Kroll Ontrack
Instrukcja Angular API – LAB 1

Api: https://jsonplaceholder.typicode.com/
1. Wstępna konfiguracja projektu
a. Uruchomienie instalacji pakietów komendą npm install
2. Wygeneruj http service za pomocą CLI (ng g s http)
3. Dodaj provider HttpService w app.module
4. Dodaj import HttpClientModule w app.module (koniecznie za BrowserModule)
5. W klasie HttpSevice wskrzyknij HttpClient
6. Dodaj metodą getPosts(), która wykona GET (https://angular.io/guide/http)
7. Wstrzyknij HttpService do app.component
8. Stwórz metodę loadData() która zasubskrybuje się do metody getPosts() z httpService i zapisze odpowiedź do
zmiennej posts
9. Wywoał metodę loadData() wewnątrz onInit
10. Za pomocą interpolacji wyświetl w pierwszej przygotowanej komórce tabeli post.title, a w drugiej post.body
11. Analogicznie stwórz metodę postPost() w httpService oraz addPost() wewnątrz komponentu
12. Przekaż do dodania obiekt zapisany w zmiennej post
13. Wywoaj metodę addPost() po kliknięciu przygotowanego buttona
14. Zaloguj w konsoli odpowiedź serwera
15. Wykonaj DELETE oraz PUT bez podpowiedzi
EXTRA

1. Obsługa błędów

Instrukcja Angular API – LAB 2

1. Zmodyfikowanie utworzonego wcześniej serwisu by komunikował się z prawdziwym backendem:
    a. Ustawienie odpowiednich endpointów (localhost którego używa project backendowy + odpowiednia ścieżka z kontrolera)
    b. Ustawienie CORS po stronie backendowej – w przypadku zablokowanych zapytań wychodzących z frontendu (https://stackoverflow.com/questions/31942037/how-to-enable-cors-in-asp-net-core)
2. Modyfikacja modelu postu po stronie frontendowej, w taki sposób by odzwierciedlał model po stronie backendowej.
3. Wyświetlenie w konsoli listy postów pobranej z serwisu.
4. Stworzenie dwóch nowych komponentów (nie zapomnij dodać referencji do app.module):
    a. Post List Component
    b. Post Component
    Propozycja Struktury projektu:
    Root Component (app component) -> Post List Component -> Post Component
5. Instalacja paczki primeNG I dodanie odpowiednich zależności:
    https://www.primefaces.org/primeng/#/setup
    a. Terminal:
    - npm install primeng --save
    - npm install @angular/animations@latest –save

    b. Plik package.json, tablica “dependencies”:
    "@angular/animations": "5.2.10",

    c. Plik styles.css:

```
@import '../node_modules/primeng/resources/themes/nova-light/theme.css';
@import '../node_modules/primeng/resources/primeng.min.css';
```

    d. Plik app.module:
    import { BrowserAnimationsModule } from '@angular/platform-browser/animations';
    W tablicy “imports” dodaj: BrowserAnimationsModule
6. Implementacja tabeli w komponencie Post List Component w której wyświetlone zostaną posty pobrane z backendu. Wykorzystanie komponentów primeNG:
    https://www.primefaces.org/primeng/#/table

    Referencja w app.module:
    import { TableModule } from 'primeng/table';
    W tablicy “imports” dodaj: TableModule
7. Dodanie przycisku w tabeli umożliającego usuwanie danego posta.
    https://www.primefaces.org/primeng/#/button
8. Dodanie przycisku pod tabelę umożliwiającego dodanie nowego posta. Przycisk ten tworzyć ma komponent Post Component który zawierać będzie logikę potrzebną do tworzenia postów (kolejny punkt instrukcji).

9. Implementacja logiki w komponencie Post Component, pozwalającej na dodawanie nowych postów (bez listy komentarzy, wystarczy tylko: id, title oraz description).
   https://www.primefaces.org/primeng/#/inputtext

   Referencja w app.module:
   import { FormsModule } from '@angular/forms';
   import { InputTextModule } from 'primeng/inputtext';
   W tablicy “imports” dodaj: FormsModule, InputTextModule
10. Modyfikacja stworzonego wyżej mechanizmu, tak by obsługiwał również edycję istniejących postów pobranych z backendu.
11. Modyfikacja Post Componentu w taki sposób, by wyświetlał pobrane z backendu listy komentarzy dla danego posta.
    https://www.primefaces.org/primeng/#/table
12. Dodanie mechanizmu pozwalającego na dodawanie oraz usuwanie komentarzy dla danego posta w Post Component.

13. Paginacja
    I.Dla <p-table> w blog-post-list.component dodać prostą paginację, bazującą na endpoint'cie wyciągającym w ramach wydarzenia ngOnInit() wszystkie istniejące BlogPosty, tak aby użytkownik widział po 3 posty na każdej stronie.

    II.Dostosować blog-post-list.component <p-table> tak aby korzystał z endpointu:

    GET api/v2/blogposts[?pageIndex=3&pageSize=10]

    W celu wyciągania jedynie tych postów, które powinny być widoczne na zaznaczonej stronie.
    Podpowiedź:

    Można użyć (onPage) do podpięcia metody, która będzie wywoływana za każdym razem kiedy zmieni się wybrana strona. Do wyliczenia numeru strony do jakiej użytkownik chce przejść można użyć następującego równania:

    pageNumber = event.first / event.rows;


Add new post
Title: (First letter capital, min 3 characters)
Post Title
Description:
Post Description
Post Comments:
Author:
Post Title:
Delete:
Author:
Comment Author
Content:
Comment Concent
Add comment!
Save changes!