UI Challenge

This challenge is intended for UI engineers, and consist of improvements and analysis of a fictional web application. The project is composed of a simple Single Page Application built in React and using Redux. The back-end is a simple Python server providing a JSON API built on Flask.

The challenge is divided in 4 parts, try to complete all of them, but if you get stuck on some of the tasks, please skip those and submit it with all that you could finish. If you have any questions please direct them to career@contunity.eu.

Relevant information:

● Please write your own code, and do not copy solutions from elsewhere.

● The README.md file on the project will provide you with instructions on how to run the application. You should start with running it and seeing how it works.

● Employ best practices when writing code. Code quality, functionality and performance are of the utmost importance for us.

● The application is protected by a login page, but for testing purposes, any user with the password “hipster” would be able to log in. Use that for your tests.

● Everything should look good on mobile.

Background

You take up a freelance job to finish a messaging web application for a company called Hipster Messenger , a messaging app targeting people that own a wide collection of flannel shirts and groom their beards obsessively.

The app is almost finished, but the company lost some of their engineers right before the big release date, and now it is up to you to deliver the final touches.

Task 1: Style and implement UI requirements

First, you are asked to carefully follow the requirements document (found on **ui_requirements.pdf**) and finish the Login and Home screens. For this task, everything that is needed on the back-end is already in-place, and your job is only to make the interface ready for the release.

Task 2: Add back-end functionality

Hipster Messenger stands by their user’s right to make mistakes, and as such, wants to implement a ‘delete message’ button to allow a user to remove their messages at any point time.
This is a last-minute change, and they do not have a requirements document for this one. It is up to you to decide on the details, as long as it allows users to permanently delete their messages on the message board.

For this feature, you will need to make some tweaks on the API.

Task 3: Freestyle

As a proud developer, you should leave your mark on the application! Make a change that you believe would add value to the project.

It can be anything, anywhere.

Task 4: Quick questionnaire

Answer a quick questionnaire below about what you saw.

Questionnaire

Please answer this questionnaire on an attached document and be brief and concise. Please do not exceed one page for the whole of your answers.

1. Have you identified any security vulnerabilities or bad practices? Why are those issues dangerous? How would you fix them? No need to write code or dig deep into it, give only a brief overview.

2. Is the project following best practices? What do you feel could be improved? Consider code style recommendations, architecture and whatever else you feel is relevant.

3. Right now the app would not work well if multiple users were to message each other at the same time, as people would have to constantly refresh the page to see new messages. Give a top-level description of what would be necessary for the app to automatically update with the messages from other users.