## Opgave 1: Frontend-Integration med Lagerstyringssystemet

Du skal nu udvikle en front-end side til det lagerstyringssystem i udviklede i sidste uge, produktet skal præsenteres for facilitator og medkursister i slutningen af ugen. Overvej hvordan du skal lave et interface der afspejler den database i har udviklet og giver brugeren mulighed for at navigere databasens indhold præsenteret på en visuelt let forståelig facon. Dlt formål forventes det at du inddrager:

1. **Din frontend-kode** (inkl. kommentarer, der forklarer vigtige dele af koden).
2. **Kort beskrivelse af din løsning**, der forklarer, hvordan frontend'en henter og viser data fra Cereal API'et.
3. **Opsætningsvejledning** til at køre frontend’en, inkl. de nødvendige afhængigheder.
4. **Kode-dokumentation** af din frontend-struktur og MVVM-implementering.

**Baggrund:**

I denne opgave skal du bygge en frontend til det lagerstyringssystem, som du tidligere har arbejdet på. Frontend'en skal hente informationer fra det Cereal API, du har udviklet, og vise dem på en webside. Formålet er at skabe en brugervenlig og responsiv grænseflade, der giver brugerne mulighed for at interagere med og filtrere data om cerealprodukter.

Opgaven skal implementeres ved hjælp af **MVVM-arkitekturen** (Model-View-ViewModel), hvilket vil sikre en klar adskillelse af applikationens data, logik og præsentation. Dette er afgørende for at opretholde en høj grad af modularitet, der gør koden lettere at vedligeholde, teste og udvide.

**API-fordele og anvendelsesområder:**

**Fordele ved API-integration:**

* **Forbedret funktionalitet:** Frontend'en kan hente data i realtid fra API'et, hvilket giver brugerne mulighed for at se opdaterede informationer uden at genindlæse siden.
* **Modularitet og fleksibilitet:** Ved at bruge et API kan du adskille backend (Cereal API) fra frontend, hvilket giver mulighed for at ændre eller udskifte backend'en uden at skulle ændre frontend-koden.
* **Bedre brugeroplevelse:** Med API-integration kan du opnå dynamisk datavisning og opdateringer, hvilket forbedrer brugerens interaktion med systemet.
* **Skalerbarhed:** API'er tillader nemt at udvide systemet med nye funktioner eller tredjepartsintegrationer, såsom rapportering eller e-handelsintegration.

**Eksempler på anvendelse:**

* **Interaktive dashboards:** Frontend kan vise realtidsdata om lagerbeholdning, salg og produkttilgængelighed, hvilket hjælper med hurtige beslutninger.
* **Datafiltrering:** Brugere kan søge og filtrere produkter baseret på parametre som kalorier, protein, sukkerindhold osv., hvilket gør det lettere at finde de ønskede produkter.

**Hvad er MVVM?**

MVVM (Model-View-ViewModel) er en softwarearkitektur, der adskiller applikationens data (Model), brugergrænsefladen (View), og logikken, der binder de to sammen (ViewModel). Dette gør det muligt at opretholde en klar opdeling af ansvar og fremmer genbrugelighed og testbarhed i koden.

* **Model:** Indeholder applikationens data og forretningslogik. I dette tilfælde vil Model-laget repræsentere cerealprodukter og hente data fra Cereal API'et samt inkludere databaseintegration for at lagre og hente data.
* **View:** Består af brugergrænsefladen, som præsenterer data for brugerne. View inkluderer de visuelle elementer som tabeller, knapper og søgefelter, der giver brugerne mulighed for at interagere med cerealdata.
* **ViewModel:** Fungerer som mellemleddet mellem Model og View. ViewModel håndterer logikken for brugerinteraktioner, såsom at hente data fra API'et, formatere det korrekt, og sende det videre til View. Det styrer også opdateringer af View, når Model-data ændres, og håndterer brugerens input og filtrering.

**Foreslået Løsningsmetode:**

1. **Frontend-opbygning:**
   * Byg en enkel og responsiv webside, som henter og viser data fra Cereal API'et.
   * Sørg for, at der er visuelle komponenter, såsom lister, tabeller og søgefelter, hvor brugeren kan interagere med dataene.
2. **Implementer MVVM-arkitekturen:**
   * **Model:** Definer, hvordan cerealprodukterne skal repræsenteres, og hvordan dataene hentes fra API'et. Sørg for at inkludere felter som navn, kalorier, protein, fedt osv.
   * **ViewModel:** Håndterer API-kald til Cereal API'et. Det skal hente data, formatere dem, og sende dem videre til View for præsentation. Det skal også håndtere brugerinput, såsom filtrering og sortering.
   * **View:** Skab et brugervenligt layout, der præsenterer data fra ViewModel og muliggør filtrering baseret på brugernes søgekriterier.
3. **Funktionaliteter:**
   * **Visning af data:** Frontend'en skal vise cerealprodukterne i en overskuelig og responsiv tabel eller liste.
   * **Filtrering:** Brugerne skal kunne filtrere dataene baseret på parametre som kalorier, sukker, fedtindhold og producent.
   * **Brugerinteraktion:** Implementer søge- og filtreringsmuligheder, der giver brugeren fleksibilitet i at finde specifikke produkter.
4. **Responsivt design:**
   * Sørg for, at layoutet tilpasser sig forskellige skærmstørrelser, så det er brugervenligt både på desktop og mobile enheder.
5. **Integration med Cereal API:**
   * Frontend'en skal hente data fra API'et gennem asynkrone HTTP-kald (GET-requests) og vise dataene på en struktureret og filtrerbar måde.

**Yderligere Tips til Opgaveløsning:**

* **Sørg for korrekt opsætning af MVVM-strukturen:** Dette sikrer, at din frontend-løsning er modulær, hvilket gør den lettere at udvide og vedligeholde.
* **Testbarhed:** Med en klar opdeling af logik, data og præsentation kan du lettere teste de enkelte komponenter i din applikation separat.
* **Brugeroplevelse:** Fokuser på, at frontend'en er nem at bruge og responsiv, så den kan tilgås fra forskellige enheder uden problemer.

**Guides til udviklingen:**

* **TypeScript basics:** [TypeScript fundamentals](https://app.pluralsight.com/library/courses/typescript-5-fundamentals/table-of-contents)
* **Web build med ASP.NET og React:** [ASP.NET Core 6 og React](https://app.pluralsight.com/library/courses/asp-dot-net-core-6-web-api-react-building-end-to-end-spa/table-of-contents)
* **HTML, CSS, JavaScript:** [Grundlæggende webudvikling](https://app.pluralsight.com/library/courses/html-css-javascript-getting-started-building-websites/table-of-contents)
* **YouTube-guide:** [Frontend udvikling med MVVM](https://www.youtube.com/watch?v=I2NNxr3WPDo)

**OBS:** Hvis du føler det er realistisk inden fortidsrammen må du gerne vælge at redesigne/arbejde videre på et ligende projekt til din front-end det vigtigste er at det understøtter de områder du bliver bedt om at have fokus på i dit front-end design.