**REMIX RAPPORT**

**Præsentation af rapport**

I denne rapport vil vi gå igennem mange punkter der forklare om hvad react er, hvorfor det bliver brugt, remix funktioneliteter og andvendelsesmulighederne ved at bruge Remix.js.

**Hvad er remix og hvorfor bruger vi det**

Remix.js er en gratis open source full-stack-framework(når en framework bliver brugt til både backend og frontend) lavet af react-router teamet der er designet til at hjælpe udviklere som mig med at at opbygge hurtige og pålidelige applikationer ved hjælp af react.

Remix giver også webuviklere som mig muligheden for at lave sider med server side rendering(SSR) som kan forbedre ydeevnen af siden ved at reducere loading-tiderne og forbedre SEO på samme tid.

Vi vil typisk bruge remix til at lave hurtige hjemmesider med mange sider der er afhængig af dynamisk data som er gemt i en database og vi gør typisk dette ved at oprette en form, som jeg ville i basic html hvor jeg så håndtere alt mit data der ligger på server siden.

I tilføjelse til server siden vil det også tilbyde os webudviklere en moderne udviklings oplevelse der inkluder hot module reloading, fejl håndtering, dynamisk routing og en hurtig udviklings server der giver gøre det nemmere og meget hurtigere at udvikle web application og ligende.

**Remix funktionaliteter**

Hvordan opbygger man så en remix side efter man har installeret alting og skal i gang med at kode? Ville en person der aldrig har arbejdet med remix højst nok spørge om, og selvom jeg er ny til remix er svaret til dette meget næm, man opbygger det med at bruge de funktionaliteter og muligheder react giver dig for at opbygge en side.

En ef remix’s mest kendte funktionaliteter er dens routing, i react når man router ville man typisk lave noget routing inde i sin ”app.jsx” eller ”main.jsx” og det ville man lave med at skrive massere linje af kode der inkludere at importere fra react-router hvor man så også skal implementere alt muligt, men i remix er så simpelt at putte dine filer inde i router mappen der automatisk bliver lavet når du installer remix.js.

Routing på remix foregår således at hvert eneste route på route mappen er ikke seperate men de er alle nested sammen som en nested route. Hvert side i route mappen er nested sammen istedet for at være seperate som gør det nemere at indlejre komponenterne ind i vores side, hvor den så vil give os mindre loading-tider.

Derimod er der også en anden form for routing der hedder dynamisk routing som sørger at vi kan tage data fra en anden route og sende det videre, og der er mange måder man kan lave dynamic routes på i remix ved hjælp a file naming convections.

File naming convections er noget der ligger under dynamic routing og der mange måder man kan lave det på fx når man skal til at lave nested urls uden layout nesting, vil man typisk bruge ”trailing\_” underscoren til at, sørge for en path segment der ikke skaber noget layout nesting.

En typisk dynamic route ville ligge inde i route mappen sammen med alle de andre route baseret ting.

Et eksempel på dette ville typisk se sådan ud, ”$notes\_.notesId.jsx”.

Endnu en af remix’s funktioner som er virklig godt at bruge er dens fejl håndtering funktionalitet kaldet ”Errorboundary”.

Errorboundary er en funktion der tager og håndtere alle vores errors.

Når vi typisk får errors vil en side eller applikation nogle gange crashe eller lukke applikationen ned, men med errorboundary vil vi få et display der giver os en fallback error besked der sørger for ar vi næmt kan gendanne os fra fejlen og få den håndteret.

De sidste to funktioner jeg vil nævne er, remix’s versioner af actions og loaders.

Loaders i remix sørger for at, når vi går ind på den afsluttede side eller klikker på noget i en side, loader den med det samme, men bag scenerne sørger remix for at alt dataen var loadet før den loadet selve siden, imens action sørger for at der er server side kode i backended der bliver henrettet.

Resultatet af Action funktionen vil til aldersidst blive til resultatet af vores enpoint.

**Remix andvendelsesmuligheder**

Efter at have gået alle de punkter igennem, kan jeg se at remix har mange andvendelsesmuliger for virksomheder der gerne gider at have applikationer der har en høj kvalitet med hurtige load-tider.

Det vil også være godt at bruge til håndtering af state, da det inkludere en state management system der gør det nemt at administrere state tilstande så vi får en sikker applikation der forbliver responsive og fungerer godt, selv når der bliver ændringer på koden vi arbejder med.

Og til aldersidst i min mening ”testing”, da remix har en række værktøjer og funktioner, der kan hjælpe med at teste vores kode og sikre, at vores applikation fungerer som det skal.

Alt i alt, selvom det er et nyt værktøj til at opbygge sider med, er det stadigvæk et kraftfuldt og hurtigt værktøj til at bygge webapplikationer med mange tilpasnings, state management og test muligheder.

I 2 uger har jeg arbejdet med remix.js og i de 2 uger jeg har arbejdet med det har jeg fået en god forståelse af alle de anvendelsesmuliger remix har og hvorfor vi burde holde et godt øje til remix, især med alle dens nye og opdaterede funktioner der er i version 2 når den udkommer.

**Afslutning af rapport**

Alt i alt er remix et godt værktøj til at bruge til webudviklere der gerne gider at have en applikation der er hurtig, takket være alle dens funktioner, serverside rendering og dens relative næmme forståelse der kan få udviklere som mig til at få en hurtig applikation, der kan anvendes på mange måder, men der ikke er for bloatet.

Jeg vil forsætte med at holde øje med remix da jeg synes det har været en spændende oplevelse, med at lære alt der har med remix at gøre og føler at mit sind er blevet åbnet op får nye muligheder for kodning.