

# React – Services

Sladana Turudić - [sladjanaturudic@uns.ac.rs](mailto:sladjanaturudic@uns.ac.rs)

Teodora Ruvčeski - [teodoraruvceski@uns.ac.rs](mailto:teodoraruvceski@uns.ac.rs)



# Servisi



# Servisi

- Servisi u Reactu predstavljaju 'klase' koje su dostupne različitim komponentama
- Osnovna uloga servisa je komunikacija sa serverskim delom aplikacije
- Servisima se takođe omogućava komunikacija između komponenti
- Servisi se kreiraju u zasebnom folderu

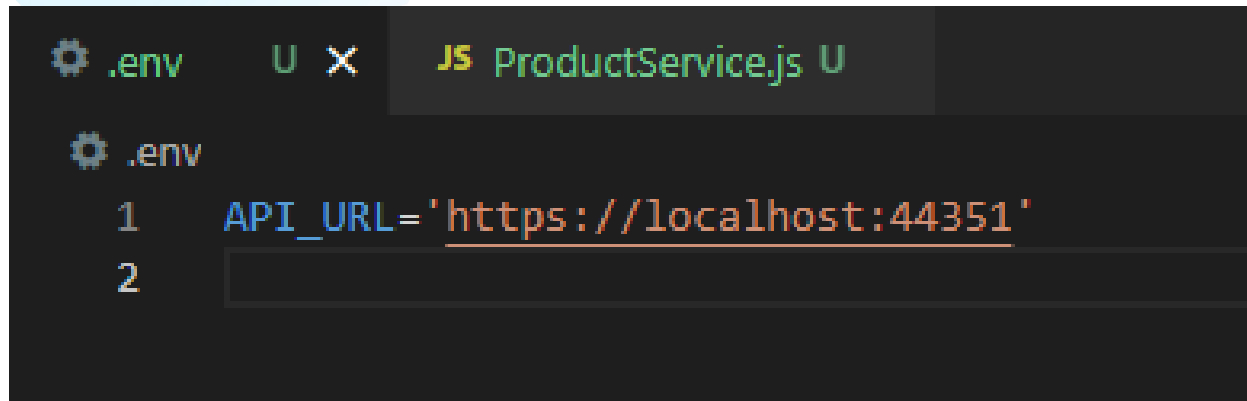


# Servisi

- URL-ovi API-ja koje servisi kontaktiraju se obično drže u `.env` fajlu i učitavaju iz njega
- Na ovaj način možemo imati više konfiguracija, jednu lokalnu i jednu kada dignemo sajt negde.
- Takođe, ako se URL servera promeni nećemo morati da menjamo na više mesta već samo na jednom.



# Servisi



The screenshot shows a code editor with two tabs: `.env` and `JS ProductService.js`. The `.env` tab is active, displaying the following content:

```
.env
1 API_URL='https://localhost:44351'
2
```

- Primer skladištenja varijabli u `.env` fajlu



# Servisi

```
.env U JS ProductService.js U X
src > services > JS ProductService.js > ...
1  import axios from 'axios';
2
3  export const GetProducts=async()=>>
4  {
5  return await axios.get(`${process.env.API_URL}/api/Products`);
6  }
7
```

```
import {useEffect,useState} from 'react';
import { GetProducts } from '../services/ProductService';
function Home(){
  const [products,setProducts]=useState([]);
  useEffect(()=>
  {
    const get=async()=>>
    {
      const resp= await GetProducts();
      console.log(resp);
      setProducts(resp.data);
    }
    get();
  },[]);
},[]);
```



# Components vs Services

- Komponente se bave view-related funkcionalnostima
  - atributi i metode za data-binding
- Servisi su zaduženi za:
  - dobavljanje podataka sa servera
  - validiranje korisničkih unosa
  - logovanje podataka na konzolu



# Export vs Export Default

- Ako u .js fajlu koristimo **export default** na jedan od navedenih načina

```
function Product(){  
  return(  
    <div>Product</div>  
  )  
}  
export default Product;
```

```
export default function Product(){  
  return(  
    <div>Product</div>  
  )  
}
```

- Prilikom importovanja komponente Product koja se nalazi u datom fajlu koristimo sledeći način importovanja:

```
import Product from './components/Product'
```





# Export vs Export Default

- Ako u .js fajlu koristimo **export** za više komponenti, funkcija ili konstanti kao u primeru:

```
export const GetProducts = async () =>
{
  return await axios.get(`${process.env.API_URL}/api/Products`);
}

export const AddProduct = async () =>
{
  return await axios.post(`${process.env.API_URL}/api/Products`, 'NewProduct');
}
```

- Prilikom importovanja komponenti ili funkcija koje se nalaze u datom fajlu koristimo sledeći način importovanja:

```
import { GetProducts, AddProduct } from '../services/ProductService';
```



# Tokeni



# Tokeni

- Ako želimo da sačuvamo tokene dobijene od nekih eksternih servisa i koristimo ih za autentifikaciju to možemo učiniti tako što ih stavimo u localStorage.

```
Login('admin','admin').then((data)=>
{
  localStorage.setItem('token',data.token);
})
```

- Na koji način je token vraćen zavisi od servisa koji ga izdaje, ali je ideja generalno ista



# Umetanje tokena

- Kako bismo se autentifikovali prilikom slanja zahteva serverima moram u zahtev umetnuti token koji nam je server izdao.
- Jedan od načina je da unutar svakog zahteva koji šaljemo ka serveru uvrstimo token unutar header-a



# Umetanje tokena

```
export const GetToken={()=>
{
  return localStorage.getItem('token');
}
const config={
  headers: {
    "Authorization" : `Bearer ${GetToken()}`
  }
};
export const GetProducts = async () =>
{
  return await axios.get(`${process.env.API_URL}/api/Products`,config);
}
```



# Guards



# Guarded Routes

```
import React from "react";
import { Route, Redirect } from "react-router-dom";
import { useSelector } from "react-redux";
import { isAuthSelector } from "../store/auth";

function PrivateRoute(props) {
  const isAuth = useSelector(isAuthSelector);

  return isAuth
    ? <Route {...props} />
    : <Redirect to="/login" />;
}

export default PrivateRoute;
```



# Termin 3 zadatak

- Na stranici registration omogućiti registraciju novog korisnika podešavanjem forme.
  - Dodati metodu u Auth servis koja omogućava registraciju, proveriti koju putanju na zadnjoj strani treba gađati i šta joj treba proslediti.
  - Nakon uspešne registracije preusmeriti korisnika na stranicu za prijavu
  - Na Home stranici omogućiti da korisnici vide proizvode koje preko servisa dobavljaju sa api-ja
  - Limitirati pristup toj strani na samo autentifikovane korisnike
- 





**HVALA NA PAZNJI!**

