Strider Web Back-end Assessment - 3.0

Briefing

View test instructions, submission instructions, and evaluation criteria in the briefing:

Strider Technical Assessment Briefing

Project Description

Overview

The Project Manager you work with wants to build a new product, a new social media application called Posterr. Posterr is very similar to Twitter, but it has far fewer features.

Posterr only has two pages, the homepage, and the user profile page, which are described below. Other data and actions are also detailed below.

Pages

Homepage

- The homepage, by default, will show a feed of posts (including reposts and quote posts), starting with the latest 10 posts. Older posts are loaded on-demand on chunks of 10 posts whenever the user scrolling reaches the bottom of the page.
- There is a toggle switch "All / Only mine" that allows you to switch between seeing all posts and just posts you wrote. For both views, all kinds of posts are expected on the feed (original posts, reposts, and quote posts).

- There is a date range filter option (start date and end date) that allows results filtering based on the posted date, both values are optional: e.g user may want to filter only posts after a certain date without defining a limit date.
- New posts can be written from this page.

User profile page

- Shows data about the user:
  - Username
  - Date joined Posterr, formatted as such: "March 25, 2021"
  - Count of number of posts the user has made (including reposts and quote posts)
- Shows a feed of the posts the user has made (including reposts and quote posts), starting with the latest 5 posts. Older posts are loaded on-demand when the user clicks on a button at the bottom of the page labeled "show more".
- New posts can be written from this page: for this assessment, when writing a post from the profile screen, the profile user should be set as the author of the new content.

More Details

Users

- Only alphanumeric characters can be used for username
- Maximum 14 characters for username
- Usernames should be unique values
- Do not build authentication
- Do not build CRUD for users (registration and sign-in will be handled by a different service, the user model should be part of your data modeling tho. You can seed the database with 4 users to help the reviewer demo your solution)
- When/if necessary to make your application function, you may hard-code the user. For example, you may need to do this to implement creating new posts.

Posts

Posts are the equivalent of Twitter's tweets. They are text-only, user-generated content. Users can write original posts and interact with other users' posts by reposting or quote-posting. For this project, you should implement all three — original posts, reposts, and quote-posting

- A user is not allowed to post more than 5 posts in one day (including reposts and quote posts)
- Posts can have a maximum of 777 characters
- Users cannot update or delete their posts
- Reposting: Users can repost other users' posts (like Twitter Retweet), limited to original posts and quote posts (not reposts)
- Quote-post: Users can repost other user's posts and leave a comment along with it (like Twitter Quote Tweet) limited to original and reposts (not quote-posts)

Phase 1, coding

Estimated time: 4 hours

- Build out a RESTful API and corresponding backend system to handle the features detailed above. This RESTful API would communicate with a single-page JS app. This API you build should enable all the features on both of the pages.
- You should implement a real, production-ready database, and queries should be performant.
- Do not implement additional features beyond what is explained in the overview.
- Write automated tests for this project.
- Make sure you provide a straightforward way to set up your app locally with proper requirements definitions (we strongly recommend a containerized solution).
- Do not build a front-end.

Phase 2, self-critique & scaling

Estimated time: 30 minutes

In any project, it is always a challenge to get the code perfectly how you'd want it. Here is what you need to do for this section:

- Reflect on this project, and write what you would improve if you had more time.
- Write about scaling.
    - If this project were to grow and have many users and posts, which parts do you think would fail first?
    - In a real-life situation, what steps would you take to scale this product? What other types of technology and infrastructure might you need to use?

This should be added as a section called "Critique" (**please provide as much detail as possible)** in the README.