CIS 557 Fall 2021 Final Project Rubric

The following document contains the different sections on which the final project will be graded. In order to pass the final project, you must complete the minimum number of requirements listed for each section. The sections are separated into two categories - sections that will be assessed during the demo and sections that will be assessed separately. Note that this is a live document that may be changed so make sure you check this regularly. If you have any questions about the rubric feel free to ask on Piazza or come to office hours. Note that the full user story is listed in parentheses with the description of the requirement we will be checking for.

Demo sections:

The app must be deployed, otherwise, you will get an incomplete for the project (and the course).

Basic Checks - need all to pass

Can register and login successfully (US 1, 2)	P/F
Account lockout if too many unsuccessful login attempts (US 2)	
Profile view has at least username, when they registered, and avatar/profile pic (US 3)	
Users should be able to change password and deactivate account (US 3)	
Users should be able to create public and private groups (US 4, 5)	
Group admins should be able to promote and demote other admins, except the creator (US 6)	
Users can request to join a public group and get notified whether or not they were accepted (US 7)	
Users in a private group can invite other users to join their group, and admins need to approve them before they can join (US 8)	
Users should be able to leave a group (US 9)	
Users should be able to filter public groups by tag (US 10)	
Users should be able to make posts with audio, video, or image attachments within groups they're in, and see all posts in a group (US 11, 15)	
Users should be able to flag posts and admins should be able to go through and delete flagged posts (US 12)	
Users should be able to delete their own posts (US 13)	
Users should be able to see a list of public groups (US 14)	

Can comment on a post and see other comments (US 16)	
Can edit or delete a comment (US 19)	
Can hide other user's posts (US 18)	
Can send asynchronous text message (US 20.0)	
Can send asynchronous audio message (US 20.1) - audio plays within the app	
Can send asynchronous picture message (US 20.2) - picture displays within the app	
Can send asynchronous video message (US 20.3) - video displays within the app	

Intermediate Checks - need minimum 5 to pass

Several ways of sorting groups (US 21)	P/F
Has group suggestion feature (US 24)	
Has mentions in posts and comments (US 23)	
Has support for hashtag filtering posts and comments (US 22)	
Has delivered + read receipts (US 27)	
Has pagination OR infinite scroll on posts (US 29/30)	
Has analytics for posts (US 26)	
Has analytics for groups (US 28)	
Has live updating messages and posts - no refresh (US 25)	
Has live updating notifications - no refresh (US 26)	
Fully responsive (US 31)	

Functional Testing Checks - need minimum 2 to pass

Tests for registration	P/F
Tests for login	
Tests for sending a message	
Tests for making a post	

Non-Demo Sections:

The following sections will be graded separately from the demo

Project Management Checks - need all to pass

Has failed no more than 3 sprints	P/F
Has API documented on Swagger (US 17)	

Wiki/Documentation Checks - need minimum 4 to pass

Has wireframes	P/F
Has one prototype	
React component description	
Has ER diagram	
ER diagram has accompanying Relational schema or NoSQL schema	
Has description of CRUD operation functions (controller functions)	

Security Checks - need minimum 5 to pass

Description of access control	
Description of input validation	
Description of account lockout policy	
Description of HTTP status codes used and reasoning	
Description of how exceptions are handled	
Description of user tokens/cookies	
Using .env files (no secure information like API keys on Github)	
For MongoDB: have access management for different accounts (under Security>Database Access) or utilize API keys (https://docs.atlas.mongodb.com/configure-api-access/#programmatic-api-keys)	
For SQL: have all the parameterized SQL statements	

(Travis) Testing Checks - need minimum 3 to pass

Have snapshots for at least half of their components	P/F
Unit tests (backend) code coverage is at least 60%	
Unit tests (frontend) code coverage is at least 60%	
Integration tests (controller/database) code coverage is at least 60%	
API tests (routes) code coverage is at least 60%	
If the sum of any combination of code coverages is at least 200% you automatically pass this section	