|  |
| --- |
| Emil Ømark Jensen  [Dato] |

|  |
| --- |
| [Firmanavn] |
| Gårdbutik styring system |
| [Dokumentets undertitel] |



|  |
| --- |
| **Elev:**  Emil Ømark Jensen |
| **Firma:**  Netcompany |
| **Projekt:**  Gårdbutik styring system |
| **Uddannelse:**  Datateknikker med special i programmering |
| **Projektperiode:**  11/04/2023 – 16/05/2023 |
| **Afleveringsdato:**  dd/mm/yyyy |
| **Fremlæggelsesdato:**  dd/mm/yyyy |
| **Vejledere:**  Navn på vejledere |
|  |

# Titelblad



Techcollege Aalborg,

Struervej 70,

9220 Aalborg

# 

# Læsevejledning

I læsevejledningen beskriver man hvordan rapporterne anbefales at læses og evt. andre skriftlige formalia, som forkortelser og engleske fagord.

*Alle overskrifter i parentes er ikke et krav at have med.  
Nogen overskrifter skal omskrives for at passe ind.*

# Forord

Hvad skal man vide om rapporterne og projektet før man går i gang med at læs

Indholdsfortegnelse

[Titelblad i](#_Toc132281297)

[Læsevejledning ii](#_Toc132281298)

[Forord ii](#_Toc132281299)

[Indledning 1](#_Toc132281300)

[Case beskrivelse 1](#_Toc132281301)

[Problemformulering 1](#_Toc132281302)

[Projektplanlægning 1](#_Toc132281303)

[Estimeret tidsplan 1](#_Toc132281304)

[Arbejdsfordeling 1](#_Toc132281305)

[Metode- og teknologivalg 1](#_Toc132281306)

[Applikation 2](#_Toc132281307)

[Frontend 2](#_Toc132281308)

[Backend 2](#_Toc132281309)

[Server 2](#_Toc132281310)

[Database 2](#_Toc132281311)

[Web server 2](#_Toc132281312)

[Asp.net core 2](#_Toc132281313)

[Entitet framework 2](#_Toc132281314)

[Blazor 2](#_Toc132281315)

[Docker 2](#_Toc132281316)

[PostgreSQL 3](#_Toc132281317)

[Væsentlige elementer fra produktrapporten 3](#_Toc132281318)

[Realiseret tidsplan 3](#_Toc132281319)

[Konklusion 3](#_Toc132281320)

[Diskussion 3](#_Toc132281321)

[(Referencer) 4](#_Toc132281322)

[(Bilag) 5](#_Toc132281323)

# Indledning

En gårdbutik er en butik, som typisk ligger på en gård, hvor man kan købe produkter, der er enten produceret på gården eller lokalt producerede varer.

I en gårdbutik vil man kunne finde et udvalg af varer, herunder både gårdenes egne produkter og andre indkøbte varer, som normalt er andre vare som er produceret lokalt i nærheden. Alle varer, som sælges i butikken, skal kunne registreres i systemet, og det kan nogle gange være nødvendigt at samle produkterne i en gavekurv eller sælge dem i kilopris i stedet for stykpris.

# Case beskrivelse

Som en lille gårdbutik vil der være mange forskellige priser og lager, der skal holdes styr på. Det kan være, om der er noget, der snart udløber, eller man skal kunne lave en faktura for en gavekurv med en masse forskellige ting i eller en bestilling af en samling af ting, hvor der skal findes ud af, hvad det koster for butikken at lave kurven, og hvad en god pris vil være for kurven. Derefter vil der også skulle holdes styr på, hvor meget salg og indtjening, man har haft.

## Problemformulering

Problemet for gårdbutikken er at holde styr på, hvad der er på lager/i butikken, og hvornår noget udløber, samt at kunne oprette faktura for større bestillinger og lave statistik over hvad der er blevet solgt over det sidste stykke tid.

# Projektplanlægning

## Estimeret tidsplan

Den estimerede tidsplan er meget fleksibel, da der forventes at blive skiftet meget imellem forskellige opgaver. Der vil sandsynligt ikke blive afsat en dag til at arbejde på backend eller en anden del af projektet, samtidig vil der nok også blive skrevet lidt på produktrapporten og processrapporten.

På grund af denne tilgang vil det være udfordrende at opdele arbejdet i individuelle dage for hver enkelt opgave. Tidsplanen vil i stedet blive brugt som en deadline for, hvornår det forventes, at alt er 100% færdigt. (Bilag)

## Arbejdsfordeling

I afsnittet "estimerede tidsplan" blev det beskrevet, at arbejdsfordelingen er meget fleksibel. Dette skyldes, at der sjældent vil være en dag, hvor der kun arbejdes på en opgave. I stedet vil man vælge at fokusere på den opgave, der mangler mest i forhold til hvornår deadline, der er blevet fastsat i tidsplanen. (Bilag)

# Metode- og teknologivalg

## Applikation

Denne applikation er en klient/server-webløsning, hvor frontenden er bygget med Blazor og backenden er bygget med ASP.NET Core. Formålet med applikationen er at håndtere og analysere data fra en database ved hjælp af en single-page-applikation (SPA). Brugerne vil kunne interagere med applikationen gennem et responsivt og brugervenligt hjemmeside udviklet med Blazor. Backenden vil fungere som et API, der giver adgang til databasen og muliggør oprettelse, læsning, opdatering og sletning af data. Samt generering af dokumenter ud fra en templates lavet a brugen med word feild codes.

### Frontend

frontenden er skrevet i Blazor, betyder det, at webapplikationen er udviklet ved hjælp af Blazor's Razor-komponenter og C#-kode, og at det kan køre direkte i browseren uden behov for ekstra JavaScript-biblioteker eller -frameworks.

Blazor er blevet valgt som grundlag for at skabe en Single Page Application (SPA), hovedsageligt på grund af dets brugervenlige sprog. Dette gør det muligt at holde det meste af koden inden for ét programmeringssprog, nemlig C#. Ved at benytte Blazor kan man dermed udvikle en mere strømlinet og effektiv applikation, som både er nem at arbejde med og vedligeholde.

### Backend

Backenden er skrevet i ASP.NET Core, som er et open-sourceweb-applikations-framework, der er designet til at udvikle moderne, cloud-baserede applikationer på tværs af platforme. Det giver en række funktioner, der gør det nemt at opbygge en skalerbare og sikker web-applikation, herunder indbygget støtte til dependency injection og en fleksibel middleware pipeline.

ASP.NET Core er blevet valgt som backend-teknologi for at kunne oprette et API, der holder sig til det samme programmeringssprog, C#. Dette gør det også nemt at modularisere og udvide funktionaliteten ved hjælp af NuGet-pakker. For API'et er der anvendt et Model-View-Controller (MVC) designmønster, som sikrer en klar adskillelse af ansvar og gør det nemt at vedligeholde og skalere applikationen. På backend-siden er Entity Framework Core også valgt som en Object-Relational Mapping (ORM) løsning, hvilket muliggør nem styring af databasen ved hjælp af C#. Det forenkler integrationen mellem kode og database, og sikrer en mere ensartet og effektiv udviklingsproces.

## Server

Applikationen vil blive hostet gennem Docker på en Linux-server, hvilket sikrer en ensartet og problemfri drift af både ASP.NET Core-webserveren og PostgreSQL-databaseserveren. Ved at anvende Docker-containere opnås en højere grad af fleksibilitet og skalérbarhed, samtidig med at man undgår eventuelle kompatibilitetsproblemer mellem forskellige miljøer. Denne opsætning er valgt for at sikre en effektiv, pålidelig og vedligeholdelsesvenlig infrastruktur for webapplikationen og databasen.

### Database

### Web server

## Asp.net core

## Entitet framework

Entity Framework Core (EF Core) er en letvægts, ekstensibel og open-source ORM (Object Relational Mapper) fra Microsoft, der er brugt i denne applikation for at forenkle dataadgang og manipulation. EF Core gør det muligt at arbejde med databaser ved hjælp af .NET objekter og

LINQ-forespørgsler i stedet for at skrive SQL-kode direkte. Det hjælper med at reducere kompleksiteten af databasetilslutning, samtidig med at det sikrer en mere vedligeholdelsesvenlig og testbar kodebase. EF Core understøtter en række forskellige databaser, hvilket gør det til et fleksibelt og alsidigt valg for udvikling af moderne webapplikationer.

## Blazor

Blazor er et frontend webudviklingsværktøj, der gør det muligt at bygge interaktive webapplikationer ved hjælp af C#-programmeringssprog og .NET-platformen. Det er en open source-teknologi, der giver udviklere mulighed for at skrive kode, der kører direkte i browseren, og som kan kommunikere med en backend API.

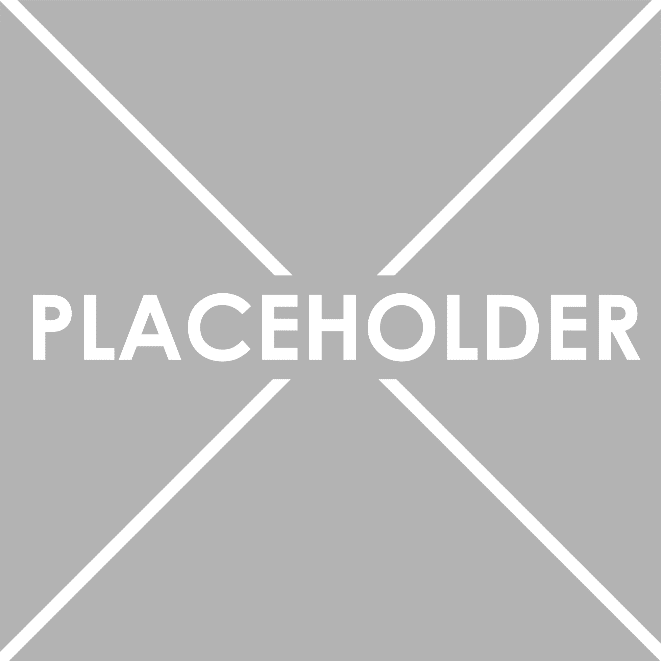
Blazor giver mulighed for at udvikle single-page applikationer (SPA) ved hjælp af Razor-komponenter, der kombinerer HTML og C#-kode på en effektiv måde. Det er også muligt at bruge Blazor til at opbygge traditionelle flersidede webapplikationer.

## Docker

Docker er blevet valgt som en central del af infrastrukturen for at kunne opnå en nem og problemfri implementering af serveropsætningen i forbindelse med projektet. Ved at bruge Docker-containere kan man garantere en ensartet og pålidelig drift af applikationen på tværs af forskellige miljøer, hvilket gør det lettere at skalere og vedligeholde systemet. Dette valg sikrer en hurtig og effektiv implementeringsproces, der minimerer både udviklings- og driftsomkostningerne.

## PostgreSQL

PostgreSQL er valgt som databaseløsning for at kunne tilbyde en gratis og omkostningseffektiv løsning for brugeren uden at gå på kompromis med ydeevne og pålidelighed. PostgreSQL er et open source objekt-relationel databasesystem, der fungerer godt sammen med ASP.NET Core og tilbyder en stærk og fleksibel databaseløsning, der er i stand til at understøtte en lang række applikationer og behov. Dette valg sikrer en solid og omkostningseffektiv infrastruktur, der kan tilpasses efter projektets krav og vækst.behov. Dette valg sikrer en solid og omkostningseffektiv infrastruktur, der kan tilpasses efter projektets krav og vækst.



Figur 1 PLaceholder billede

# Væsentlige elementer fra produktrapporten

Eks. henvisning til afsnit og punkter i produktrapporten

Find relevante overskrifter baseret på indhold

# Realiseret tidsplan

# Konklusion

Skal hænge sammen med case beskrivelsen og problemformuleringen.

## Diskussion

### Tilføjelser

Som tilføjelse vil man kunne lave en side der kan bruges af kunder til at scanne vare tilføje for en samled prise samt automatisk at åbne mobile pay med mængd

# (Referencer)

ASP.NET Core blazor: <https://learn.microsoft.com/da-dk/aspnet/core/blazor/?WT.mc_id=dotnet-35129-website&view=aspnetcore-7.0>

ASP.NET Core blazor historie: <https://en.wikipedia.org/wiki/Blazor>

ASP.NET Core https://learn.microsoft.com/en-us/aspnet/core/introduction-to-aspnet-core?view=aspnetcore-7.0

PostgreSQL https://www.postgresql.org/about/licence/

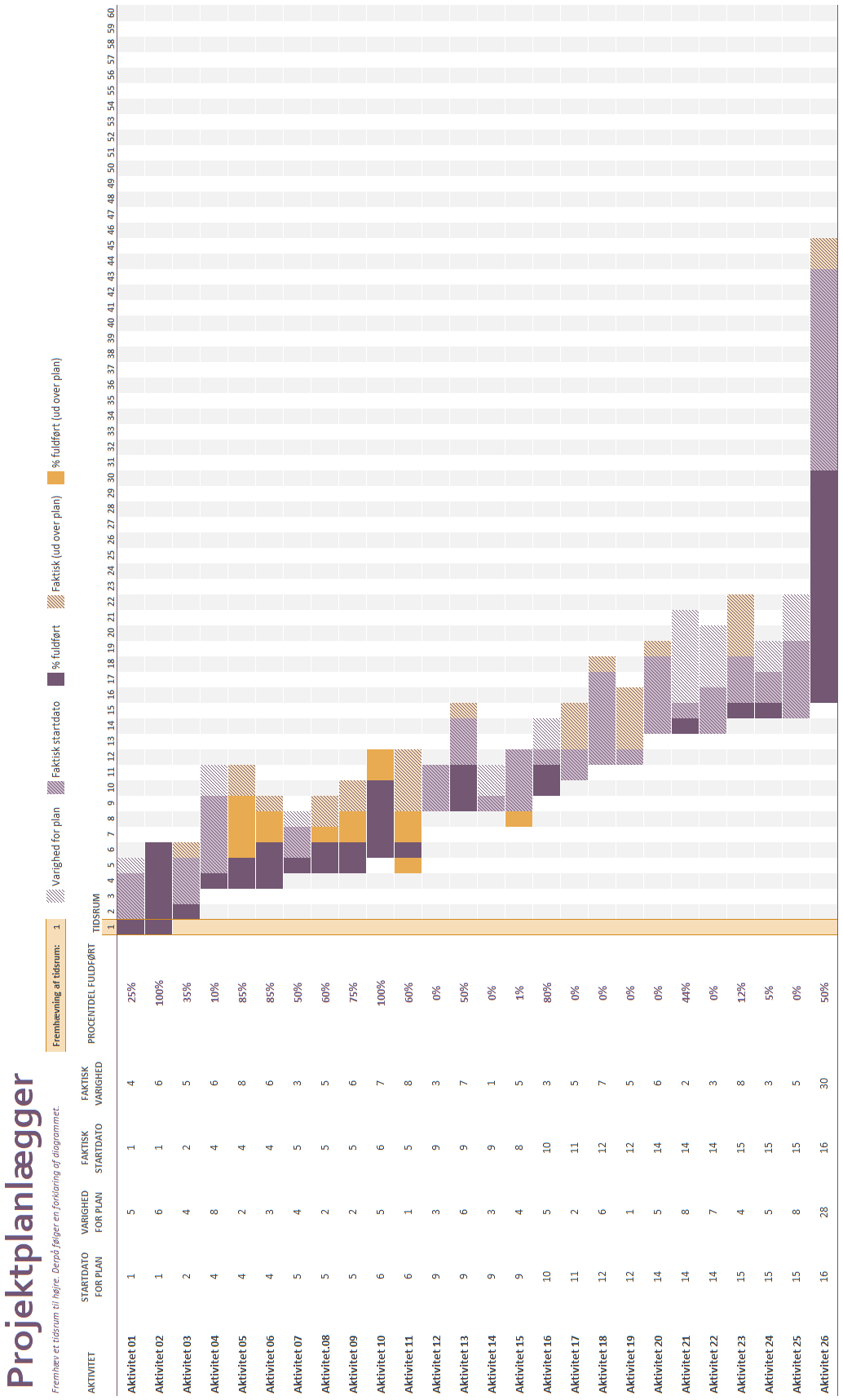
# (Bilag)

[Figur 1 PLaceholder billede 1](file:///C:\Users\emilo\Downloads\Processraport%20template%20(1).docx#_Toc80946617)

[Figur 2 Biag A: Estimeret Tidsplan 5](file:///C:\Users\lkri\Documents\Undervisningsmateriale\H6\Rapport%20materiale\Procesraport%20template.docx#_Toc80946618)

#### Estimeret Tidsplan

Figur 2 Biag A: Estimeret Tidsplan



#### Dagbog/Logbog

Den skal føres fra dag 1 i forløbet.