# Fagprøve IT-Utviklerfaget

Rasmus Løvli Netland

13.08.2025 - 21.08.2025

Innholdsfortegnelse

[Fagprøve IT-Utviklerfaget 1](#_Toc205988656)

[Planleggingsdel 3](#_Toc205988657)

[Målet med oppgaven 3](#_Toc205988658)

[Fremgangsmåte 4](#_Toc205988659)

[Skisse av løsningen 6](#_Toc205988660)

[Tidsskjema 7](#_Toc205988661)

[Oversikt over utstyr 8](#_Toc205988662)

[Kostnadsoverslag 9](#_Toc205988663)

[Infrastrukturkostnader 9](#_Toc205988664)

[Engangskostnader 9](#_Toc205988665)

[Arbeidskostnader 9](#_Toc205988666)

[Totale kostnader 9](#_Toc205988667)

[Informasjonskilder og Samarbeidspartnere 10](#_Toc205988668)

[Forenklinger, Avgrensninger og Forbehold 11](#_Toc205988669)

# Planleggingsdel

## Målet med oppgaven

Målet med oppgaven er å utvikle en funksjonell og sikker handleliste-applikasjon som gjør det enkelt for brukere å planlegge, og gjennomføre handleturer. Løsningen skal være tilgjengelig på PC og mobil, slik at den kan brukes hjemme for planlegging og i butikken under handling.

Applikasjonen skal tilby:

* **Brukerhåndtering** – Registrering og innlogging med sikker passordlagring, slik at kun autoriserte brukere får tilgang til egne handlelister.
* **Fleksible handlelister** – Muligheten til å opprette én eller flere lister, legge til, redigere og slette varer.
* **Kjøpsstatus** – En tydelig måte å markere hvilke varer som er kjøpt, eller lagt i handlekurven. Slik at brukeren slipper å tenke over dette selv under en handletur.
* **Sikker behandling av data** - Person- og handledata skal være beskyttet via sikker dataoverførsel (HTTPS), kryptering og sikker databasehåndtering.

Løsningen skal være enkel å bruke for alle, rask å lære, og skal kunne utvides med flere funksjoner i fremtiden om det skulle være relevant.

## Fremgangsmåte

For å nå målet vil arbeidet organiseres i flere faser som bygger på hverandre. Hver fase har tydelige leveranser og milepæler for å sikre fremdrift.

**Fase 1** **– Planlegging og design (Dag 1)**

* Analysere kravene i oppgaven og lage en løsning som dekker alle funksjoner.
* Lage skisser av arkitekturen, databasestrukturen, kommunikasjon og brukergrensesnitt.
* Planlegge databasestruktur.
* Velge rammeverk og hosting for backend og frontend.
* Lage en tidsplan og ressursoversikt.

**Fase 2 – Oppsett av utviklingsmiljø og database**

* Installere og sette opp nødvendige utviklingsverktøy.
  + Visual Studio
  + Microsoft SQL Server Management Studio
  + .NET 8
  + Docker Desktop (for testmiljø)
* Opprette database med tabeller.
* Definere relasjoner og sikre dataintegritet med fremmednøkler.
* Definere Stored Procedures for database operasjoner.

**Fase 3 – Backend Utvikling**

* Sette opp CRUD- endepunkter for oppretting, redigering, sletting og innhenting av:
  + Lister
  + Varer i listene
* Lage registrerings og påloggings endepunkter.
  + Returnerer en «token» ved pålogging, token genereres med «utløpsdato».
  + Token kan caches på klientside, og oppdateres via API for å slippe at brukeren må logge på konstant.
* Implementere sikker passordbehandling, via bruk av salt og hashing.
  + Å bruke salt (tilfeldige karakterer lagt til passordet) øker sikkerheten ved at brukere ikke kan ha samme passord hash.
* Implementere grunnleggende validering av data, og feilhåndtering.
  + Innebakt logger, for å beholde feilmeldinger for analyse. Logges til disk eller Azure ved behov.
* Sette opp Unit tester for å sikre API funksjonalitet ved re-deployement etter potensielt fremtidig arbeid.

**Fase 3 – Frontend Utvikling**

* Implementere registrerings- og innloggingssider med kobling opp mot API.
  + Lagre bruker «token» i ett «cache»
* Lage listeoversikt med mulighet for å opprette nye handlelister.
* Lage visning av varer i liste, med mulighet til å legge til, redigere, slette og markere vare som «kjøpt».
* Gjøre løsningen responsiv for PC og mobil.
* Lage ett brukergrensesnitt som er responsivt, brukervennlig og lett å forholde seg til.

**Fase 4 – Sikkerhet og optimalisering**

* Utnytte HTTPS (SSL- sertifikat) for kryptert og sikker kommunikasjon.
* Forbedre input-validering for å unngå mulig SQL-injection.
* Sikre at API-et kun gir tilgang til brukerens egne data.

**Fase 6 – Testing og dokumentasjon**

* Gjennomføre funksjonstester av alle hovedfunksjoner.
* Brukerteste på støttede enheter.
* Fikse eventuelle feil som oppstår, eller fornedrelser som må utføres.
* Skrive ferdig teknisk dokumentasjon og brukerveiledning.

**Fase 7 – Presentasjon**

* Presentere tjeneste og oppgave for prøvenemda.

## Skisse av løsningen

A diagram of a computer

AI-generated content may be incorrect.

## Tidsskjema

Estimert timebruk 45 timer. Ekskluderer fremføring av fagprøven, og gjennomgang av oppgaven.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Tid | 13.08 | 14.08 | 15.08 | 18.08 | 19.08 | 20.08 | 21.08 |
| 08:30-  09:30 | Se gjennom oppgave med prøvenemda. | Sette opp programvare som trengs | Sette opp autentifiserings system for brukere og API. | Definere basic grensesnitt i frontend. | Ferdigstille «hoved» siden, med lister og varer. | Starte testing av hele systemet. | Fremføre fagprøve |
| 09:30-  11:00 | Planlegging av gjennomføring | Sette opp database, og definere tabeller. |  | Etabler kommunikasjon mellom frontend og backend |  |  |  |
| LUNSJ | LUNSJ | LUNSJ | LUNSJ | LUNSJ | LUNSJ | LUNSJ | LUNSJ |
| 11:30-13:00 | Lage en oversikt over utstyr og kostnader | Definere påkrevde Stored Procedures for database opreasjoner. | Fullføring av API logikk. |  | Finpusse grensesnitt og styling. | Rett opp i evt. feil og forbedre det som behøves. |  |
| 13:00-14:00 | Lage skisser for løsning, og arkitekturtegninger | Sette opp API miljø, definer endepunkter. |  | Ferdigstille registrering og påloggings sider. | Etablere hosting med HTTPS kryptering, og SSL. | Ferdigstill hele prosjektet. |  |
| 14:00-16:30 | Ferdigstille og planleggings-dokument og sende inn til prøvenemda. | Opprette database kommunikasjon, og bygg logikk for endepunktene. | Starte opprettelse av frontend prosjekt. |  | Implementere sikkerhets-tiltak. Eks. input validering |  |  |

## Oversikt over utstyr

Programvare:

* **Utviklingsverktøy**
  + **Visual Studio 2022** – Hovedverktøy for utvikling.
  + **Postman** – For testing av API-endepunkter.
  + **TortoiseGit** – For versjonskontroll og sikker lagring av kildekode i GitHub.
  + **SQL Server Management Studio (SSMS)** – For databaseadministrasjon.
  + **ReSharper** – Verktøy som analyserer koden i VS, og legger til formatting på kodestilen.
  + **Docker Desktop** – For lokal deployement av programvare og miljø.
* **Backend**
  + **ASP.NET Web Api** **.NET 8**– Rammeverket for Web APIet som håndterer databehandling for web appen.
  + **MediatR** – Bibliotek for å håndtere logikken til endepunktene. Etablerer en «mellommann» mellom komponentene i APIet, som forminsker direkte avhengigheter og gjør kodebasen lettere å vedlikeholde. (CQRS pattern)
  + **System.Data.SqlClient –** Bibliotek for å la APIet kommunisere med database.
  + **NUnit –** Testbibliotek, slik at API kan kjøre tester på funksjonalitet ved deployement.
* **Frontend**
  + **Blazor .NET 8 -** Rammeverket webappen vil bli utviklet i.
* **Database**
  + **Microsoft SQL Server –** relasjonsdatabase for lagring av brukere, lister og varer.
* **Sikkerhet** 
  + **HTTPS/SSL** – kryptert kommunikasjon mellom klienter og server.

Utstyr:

* Pc med nødvendig programvare, og tilstrekkelige ressurser til å kjøre prosjektet og programvarene. (DELL Latitude 7400)
* Mobiltelefon for testing. (SM-S24)
* Stabil internettforbindelse.
* Azure cloud for hosting.

## Kostnadsoverslag

### Infrastrukturkostnader

|  |  |  |
| --- | --- | --- |
| Tjeneste | Beskrivelse | Pris (NOK/mnd) |
| Azure MsSQL database | Database i Azure med 2vCPUs, og 10GB Minne. (Kan måtte skaleres opp i fremtiden) | 4.293kr |
| Azure App Service (P1v3) | Hosting for API, 2 vCPUs, 8GB Minne, 250GB lagring. | 1.749kr |
| Azure App Service (P1v3) | Hosting for API, 2 vCPUs, 8GB Minne, 250GB lagring. | 1.749kr |
| SSL-sertifikat | Gratis via Azure | 0kr |
| Sum infrastruktur |  | 7.791kr |

### Engangskostnader

|  |  |  |
| --- | --- | --- |
| Utstyr | Beskrivelse | Pris |
| Dell Latitude 7400 | Laptop brukt for utvikling | 7.000kr |
| Samsung S-24 | Mobiltelefon brukt for testing | 7.450kr |
| Sum Engangskostnader |  | 14.450kr |

### Arbeidskostnader

I tillegg til disse kostnadene kommer kostnadene av arbeidet. Har estimert å bruke ca. 45 timer på dette arbeidet. Med utgangspunkt i en timelønn på 500kr, tilsvarer dette 22 500kr.

### Totale kostnader

Totale kostnader på dette, beregnet fra infrastrukturkostnader (7 791kr), engangskostnader (14 450kr), og arbeidskostnader (22 500kr). Blir de totale kostnadene 44 741kr. Med løpende månedlige kostnader på 7 791kr (infrastruktur).

## Informasjonskilder og Samarbeidspartnere

**Informasjonskilder**

* **Stack Overflow** – For løsninger eventuelle problemer som kan oppstå.
* **GitHub** – For løsninger på eventuelle problemer.
* **Microsoft Docs** – Dokumentasjon for rammeverkene jeg skal benytte.
  + **SQL Docs**
  + **ASP.Net Docs**
  + **Blazor Docs**

**Samarbeidspartnere**

* **Kollegaer –** Ved eventuell nytte så kan kollegaer benyttes for informasjon, eller hjelp med problemer.

## Forenklinger, Avgrensninger og Forbehold

Grunnet naturen av oppgaven så har jeg valgt å kjøre opp miljøet mitt i Docker Desktop, for å slippe store kostnader i Azure portalen. I prinsippet vil prosjektet være identisk, jeg vil benytte de samme tjenestene, og om prosjektet engang skulle bli deployed i Azure så ville det ha krevd minimale endringer. Eneste forskjellen vil at ettersom dette kjøres lokalt, så vil ikke HTTPS kryptering inkluderes, men det ble vurdert og tenkt nøye på under planlegging, og hadde vært veldig relevant i en reell deployement. Dette ville ha blitt gjort via Azure sin gratis Let’s Encrypt tjeneste.

Jeg gjør også forbehold om at det kan oppstå endringer undervis i oppgaven, grunnet problemer eller endringer fra prøvenemda. Men grunnet hvor modulært planleggingen min er, og mtp. at alle «komponenter» i prosjektet har ansvar for sin egen logikk og sin egen del så skal dette gå fint. Målet med planen min er å ha ett mest mulig fleksibelt design, som kan ta imot endringer, og lett bli utvidet i fremtiden.