

# Tverrspråklig Prosjektoppgave – Avansert Qt-applikasjon

## Mål

Studentene skal utvikle en **applikasjon for hverdagsbruk** (f.eks. handleliste, oppgaveplanlegger, budsjett-app, treningslogg, notatbok).

Prosjektet skal utvikles **over flere uker**, og skal inkludere **avanserte funksjoner** slik at sluttproduktet blir en **profesjonell applikasjon**.

Studentene får velge programmeringsspråk:

- **C++ (Qt Widgets)**
- **Python (PyQt5 / PySide6)**
- **Java (Qt Jambi)**
- **Etc...**

---

## Minimumskrav (grunnlag)

### 1. GUI med Qt Widgets

- Bruk QMainWindow eller QWidget med knapper, tekstfelt, tabeller/lister.
- Bruk layouts (QVBoxLayout, QHBoxLayout, QGridLayout).

### 2. Datahåndtering

- Bruk fil (TXT/CSV/JSON) **eller** SQLite-database.
- Implementer CRUD (Create, Read, Update, Delete).

### 3. Interaksjon

- Bruk signals/slots eller tilsvarende (knappetrykk → handling).
- Minst én dialog (QMessageBox, QFileDialog, etc.).

### 4. Funksjonalitet

- Brukeren skal kunne legge inn, endre og slette data.
  - Data skal kunne vises oversiktlig (liste/tabell).
-

## **Utvidede krav (avansert nivå)**

### **1. Database**

- Migrer fra enkel fil til SQLite (via QSqlDatabase og QSqlTableModel).
- Implementer søk og filtrering.
- Støtt import/eksport (CSV, JSON, XML).

### **2. GUI og brukeropplevelse**

- Flere vinduer eller faner (QTabWidget).
- Innstillinger/preferanser lagret i QSettings.
- Tema (lys/mørk modus) med Qt Style Sheets (QSS).
- Dra-og-slipp og kontekstmenyer.

### **3. Statistikk og visualisering**

- Vis grafer (QtCharts, Matplotlib i Python, etc.).
- Statistikk: summeringer, fordelinger, tidslinjer.

### **4. Avansert funksjonalitet**

- Timer/varsling med QTimer.
- Multithreading for tunge operasjoner (QThread / QtConcurrent).
- Nettverk (REST API via QNetworkAccessManager – f.eks. synkronisering med server).

### **5. Brukerstøtte**

- Innlogging med brukere/roller.
- Egne data for hver bruker.

### **6. Profesjonelle krav**

- Dokumentasjon (README/manual).
  - Versjonskontroll med Git.
  - Enhetstesting (QTest i C++, pytest-qt i Python, JUnit i Java).
-

## Vurderingskriterier

- **Funksjonalitet:** Oppfyller både grunn- og utvidede krav.
- **Teknisk kvalitet:** Bruker Qt riktig i valgt språk.
- **Brukervennlighet:** Godt design, oversiktlig GUI.
- **Profesjonalitet:** Dokumentasjon, testing, Git-bruk.
- **Kreativitet:** Egne idéer utover kravene.