ORM – case study

Opgave: ORM Case Study

Mål:

- **Basal viden om ORM:** Lærlingen skal opnå en grundlæggende forståelse af, hvad ORM er, hvordan det fungerer, og hvorfor det bruges i moderne softwareudvikling.
- **Anvendelse af en færdig database:** Lærlingen kan anvende en færdig database og lade ORM generere domain-modellen.
- **Oprettelse af en domain-model:** Lærlingen kan oprette en domain-model og lade ORM generere databasen og forespørgsler.

1. **Udforsk:** Først og fremmest skal du og din makker vælge et ORM-framework at undersøge. Det kunne være Entity Framework (for .NET), Hibernate (for Java), Django ORM (for Python) eller et andet, du er interesseret i.

2. **Forskning:** Næste skal I bruge tid på at undersøge og lære om ORM-frameworket. Nogle spørgsmål, du måske vil besvare, inkluderer:
    - Hvad er de vigtigste funktioner i dette ORM-framework?
    - Hvordan håndterer dette ORM-framework relationer mellem forskellige tabeller (one-to-one, one-to-many, og many-to-many)?
    - Hvordan bruger dette ORM-framework migrations til at styre ændringer i databasens skema?
    - Hvordan kan man optimere ydeevnen med dette ORM-framework?
    - Hvordan håndterer dette ORM-framework arv i både Code-First og Database-First tilgange?

3. **Anvendelse:** Derefter skal I anvende, hvad I har lært ved at bruge både Database-First og Code-First tilgange i et lille projekt. I begge tilfælde skal I inkorporere brugen af arv, implementere CRUD-operationer (Create, Read, Update, Delete), og oprette relationer mellem tabellerne (one-to-one, one-to-many, og many-to-many). For eksempel kan I:

    - **Database-First:** Opret en lille database for en fiktiv virksomhed med forskellige tabeller (f.eks. "Medarbejdere", "Afdelinger", "Projekter") og brug ORM-frameworket til at generere den tilsvarende kode. Sørg for, at mindst en af jeres tabeller demonstrerer et eksempel på arv (f.eks. en "Manager" tabel, der arver fra "Medarbejder"), skaber relationer mellem tabeller, og implementerer CRUD-operationer for denne tabel.
    - **Code-First:** Skriv kode for at definere en anden virksomheds data (f.eks. "Kunder", "Ordrer", "Produkter") og brug ORM-frameworket til at

generere den tilsvarende database. Også her skal mindst en af jeres klasser demonstrere et eksempel på arv (f.eks. en "VIPKunde" klasse, der arver fra "Kunde"), skabe relationer mellem klasser, og I skal implementere CRUD-operationer for denne klasse.

4. **Præsentation:** I skal forberede en præsentation for klassen om jeres fund og projekt. Præsentationen skal inkludere en forklaring af, hvordan ORM fungerer, en demonstration af jeres kode og CRUD-operationer, og en diskussion af fordele og ulemper ved at bruge ORM, og ved at bruge Database-First og Code-First tilgange. I bør også forklare, hvordan arv kan håndteres i ORM og hvordan forskellige typer af relationer kan repræsenteres. Præsentationen skal ikke vare mere end 20 minutter.