Prosjekt: Musikkregister (Python + MariaDB)

Sammendrag

Du skal lage et lite konsollprogram i Python som registrerer **artister** og **sanger** i en **MariaDB**-database. Programmet har en enkel meny for å legge til, liste og oppdatere data. Dette er en praktisk oppgave der du øver på grunnleggende databasebruk fra Python.

Læringsmål

Etter oppgaven skal du kunne:

- Opprette og koble til en database fra Python (MariaDB/MySQL-connector).
- Lage tabeller med primærnøkler og fremmednøkler.
- Skrive og kjøre enkle SQL-spørringer fra Python (INSERT, SELECT, UPDATE).
- Strukturere et lite konsollprogram med funksjoner og meny.
- Bruke parameteriserte spørringer for tryggere kode.
- Bruke virtuelt miljø (venv).

Forberedelser

1. Opprett ny mappe/folder for prosjektet og aktiver virtuelt miljø:

   ```
   python3 -m venv venv
   source venv/bin/activate   # Windows: venv\Scripts\Activate.ps1
   ```

2. Installer database-driveren:

   ```
   pip install mysql-connector-python
   ```

   Har du tidliger bruk mariadb connector eller pysql connector, bruker du den også her.

3. Opprett database i MariaDB:

   ```
   CREATE DATABASE musikk CHARACTER SET utf8mb4 COLLATE utf8mb4_unicode_ci;
   ```

4. Lag filen `musikkregister.py` (beskrivelse er gitt i oppgaveteksten).

> **Tips:** Dersom databasen er på en annen maskin (Raspberry PI) i lokalnettet, endre `host` i `DB`-oppsettet til IP-adressen til databaseserveren, og sørg for at serveren aksepterer eksterne tilkoblinger (bind-

> address, bruker/tilgang, brannmur. Dette har de fleste av dere allerede gjord i forrige uke). Bruk databasebrukeren du lagde forrige uke.

Oppgavebeskrivelse

1. **Kjør programmet** og verifiser at tabellene opprettes automatisk ved oppstart.
2. **Legg til minst 3 artister** via menyen.
3. **Legg til minst 5 sanger**, fordelt på minst 2 ulike artister.
4. **List artister** og **list sanger** fra menyen og kontroller at data vises riktig.
5. **Oppdater** (minst) én artists navn og (minst) én sangtittel via menyen.
6. **Restart programmet** og sjekk at data fortsatt ligger i databasen.

Vurderingstrapp (kompetansenivåer)

Prosjektet vurderes etter IM's vurderingstrapp for kompetanseutvikling:

NOK KOMPETANSE I FAGET – HVA

Du viser at du forstår **hva** du skal gjøre, og følger en veiledning for å løse oppgaven.

- Du får programmet til å kjøre.
- Du oppretter databasen og kobler deg til den fra Python.
- Du kan legge til og vise artister og sanger.
- Du jobber etter en gitt oppskrift eller brukerveiledning.

GOD KOMPETANSE I FAGET – HVORDAN

Du viser forståelse for **hvordan** løsningen fungerer og kan gjøre endringer selv.

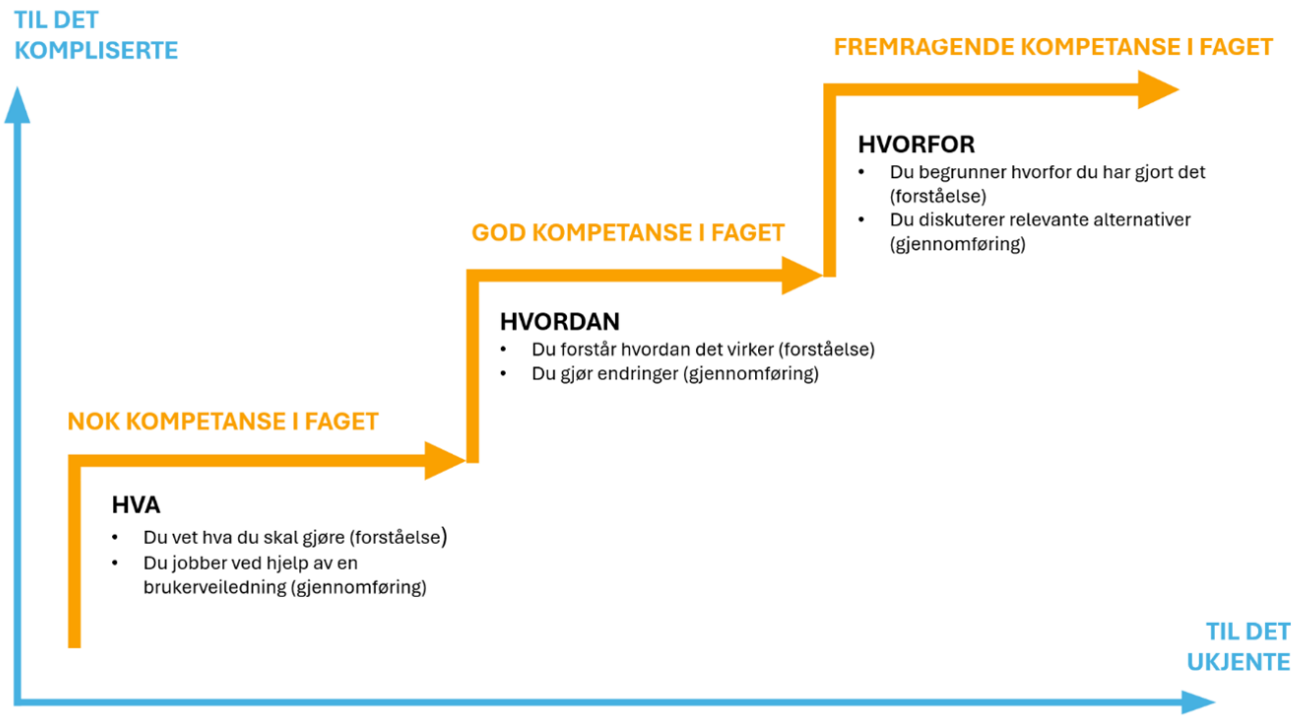
- Du forstår hvordan Python kommuniserer med databasen.
- Du kan forklare hvordan tabellene `artist` og `sang` henger sammen (fremmednøkler).
- Du klarer å gjøre endringer i koden (for eksempel legge til ny menyfunksjon eller endre utskrift).
- Du forstår hvordan SQL-spørringene brukes i koden.

FREMRAGENDE KOMPETANSE I FAGET – HVORFOR

Du kan forklare og begrunne **hvorfor** du har valgt løsninger, og reflektere over alternativer.

- Du begrunner hvorfor programmet er strukturert slik det er (for eksempel bruk av funksjoner og databasekobling).
- Du diskuterer forbedringer, som feilhåndtering, brukervennlighet eller sikrere kode.
- Du sammenligner løsningen din med andre måter å løse oppgaven på.
- Du viser evne til selvstendig utforsking og videreutvikling.

Vurderingsrubrikk



Minimumskrav

- Programmet starter uten feil og viser meny.
- Automatisk opprettelse av tabellene `artist` og `sang`.
- Legge til og liste artister og sanger.
- Oppdater minst én post.

Bonus / videreutvikling

- Legg til sletting av artist/sang.
- Lag søkefunksjon.
- Bruk `.env`-fil for å lagre passord.

Innlevering

- Python-fil (`musikkregister.py`).
- Skjermbilder av kjøring (legg til, liste, oppdatere).
- Kort refleksjon: Hva lærte du? Hva ville du forbedret?
- eventuelt lenke til GitHub-repo. Da må du ikke legge ut passord og brukernavn. Lag en .env-fil som ignoreres i .gitignore