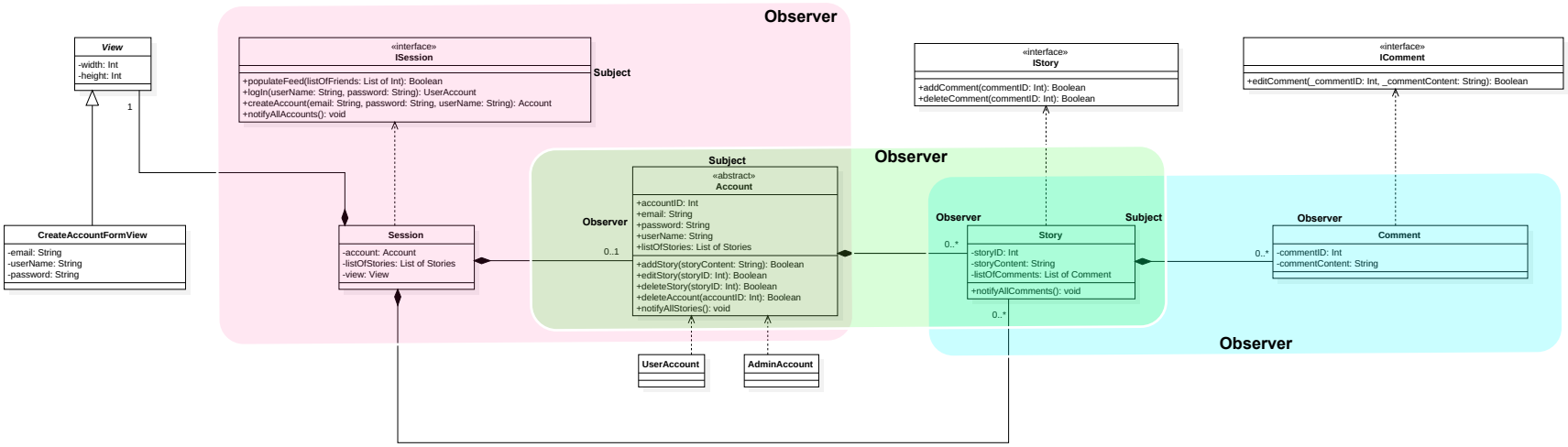
Part 2 Class Diagram



Part 3 Class Diagram



Ross Blassingame, Michael Tang, Josh Jenkins, Griffith Neumark

CSCI 4448

November 16, 2017

Part 3: Refactoring

Our project is a social media app where users can create accounts, log in, post statuses, send direct messages, add friends, view other users' profiles, etc. To implement this, we will have an "Account" abstract class, with the different types of accounts as concrete classes ("user", "admin", etc). We have a separate class for the statuses, or "Stories", in a "Story" class. Users will be able to comment on stories, so we have a "Comment" class. In order for updates to occur when a user posts or edits a story or comment, we implemented several instances of the Observer design pattern. This pattern will allow the observers to see when a change occurs and act accordingly.

In our class diagram, we have three instances of the Observer design pattern. In the first instance, the account(s) in the session will be observing the session for updates. In the second instance, the account class will notify the list of stories when a story is posted, so the post can be appended to the global story list, or other behavior can occur. The last instance has the comment class as the observer(s), observing the story class for updates. When a story is edited, posted, or deleted, comments will have to behave accordingly.