

Eksamensoppgave

Emne 6 Software design



TID: Start: 2025-12-8, Innlevering: 2025-12-12

Hjelpebidrag: Alle

Innlevering:

- Zip-fil: eksamen_emne5_<navn>.zip
- README.md (kort oversikt + hvordan kjøre kode + eventuelle KI-prompter referert til)
- Videofil
- Legg ved Mappe 1 ved din innlevering! (**Du kan forbedre koden din**)

Oppgave 1

Studenten skal levere **Mappe 1** på nytt som del av eksamen.

OPS! Viktig

Selv om mappen allerede ble **levert** tidligere i semesteret, skal du levere den på nytt i Wiseflow som del av eksamen.

- Du kan forbedre koden før innlevering (anbefales)
- Du trenger ikke gjøre store endringer kun det som gjør applikasjonen tydeligere og mer profesjonell
- **Den mappen som leveres her vil være grunnlaget for vurderingen av videoen i oppgave 3**

Oppgave 2 (30%)

Som tillegg til mappe 1 skal du implementere en liten ekstra funksjon som viser at du behersker API-integrasjon i Blazor.

Du kan velge en av følgende forbedringer:

Alternativ A – Hente data fra et eksternt API

Legg til en ny komponent i prosjektet som gjør følgende:

- Sender en **GET-request** med HttpClient
- Henter en liste med objekter (f.eks. produkter, todos eller brukere)
- Viser dataene i UI med liste- eller tabellvisning
- Viser “Loading...” mens data hentes
- Viser feilmelding hvis API-kallet feiler **Krav:**
- Bruk async / await
- Bruk try/catch for feilhåndtering
- Bruk Dependency Injection for HttpClient

Alternativ B – Legge til data (POST)

Lag en komponent der brukeren kan legge inn et nytt element (f.eks. et produkt eller en todo), og:

- Sender data til API med HttpClient.PostAsJsonAsync
- Viser responsen i UI
- Validerer input før sending
- Håndterer feil og viser dem på en god måte

Alternativ C – Forbedret feilhåndtering

Lag en funksjon som demonstrerer robust API-håndtering:

- Implementer håndtering av HTTP-statuskoder (200, 400, 404, 500)
- Implementer timeout (f.eks. 30 sekunder)
- Vis passende feilmeldinger i UI
- Logg feiltyper i konsollen eller UI

Oppgave 3 (70%) - Presentasjon og Refleksjon

Målet med denne oppgaven er å reflektere over løsningen av Oppgave 1,2 og formidle denne refleksjonen på en klar og sammenhengende måte. **Instruksjoner:** A.

Lage en Video:

- Lag en video hvor du forklarer hvordan du tenkte når du løste oppgaven og hvordan du har implementert løsningen.
- Videoen bør være mellom 5-10 minutter lang. B. Inkluder følgende i videoen:
 - Kodegjennomgang:
 - Vis og forklar de viktigste delene av koden din. Du trenger ikke å gå gjennom hver eneste linje, men koncentrer deg om de mest sentrale delene i forhold til oppgavens funksjonalitet.
 - Problemløsning:

- Diskuter eventuelle utfordringer du støtte på mens du løste oppgaven og hvordan du overkom dem.
- Demonstrasjon:
 - Kjør programmet og demonstrerer hvordan det fungerer. Fokuser på hvordan programmet reagerer på forskjellige brukerinndata.

C. Vurderingskriterier:

- Klarhet: Er din forklaring klar og lettfattelig?
- Fullstendighet: Har du dekket alle aspekter av oppgaven?
- Teknisk dybde: Går du i dybden på hvordan koden fungerer, ikke bare hva den gjør? o
Presentasjon: Er videoen organisert på en måte som er lett å følge?

Tips:

- Planlegg hva du skal si før du begynner å filme for å sikre en strømlinjeformet og konsistens i presentasjonen.
- Du kan benytte skermopptaksverktøy som OBS, QuickTime (for Mac), eller andre tilgjengelige programvare for å lage videoen.
- Vær deg selv når du presenterer. Mindre feil her og der er helt akseptable, det viktigste er at du klart formidler din forståelse av problemet og løsningen

Vedlegg

Vurderingskriterier, Hva bør du tenke på før innlevering

Kjørbarhet

- Scriptene fungerer uten feil
- Delvis funksjonelle (inneholder mindre feil som påvirker output)
- Ikke kjørbare
 1. Syntax error
 2. Koden er kommentert ut

Fullførelse

- Alle oppgaver er fullført
- Oppgavene er delvis fullført
- Ingen av oppgavene er fullført

Korrektethet

- Output er i tråd med oppgaveteksten
- Mindre avvik fra oppgaveteksten
- Betydelige avvik (oppfattes som et annet arbeid)

Kodekvalitet (Lesbarhet og struktur)

Kodekvalitet (Lesbarhet og struktur) Høy kvalitet

- God kodeorganisering og ryddig struktur
- Konsistent og korrekt formattering

- Gode, beskrivende variabelnavn
- Relevante og presise kommentarer
- Koden er enkel å lese og forstå **Middels kvalitet**
- Delvis ustrukturert kode
- Manglende eller svake kommentarer
- Variabelnavn er lite presise eller blander språk
- Noe vanskelig å lese, men logikken kan følges

Dårlig kvalitet

- Uorganisert og uoversiktlig kode
- Uklare eller misvisende variabelnavn
- Manglende kommentarer
- Koden er vanskelig å lese og forstå

Video

- **Klarhet:** Er forklaringen klar og enkel å forstå?
- **Fullstendighet:** Dekker videoen alle aspekter av oppgaven?
- **Teknisk dybde:** Gir den en forståelse av hvordan koden fungerer, ikke bare dens funksjon?
- **Presentasjon:** Er videoen godt organisert og lett følgelig?

Bruk av KI (Kunstig intelligens)

Bruk av kunstig intelligens (KI) er **tillatt**, men kun under forutsetning av korrekt og fullstendig dokumentasjon.

Krav til dokumentasjon

- Du må levere alle forespørslar (prompter) og svar som er brukt i besvarelsen.
- Dokumentasjonen skal være en del av innleveringen — enten i README.md eller i en egen fil.
- Du skal kun inkludere forespørslar og svar som helt eller delvis er kopiert inn i, eller som direkte har påvirket, løsningen din.
- **Konseptuelle eller generelle samtaler** med KI som ikke har påvirket innleveringen, trenger ikke dokumenteres.
- Alternativt kan du oppgi en eksplisitt erklæring om at du ikke har brukt KI i den aktuelle innleveringen.

Viktig

- **Manglende dokumentasjon av KI-bruk regnes som brudd på eksamensreglene.**
- **Ved mistanke om udokumentert KI-bruk kan** kandidaten bli innkalt til en **mntlig kontroll** for å verifisere egen forståelse av innleveringen.
- Dersom kandidaten ikke kan gjøre rede for arbeidet sitt, eller dersom KI-bruk viser seg å være udokumentert, kan det bli behandlet som juks i tråd med skolens retningslinjer