**Questions:**

* What is the intended purpose of your application?

Text based social media application, similar to Threads or X

* What work has been completed for this milestone?
  1. Server endpoints are almost all setup
  2. Front end functionality for tweeting and seeing your feed
  3. Blocked out component placement with columns and divs
* What work is left, and how do you plan to complete it?
  + What does your timeline/roadmap look like to finish on time?
    - 1. Add functionality to use /getAllTweets to allow users to view all the posted tweets (not just there own)
      2. Implement twitter premium for profit model
      3. Finish styling the pages to have visual appeal
* How are you using React?
  + What components have you made?
    1. Tweet entry
    2. Tweets feed (users tweets)
    3. Password changer
    4. Account info
  + What components do you still plan to add?
    1. Tweet feed (everyones tweets)
    2. Paid tweet that will always show up at top of the tweet feed
* What data are you storing in MongoDB?
  1. User tweets
  2. User Accounts
  + What data do you still need to store?
    1. Section of user account to say if they have paid for premium
    2. Section of tweet to say if it should be private
* What is your profit model?
  1. Premium version that gives the user a symbol next to there name to show that they are better than everyone else
  2. Sponsored tweet that will show up at the top of everyone’s public feed (for now will just be a stand in tweet but would be hooked up to an ad API if I were to continue the project)
  + Have you implemented it yet?
    - If so, how?
      1. So far all Ive done is get a button in the account section that will trigger the purchase
    - If not, what is your plan to implement it?
      1. Add a bool to the users account to say if it has been purchased or not
      2. Have a hidden symbol in the react tweet that will show up if the user has purchased premium
      3. Add a react component to the public feed that will show an add styled as a tweet
* Do you have a plan for going above and beyond? If so, what is it?

Implement socket.io to allow for real time tweet updating in the public feed

* If you used any borrowed code or code fragments, where did you get them from? What do the code fragments do? Where are they in your code?

See “Sources” section

**Endpoints:**

URL: /getTweets  
Supported Methods: GET   
Middleware: Requires Login  
Query Params: N/A  
Description: Retrieves all of the current users tweets  
Return Type(s): JSON

URL: /getAllTweets  
Supported Methods: GET  
Middleware: N/A  
Query Params: N/A  
Description: retrieves all tweets from every user  
Return Type(s): JSON

URL: /login  
Supported Methods: GET POST  
Middleware: requiresSecure, requiresLogout  
Query Params: N/A  
Description: get will return the login page, post will log the user in  
Return Type(s): HTML, JSON

URL: /signup  
Supported Methods: POST  
Middleware: requiresSecure, requiresLogout  
Query Params: N/A  
Description: Will create a new user account if both passwords match and the username has not been used  
Return Type(s): JSON

URL: /logout  
Supported Methods: GET  
Middleware: requiresLogin  
Query Params: N/A  
Description: will log the user out and return them to the login page  
Return Type(s): JSON

URL: /changePassword  
Supported Methods: GET POST  
Middleware: requiresLogin, requiresSecure  
Query Params: N/A  
Description: post will change the password of the user as long as old password is correct and new passwords match. Get used to return change password page but that no longer exists so just needs to be removed at this point  
Return Type(s): JSON

URL: /account  
Supported Methods: GET  
Middleware: requiresLogin  
Query Params: N/A  
Description: will return the account page  
Return Type(s): HTML

URL: /accountInfo  
Supported Methods: GET  
Middleware: requiresLogin  
Query Params: N/A  
Description: returns username and creation date of the user  
Return Type(s): JSON

URL: /tweet  
Supported Methods: GET POST  
Middleware: requiresLogin  
Query Params: N/A  
Description: get will return the main tweet page post will create a new tweet on the database  
Return Type(s): HTML JSON

URL: /  
Supported Methods: GET  
Middleware: requiresSecure, requiresLogout  
Query Params: N/A  
Description: will send the user to the login page  
Return Type(s): HTML

**Sources:**

* Starting code was taken from domo maker part e
* Gulp implementation was taken from the sass/gulp demo code
* Favicon: <https://fontawesome.com/icons/comment?f=classic&s=solid>
* Premium Check: <https://fontawesome.com/icons/square-check?f=classic&s=solid>
* Searching for javascript functions [W3School](https://www.w3schools.com/) and [MDN Web Docs](https://developer.mozilla.org/en-US/) used these sites to find functions for various different tasks in my code such as reversing the array of tweets in app.jsx
* Making the tweet div scrollable: <https://www.geeksforgeeks.org/making-a-div-vertically-scrollable-using-css/>