**PRODUKTRAPPORT**

Sidernes Verden – Webshop til salg af eventyrlige bøger



Michael Aggerholm | Svendeprøve | 9. april - 14. maj 2024

Datatekniker m. speciale i programmering

Indhold

[Indledning 2](#_Toc164241831)

[Læsevejledning 2](#_Toc164241832)

[Case 3](#_Toc164241833)

[Problemformulering 3](#_Toc164241834)

[Arkitektur 4](#_Toc164241835)

[Teknisk produktdokumentation 5](#_Toc164241836)

[Frontend 5](#_Toc164241837)

[Laravel Blade 5](#_Toc164241838)

[Bootstrap 5](#_Toc164241839)

[Backend 5](#_Toc164241840)

[Database 6](#_Toc164241841)

[Server 7](#_Toc164241842)

[Versionsstyring 7](#_Toc164241843)

[Kravspecifikation 8](#_Toc164241844)

[Frontend 8](#_Toc164241845)

[Database 9](#_Toc164241846)

[Backend 10](#_Toc164241847)

[Testspecifikation 12](#_Toc164241848)

[Frontend 12](#_Toc164241849)

[Database 13](#_Toc164241850)

[Backend 14](#_Toc164241851)

[Sikkerhed 17](#_Toc164241852)

[Test 18](#_Toc164241853)

[Laravel Applikationstest Workflow 18](#_Toc164241854)

[Tredjeparts biblioteker sikkerhedstest 18](#_Toc164241855)

[Larastan workflow 18](#_Toc164241856)

[Installationsvejledning 19](#_Toc164241857)

[Brugervejledning 20](#_Toc164241858)

[Konklusion 21](#_Toc164241859)

[Referencer 22](#_Toc164241860)

[Bilag 23](#_Toc164241861)

# Indledning

SidernesVerden er udarbejdet som svendeprøve projekt under uddannelsen som datateknikker med speciale i programmering på Tech College i Aalborg.

I rapporten beskrives udviklingsprocessen fra projektstart til endelig aflevering samt hvilke teknologier og metoder der er valgt.  
Der vil være dokumenteret hvilke udfordringer som er opstået undervejs, hvordan de er løst, samt hvilke alternativer der eventuelt ville kunne være valgt.

Formålet med rapporten er at give læseren klart indblik i den udviklingsproces og de valg som er foretaget under udviklingen af SidernesVerden.

Under læsningen kan det være en fordel at have adgang til projektet, det findes frit tilgængeligt via GitHub url:  
<https://github.com/MichaelAggerholm/Apprentice_Test>

# Læsevejledning

Denne del er projektets produktrapport, den bør læses efter procesrapporten da den indeholder beskrivelsen af det endelige produkt.

Produktrapporten indeholder den tekniske produktdokumentation, beskrivelsen af funktioner, specifikationer og tekniske detaljer samt installationsvejledning og brugervejledning.

# Case

Firmaet 'SidernesVerden' ønsker en webshop til salg af bøger om eventyrlige fortællinger.

Webshoppen skal give en hurtig og intuitiv oplevelse for kunderne. Det er vigtigt, at kunderne nemt kan finde de ønskede bøger og at grænsefladen er brugervenlig.

Ejeren og medarbejderne hos 'SidernesVerden' skal have mulighed for at få et fuldt overblik over alle aspekter af webshoppen, herunder bøger, forfattere, udgivere, salg og brugere. Desuden skal medarbejderne have adgang til en log, hvor de kan se, hvilke ændringer der er foretaget på shoppen, og af hvem.

Det er også vigtigt for virksomheden at have mulighed for at importere nye bøger fra Excel-filer, både før lanceringen af webshoppen og på sigt, hvis der skulle komme aftaler med nye leverandører.

# Problemformulering

Er det muligt at udvikle en webshop, hvor forbrugere kan registrere sig og foretage køb af bøger, samtidig med at medarbejdere kan logge ind, administrere og have overblik over webshoppen?

# Arkitektur

Produktet er struktureret med tre hovedkomponenter, som hver især opfylder specifikke funktioner inden for systemets struktur. En frontend, hvor brugerne kan interagere med systemet, en backend for administration, og en database til opbevaring af data.

**Frontend:**

Frontend-komponenten udgør det brugerorienterede interface, som muliggør interaktion med systemet gennem et web interface. Denne del er designet til at give brugeren en intuitiv og responsiv brugeroplevelse på diverse enheder og skærmstørrelser.

**Backend:**

Backend-komponenten fungerer som systemets administrative kerne, hvor det er muligt for administratorer at håndtere og overvåge systemets forskellige funktioner.  
Denne del af systemet tilbyder administrering af alle webshoppens entiteter.

**Database:**

Databasen er ansvarlig for lagring af alle data og giver en struktureret og effektiv håndtering af webshoppens data som er essentielle for systemets drift.

**Modulær struktur:**

Arkitekturens modulære design betyder, at de tre hovedkomponenter er adskilte og kan opdateres, testes og vedligeholdes uafhængigt af hinanden. Dette giver en høj grad af fleksibilitet og skalerbarhed for webshoppen, hvilket gør det muligt at tilpasse og forbedre eller udskifte hver del uden at påvirke resten af systemets elementer.

# Teknisk produktdokumentation

Beskrivelse af den tekniske opbygning og miljøet for de tre udviklede komponenter.

Frontend

Webshoppens frontend er bygget i Laravel Blade med Bootstrap, hvilket danner et solidt grundlag for en web frontend, kombinationen giver en effektiv udviklingsproces som prioriterer genbrug af kode, høj modularitet og et responsivt design.

### Laravel Blade

Blade er et templatingværktøj som integreres sømløst med Laravel-frameworket. Det giver mulighed for at opdele frontend kode i genanvendelige komponenter, samt tilbyder brugen af PHP-logik direkte i HTML skabeloner, eksempelvis løkker, betingelser, variabler.

### Bootstrap

Bootstrap er et CSS-framework som tilbyder en stor samling af foruddefinerede komponenter og layouts. Bootstrap komponenter understøtter responsiv visning på tværs af forskellige enhedsstørrelser.

## Backend

Valget af Laravel som backend framework er baseret på en række tekniske overvejelser, blandt andet er hele kombinationen af Laravel / Blade / SQLite en meget stærk og elegant sammensætning som giver et hurtigt og let arbejdsmiljø, med et stort antal foruddefinerede funktionaliteter.

Laravel benytter MVC arkitektur (Model-View-Controller), som giver en struktureret tilgang til at organisere og separere logikken i backenden, det bidrager til at skabe en mere skalerbar og testbar kodebase.

Eloquent ORM kommer inkluderet i Laravel og tilbyder en let måde at interagere med databasen på, ved hjælp af PHP-objekter.

Migrations er endnu et stærkt element i Laravel som giver mulighed for at versionsstyre og administrere databaseændringer, der kan blandt andet rulles tilbage til forrige database migreringer hvis der skulle opstå fejl eller mangler i nyere migrationer.

Laravel kommer med indbygget sikkerhed som CSRF-beskyttelse, inputvalidering og XSS (Cross-site scripting) beskyttelse som reducerer risikoen for sikkerhedsbrud.

Database

Webshoppens backend er konfigureret til at benytte en SQLite database, SQLite er et letvægtsdatabasesystem hvilket er ideelt til brug for mindre til mellemstore webapplikationer som denne.

I Laravel genereres tabeller ved hjælp af migrations, hvilket giver en struktureret tilgang til håndtering af databaseændringer, databasen defineres i koden og kan opdateres til databasen ved hjælp af Artisan-kommandoer.  
Artisan kommandoer er Laravel’s indbyggede kommando sprog som letter arbejdet for eksempelvis at slette en hel database og køre den frisk ind igen, eller opdatere en eksisterende database.

SQLite er en standard understøttelse i Laravel hvilket gør det let at interagere med tabeller og data som PHP-objekter gennem Laravel's Eloquent ORM.

For at sikre kvaliteten af koden kan SQLite også bruges til at køre tests i et isoleret miljø.

SQLite og Laravel som en kombination giver en strømlinet og effektiv tilgang til databaseadministration via Laravel’s backend ved brug af code-first princippet.

****

# Server

Webshoppens servermiljø er sat op ved hjælp af Laravel Sail på en Ubuntu-server, som kører lokalt via WSL2 (Windows Subsystem for Linux 2). Dette miljø udnytter Docker til at skabe en ensartet udviklingsplatform, hvor alle nødvendige komponenter er containeriserede.

Laravel Sail er optimeret til Laravel-applikationer og automatiserer opsætningen af Docker-miljøet. Under opsætningen kan konfigurationer tilpasses, herunder valget af web-server og database.

Sail bruger PHP til at køre applikationen, PHP er et serverside scripting sprog, her har man under opsætningen selv mulighed for at vælge hvilken version af PHP der skal tages i brug, ellers bruger opsætningen som standard den seneste stabile (LTS) version.

Nginx bliver benyttet som web-server samt reverse proxy, det vil sige at Nginx står for at levere indhold til klienter, men også at håndtere anmodninger fra klienter til applikationen.

Xdebug et værktøj til fejlfinding i PHP-kode som er integreret i servermiljøet for at lette udviklingsprocessen.

# Versionsstyring

Projektet anvender Git som versionsstyringsteknologi, valget er grundet Gits pålidelighed og effektive måde at administrere og opretholde en velorganiseret og pålidelig kodehistorik.

Der anvendes GitHub som platform til visuelt at danne overblik over projektet samt at vedligeholde diverse pipelines, også kaldet GitHub Actions / Workflows som projektet benytter til tests og at finde kendte sikkerhedsbrud i koden samt i inkluderede tredjeparts biblioteker.

# Kravspecifikation

Kravspecifikation opdelt i henholdsvis Frontend, Backend og Database.

## Frontend

Frontend delen af projektet skal sikre overskuelig og brugervenlig præsentation for sidens kunder, samt give medarbejdere adgang til et struktureret administratorpanel via login.

|  |  |  |  |
| --- | --- | --- | --- |
| ID | Krav | Prioritet | Type |
| F1 | Funktionalitet til at søge efter bøger | Høj | Funktionelt |
| F2 | Købsfunktionalitet | Høj | Funktionelt |
| F3 | Brugerregistrering og login | Høj | Funktionel |
| F4 | Adminpanel til webshopstyring | Høj | Funktionel |
| F5 | Produktvisning | Høj | Funktionel |
| F6 | Betalingsgateway | Høj | Funktionel |
| F7 | Responsiv design | Høj | Ikke-funktionel |
| F8 | Sikkerhed | Høj | Ikke-funktionel |

## Database

Databasekravspecifikationen skal sikre en struktureret og effektiv lagring af data for webshoppen.

|  |  |  |  |
| --- | --- | --- | --- |
| ID | Krav | Prioritet | Type |
| D1 | Brugertabel | Høj | Funktionelt |
| D2 | Udgivertabel | Høj | Funktionelt |
| D3 | Formattabel | Høj | Funktionel |
| D4 | Tilstandstabel | Høj | Funktionel |
| D5 | Genretabel | Høj | Funktionel |
| D6 | Sprogtabel | Høj | Funktionel |
| D7 | Forfattertabel | Høj | Funktionel |
| D8 | bogtabel | Høj | Funktionel |
| D9 | Bog-Genre relations tabel | Høj | Funktionel |
| D10 | Bog-forfatter relations tabel | Høj | Funktionel |
| D11 | Aktivitetslog tabel | Høj | Funktionel |
| D12 | Ordretabel | Høj | Funktionel |

## Backend

Backend-delen af projektet skal sikre en robust og sikker håndtering af data, forretningslogik og administration af webshoppen.

|  |  |  |  |
| --- | --- | --- | --- |
| ID | Krav | Prioritet | Type |
| B1 | Fejlhåndtering og logging | Høj | Funktionelt |
| B2 | Autentificering og autorisation | Høj | Funktionelt |
| B3 | API ‘er til frontend | Høj | Funktionel |
| B4 | Administrationspanel | Høj | Funktionel |
| B5 | Ordrehåndtering | Høj | Funktionel |
| B6 | Brugerhåndtering | Høj | Funktionel |
| B7 | Udgiverhåndtering | Høj | Funktionel |
| B8 | Formathåndtering | Høj | Funktionel |
| B9 | Tilstandshåndtering | Høj | Funktionel |
| B10 | Genrehåndtering | Høj | Funktionel |
| B11 | Sproghåndtering | Høj | Funktionel |
| B12 | Forfatterhåndtering | Høj | Funktionel |
| ID | Krav | Prioritet | Type |
| B13 | Boghåndtering | Høj | Funktionelt |
| B14 | Administratorhåndtering | Høj | Funktionelt |
| B15 | Fejlhåndtering | Høj | Funktionel |
| B16 | Hændelseslogging | Høj | Funktionel |
| B17 | Søgefunktionalitet | Høj | Funktionel |
| B18 | Sikkerhed | Høj | Funktionel |
| B19 | Gendannelse af slettede objekter | Middel | Funktionel |

# Testspecifikation

Testspecifikation opdelt i henholdsvis Frontend, Backend og Database.

## Frontend

Oversigt over test krav til frontend.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ID | Acceptkrav | Kritisk | Godkendt | Afvist |
| F1 | Brugere kan søge efter bøger ved hjælp af forskellige søgekriterier som titel, forfatter, genre samt at søgeresultaterne er nøjagtige og relevante for søgningen | Ja | Ja |  |
| F2 | Brugere kan tilføje varer til deres indkøbskurv og gennemføre købet korrekt samt betalingsprocessen fungerer problemfrit og sikkert | Ja | Ja |  |
| F3 | Brugere kan registrere sig med succes og logge ind på deres konto | Ja | Ja |  |
| F4 | Administratorer kan logge ind på Adminpanel og har adgang til nødvendige funktioner til styring af bøger, brugere, ordrer osv. Og at ændringer foretaget i Adminpanel afspejles korrekt i webshoppen | Ja | Ja |  |
| F5 | Produkter vises korrekt på webshoppen med alle relevante oplysninger såsom titel, forfatter, pris, beskrivelse osv. | Ja | Ja |  |
| F6 | Kunder kan gennemføre betalinger uden problemer og betalingen kan findes i Stripe | Ja | Ja |  |
| F7 | Webshoppen reagerer korrekt på forskellige enheder, herunder desktopcomputere, bærbare computere, tablets og smartphones | Ja |  | Ja |
| F8 | Der anvendes HTTPS-protokollen for at kryptere kommunikationen mellem webbrowseren og web-serveren | Ja | Nej | Ja |

## Database

Database testkrav for at verificere at databasen understøtter webshoppen og opfylder de krav som er nødvendige for at kunne levere data.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ID | Acceptkrav | Kritisk | Godkendt | Afvist |
| D1 | brugertabelen indeholder alle nødvendige felter til lagring af brugeroplysninger | Ja | Ja |  |
| D2 | Indeholder alle nødvendige felter til lagring af udgivere | Ja | Ja |  |
| D3 | Indeholder alle nødvendige felter til lagring af bog formater | Ja | Ja |  |
| D4 | Indeholder alle nødvendige felter til lagring af bog tilstande | Ja | Ja |  |
| D5 | Indeholder alle nødvendige felter til lagring af genre | Ja | Ja |  |
| D6 | Indeholder alle nødvendige felter til lagring af sprog | Ja | Ja |  |
| D7 | Indeholder alle nødvendige felter til lagring af forfattere | Ja | Ja |  |
| D8 | Indeholder alle nødvendige felter til lagring af bøger | Ja | Ja |  |
| D9 | Indeholder bog id og genre id for at lave en bog-genre relation | Ja | Ja |  |
| D10 | Indeholder bog id og forfatter id for at lave en bog-forfatter relation | Ja | Ja |  |
| D11 | Indeholder alle nødvendige felter til lagring af administrator aktiviteter | Ja | Ja |  |
| D12 | Indeholder alle nødvendige felter til lagring af ordreoplysninger | Ja | Ja |  |

## Backend

Backend testspecifikation for at sikre at webshoppen fungerer korrekt og opfylder specifikke krav og forventninger.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ID | Acceptkrav | Kritisk | Godkendt | Afvist |
| B1 | backend håndterer alle typer fejl og undtagelser korrekt samt logger hændelsesaktivitet til databasen. | Ja | Ja |  |
| B2 | brugere kan logge ind ved brug af brugernavn/adgangskode og systemet kan skelne mellem normal bruger og administrator brugere | Ja | Ja |  |
| B3 | API 'er til frontend er korrekt implementeret og leverer data som vises på websitet samt håndtere fejl korrekt og returnere passende fejl/succes beskeder | Ja | Ja |  |
| B4 | Administrationspanelet indeholder alle nødvendige funktioner og værktøjer til at administrere bøger, kunder, ordrer osv. Samt se hændelseslog for admin aktivitet | Ja | Ja |  |
| B5 | Applikationen kan modtage, behandle og opdatere ordrer korrekt | Ja | Ja |  |
| B6 | Applikationen giver mulighed for oprettelse, redigering og sletning af brugere med tilhørende informationer | Ja | Ja |  |
| B7 | Applikationen giver mulighed for oprettelse, redigering og sletning af udgivere med tilhørende informationer | Ja | Ja |  |
| B8 | Applikationen giver mulighed for oprettelse, redigering og sletning af formater med tilhørende informationer | Ja | Ja |  |
| B9 | Applikationen giver mulighed for oprettelse, redigering og sletning af tilstande med tilhørende informationer | Ja | Ja |  |
| B10 | Applikationen giver mulighed for oprettelse, redigering og sletning af genre med tilhørende informationer | Ja | Ja |  |
| B11 | Applikationen giver mulighed for oprettelse, redigering og sletning af sprog med tilhørende informationer | Ja | Ja |  |
| B12 | Applikationen giver mulighed for oprettelse, redigering og sletning af forfattere med tilhørende informationer | Ja | Ja |  |
| B13 | Applikationen giver mulighed for oprettelse, redigering og sletning af bøger med tilhørende informationer | Ja | Ja |  |
| B14 | Mulighed for at give brugere administrator adgang samt at fratage admin privileger igen | Ja | Ja |  |
| B15 | Backend håndterer fejl og sender relevante fejlbeskeder videre til frontend hvis muligt | Ja | Ja |  |
| B16 | Alle admin aktiviteter logges til hændelsesloggen og vises korrekt i admin panelet med brugernavn samt hvilken aktivitet der er foretaget | Ja | Ja |  |
| B17 | Korrekt håndtering af brugerens søgeforespørgsler og returnering af relevante resultater baseret på søgekriterier som titel, forfatter, kategori | Ja | Ja |  |
| B18 | Der er begrænsning af adgang til beskyttede ressourcer og funktioner baseret på brugerens rolle samt inputvalidering for at forhindre SQL-injektioner, XSS og lignende former for angreb, alle forms er beskyttet med CSRF beskyttelse | Ja | Ja |  |
| B19 | Efter sletning af ethvert objekt som forfatter, sprog eller udgiver, skal det slettede objekt kunne gendannes fra administratorpanel. | Ja | Ja |  |

# Sikkerhed

Sikkerhed udgør en fundamental del af enhver udviklingsproces, især når der er involveret brugerlogin med opbevaring af følsomme oplysninger. Det åbner op for en lang række muligheder for hackere at udnytte, og derfor er det af afgørende betydning at implementere effektive sikkerhedsforanstaltninger for at beskytte mod potentielle trusler. Webshoppen tager dette ansvar alvorligt og benytter en række værktøjer og teknologier til at opretholde en høj sikkerhed.

En central del af sikkerheden er adgangskontrol, som administreres gennem brug af forskellige roller og tilladelser. Dette sikrer, at hver bruger kun har adgang til de funktioner og ressourcer, der er relevante for deres rolle i systemet. Samtidig anvendes CSRF-tokens til at forhindre Cross-Site Request Forgery (CSRF), som er et angreb hvor uautoriserede handlinger kan udføres af en bruger, der er blevet narret til at udføre en handling på en anden hjemmeside, eksempelvis en falsk hjemmeside eller en forklædt url.

Passwords bliver sikret med bcrypt algoritmen, som tilføjer en unik 'salt' til hver adgangskode for at øge tilfældigheden og sikkerheden. Dette sikrer, at selv om to brugere har samme adgangskode, vil deres hashes - eller digitale signaturer - være forskellige. Denne metode hjælper med at beskytte passwords mod hacking.

Laravel’s indbyggede Middlewares giver mulighed for at filtrere og håndtere indgående anmodninger, hvor brugeren sendes hen ift. Rolle eller ip-adresse, eksempelvis hvis webshoppens administration ikke skal være tilgængelig via udenlandsk ip-adresse.  
På den måde kan der skræddersyes sikkerhedsregler alt efter specifikke krav og behov.

Et vigtigt sikkerhedsaspekt er beskyttelse mod SQL-injektioner, en almindelig form for angreb hvor hackere indsætter SQL kode igennem inputfelter, dette forhindres gennem en kombination af Laravel’s indbyggede SQL-Injektionsbeskyttelse og inputvalidering samt rense data (også kaldet at data-sanitizing) i Controllers inden data bruges til eksempelvis at oprette eller opdatere data på siden.

Ved go-live af webshoppen vil der skulle benyttes HTTPS (Hypertext Transfer Protocol Secure) som er en afgørende faktor for at overholde høj sikkerhedsstandard. HTTPS er en krypteret protokol som sikrer at kommunikationen mellem brugerens browser og web-serveren er krypteret og dermed beskyttet mod potentielle angreb eller aflytninger af data under kommunikation mellem browser og web-serveren.  
Ved at bruge HTTPS sikres følsomme oplysninger, såsom adgangskoder og betalingsoplysninger. HTTPS giver også brugerne tillid til, at webshoppen er autentisk og legitim, da det er forbundet med et SSL-certifikat, der bekræfter webstedets identitet, hvilket også bidrager til en god brugeroplevelse.

Projektet benytter en environment fil (.env), som gavner projektet af flere sikkerhedsmæssige årsager, her isoleres konfigurationsnøgler ved at adskille oplysninger fra kodebasen som blandt andet API nøgler, MYSQL-adgang og Stripe nøglen.  
Det reducerer fejlrisikoen for utilsigtet eksponering af oplysninger, .env filen er eksempelvis heller ikke en del af versionsstyringen, men skal opsætte på ny ved hver installation.

# Test

Webshoppen er opsat med en testproces for at sikre en høj kvalitet, sikkerhed og forhindrer nedbrud. Der er implementeret tre GitHub Action-workflows for at automatisere testprocessen og sikre at kodebasen forbliver stabil og pålidelig efter enhver ændring.

(Se bilag: Workflow Runs, s. 213123)

## Laravel Applikationstest Workflow

Dette workflow tester Laravel applikationen ved at bygge den og verificere at den bygges korrekt.   
Først installeres alle tredjeparts biblioteker ved hjælp af composer og NPM, Derefter oprettes der forbindelse til en SQLite test database, efterfølgende migreres der til databasen, alle PHPunit tests køres. Såfremt alt kører igennem med succes, meldes der gennemført workflow tilbage.

## Tredjeparts biblioteker sikkerhedstest

Her fokuseres der på at teste tredjeparts biblioteker for kendte sikkerhedsbrud eller forældede versioner.  
Kørslen starter med at scanne alle tredjeparts biblioteker for kendte sikkerhedsbrud, efterfølgende kontrolleres det om der i projektet benyttes forældede versioner samt sikrer at de versioner som benyttes, er sikkerhedsmæssigt opdateret.

## Larastan workflow

Larastan er en udvidelse af Phpstan, tilpasset Laravel miljøet, det er et PHP analyseværktøj som benyttes til at opdage fejl og potentielle problemer i Laravel koden og PHP-koden (Se bilag: Larastan, s. 123).

Udvidelsen er tilpasset til at forstå Laravel specifikke elementer som Eloquent ORM ‘en, migrationer og mange andre indbyggede funktioner.

Dette workflow installerer alle tredjeparts biblioteker, validerer PHP-koden for eventuelle syntaksfejl, efterfølgende køres der en dybdegående analyse af kodebasen for at identificere potentielle problemer eller brug af indbyggede forældede PHP-funktionaliteter, der tjekkers også for kendte sikkerheds fejl eller fejllogikker i koden.  
Sidst meldes der tilbage med en detaljeret rapport over eventuelle fundne problemer, hvilket gør det let at forbedre kvaliteten og forhindre fejl i koden.

Larastan har ikke en officiel website men er open source og kan findes på <https://github.com/larastan/larastan> - Larastan er baseret på PHPStan som kan tilgås via <https://phpstan.org/>

# Installationsvejledning

For at begynde med skal du sikre dig, at dit system opfylder de nødvendige krav:

* **Windows Subsystem for Linux (WSL2)** på Windows maskiner
* **Docker Desktop**: Installeret og kørende
* **Git**: Installeret til versionskontrol
* **Composer**: Til PHP afhængighedshåndtering
* **Node.js og npm**: Til frontend afhængighedshåndtering

Herefter følges guide trin for trin for opsætning af projekt på lokal maskine:

1. Åbn en terminal (Brug WSL2 terminalen på Windows).
2. Naviger til den mappe, hvor du ønsker projektet skal være.
3. Kør git clone kommando for at klone projektet ned i ønsket mappe:

~ git clone <https://github.com/MichaelAggerholm/Apprentice_Test>

1. Skift til projektets rodmappe:

~ cd Apprentice\_Test

1. Installer PHP-afhængigheder:

~ composer install

1. Installer Nods.js afhængigheder med NPM:

~ npm install

1. Opsætning af Laravel, kopier miljø fil med cp kommando:~

~ cp .env.example .env

1. Generer Laravel applikationsnøgle:

~ php artisan key:generate

1. Start docker containers i ”detached mode” med sail kommando:

~ ./vendor/bin/sail up -d

1. Kør migrationer for at opsætte databasestrukturen:

~ .vendor/bin/sail artisan migrate

1. Seed databasen med test data:

~ .vendor/bin/sail artisan db:seed

1. kør frontend miljø for at kunne tilgå webshoppen med følgende npm kommando:

~ npm run build

1. Tilgå nu webshoppen via computerens localhost adresse på følgende url:

<http://localhost>

# Brugervejledning

I dette afsnit beskrives hvordan websitet benyttes og hvordan der navigeres i henholdsvis frontend for forbrugeren samt i administrationspanelet for medarbejdere.

## Forbrugermanual

## Administrator manual

# Konklusion

Webshoppen er nu fuldt implementeret og fungerer som en integreret helhed bestående af en selvstændig backend, frontend og database. Dette design sikrer høj modularitet og skalerbarhed, hvilket betyder, at individuelle dele af projektet kan udvides eller tilpasses uden at påvirke de andre komponenter.

Administrationen tillader vedligeholdelse af alle entiteter. Det er muligt at søge efter bøger baseret på søgeord, genrer eller tilstande og at filtrere resultater på bogsiden. Brugere kan registrere sig, logge ind og gennemføre køb, mens administratorer kan tilgå et dedikeret Adminpanel, Her kan de overvåge Admin hændelser, se og tilpasse ordrestatusser. Admin kan desuden tilgå betalingsplatformen Stripe for detaljeret overvågning af transaktioner.

Til trods for disse avancerede funktioner er der visse mangler. Webshoppen er ikke fuldt responsiv på mobile enheder, og den understøtter ikke Cascade deleting med soft deletes. Disse funktioner er blevet nedprioriteret for at prioritere yderligere funktionalitet.

På den positive side er systemet designet med høj fejltolerance. Administratorer har mulighed for at slette og gendanne entiteter, hvilket forbedrer systemets robusthed. På frontend vises brugte bøger med et tydeligt skråbanner.

For at forbedre webshoppen yderligere og sikre en optimal brugeroplevelse er der nogle nøgleområder som bør forbedres i fremtiden, her er der blandt andet tale at implementere avanceret søgefunktionalitet i administrationen for at øje brugervenligheden og effektiviteten for administratorer, således de hurtigt kan lokalisere og redigere specifikke bøger. Ligeledes vil det gavne forbrugeren at få udvidet filtreringen af bøger til forfattere, udgivere, priser mm. På frontenden for at forstærke den samlede købsoplevelse for kunder.

Det nuværende design af webshoppen giver plads til forbedringer for at kunne tiltrække og fastholde kunder, der bør nok investeres i en professionel designer som kan levere et mere tiltalende og moderne layout.

Grundet projektets ridsramme har der heller ikke været fokuseret nok på den fordel som blade templating tilbyder med komponent baseret kodning og genanvendelse, her kunne en hel del af koden væres sparret ved at være defineret og kaldt som komponenter fremfor duplikeret kode.

# Bilag

Database diagram  
Kan også findes via GitHub url:  
<https://github.com/MichaelAggerholm/Apprentice_Test/blob/main/Assets/Documents/database_diagram.drawio.pdf>



Workflow runs  
Kan også findes via GitHub url:  
<https://github.com/MichaelAggerholm/Apprentice_Test/actions>

