React Test

Overview

Create a React application that shows a list of Posts and associated post Comments made by a User that are fetched via provided API.

Features:

- Create 3 routes '/' route, '/app' route and 'post/<id>' route. Base route should have a login form with email and password. The actual authentication is not needed, hardcode any valid email / password combination. On correct email / password combination redirect to /app route. If user is not authenticated successfully '/app' route must not be accessible.
- App route should show a list of posts and associated comments. Every post should have a user's name associated.
- Create a search input and filter posts by user based on the input
- Clicking a Post will open it in a new page
- MUST! EVERY component once rendered must log in the console 'Hello from <insert component name>'. The part 'Hello from' must be sent to a component via props and defined only once within the scope of the application. So it looks something like console.log(`${props.message} ${componentName}`). I made up variable names, feel free to use anything you want but make sure EVERY component shows this on render.

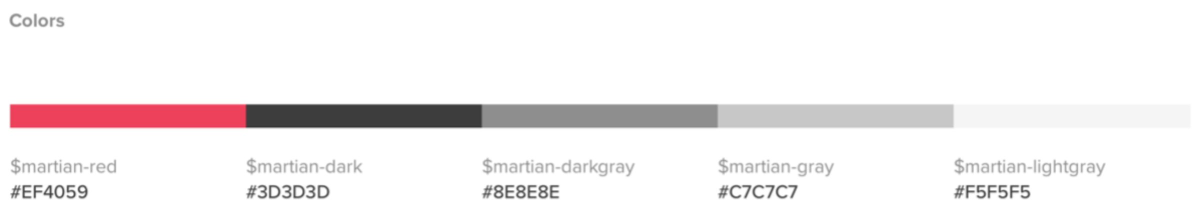
Considerations:

- The UI is up to you. This is a React oriented test but at least a minimally usable layout that does not break and is appealing to the eye is required.
- Do <u>not</u> use any 3rd party state management solution. Again this is React oriented test :) That does not mean state management can't be handled in a well structured way
- Do <u>not</u> use any 3rd party UI Component libraries. UI of the app can be very minimal and does not require 3rd party Component Libraries. CSS libraries are allowed (e.g. Tailwind)
- When creating components try to find a way to make them reusable and resilient, meaning they can easily be integrated into other applications. This part of the test is very important.
- Try to use some of the more advanced concepts like Hooks, HOC, Typechecking With PropTypes, Compound components etc.
- You are free to use any React build tool, create-react-app is recommended
- You are free to structure the code in any way you like (folder structure) but try to make it as real-world as possible, e.g. if you have a preferred folder structure / component hierarchy use it so we can see what are your preferences.
- You are free to include any tools that you use in general that can help you work on this task like linters, code style checkers, UI component testers etc
- Unit tests are not mandatory but highly recommended

API

URL: https://demo.martian.services/api
Docs: https://demo.martian.services
Access token: bWFydGlhbmFuZG1hY2hpbmU=
*send “X-Auth” header containing access token
*curl -i -H "X-Auth: bWFydGlhbmFuZG1hY2hpbmU=" "https://demo.martian.services/api/posts"

M&M CI scheme



Colors

🟥	⬛	▨	▨	⬜
$martian-red	$martian-dark	$martian-darkgray	$martian-gray	$martian-lightgray
#EF4059	#3D3D3D	#8E8E8E	#C7C7C7	#F5F5F5