PROJEKT WEBSHOP

Opgaven består af en Server del, og en Klient del.

Serveren er EF Core 5 Web Api baseret på en Controller – Service – Repository struktur.

Klienten er en Angular 13 app, der via komponenter og services arbejder med API'en

Projektets tema er en hjemmeside baseret webshop, hvor I selv vælger hvilke produkter butikken skal sælge. Dog er det et krav der er flere forskellige kategorier, hvor hver kategori har flere forskellige produkter.

Der er et par krav til hvad hjemmesiden skal indeholde og de funktioner der som minimum skal være til stede.

Gennemgående elementer, fælles for alle sider

Der skal være en menu så man kan navigere rundt på siden, og et gennemgående design så siderne fremstår som en samlet løsning. Tænk header, footer, nav, main.

Kurven skal være synlig fra alle sider, så den kan tilgås når der er puttet varer i den.

Forsiden

Her vises et udvalg af butikkens produkter. Det kan f.eks. være de 3 nyeste produkter. Eller måske 3 tilfældige produkter.

Man skal kunne klikke på de viste produkter, og så blive ledt til en produktside.

Produkt Side

Her vises alt information om det specifikke produkt, gerne med billeder. Herfra skal man kunne vælge hvor mange man vil købe, og derefter "putte i kurven" som opdateres automatisk,

Kategori Side

Alle produkter hører til i en kategori, menuen skal indeholde links til alle kategorierne.

På kategorisiden vises alle de produkter som hører til den valgte kategori, man kan klikke på et produkt og blive sendt til produkt siden.

Vis Kurv Side

Når der er varer i kurven, skal man kunne klikke ind og se hele kurven, hvor hvert produkt vises med antal og pris, samt en total pris for hele kurven.

Man skal kunne ændre på hver varelinje i kurven, og man skal kunne klikke ”Køb”

Ved køb skal brugeren være logget på, så enten ledes man til en loginside, ellers hvis man allerede er logget ind afsluttes købet (vi antager betaling og levering er helt ok)

Når købet er ”gennemført” vises en bekræftelse på siden om at købet er gennemført.

Login side

Her skal man kunne logge på som kunde.

Hvis man ikke allerede har en kundekonto, skal man kunne oprette en konto.

Opret Kunde

Her skal man kunne angive kunde data, som minimum email og kodeord. Efter oprettelse sendes man til login, eller systemet logger den nye kunde på automatisk.

Søge Side

Her kan man udføre søgninger som viser de produkter der opfylder søgekriterierne.

F.eks. kunne det være fritekst søgning på navn og beskrivelse, og søgning med minimum / maksimum pris.

Administrations sider.

Dette område skal være gemt bag et login, så det kun er medarbejdere der kan få adgang. Loginformularen, kan sagtens være den samme som kundelogin.

Administrationen kan være et helt nyt design, det behøver ikke reflektere butikkens design.

Der skal være en menu, så man nemt kan navigere rundt på administrationssiderne.

Som minimum skal man kunne følgende:

- Fuld CRUD på kategorier
- Fuld CRUD på produkter
- Fuld CRUD på kunder/medarbejdere
- Overblik over ordrer der er oprettet igennem kurven.

Bonus funktionalitet

Hvis ovenstående er opfyldt, og der er overskud til at udvide med lidt ekstra, så er der rig mulighed for at tilføje en masse lækkert, her er en liste af forslag

- Glemt Kodeord funktionalitet, hvor man kan klikke på et link der er sendt til en mail (simuleret mail afsendelse ”smtp4dev”/”mailtrap”)
- Aktiver kunde via link sendt på mail (simuleret mail afsendelse ”smtp4dev”/”mailtrap”)
- Produkter har et ”antal på lager” så man kan vise udsolgte produkter
- Produkter kan sættes på tilbud, f.eks sættes en særlig pris i en bestemt tidsperiode. Produkter på tilbud vises med særlig formattering på hjemmesiden.
- Produkter kan have forskellige varianter, f.eks. t-shirt i 5 forskellige farver. Samme produkt, men forskellige varianter.

Tekniske krav til projektet

Serveren baseres på et ASP.NET Core Web API projekt.

Alle controllers, Services og Repositories skal have unit-tests som bekræfter at koden gør det den bør gøre.

Controller testes om endpoint metoderne returnerer korrekte statuskoder.

Services testes om de kan returnere korrekte data eller fejler hvis data mangler.

Repositories testes om data der kommer fra databasen, opfylder bestemte kriterier.

Alle repositories og services implementeres med et interface. Dette hjælper jer med at arbejde sammen, og det hjælper jer i Xunit test scenarierne.

Alle DTO's sørger for validering af bruger input, dvs om feltet er required og om data opfylder nødvendige kriterier f.eks email og lign.

Klienten sættes op med Routing, og al kommunikation til API foregår via httpServices.

Arbejdsproces

I arbejder i grupper, optimalt på 3 personer.

Inden kodningen kan gå i gang, er det nødvendigt at planlægge dele af projektet.

- Der skal udarbejdes et E/R diagram over hele databasen.
- Alle interfaces planlægges, dvs:
  - Repositories
  - Services
- Alle endpoints planlægges.

Kodebasen lægges på github, i det assignment I skal acceptere når vi går i gang med webshop projektet.

Alle sørger for at committe og pushe deres koder dagligt, i branches.

Master branch er den branch hvor koden fungerer, dvs ingen udvikler direkte på master.

Det er helt fint at en branch committes med kode der ikke er færdig implementeret.

Commit ofte, push ofte.

Merge (pull-request) når koden virker.

Aflevering

Afleveirngen vil være det sidste commit der ligger på github, den sidste dag.