Twitter Clone

Task Preview:

This is a twitter API simulation project to simulate mainly all the features for twitter.

Tools

- Laravel Framework.
- Visual code editor.
- Laravel Socialite library for Facebook login.

Features

- User logins and registrations.
- Search for users by username.
- Follow user.
- Create tweets.
- Like tweets.
- Delete own tweets.
- Mention user by username.
- News feeds.
- Activity feed.
- Facebook login.

API calls

- User Registration
    - Post request by passing name, email, password and confirmation password inside the body.
        - Key: name ,value: your name
        - Key: email ,value: your email
        - Key: password, value: your password
        - Key; password_confirmation , value: confirmation password
    - The password length should be more than or equal to 8.
    - URL:
        - **http://localhost:8000/api/register**
- User login
    - Post request by passing email and password inside the body.
        - Key: email , value: your email
        - Key: password, value: your password
    - URL
        - **http://localhost:8000/api/login**
- Search
    - Post request by passing the username inside the body with key username
        - Key: username , value: the username you are searching for
    - URL
        - **http://localhost:8000/api/search**
    - Will return empty data if this username doesn’t exist.

- Follow
  - Post request by passing the user ID you want to follow in the body.
    - Key: userID , value: user ID you want to follow
  - URL
    - **http://localhost:8000/api/follow**
  - Unfollow if you called the API again.
- Create tweet
  - Post request passing the tweet text inside the body
    - Key: tweet, value: text of the tweet
  - URL
    - **http://localhost:8000/api/tweet**
- Like tweet
  - Post request by passing the ID for the tweet you want to like in the body
    - Key tweetID, value: the id of the tweet you want to like
  - URL
    - **http://localhost:8000/api/like**
  - User can like its own tweet
  - Unlike if you called the API again.
- Delete tweet
  - Post request by passing the tweet ID you want to delete in the body
    - Key: tweetID, value: the id of the tweet you want to delete.
  - URL
    - http://localhost:8000/api/delete
  - Even if the tweet doesn't exist it will keep showing you cannot delete tweet that is not yours.
- Mention
  - Post request by passing the username and tweet ID in the body
    - Key: tweetID, value: the id of the tweet you want to mention the user at.
    - Key: username, value: name of the user you want to mention
  - URL
    - **http://localhost:8000/api/mention**
  - You cannot mention people you are not following
  - Mentions made on existing tweets only
  - Possible enhancements
    - Ability to mention in new tweets
    - Provide the username within a tweet body then parsing at "@" to get the username.
- News Feeds
  - Get request
  - URL
    - **http://localhost:8000/api/newsFeeds**
  - It gets the tweets for the people I am following and my own tweets.
  - Possible Enhancements :
    - Calling one sql query to get both results in one query
    - Filter by date
- Activity Feeds
  - Get request
  - URL
    - **http://localhost:8000/api/activityFeeds**
  - It gets the activity for the people I am following.
  - Possible Enhancements :
    - Paginating results

Login with Facebook:

- Facebook Socialite has been installed.
- Write http://localhost:8000 and click login
- You will find login with Facebook.
- This table is connected with the users table to check if this user is already existed or not.
- The only thing I noticed that may be wrong that it doesn't hash the password the same way Laravel do.
- I couldn't make it as a restful API call I implemented it with front-end

Consuming the API

All the API calls are made through postman after running php artisan serve command inside the project

Database Structure:

- **Users Table**
- **Activity_feeeds**
    - User ID
        - The ID of the currently logged in user
    - Following ID
        - The ID of the user that the activity has been applied on
        - It can be zero if the activity is tweeted
    - Tweet ID
        - The tweet ID that the activity takes place at
    - Activity
        - Type of activity (like, mention, tweeted)
- **Followers**
    - User ID
        - The user ID that I will follow
    - Follower ID
        - The ID for the currently logged in user.
- **Social_facebook_accounts**
    - Table fo facebook logins
- **Tweets**
    - Tweet
        - The body of the tweet
- **Tweets_likes Table**
    - Tweet ID that has been liked
    - User ID that liked the tweet (currently logged user)

- **Mentions Table**
    - User ID
        - The one who mention (currently logged in user)
    - MentionedUserID
        - The user that has been mentioned. (he should be following)
    - TweetID
        - The tweet ID that the user has mentioned others at.

General Notes

- You will find all the logic inside Repo/TwitterRepo.php
- All the API calls are made with authorization by passing the api_token inside the Authorization field with the request in headers.
- All tables has id as primary key and timestamps for records.
- There are a lot of dummy data in the attached sql file so you need to take the journey from the beginning.

Testing Simulation example:

- Call the register API.
- Login in with the credentials
- Copy the API token that is returned from the response
- Create tweet
- Register with different account
- Login in with the new credentials
- Follow the existed user
- Create tweet
- Mention user you are following in your tweet
- Get news feeds
- Like the tweet made by the user you are following
- Log in with the other account and repeat mention and likes
- Get activity feeds
- Register new account
- Search for existing usernames you given before
- Login in the browser with Facebook.