**Stages building a React app**

**סעיף 1 – יצירת פרויקט חדש – ללא ניקוד**

צרו פרויקט SPA חדש, התקינו dependencies ככל שתידרשו כפי שנלמד בכיתה

Basic

1. Create the basic app
   1. In the address bar of the folder you wish to place your project:

**cmd**

* 1. In the cmd create the project:

**create-react-app countries --template typescript**

**cd countires**

**code .**

1. Open development server – run your project – in a new terminal:

**npm start**

1. לנקות קצת... Clean the template of React:
   1. In App.tsx:
      * Delete the Header section and the unnecessary Import
   2. In App.css:
      * Delete all the unnecessary styling
2. Install necessary packages – in a different terminal:

**npm i react-router-dom @types/react-router-dom**

**npm i axios**

**npm i react-icons**

1. Create new components, as your project require – in the terminal:
   1. Layout components

**create fc Layout/Header**

**create fc Layout/Main**

**create fc Layout/Menu**

**create fc Layout/Routing**

* 1. Pages

**create fc Pages/Home**

**create fc Pages/Page404**

**create fc Pages/CountriesCard**

**create fc Pages/CountriesTable**

**create fc Pages/About**

* 1. Shared

**create fc Shared/CountryCard**

**create fc Shared/CountryTable**

1. Insert the Layout components into the App component
   1. In App.tsx
   2. Make sure each component is imported

**<Header/>**

**<Menu/>**

**<Main/>**

1. Format the App.css:
   1. Reset the style and create a grid:

**.App{**

**height: 100%;**

**width: 100%;**

**display: grid;**

**grid-template-rows: repeat(12, 1fr);**

**grid-template-columns: repeat(12, 1fr);**

**}**

* 1. Create the grid for all the components of the App:

**.App>.Header {**

**grid-column: 12 span;**

**grid-row: 1 span;**

**}**

**.App>.Menu {**

**grid-column: 2 span;**

**grid-row: 11 span;**

**}**

**.App>.Main {**

**grid-column: 10 span;**

**grid-row: 11 span;**

**}**

1. In Index.css:
2. Import a font if you'd like

**@import url('https://fonts.googleapis.com/css2?family=Poppins:ital,wght@0,100;0,200;0,300;0,400;0,500;0,600;0,700;0,800;0,900;1,100;1,200;1,300;1,400;1,500;1,600;1,700;1,800;1,900&display=swap');**

**body {**

**font-family: Poppins;**

**-webkit-font-smoothing: antialiased;**

**-moz-osx-font-smoothing: grayscale;**

**}**

1. Reset the styles

**html, body, #root{**

**height: 100%;**

**width: 100%;**

**padding: 0px;**

**margin: 0px;**

**}**

1. Create basic classes that will be needed across the project

**.center {**

**display: flex;**

**justify-content: center;**

**align-items: center;**

**}**

**.centerColumn}**

**display: flex;**

**flex-direction: column;**

**justify-content: center;**

**align-items: center;**

**}**

* 1. Determine variables for colors, fonts etc (you can select color themes in colorhunt or cooler).

:**root {**

**--primary-color: #** **FFF4E0;**

**--secondary-color: #** **B46060;**

**--background-color: #** **4D4D4D;**

**--font-small: 20px;**

**--font-medium: 35px;**

**--font-large: 45px;**

**}**

* 1. If you don't have a light/dark mode, set the color and background color

**body {**

**font-family: Poppins;**

**-webkit-font-smoothing: antialiased;**

**-moz-osx-font-smoothing: grayscale;**

**color: var(--primary-color);**

**background-color: var(--background-color);**

**font-size: var(--font-medium);**

**}**

**סעיף 3 – יצירת מנגנון Routing – 30 נקודות**

יצירת מנגנון Routing מתאים המאפשר לגלוש לדף הבית ולדף המדינות ע"י התפריט הראשי.

\*\* שימו לב – אני פותרת את סעיף 3 לפני סעיף 2 כיוון שבלעדיו לא אראה בדפדפן אף אחד מהשינויים שאערוך בסעיף 2.

Routing

1. Install react-router-dom library (the latest version):
   * + **npm i react-router-dom @types/react-router-dom**
2. Create content to the Home and Page404 components
   1. In Home.tsx:

**<div className="Home centerColumn">**

**<h2>Welcome to the Countries platform</h2>**

**<div className="centerColumn">**

**<iframe**

**src="https://giphy.com/embed/llJ72v9vCcVgbUXSss"**

**width="300"**

**height="500"**

**frameBorder="0"**

**allowFullScreen**

**title="home"></iframe>**

**</div>**

**</div>**

* 1. In Page404.tsx:

**<div className="Page404">**

**<h3>Sorry, the page you were looking for is unavailable.</h3>**

**<h3>But do not worry! Our sharpest minds are working on it right now...</h3>**

**<div className="centerColumn">**

**<iframe**

**src="https://giphy.com/embed/Zk9mW5OmXTz9e"**

**width="480"**

**height="314"**

**frameBorder="0"**

**allowFullScreen**

**title="page404"></iframe>**

**</div>**

**</div>**

1. Create content and style to the Menu component
   1. In Menu.tsx
   2. Make sure the Link is imported
      * **<div className="Menu">**

**<Link to="home">Home</Link>**

**<Link to="tasks">Tasks</Link>**

**<Link to="about">About</Link>**

**<Link to="credits">Credits</Link>**

**</div>**

1. Create content to the Routing component
   1. In Routing.tsx
   2. Make sure that the Routes and Route are imported
   3. Make sure every component is imported.
   4. Make sure you have an index (for Home) and an \* (for Page404) elements.
   5. The name of the page in the path needs to be identical to the one in the menu component (in the Link):

**<div className="Routing">**

**<Routes>**

**<Route path="/" element={<App/>}/>**

**<Route path="home" element={<Home />}/>**

**<Route index element={<Home />}/>**

**<Route path="about" element={<About/>}/>**

**<Route path="countriesCard" element={<CountriesCard/>}/>**

**<Route path="countriesTable" element={<CountriesTable/>}/>**

**<Route path="\*" element={<Page404/>}/>**

**</Routes>**

**</div>**

1. Support URL & Components in index.tsx
   1. Add and import <BrowserRouter>
   2. Comment <React.StrictMode> for later

**ReactDOM.render(**

**//<React.StrictMode>**

**<BrowserRouter>**

**<App />**

**</BrowserRouter>**

**//</React.StrictMode>**

**document.getElementById('root') );**

1. Create content to the Main component
   1. In Main.tsx
   2. Make sure Routing and Outlet are imported
      * **<div className="Main">**

**<Routing/>**

**<Outlet/>**

**</div>**

* 1. Style the Main component, including treating the Routing (i.e., son) component in Main.css:

**.Main {**

**height:100%;**

**width: 100%;**

**}**

**.Routing{**

**height:100%;**

**}**

1. Insert the Layout component to App.tsx and make sure each component is imported in the top of the page:
   * + **<div className="App">**

**<Header />**

**<Menu />**

**<Main/>**

**</div>**

**סעיף 2 – הסדרת Layout – 10 נקודות**

צרו את ה-Layout הכללי בקומפוננט App שיכיל רכיבים כמו Header, Menu, Main

אין צורך ליצור Footer.

* יצירת Component Header המכילה את כותרת האתר ותפריט
* יצירת Component Main שתציג תוכן מתחלף
* יצירת Component Menu שתציג את אפשרויות הניווט באתר
* יצירת About Component שתציג מידע כללי על האתר (2-3 שורות)

Layout

1. Creating a header component the includes headline and a small menu:
   1. Creating the content in Header.tsx:
   2. Make sure the Link is imported

**<div className="Header">**

**<h1>Countries</h1<**

**<div className="center menuList">**

**<Link to="home">Home</Link>**

**<Link to="about">About</Link>**

**</div>**

**</div>**

* 1. General style in index.css:

**.menuList{**

**font-size: var(--font-medium);**

**}**

**a{**

**text-decoration: none;**

**color: var(--secondary-color);**

**}**

**a:visited, a:active{**

**color: var(--secondary-color);**

**}**

**a:hover{**

**font-weight: 800;**

**}**

* 1. Specific style in Header.css:

**.Header {**

**text-align: center;**

**}**

**.Header .menuList{**

**justify-content: space-evenly;**

**}**

**.Header \*{**

**margin: 0.2rem;**

**}**

1. Creating a Main component that will present a changing content – see Routing above.
2. Creating a Menu component
   1. In Menu.tsx:

**<div className="Menu">**

**<div className="centerColumn menuList">**

**<Link to="countriesCard">Countries Card</Link>**

**<Link to="countriesTable">Countries Table</Link>**

**</div>**

**</div>**

* 1. In Menu.css:

**.Menu \*{**

**margin-top: 3rem;**

**margin-left: 0.5rem;**

**}**

1. Creating a Home page and Page404 - content – see Routing above.
2. Creating an About component
   1. In About.tsx:

**import "./About.css";**

**import { MdOutlineWavingHand } from 'react-icons/md';**

**function About(): JSX.Element {**

**return (**

**<div className="About">**

**<h2>Welcome to the Countries platform <MdOutlineWavingHand/></h2>**

**<p>This platform presents all the countries listed in our database, with the following information: the name of the country, the name of the capital, the size of the population and the flag.</p>**

**<p>The data can be presented in the form of cards or a table. You can switch between forms of presentations by pressing the menu on the left side.</p>**

**<p>The platform also allows you to filter the the countries based on their names, using the search bar above the presentation. When filtering the presentation include only countries that have the letters you've entered in their name.</p>**

**</div>**

**סעיף 4 – משיכת מידע משרת מרוחק והצגתו באתר – 30 נקודות**

לצורך הבאת המידע, יש לגלוש ב-AJAX לכתובת הבאה, המחזירה JSON של כל המדינות: [https://res tcountries.com/v2/all](https://restcountries.com/v2/all)

הציגו את המידע בתצורת טבלה או כרטיסיות, עבור כל מדינה יש להציג:

1. name – שם המדינה.
2. capital – עיר הבירה של המדינה.
3. population – מספר התושבים במדינה.
4. flag – דגל המדינה (רוחב 80 פיקסלים, גובה 50 פיקסלים).

השמות הללו אלו בדיוק השמות המופיעים ב-JSON המוחזר.

Networking

1. Import axios

**npm i axios**

1. Create the interface of the data set:
   1. Copy the data set from the server
   2. Find an on line website that transform JSON to typescript and paste the data there. This will create an interface of the data set, i.e. a model of the way the data is build.
   3. In VScode, in the src folder, create a new folder name Models (right click on the folder, select the "new folder" option and write the name Models).
   4. In VScode, in the Models folder, create a new ts file and name it Country.ts (right click on the folder, select the "new file" option and write the name you want with a postfix of ts).
   5. Copy the interface from the website (without the two upper rows with the Root) to the ts file and change the name of the interface to a meaningful one, such as CountriesModel.

**export interface CountriesModel {**

**name: string**

**topLevelDomain: string[]**

**alpha2Code: string**

**alpha3Code: string**

**callingCodes: string[]**

**capital?: string**

**altSpellings?: string[]**

**subregion: string**

**region: string**

**population: number**

**latlng?: number[]**

**demonym: string**

**area?: number**

**timezones: string[]**

**borders?: string[]**

**nativeName: string**

**numericCode: string**

**flags: Flags**

**currencies?: Currency[]**

**languages: Language[]**

**translations: Translations**

**flag: string**

**regionalBlocs?: RegionalBloc[]**

**cioc?: string**

**independent: boolean**

**gini?: number**

**}**

**export interface Flags {**

**svg: string**

**png: string**

**}**

**export interface Currency {**

**code: string**

**name: string**

**symbol: string**

**}**

**export interface Language {**

**iso639\_1?: string**

**iso639\_2: string**

**name: string**

**nativeName?: string**

**}**

**export interface Translations {**

**br: string**

**pt: string**

**nl: string**

**hr: string**

**fa?: string**

**de: string**

**es: string**

**fr: string**

**ja: string**

**it: string**

**hu: string**

**}**

**export interface RegionalBloc {**

**acronym: string**

**name: string**

**otherNames?: string[]**

**otherAcronyms?: string[]**

**}**

1. Create a component that holds the data from the data set in the form of cards and another to hold the card of each item, and another to hold the data in the form of a table and another to hold the table.
   1. Create a relevant component in the Pages folder

**create fc Pages/CountriesCard**

**create fc Pages/CountriesTable**

* 1. Create a relevant component in the Shared folder

**create fc Shared/CountryCard**

**create fc Shared/CountryTable**

1. Create a prop for the data in the card component and insert the data into the component itself
   1. In the card component, CountryCard.tsx
   2. Before the function create an interface that accept the CountryModel. from the CountryModel.ts file and make sure that the model is imported

**import { CountryModel } from "../../../Models/Country";**

**import "./CountryCard.css";**

**interface CountryCardProps{**

**country: CountryModel;**

**}**

* 1. After the fonction, insert the model into the card component, including the img, and make sure to pass the props as an argument for the function

**function CountryCard(props: CountryCardProps): JSX.Element {**

**return (**

**<div className="CountryCard">**

**<h3>{props.country.name}</h3>**

**<p>Capital: {props.country.capital}</p>**

**<p>Population: {props.country.population}</p>**

**<p className="center">**

**<img className='flag' src={props.country.flags.png} alt={`${props.country.name}'s flag`} />**

**</p>**

**</div>**

**);**

**}**

1. Format the card component.
   1. In CountryCard.css.

**.CountryCard {**

**margin: 1rem;**

**border: 1px solid var(--secondary-color);**

**border-radius: 10px;**

**padding: 1rem;**

**box-shadow: 5px 5px 10px var(--primary-color);**

**font-size: var(--font-small);**

**}**

* 1. In Index.css:

**h2{**

**margin: 1rem;**

**}**

**h3{**

**margin: 0.5rem;;**

**}**

**.flag{**

**width: 80px;**

**height: 50px;**

**}**

1. Import the data and insert it into the Component in cards form:
   1. In the parent component, CountriesCard.tsx
   2. After the function and before the return – create a state and import the data from the online json file.
   3. The state initialized with an empty array.
   4. Make sure to close the useEffect hook with []
   5. Make sure that the model is imported from the ts file.
   6. Make sure the hooks and the axios are imported.

**import { useEffect, useState } from "react";**

**import "./CountriesCard.css";**

**import { CountryModel } from "../../../Models/Country";**

**import axios from 'axios';**

**import CountryCard from "../../Shared/CountryCard/CountryCard";**

**function CountriesCard(): JSX.Element {**

**const [country, setCountry] = useState<CountryModel[]>([]);**

**useEffect(() => {**

**axios.get('https://restcountries.com/v2/all')**

**.then(res => {setCountry(res.data);})**

**.catch(err => {console.log('The data did not load');});**

**}, []);**

* 1. After the return – call the card component
  2. Make sure to write the key

**return (**

**<div className="CountriesCard">**

**<h2>Countries</h2>**

**<div className="cards center">**

**{country.map(c => <CountryCard key={c.name} country={c}/>)}**

**</div>**

**</div>**

**);**

* 1. NOTE: this component will change in the next section.

1. Format the component:
   1. In the ContriesCard.css:

**.CountriesCard {**

**}**

**.CountriesCard .center{**

**justify-content: space-evenly;**

**}**

**.CountriesCard .cards{**

**flex-wrap: wrap;**

**}**

1. Same thing for the CountryTable: rreate a prop for the data in the card component and insert the data into the component itself
   1. In the table component, CountryTable.tsx
   2. Before the function create an interface that accept the CountryModel from the CountryModel.ts file and make sure that the model is imported.
   3. Note that here we import the data in the form of an array

**import { CountryModel } from "../../../Models/Country";**

**import "./CountryTable.css";**

**interface CountryTableProps{**

**country: CountryModel[];**

**}**

* 1. After the fonction, insert the model into the card component, including the img, and make sure to pass the props as an argument for the function

**function CountryTable(props: CountryTableProps): JSX.Element {**

**const header = ['Name', 'Capital', 'Population', 'Flag'];**

**return (**

**<div className="CountryTable">**

**<table>**

**<thead>**

**<tr>**

**{header.map(h => <th key={h}>{h}</th>)}**

**</tr>**

**</thead>**

**<tbody>**

**{props.country.map(c =>**

**<tr key={c.name}>**

**<td>{c.name}</td>**

**<td>{c.capital}</td>**

**<td>{c.population}</td>**

**<td>**

**<img className='flag' src={c.flags.png} alt={`${c.name}'s flag`} />**

**</td>**

**</tr>**

**)**

**}**

**</tbody>**

**</table>**

**</div>**

**);**

1. Format the table component.
   1. In CountryTable.css:

**.CountryTable {**

**font-size: var(--font-small);**

**}**

**table td, table th {**

**border: 1px solid var(--secondary-color);**

**padding: 8px;**

**}**

**table th{**

**text-align: left;**

**background-color: var(--secondary-color);**

**color: var(--background-color);**

**}**

1. Import the data and insert it into the Table component.
   1. In the parent component, CountriesTable.tsx
   2. After the function and before the return – create a state and import the data from the online json file.
   3. The state initialized with an empty array.
   4. Make sure to close the useEffect hook with []
   5. Make sure that the model is imported from the ts file.
   6. Make sure the hooks and the axios are imported.

**import { useEffect, useState } from "react";**

**import "./CountriesTable.css";**

**import { CountryModel } from "../../../Models/Country";**

**import axios from "axios";**

**import CountryTable from "../../Shared/CountryTable/CountryTable";**

**function CountriesTable(): JSX.Element {**

**const [country, setCountry] = useState<CountryModel[]>([]);**

**useEffect(() => {**

**axios.get('https://restcountries.com/v2/all')**

**.then(res => {setCountry(res.data);})**

**.catch(err => {console.log('The data did not upload');})**

**},[]);**

* 1. After the return – call the table component
  2. Note that here there is no need form map or key because the CountryTable component in itself maps the data.

**return (**

**<div className="CountriesTable">**

**<h2>Countries</h2>**

**<div className="table center">**

**<CountryTable country={country}/>**

**</div>**

**</div>**

**);**

* 1. NOTE: this component will change in the next section.

**סעיף 5 – תמיכת חיפוש באתר – 30 נקודות**

יצירת תיבת חיפוש מדינה, כך שבכל הקלדה הטבלה תתעדכן ותציג רק את המדינות המכילות את מחרוזת החיפוש – בשם המדינה או בעיר הבירה שלה.

Filtering

1. For the cards display add state for the search, search input and conditional rendering with the filter function
   1. In the CountriesCard.tsx
   2. After the function but before the return – add a state for the search

**const [search, setSearch] = useState<string>('');**

* 1. After the return – add an input and conditional rendering
  2. Note that for display purposes I've changed the h2 and put it inside a header

**return (**

**<div className="CountriesCard">**

**{/\* <h2>Countries</h2> \*/}**

**<header className="center">**

**<h2>Countries</h2>**

**<input type="text" placeholder="Search country's name or capital" className="CountryCard"**

**value={search} onChange={(e) => setSearch(e.target.value.toLowerCase())}/>**

**</header>**

**<div className="cards center">**

**{**

**(search.length > 0) ?**

**country.filter(c => c.name.toLowerCase().includes(search) || c.capital?.toLowerCase().includes(search)). map(c => <CountryCard key={c.name} country={c}/>)**

**:**

**country.map(c => <CountryCard key={c.name} country={c}/>)**

**}**

**</div>**

**</div>**

**);**

1. Format the component:
   1. In CountriesCard.tsx:

**.CountriesCard .center{**

**justify-content: space-evenly;**

**}**

1. Similarly, for the CountriesTable. But not that the filter function is a little different from the cards, since the mapping of the array takes place inside the CountryTable component and not the parent component, namely CountriesTable.
   1. In CountriesTable.tsx:
   2. After the function and before the return – add a state for the search

**const [search, setSearch] = useState<string>('');**

* 1. After the return – add a search input and conditional rendering
  2. Note that for display purposes I've changed the h2 and put it inside a header

**return (**

**<div className="CountriesTable">**

**<header className="center">**

**<h2>Countries</h2>**

**<input type="text" placeholder="Search country's name or capital" className="CountryCard"**

**value={search} onChange={(e) => setSearch(e.target.value.toLowerCase())}/>**

**</header>**

**<div className="center">**

**{**

**(search.length > 0) ?**

**<CountryTable country={country.filter(c => c.name.toLowerCase().includes(search) || c.capital?.toLowerCase().includes(search))}/>**

**:**

**<CountryTable country={country}/>**

**}**

**</div>**

**</div>**

**);**

1. Format the component.
   1. In CountriesTable.tsx:

**.CountriesTable .center{**

**justify-content: space-evenly;**

**}**