Assignment
Create a Sign Language Translator using React

Lost in Translation

Build an online sign language translator as a Single Page Application using the React framework.

You have freedom to be as creative as you wish, if it meets the minimum requirements described in Appendix A.

1) Set up the development environment

Make sure you have the following tools available:

- Figma
- NPM/Node.js (LTS – Long Term Support version)
- React CRA (create-react-app)
- Visual Studio Code Text Editor/ IntelliJ
- Browser Developer Tools for testing and debugging
    - React and Redux Dev Tools
- Git
- Rest API: https://github.com/dewald-els/noroff-assignment-api
- Heroku

2) Design a component tree

Use Figma to create a component tree of the application. The component tree should show the pages and feature components you plan to create in your application. This will count towards the overall grade for the application. It should be done BEFORE a single line of code is written.

3) Write HTML & CSS as needed

a) **Colours:** If you have trouble choosing colours, use a free resource like https://coolors.co to browse and experiment with colour combinations.
b) **Animations:** If you want to use animations to bring your design to life, use https://animate.style/.
c) **Free graphics for your web applications:** https://www.justinmind.com/blog/35-places-to-get-free-vector-images-for-your-designs/

4) Use the React framework to build the following screens into your translator app (See Appendix A for detailed specs):

a) The *Startup* Page
b) The *Translation* Page
c) The *Profile* Page

5) Submit

a) Export the component tree to PDF, upload the file to the project's Git repository and submit a link to your file.
b) Publish your Single Page Application on Heroku and submit a link to your app and the source code on your Git repository. Use https://gitlab.com/javascript-project-examples/heroku-deployment-guides/-/blob/main/guides/heroku-react-deployment.md to learn how to deploy React apps to Heroku.

Appendix A: Requirements for the sign language translator

The application will have one main feature: to act as a "translator" from regular text to sign language. The application must be able to translate English words and short sentences to American sign language. The images for the sign language will be provided.

1) Startup/Login Page

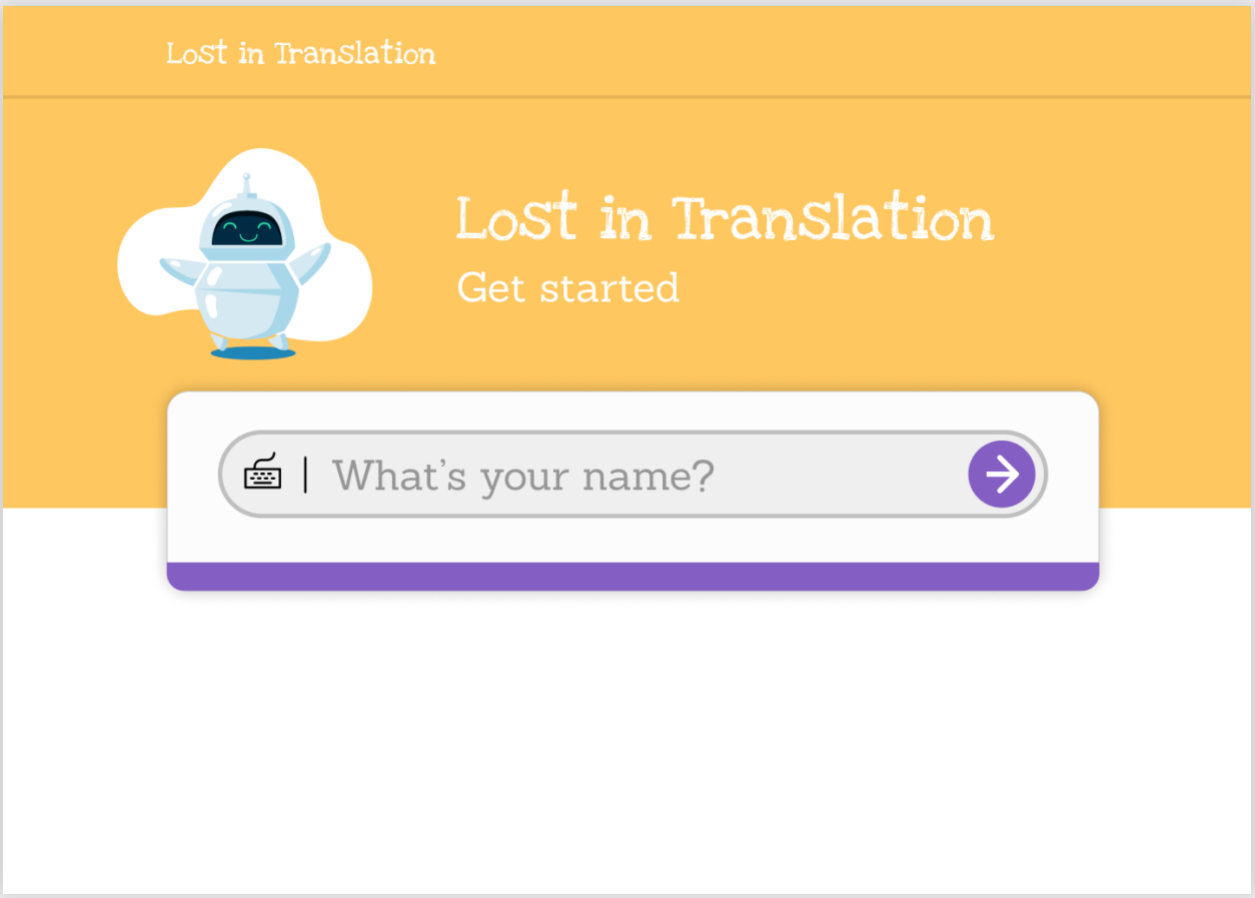

Figure 1: Example of the Startup/Login page

The first thing a user should see is the "Login page" where the user must be able to enter their name.

Save the username to the Translation API. Remember to first check if the user exists. Once the name has been stored in the API, the app must display the main page, the translation page.

Users that are already logged in may automatically be redirected to the Translation page. You may use the browsers' local storage to manage the session.

NB! NEVER STORE passwords or sensitive information in the browser storage.

2) Translation Page

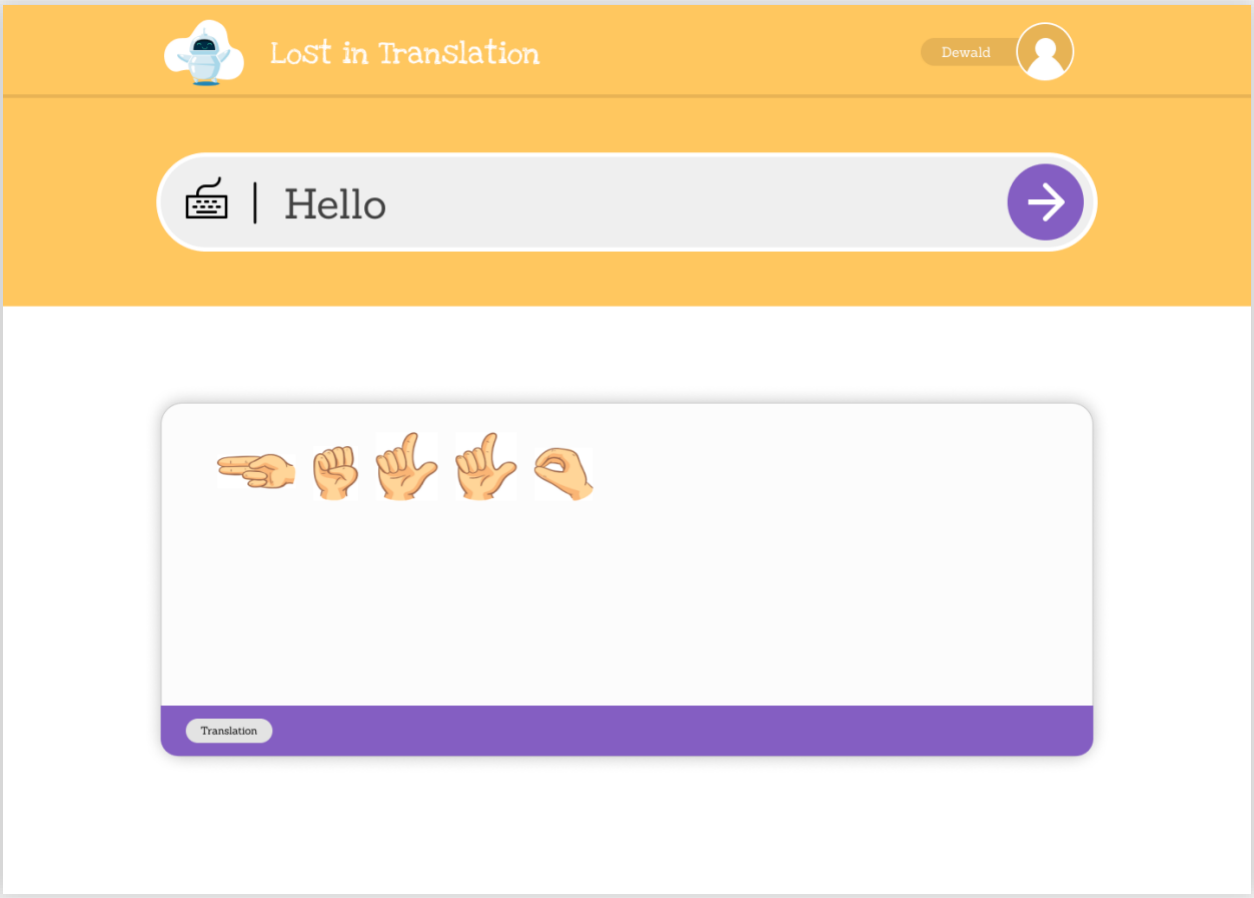

Figure 2: Example of the Translation page

A user may only view this page if they are currently logged into the app. Please redirect a user back to the login page if now active login session exists in the browser storage.

The user types in the input box at the top of the page. The user must click on the “translate” button to the right of the input box to trigger the translation.

Translations must be stored using the API (See Required features for more information). The Sign language characters must appear in the “translated” box. You may choose to limit the input to 40 letters. (Not required).

You may ignore special characters and spaces.

IMPORTANT NOTE:
ONLY the text must be stored, not the sign language images.

3) Profile page

The profile page must display the last 10 translations for the current user. You only need to display the text of the translation.

There must also be a button to clear the translations. This should “delete” in your API and no longer display on the profile page. To simplify things, you may simply delete the records, but you should NEVER delete data from a database.

The Logout button should clear all the storage and return to the start page. You may design this however you’d like.

Required features

The following features/tools must be present in the application:

- **React framework**
- **React Router** to navigate between pages
- **API** to store the users and their translations

1. IF YOU DO NOT USE REDUX, make use of the ContextAPI to manage state

Optional features

The following features/tools are **optional** in the application:

- **React-Redux**
    - https://react-redux.js.org
    - I highly recommend using the Redux Toolkit: https://redux-toolkit.js.org