2022

Svendeprøve dokumentation

ONA GIRDZIJAUSKAITĖ

Ona Girdzijauskaite
10/10/2022

Tech-Stack:

• React
• React-route
• Tailwinds
• axios

Grundlæggende teknologier:

HTML : struktur
CSS : presenation
JavaScript : logik

Core Frameworks:

Tailwind CSS – Utility based css classes

Core libraries:

- *React.js* — det et kernebibliotek open source Java-Script bibliotek , der bruges til at opbygning af UI – elementer.

Der hjælper mig at bruge midre til på koding når jeg har komponenter fra et bibliotek at gå ud fra. (mulgit for mig at henter funktionalitet fra noget paker.npm.

- *Framer* – Et produktionsklart bevægelsesbibliotek til React. Udnyt kraften bag Framer, det bedste prototypeværktøj til teams.

- React-icons

Core API's:
Context – Global stage håndtering.

Core packet:

react-error-boundary — fejlgrænse for at fange gengivelsesfejl.
react-hook-form — er et bibliotek, der hjælper dig med at validere formularer i React. Det er et minimalt bibliotek uden andre afhængigheder, mens det er effektivt og ligetil at bruge, hvilket kræver, at udviklere skriver færre linjer kode end andre formbiblioteker.
Axios – er en letvægts HTTP-klient baseret på XMLHttpRequests-tjenesten. Det ligner Fetch API og bruges til at udføre HTTP-anmodninger.
Yup – Yup er en JavaScript-objektskemavalidator.

Versionsstyring:

Github:

Brug af GitHub har mange fordele, herunder lettere samarbejde med kolleger og kolleger, evne til at se tilbage på tidligere versioner og mange nemme integrationsmuligheder.

- Få din kode gennemgået af fællesskabet

- GitHub er et depot

Dette var allerede nævnt før, men det er vigtigt at bemærke - GitHub er et depot.

Hvad det betyder, at det giver dit arbejde mulighed for at komme ud der foran offentligheden. Desuden er GitHub et af de største kodende samfund omkring lige nu, så det er bred eksponering for dit projekt.

- Samarbejd og spor ændringer i din kode på tværs af versioner

  Meget som at bruge Microsoft Word eller Google Drive, kan du have en versionshistorik for din kode, så tidligere versioner ikke går tabt ved hver iteration.

  GitHub sporer også ændringer i en changelog, så du kan få en nøjagtig ide om, *hvad* der ændres hver gang. (Dette er især nyttigt for at se tilbage i tiden.)

Deploy process:

Tech-Stack perspektivering:

React vs vanile

Jeg har haft to muligheder hvordan skulle jeg lave min examen projekt.

Jeg har brugt at bruge Ract fordi :

Jeg kunne godt bruge Vanile JavaScript eller React. Jeg har valgt React fordi for mig det var nemmere på grund af:

- Npm paker som hjæleper maget
- Jsx jeg kan bruge – JavaScript Syntax Extention , som oftes anbefales at bruge i React JS
- Komponenter – React.Js bruges som en komponentstruktur, der bruges til at arbejde i et stort project med en anden modulopbygget struktur.
- Enkeltvejsstrøm – Den implementerer envejs dataflow ved hjælp af Flux-mønster for at holder dataener i en retning.

Jeg vil i min præsentation kommer nærmere ind på …

```
useEffect(()=>{
    // todo : implement proper middlewares to handle auth
    // if the user is logged in redirect him to home page
    const userData = window.localStorage.getItem('user')
    if(userData) redirectToHomePage()
}
    ,[])


    // Login api + safe in local storage

    const loginHandler = async({username , password}) =>{
       try {
        // the link im senden user name and password to this link
        //I get the respoce and than i get the token

            const response = await axios.post('http://localhost:4000/auth/token' , {username , password})

        // Here im sving dATA TO MY LOCAL STORAGE
        // than we  redirectToHomePage()
        window.localStorage.setItem('user' , JSON.stringify(response.data))
        setTimeout(()=>{
            redirectToHomePage()
        },300)
        } catch (error) {
            //
        }
        }
```