Du er blevet hyret som konsulent til at udvikle et mindre lagerstyringssystem i Python med en database til at gemme data i (MySQL).

Kunden ønsker at du opbygger systemet med anvendelsen af OOP-principper som klasser, arv, og polymorfisme og relevante design pattens.

Kunden ønsker at al data gemmes i en MySQL database, denne database må gerne ligge lokalt på jeres computer, i dette tilfælde.

Krav til systemet

I lagersystemet skal det være muligt at:

- Tilføje, fjerne, og opdatere varer
- Vise den aktuelle lagerbeholdning
- Generere relevante rapporter – *Her har kunden behov for jeres hjælp til forslag*
- Systemet ska tilgås via en menu i en terminal

Forslag til opbygning

Varer: Hver vare kan repræsenteres som et objekt med følgende egenskaber:

- Vare-ID (unik identifikator)
- Navn
- Beskrivelse
- Kategori
- Pris
- Antal på lager

Kategorier: Varer kan inddeles i kategorier for lettere organisering og søgning. En kategori kan have følgende attributter

- Kategori-ID
- Kategorinavn
- Beskrivelse

Transaktioner: For at spore bevægelser af varer ind og ud af lageret, kan transaktioner registreres. En transaktion kan indeholde:

- Transaktions-ID
- Vare-ID
- Dato og tid
- Antal (P*ositiv for indkøb/tilføjelse, negativ for salg/fjernelse*)
- Transaktionstype (*F.eks. indkøb, salg*)

Dette er et forslag i er meget velkommende til at ændre/tilføje/forbedre dette oplæg.!

Funktioner

- **Tilføjelse af varer**: Mulighed for at tilføje nye varer til lageret, inklusive alle relevante oplysninger.
- **Opdatering af lagerstatus**: Opdater antallet af varer på lager, enten ved at tilføje (*ved indkøb*) eller fjerne (*ved salg*)
- **Søgning efter varer**: Søg i lageret baseret på forskellige kriterier som vare-ID, navn, eller kategori.
- **Rapportgenerering**: Generer rapporter om lagerstatus, såsom en liste over alle varer, varer sorteret efter kategori, eller varer med lav lagerbeholdning.
- **Håndtering af kategorier**: Opret, Rediger og Slet kategorier for at organisere varerne
- **Transaktionshistorik**: Se en historik over alle transaktioner, herunder indkøb og salg af varer, for at spore lagerbevægelser over tid. Implementering
- Osv. ....

I OOP-kontekst skal du oprette klasser for hver af de ovennævnte dataobjekter (*Varer, Kategorier, Transaktioner*) og implementere de nødvendige metoder (*funktioner*) til at manipulere disse objekter.

For eksempel kan **klassen Vare** have metoder som **tilføj_vare()**, **opdater_lager()**, og **slet_vare()**, mens en Lager klasse kan administrere samlingen af alle varer og håndtere søgning og rapportgenerering.

Du skal sikre at data er beskyttet og kun kan ændres gennem veldefinerede grænseflader (*metoder*).

Arv kan bruges til at udvide funktionaliteten af basale klasser, hvis der er behov for mere specialiserede varer eller kategorier. Polymorfisme ? er dig at interagere med alle disse objekter på en fleksibel måde, hvilket gør dit system mere skalerbart og lettere at vedligeholde.

Det er vigtigt at lagersystemet har en solid struktur og at det er let at tilføje nye funktioner eller ændre eksisterende funktionalitet i fremtiden.

Database udvidelse

Hvis du vil afprøve noget mere avanceret SQL-programmering kunne en udvidelse være

- Stored Procedure til at alle database handlinger, UPDATE, INSERT, DELETE. *Har det nogen fordele? – F.eks. i forhold til sikkerhed?*
- Opret en Log tabel med brugere handlinger (UPDATE, INSERT, DELETE). Brug Triggere til dette
- Opret forskellige brugere og giv dem forskellige rettigheder

Aflevering

- I skal aflevere følgende, som aftalt på Teams mødet, på GitHub.
- Kort beskrivelse af systemet og jeres "*design*" overvejelser, meget gerne i README filen på GitHub
- Class diagram over jeres kode, som PDF
- Andre relevante UML diagrammer som PDF, kunne være; *Sequence Diagram, Activity Diagram, Use Case Diagram, ...*
- SQL-filer til oprettelse af jeres database, som lagersystem.sql fil
- ER-diagram over jeres database, som PDF
- Alle Python kodefiler
- requirements.txt fil så det muligt at afvikle jeres kode i et Virtual Environment

Kom så langt som muligt, jeg forventer ikke at alle når helt i mål med denne opgave, men jeg tænker at den indeholder rigtig mange nye, "gamle" og relevante problemstillinger og teknikker.

Last modified: Friday, 15 March 2024, 10:33 AM