Eksamensopgave

Kursus: SW4FED-02 Front-end Development
Varighed: 24 timer
Eksamenstermin: Eksamen juni 2024
Praktiske informationer: Eksamensbesvarelsen skal afleveres i ZIP-format. Husk at aflevere før deadline som er angivet i Wiseflow, hvor opgaven skal afleveres.
Hjælpemidler: Alle hjælpemidler må benyttes, herunder internettet som opslagsværktøj, men opgaven skal besvares individuelt. Og husk referencer hvis du genbruger kode fra andre opgaver eller projekter – skal indsættes som en kommentar. Både brug af ChatGPT og AI integreret i en codeeditor så som Intellisense og GitHub CoPilot er tilladt.
Ansvarlig underviser: Poul Ejnar Rovsing

Indledning
Ved besvarelsen af opgaverne skal du huske, at du ved fremlæggelsen skal kunne demonstrere opfyldelse af kursets læringsmål ud fra opgaverne. Din opgavebesvarelse skal derfor dække så mange læringsmål som muligt.

Opgave 1
Du skal udvikle en applikation til at registrere og persistere opgaver for et bilværksted.

Programmet skal kodes i C# til .Net-platformen med anvendelse af MAUI og køre på en device efter eget valg. Du skal selv fastlægge programmets brugergrænseflade og softwarearkitektur, samt hvilken funktionalitet der eventuelt implementeres ud over den grundlæggende funktionalitet specificeret her.

Grundlæggende funktionalitet:

Appen skal bruges af bilværkstedet til at registrere hvad en kunde ønsker at få lavet ved sin bil. Så som service og olieskift, skift sommer-/vinterdæk, nye dæk, reparation mv. Samt nogle andre oplysninger som anført herunder.

1: Book ny opgave
Brugeren af appen (værkstedet) skal kunne indtaste:
Kundens navn og adresse
Bilmærke og model
Indregistreringsnummer
Data og klokkeslæt for indlevering af bilen.
Hvad der skal laves

2: Oversigt over opgaver for en valgt dato
Brugeren navigerer til kalendersiden, hvor der kan vælges en dato, hvorefter appen viser de opgaver (ordrer) der er planlagt den pågældende dato.

3: Specifikation af udført arbejde
Brugeren navigerer til fakturasiden, hvor der kan registreres: mekanikerens navn, hvilke materialer der er brugt og deres pris, samt antal timer og deres pris.

Alle data persisteres. Du bestemmer selv hvordan. Du kan f.eks. benytte en SQLite database, men du kan også vælge at benytte filer eller en json-server (som i opgave 2, som du kan tilgå med HttpClient).
Der er ikke krav om udskrift af faktura.

Opgave 2 er på næste side

Opgave 2
Du skal udvikle en front-end til en Web applikation som bilværkstedets kunder kan bruge til selv at booke en tid på værkstedet.

Front-enden skal udvikles som en React app. Du skal selv fastlægge brugergrænsefladen, men funktionaliteten skal være som anført nedenfor. Eksemplet på hvordan data kunne struktureres i "dokumentdatabasen" (json-filen), som er vist i bilag1, er kun vejledende, du må gerne lave din løsning anderledes.
Der er ikke krav om login, og som server bruges en lokal json-server som vist i lektion 19 "React Fetching data" samt i lab 22 (https://github.com/typicode/json-server).

Grundlæggende funktionalitet
Bemærk at alt indtastet skal persisteres via et REST api som tilbydes af json-server.

1: App'en skal have 4 sider
Home (landing page), book ny aftale, ret aftale og vis aftaler.

2: Home
Siden home indeholder kun statisk tekst om værkstedet.

3: Book ny aftale
Her kan brugeren udfylde en formular med de oplysninger værkstedet har brug for, og så trykke book tid, hvorefter der oprettes en booking hos værkstedet (i "databasen" på json-serveren).
Hint: Man kan lave en søgning ved brug af en query streng. Feks.
GET /appointments?licensePlate=AL12345

4: Ret aftale
Her vises øverst en mulighed for at søge efter enten et navn eller en nummerplade (f.eks. ved brug af en listbox) samt et input felt til den værdi, der skal søges efter.
Den fundne aftale vises under søgefelterne, og brugeren har så mulighed for at ændre i aftalen, og få ændringerne persisteret ved tryk på knappen opdater.

5: **Vis aftaler**
Her vises øverst en mulighed for at søge efter en dato, og under disse søgefelter vises en liste med de fundne aftaler for den pågældende dato.

Forslag til strukturering af datafil, som definerer API'et (db.json), er vist i bilag 1. Bemærk, at dette kun er vejledende.