**PROCESRAPPORT**

Sidernes Verden – Webshop til salg af eventyrlige bøger



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

Datatekniker m. speciale i programmering

Indhold

[Indledning 2](#_Toc164579637)

[Læsevejledning 2](#_Toc164579638)

[Forord 3](#_Toc164579639)

[Case 4](#_Toc164579640)

[Problemformulering 4](#_Toc164579641)

[Projektplanlægning 5](#_Toc164579642)

[Metodevalg og teknologi 6](#_Toc164579643)

[PHP 6](#_Toc164579644)

[GitHub 6](#_Toc164579645)

[Laravel 6](#_Toc164579646)

[Laravel's Blade Framework 7](#_Toc164579647)

[Bootstrap 7](#_Toc164579648)

[NPM 7](#_Toc164579649)

[Composer 7](#_Toc164579650)

[Laravel Sail 8](#_Toc164579651)

[Stripe 8](#_Toc164579652)

[Laravel Debugbar 8](#_Toc164579653)

[ActivityLog 8](#_Toc164579654)

[Design og implementering 9](#_Toc164579655)

[MVC designmønster 9](#_Toc164579656)

[Service Provider 9](#_Toc164579657)

[Middleware 9](#_Toc164579658)

[Observer Pattern 9](#_Toc164579659)

[Soft Deletes 10](#_Toc164579660)

[Factories og Seeders 10](#_Toc164579661)

[Konklusion 12](#_Toc164579662)

[Diskussion 13](#_Toc164579663)

[Referencer 14](#_Toc164579664)

[Bilag 15](#_Toc164579665)

[Logbog 22](#_Toc164579666)

# 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 procesrapport, den bør læses før produktrapporten da den indeholder beskrivelsen af processen der er gennemgået for at ramme det endelige produkt.

Procesrapporten indeholder information om projektets planlægning, design, udvikling, test og implementering. Procesrapporten giver også en detaljeret beskrivelse af de udfordringer der er stødt på undervejs, samt hvordan jeg har valgt at løse dem.

# Forord

Sidernes Verden – En webshop til salg af eventyrlige bøger, er et projektet der forener passionen for eventyrlige fortællinger med den moderne digitale handelsverden. Firmaet 'SidernesVerden' søger en løsning som lever op til kundernes forventninger, når de træder ind i den virtuelle webshop af bøger.

Målet for projektet er at udarbejde en webshop som er enkel og funktionel for kunder, samtidig med at der er overskuelighed og gennemtænkt funktionalitet for firmaets ansatte som skal vedligeholde webshoppen. Til dette formål er valget faldet på Laravel, et kraftfuldt og moderne PHP-framework, med blade frameworket som frontend og Laravel backend. Dette sikrer en robust og skalerbar platform, der kan imødekomme både nutidige og fremtidige behov.

Med anvendelsen af en SQLite-database opnås der hurtig og effektiv datalagring Dette er særligt relevant for et projekt i denne størrelsesorden, hvor ressourceoptimering er en central faktor.

der blive implementeret en detaljeret hændelseslog, som dokumenterer alle administrative interaktioner på webshoppen som gør det muligt at identificere, hvilke medarbejdere der har foretaget ændringer i forbindelse med bøger eller andre elementer, hvilket kan være afgørende for at identificere fejl og sikre en hurtig rettelse af fejlen, hvilket vil bidrage til højere kvalitet af webshoppen samt bedre brugeroplevelse for kunder samt ansatte.

Endvidere vil der blive fokuseret på implementeringen af en funktion til import af bøger via Excel-ark. Dette vil ikke blot lette processen med at opdatere sortimentet før lanceringen af webshoppen, men også sikre, at virksomheden kan udvide sit udvalg effektivt og i takt med kundernes efterspørgsel.

Gennem denne rapport bliver der dykket ned i de tekniske aspekter af projektet, samt undersøgt hvordan valget af Laravel, blade frameworket og valget af SQLite databasen kan bidrage til at opfylde fyldestgørende målsætning for webshoppen til 'SidernesVerden'.

# 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?

# Projektplanlægning

Under projektudviklingen bliver der benyttet Kanban til projektstyring, det er valgt da Kanban er en agil metode som er nem at forstå og anvende til mindre projekter som dette, det giver overblik over opgaver samt sikrer at man ikke overser lavere prioriterede opgaver. Der er valgt at benytte GitHubs Kanban board da projektet i forvejen ligger i GitHub, det giver også let adgang til at kunne uddelegere opgaver på tværs af udviklere på sigt. Igennem projektet bliver opgaver fulgt via Kanban boardet og alle nye opgaver får hver sin ”task” som bliver fulgt til ende. Projektstyringen er opstillet således at der afholdes stand-up møde ved Kanban boardet med status og opdatering til opgaver 1 gang om ugen. Der er taget billede undervejs af Kanban processen, de kan findes via bilag her:  
Bilag Kanban fase 1 s. 11  
Bilag Kanban fase 2 s. 12  
Bilag Kanban endelig fase s. 13

Inden udviklingsopstart er der truffet nogle valg for at komme frem til blandt andet det valgte programmeringssprog, frameworks og tredjeparts libraries som er valgt til projektet, dette er blandt andet valgt ud fra mange muligheder som vist via mindmap (se bilag side 213123), i mindmap er der skrevet op til flere frameworks, databaser og diverse muligheder for at skabe lige præcis det opnået udgangspunkt, efterfølgende har teamet nøje udvalgt elementer som arbejder godt sammen ift. Projektets størrelse og det ønskede resultat samt en del baseret på teamets erfaring og den mængde tid der er sat til projektet.

Herefter bliver hver opgave defineret og beskrevet i kravspecifikationen (se dokumenteret kravspecifikation i produktrapport s. 11111), i kravspecifikationen er basal funktionalitet slået sammen for at skabe overblik, det gør sig gældende blandt andet for håndtering af bøger, herunder CRUD-funktionalitet mm. herefter bliver hver opgave opstillet i testspecifikation (se dokumenteret test specifikation i produktrapport s. 2222).

Ved projektstart er der udarbejdet en estimeret tidsplan (se bilag estimeret tidsplan s. 123) som teamet baseret på tidligere erfaring forventer at processen vil forløbe efter, sideløbende udfyldes en realiseret tidsplan for at løbende at følge og dokumentere den faktiske fremdrift (Se bilag: realiseret tidsplan, s. 12313).

I begyndelsen af udviklingsprocessen er der udarbejdet et database-diagram for at sikre, at relationerne og strukturen giver et solidt fundament (se bilag: database diagram s. 21323). Dernæst et klassediagramfor at give et overblik over de controllers og klasser der skal oprettes i backend (se bilag: klassediagram, s. 213213123).

Efter fastlæggelse af database og klassestruktur går udviklingsfasen i gang, som beskrevet i Kanban-boardet og tidsplanen, opdelt i tre faser: Frontend, Database og Backend. Gennem hele projektforløbet bliver der ugentligt på stand-up lavet status og tilbageblik på opgaver, udfordringer og genovervejelser. Denne viden bliver brugt til refleksion og læring fra de forskellige situationer og udfordringer som er opstået undervejs.

Under hele projektforløbet bliver der ført en logbog, hvor daglige opgaver og eventuelle udfordringer bliver dokumenteret (se bilag: logbog, s. 213123).

# Metodevalg og teknologi

I forbindelse med projektet er der truffet beslutninger vedrørende valg af metoder og teknologier. Disse valg og overvejelserne bag dem vil blive uddybet i det følgende afsnit.

## PHP

PHP er blevet valgt til udviklingen af dette projekt af flere årsager. For det første har PHP en lang historie inden for webudvikling og har vist sig at være pålideligt og skalerbart til opbygning af komplekse webapplikationer. Derudover har PHP et stort udviklermiljø og et væld af dokumentation og ressourcer, hvilket gør det nemt at finde løsninger og få hjælp, når det er nødvendigt.

## GitHub

GitHub er en onlineplatform, der tilbyder en bred vifte af værktøjer til projektstyring, herunder lagring af projekter og versionsstyring. Dets anerkendte ry og høje kvalitet gør det til det ideelle valg for projektet. Et af de centrale værktøjer som vi benytter os af, er GitHub Actions, også kaldet ”Pipelines”, disse kører automatisk ved hver eneste ændring som bliver pushet op via Git, som sikrer en konstant kvalitetskontrol og fejlfinding i projektet.

Derudover leverer GitHub også et intuitivt Kanban board, som bliver anvendt til projektstyring. GitHub leverer alt i alt en omfattende pakke af funktioner og værktøjer som bidrager til en effektiv og struktureret gennemførelse af projektet.

## Laravel

Laravel er valgt som det primære framework til udviklingen af flere vigtige årsager.   
For det første tilbyder Laravel en omfattende og velstruktureret løsning til webudvikling.  
Dets arkitektur og indbyggede funktioner gør det muligt at opbygge applikationer på en effektiv måde.

Laravel tilbyder en bred vifte af indbyggede funktioner og værktøjer, herunder en kraftfuldt ORM (Object–Relational Mapping) kaldet Eloquent, migrations, og et omfattende sæt af kommandoer til at lette udviklingsprocessen og øge produktiviteten.

Laravel er et anvendt framework med en gennemarbejdet dokumentation som gør det nemt at finde hjælp når det er nødvendigt.

## Laravel's Blade Framework

Til frontend-delen af projektet er der valgt Laravel's Blade framework. Blade har en række stærke funktioner, herunder Blade's evne til at lette integrationen af PHP-kode, der sendes fra controlleren. Dette gør det muligt at opretholde en klar adskillelse mellem frontend og backend, samtidig med at udnytte PHP's styrke til dynamisk at generere indhold.

Blade har en stærk evne til at genanvende komponenter, hvilket betyder at der kan opbygges og vedligeholdes frontend-kode på en mere struktureret og effektiv måde. Dette resulterer i en mere skalerbar og vedligeholdelsesvenlig kodebase. Samlet set gør Blade frameworket det muligt at opnå en stilfuld og funktionel frontend som integreres sømløst med Laravel's backend, hvilket sikrer en sammenhængende og robust webapplikation.

## Bootstrap

Bootstrap er et moderne css framework som giver mulighed for effektivt at opbygge en responsiv brugergrænseflade. Med Bootstrap kommer der et bredt udvalg af færdiglavede komponenter, layouts og css klasser som kan spare tid og reducere behovet for meget nyudvikling af kode. Desuden tilbyder Bootstrap mange tilpasningsmuligheder kun ved hjælp af tilføjelser af klasser, som gør det simpelt at opstille design.

## NPM

Projektet benytter NPM (Node Package Manager) i Laravel for at håndtere pakker, eksempelvis installationen af Bootstrap. Ved hjælp af NPM kan der nemt installeres / afinstalleres og administreres forskellige frontend biblioteker, værktøjer og versioner.

En af de primære fordele ved at bruge NPM er muligheden for at se live kodeændringer direkte browseren under udviklingsfasen ved hjælp af ”npm run dev” kommandoen.

Når projektet er klar til at blive lanceret, benyttes kommandoen ”npm run build” i stedet, som genererer en optimeret version af frontend koden, eksempelvis ved at minify CSS og JavaScript, som reducerer filstørrelsen og forbedrer ydeevnen i produktionsmiljøet.

## Composer

Composer er et kraftfuldt værktøj til PHP, som gør det muligt at installere, opdatere og administrere tredjeparts biblioteker og frameworks.

Desuden gør Composer det muligt at udnytte et stort økosystem af open-source PHP-pakker, der spænder over alt fra frameworks til utilities og libraries som eksempelvis det logging library der er gjort nytte af i webshoppens administration (Link til logging library), eller den specielt integrerede debug bar som der er gjort nytte af under udviklingsfasen for effektivt at fejlsøge under udfordringer (link til debug bar library)

Ved hjælp af kommandoer som ”composer install” og ”composer update” holdes projektet opdateret og korrekt konfigureret, hvilket giver en mere stabil og pålidelig webapplikation.

## Laravel Sail

Sail er Laravel’s indbyggede Docker-miljø som gør det nemt at opsætte og køre projektet i et isoleret og konsistent udviklingsmiljø uanset hvilket system der arbejdes på, samt når projektet skal gå live på en web server.

Ved hjælp af Sail er det muligt at starte hele Laravel projektet op, med installation of composer pakker og lignende afhængigheder, alt sammen ved hjælp af én enkelt kommando og på ganske kort tid.

## Stripe

Stripe er valgt som betalingsløsning for webshoppen. Stripe tilbyder en brugervenlig platform og gør det nemt at integrere betalingsfunktionalitet i webshoppen.

Med Stripe kan kunder betale via kreditkort, dankort og debetkort. En af de mest tiltalende funktioner ved Stripe et meget brugervenlige onlinepanel, her kan man via en token logge ind og dermed have overblik over alle salg, betalinger og lignende som er foretaget fra webshoppen (Link til Stripe dokumentation / panel).  
(Se Stripe-PHP reference s.2131243)

## Laravel Debugbar

For at effektivisere fejlfinding under udviklings benyttes der Laravel Debugbar. Dette værktøj giver en dybere indsigt i applikationens SQL-kald, diverse views tilknytning til controllers, routes og meget mere, direkte i browseren.  
(Se Laravel-Debugbar reference s.2131243)

## ActivityLog

For at lette integrationen med at logge brugeraktivitet og vigtige hændelser på shoppen, anvendes der ActivityLog biblioteket. Dette giver mulighed for at logge diverse detaljer og hændelser i hele applikationen og sikrer at logning er skalerbart og vedvarende tilgængelig, også i tilfælde af at webshoppen udvides eller ændrer størrelse.  
(Se ActivityLog reference s.2131243)

## laravel-Excel

# Design og implementering

En kort beskrivelse af punktet her (noget med at der er implementeret MVC designmønster, Service Provider mønstret, middleware mønster og observer pattern).

## MVC designmønster

Laravel anvender Model-View-Controller (MVC) arkitekturen, et designmønster som øger separation og giver kodebasen høj modularitet, webshoppen er bygget omkring dette designmønster og er derfor påvirket både på design og implementering heraf.

**Model**

Modellerne håndterer al data logik og database interaktion, dette omfatter hentning, indsætning, opdatering og sletning af data i databasen. I projektet er modeller som ’Produkt’, ’Order’ og ’User’ lavet for at håndtere entiteter og relationer.

**View**

Views er ansvarlige for præsentation og indeholder den HTML som bliver sendt til klientens browser, baseret på den data som modtages fra controlleren. Ved hjælp af Laravel’s Blade templating engine bliver der leveret dynamisk brugergrænseflade og produktvisning til klienten.

**Controller**

Her bearbejdes indgående forespørgsler og udføres logikken som håndterer data inden den videresendes til views.

## Service Provider

Denne del er essentiel i Laravel for at binde forskellige komponenter af applikationen sammen, det er blandt andet for at registrere serviceklasser i Laravel’s service container som resulterer i en ren måde at implementere Dependency Injection på. Dependency Injection sørger automatisk for at give applikationen de ressourcer og tjenester det har brug for, for at fungere effektivt.

## Middleware

I Laravel giver Middlewares mulighed for at håndtere handlinger i Request Lifecycle. Projektet anvender middleware til at håndtere autentifikation og sikre at kun autentificerede brugere kan tilgå bestemte sektioner af webshoppen, som eksempelvis administrationspanelet.

## Observer Pattern

Dette benyttes via event og listener klasser for at håndtere handlinger som reagerer på visse hændelser, såsom at logge en hændelse når en administrator logger ind eller ændrer en bog fra administrationen.

## Soft Deletes

Der er implementeret en funktion kendt som ’Soft Deletes’ gennem Laravel’s Eloquent ORM. Dette gør at vi kan bevare data i databasen, selv efter at den er slettet af en administrator. Faktisk bliver denne data ikke fysisk fjernet fra databasen, i stedet markeres dataen med tidspunkt for sletning, hvilket gør det muligt at gendanne data på et senere tidspunkt, dette er også muliggjort fra administrationspanelet, hvis nødvendigt.

Denne del er særlig nyttig i håndtering af data hvor historiske data kan være vigtige. Ved at benytte Soft Deletes, sikrer vi at vigtige data ikke tabes, dette giver os en mere fejltolerant arkitektur.

## Factories og Seeders

For at effektivisere udviklingsprocessen og sikre grundig testning af webshoppen er der implementeret brug af Factories og Seeders, som giver mulighed for nemt og hurtigt at kunne oprette realistisk testdata på shoppen.

**Factories**

Laravel’s model factories tillader at definere skabeloner for hvordan hver type af data, såsom brugere og bøger skal genereres.  
På den måde kan der opsættes modeller for at skabe konsistent og realistisk data som kan bruges både til udvikling, tests og automatiserede tests på shoppen.

**Seeders**

Laravel Seeders avendes til at indlæse data ud fra de factories der er defineret, ind i databasen, hvilket gør det muligt at berige webshoppen med relevant data ved hjælp at kun en enkelt kommando. Denne del er især vigtig i de tidlige stadier af udviklingen da det giver mulighed for præsentation og giver adgang til hurtig gendannelse af en beriget webshop, hvilket er ideelt for tests og kvalitetssikring.

Brugen af factories og seeders sikrer at udviklingsteamet kan arbejde effektivt, samt at webshoppen kan testes grundigt på alle tidspunkter af udviklingscyklussen, hvilket fører til et mere robust og pålideligt slutresultat.

# Konklusion

fortælle at siden ikke er nået at blive gjort fuldt ud responsiv, at ikke alt er færdig ift. Opsætning af edit og permanent delete, og at softdelete p.t ikke er opsat til at understøtte cascading soft-delete? Men permanent delte cascader.

AJAX søgning med parametre i url osv. Osv.

# Diskussion

# Referencer

Laravel debugbar

Stripe-php

ActivityLog

https://laravel-excel.com/

# Bilag

Herunder findes bilag.

**Mindmap**

Også tilgængeligt via url: <https://github.com/MichaelAggerholm/Apprentice_Test/blob/main/Assets/Documents/mindmap.drawio.pdf>

****

**Databasediagram**

Også tilgængeligt via url: <https://github.com/MichaelAggerholm/Apprentice_Test/blob/main/Assets/Documents/database_diagram.drawio.pdf>



**Klassediagram**

Også tilgængeligt via url: <https://github.com/MichaelAggerholm/Apprentice_Test/blob/main/Assets/Documents/klasse_diagram.drawio.pdf>



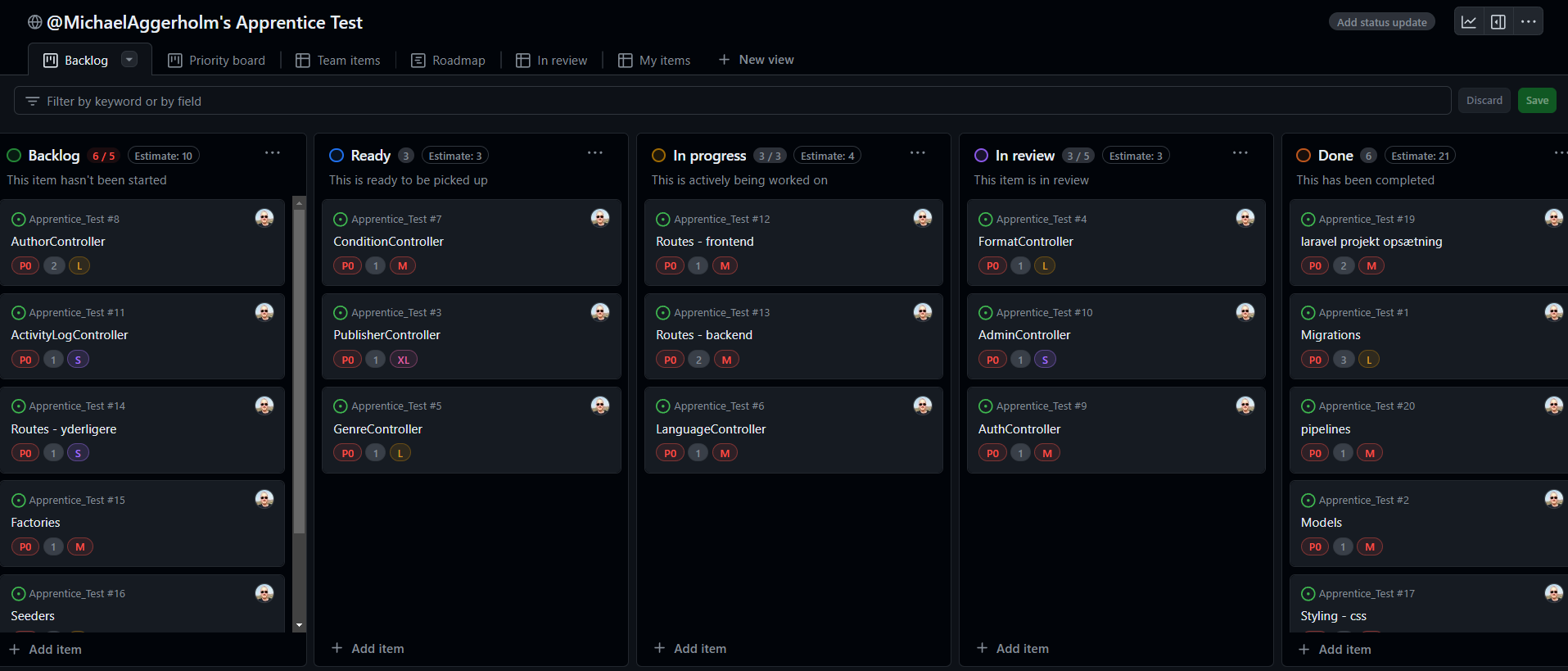
**Kanban Fase 1 – opstart af projekt**

Dette billede fra Kanban boardet er taget ved projektets opstart, lige inden opsætningen af Laravel projektet.

For at se Kanban boardet ’s nuværende fase, kan det tilgås via GitHub URL:  
<https://github.com/users/MichaelAggerholm/projects/2/views/1>

****

**Kanban Fase 2 – midtvejs i projekt**

****

**Kanban Fase 3 – Afslutning af projekt**

**Pipelines kørsel**

****

# Logbog

Logbog føres primært for at dokumentere daglige opgaver, udfordringer og løsninger. Det giver oversigt over projektets udvikling og muliggør refleksion over processen.

**09/04 – Tirsdag**

Intro til svendeprøve på Tech College.  
Efterfølgende er der valgt udarbejdelse af projekt fra delvis hjemmekontor/arbejdsplads fremadrettet.  
Opstart/opstilling af produktrapport samt procesrapport.  
Opsætning af Git repository til lagring af alle projektfiler.   
Mindmap, valg af metoder og teknologier.  
Udarbejdelse af Case beskrivelse og problemformulering.  
Klargøring af Kanban board til projektstyring via GitHub.

**10/04 – Onsdag**

Arbejde på produktrapport samt procesrapport.  
Udarbejdelse af Estimeret tidsplan.  
Opsætning af Kravspecifikation og Testspecifikation.  
Opsætning af databasediagram  
Opsætning af klassediagram

**11/04 – Torsdag**

Udarbejdet begrundelse for valg af metoder og teknologier  
Beskrivelse af produktets arkitektur i produktrapporten  
Beskrivelse af projektplanlægningen og det fremtidige forløb i procesrapporten.

**12/04 – Fredag**

Opstilling af udviklingstasks på Kanban board.  
Opsætning af Laravel applikation  
Opsætning af GitHub Actions til automatisk test workflow.

**15/04 – Mandag**

Yderligere tests af GitHub workflow.  
Påbegyndt design af frontend.  
Påbegyndt design af admin panel.  
Opsætning af migrations.  
Oprettet modeller, controllers og migreret til database.  
Påbegyndt opsætning af authentication.

**16/04 – Tirsdag**

Opsætning af administrerings funktionalitet for formater.  
Opsætning af authentication og fejlhåndtering ved un-authenticated adgang til panel.  
Opsætning af Adminpanel og routes samt middleware for adgangsstyring.  
Test og gennemgang af formater, opret, slet, genskab.   
Påbegyndt opsættelse af administrative funktioner for tilstande  
Udfordringer med at få logging til at virke korrekt, Events bliver ikke triggered korrekt og event listeners er ændret i Laravel 11 som nyligt er udkommet.

**17/04 – Onsdag**

Udfordringer med Logging løst, event listeners skulle registreres anderledes i Laravel 11.  
Færdiggørelse af administrativ funktionalitet for tilstande.  
Færdiggørelse af administrativ funktionalitet for genre.  
Færdiggørelse af administrativ funktionalitet for sprog.  
Færdiggørelse af administrativ funktionalitet for forfattere.  
Færdiggørelse af administrativ funktionalitet for udgivere.  
Påbegyndt opsætning af funktionalitet for administration af bøger.

**18/04 – Torsdag**

Færdiggjort administrative funktioner for bøger.  
Oprettelse af kurv.  
Designtilpasninger for visning af bøger.   
Funktionalitet og logik for kurv, mulighed for tilføjelse af antal, pris, samlet pris, osv.  
Test af kurv og bøger samt opsætning af multi-select for bøgers mange-til-mange.  
Påbegyndt opsætning af stripe integration og checkout funktionalitet.

**19/04 – Fredag**

Færdiggjort opsætning af stripe integration og checkout funktionalitet.  
Opsætning af factories og seeders for nem adgang til test data ved ny migration eller skift til anden pc.   
Opsætning af views og styling heraf på frontend / Adminpanel.   
Gennemgang af Kanban board, opgaver og refleksion over ugens arbejde.

**22/04 – Mandag**

Påbegyndt opsætning af bulk-import, forsøgt komplet import med bøger, forfattere, udgivere, genre osv. Alt sammen i ét go, det bliver for omfattende, der er ændret retning til at lave bog-import som er afhængig af at forfatter og yderligere allerede eksisterer på siden, hertil bør der muligvis laves en log over hvilke bøger der ikke blev importeret, og hvorfor.  
Kort test af UX, virker rigtig overskueligt og enkelt for både forbruger samt administrator.  
Kort test af UI, overholder ikke responsiv krav på nuværende tidspunkt, tilsidesættes til efter opgaver med højere prioritet færdiggøres.

**23/04 – Tirsdag**

Færdiggjort søgefunktion på forsiden, med mulighed for at søge bøger frem på titel, format, tilstand osv.  
Grundig test og tilpasning af søgefunktionalitet da der var mere omfattende SQL-Joins end forventet.

**24/04 – Onsdag**

Færdiggørelse af bulk-importer samt opsætning af hændelses-log eksporter til Excel ark download i browseren.  
Opsat administration af brugere fra Adminpanel med mulighed for at give brugere admin.   
Opfølgning på libraries og implementering i procesrapporten.

**25/04 – Torsdag**

**26/04 – Fredag**

**29/04 – Mandag**

**30/04 – Tirsdag**

**01/05 – Onsdag**

**02/05 – Torsdag**

**03/05 - Fredag**