Vitec - Hjemmesideprojekt

Meldgaard, Mike Holst
mike2860@edu.ucl.dk

Rasmussen, Anders Fredensborg
ande714b@edu.ucl.dk

Sandbøl, Kaare Veggerby
kaar1498@edu.ucl.dk

Thomsen, Johannes Ehlers Nyholm
joha321j@edu.ucl.dk

13. december 2019

Indhold

1 Formål

Vi har i dette projekt arbejdet med Vitec MV, som leverer software, der hjælper ordblinde. Lige nu er deres primære kunder skoler og kommuner. De ønsker dog at kunne sælge deres produkter til privatpersoner, da ordblindhed er en lidelse man har gennem hele livet. Derfor har vores opgave været at bygge en platform, der henvender sig til privatpersoner.

Vitec MV ønsker at sælge CDORD og Intowords til privatpersoner. Vores formål var at lave et proof of concept på, hvordan man kan henvende sig til det ønskede kundesegment. Vitec MV vil gerne have, at der fokuseres på brugervenlighed.

2 Hjemmeside

Vi har valgt at prøve at opfylde Vitec MVs krav ved at bygge en hjemmeside, som kan findes på URL'en `http://skilsmis.se/`.

Hjemmesiden er lavet med ASP.NET Core. Fordelen ved ASP.NET Core er, at det giver os Razor Page filformaten .cshtml, hvilket betyder, at vi kan skrive C#i vores HTML. Det betyder, at vi kan lave interaktive hjemmesider uden at skulle skrive Javascript.

3 Arkitektur

ASP.NET Core bruger to lagmodeller: Razor Pages, hvilket desværre har samme navn som filformaten, og MVC. Razor Pages kører med arkitektur mønstret model-view-viewmodel(MVVM), og MVC bruger selvfølgelig arkitektur mønstret model-view-controller.

Begge arkitekturmønstre har fordele og ulemper. Det er meget nemt at overholde designprincippet Single Responsibility i Razor Pages, da en side kun kan have ansvaret for, hvad der skal vises på den side. Derudover er det meget hurtigt og nemt at lave simple hjemmesider, der blot skal lave CRUD på datamodeller. Consensus er dog, at hvis din hjemmeside skal være mere kompleks, som f.eks single page applications, eller hvis den gør brug af en REST api, er MVC bedre at bruge.

Da vi har delt vores hjemmeside op i flere dele:

- En database
- En api
- En hjemmeside, som brugeren tilgår.

har vi også valgt at køre med MVC arkitekturen. For at undgå god classes har vi valgt at have flere controllers, som hver har ansvar for dele af vores hjemmeside.

4 API