CMPS 356 – Fall 2022
Web Applications Design and Development

Lab Final

Instructions

1. Skim through the whole document before you start writing any code
2. Develop the solution in your private repository under `final`
3. Commit your changes at the beginning of the session and every 30 minutes
4. Fix all warnings and push your final work before the end of the session
5. Any form of plagiarism will be treated seriously and reported

1. Favorite Countries using Next

1. Create a new Next 13 application in your private repository under `final` using `npx create-next-app@latest --experimental-app`.
2. Install React Query 4+ and MUI 5+.
   2.1. Use React Query for fetching data throughout the application: `npm install @tanstack/react-query`. A query client must be created and provided.
   2.2. Use MUI for interface components throughout the application: `npm install @mui/material @mui/icons-material @emotion/react @emotion/styled`.
3. Update the root layout to include a header, main content, and a footer.
   3.1. Define and use `Header` and `Footer` components.
   3.2. Create a `Head` component that receives a `title` property in `head.jsx`.
   3.3. Display the emblem of the United Nations from `https://upload.wikimedia.org/wikipedia/commons/e/ee/UN_emblem_blue.svg` as an icon in the header with a link to the homepage.
   3.4. Display the flag of the United Nations from `https://upload.wikimedia.org/wikipedia/commons/2/2f/Flag_of_the_United_Nations.svg` in the center of the homepage.
   3.5. Add basic copyright information to the footer.
4. Create API endpoints to serve the country data available from `https://stefangabos.github.io/world_countries`:
   4.1. Create an endpoint, `/api/countries` to serve the list of countries from `data/countries.json`.
   4.2. Create an endpoint, `/api/subdivisions/[country]` to serve the list of subdivisions for a given country from `data/subdivisions.json`.
   4.3. Create an endpoint, `/api/flags/[alpha2]`, to serve a flag for a given country in 128*128 PNG format from `data/flags.json`.

5. Create a page, `/countries`, that displays to list of countries in a responsive grid format using a card for each country with its code, flag, and names in English and Arabic.
6. Create a page, `/countries/[country]`, that displays the list of subdivisions of a given country.
7. Add an action to favorite a country (to be visited in the future) and updates the list of favorites on the server side.
    7.1. Create/update an endpoint and use a mutation to keep the client/server states synchronized.
    7.2. Create a page, `/favorites?order=asc|des`, to display the list of favorite countries. The query parameter `order`, when provided, specifies the sorting order of the list of favorite countries.
    7.3. Update your endpoint to support the `order` parameter, that is, do not sort the list on the client side.
8. Add a navigation bar within `Header` with links to every page in your application and highlight the active page.
9. Create a page, `/egg`, that is only visible in the navigation bar after the user has visited `/countries` and `/favorites`. The user should be redirected to `/` if they try to visit `/egg` without having visited both `/countries` and `/favorites`. This page should display the elapsed time since the user first visited the application.
10. Add loading interfaces, error handlers, and custom heads to all your pages. The title for every page should be set for every page in head.