

Stage dotNetLab

Realisatiedocument

Stan Dens

Student Bachelor in de Toegepaste Informatica – Applicatieontwikkeling

Inhoudsopgave

1. INLEIDING	3
2. ANALYSE	4
2.1. Onderzoekscontext	4
2.2. Functionele en niet-functionele requirements	5
2.2.1. Functionele requirements	5
2.2.2. Niet-functionele requirements	6
2.3. Technologiekeuze	7
2.3.1. Criteria	7
2.3.2. Back-end vergelijking	7
2.3.3. Front-end vergelijking	8
2.3.4. Databank vergelijking	8
3. ONTWERP & ARCHITECTUUR	9
3.1. Aanpak en werkwijze	9
3.2. Structuur en datamodel	9
3.3. Technische opbouw	11
4. REALISATIE	13
4.1. Managementportaal	13
4.1.1. Lego-sets	13
4.1.2. Bibliotheekbeheer	14
4.1.3. Collecties	16
4.1.4. Statistieken	17
4.1.5. Mijn realisaties	18
4.2. Bibliotheekportaal	18
4.2.1. Inventaris	18
4.2.2. Bedden	20
4.2.3. Statistieken	21
4.2.4. Mijn realisaties	22
4.3. Reservatieportaal	23
4.3.1. Overzicht lego-sets	23
4.3.2. Detailpagina lego-set	24
4.3.3. Informatiepagina Appies Legobib	24
4.3.4. Mijn realisaties	26
5. VOORTGANG EN PLANNING	27
5.1. Initiële planning	27
5.2. Werkelijke uitvoering	27
5.3. Toekomst van de applicatie	28
6. BESLUIT	29
LITERATUURLIJST	30
BIJLAGEN	31

1. Inleiding

Dit realisatiedocument gaat over mijn stage bij dotNET lab. Tijdens deze stage werkte ik mee aan Appies Legobib, een project binnen Bricks & More waarbij lego-sets worden uitgeleend aan kinderen in ziekenhuizen. Bricks & More is een organisatie die jaarlijks grote lego-expo's organiseert en binnen Appies Legobib instaat voor het beheer van de lego-sets.

In mijn projectplan werd de beginsituatie van Appies Legobib beschreven. Daarbij kwamen ook de doelstellingen, de meerwaarde voor de verschillende gebruikers en de eerste planning van de stage aan bod. Naarmate Appies Legobib verder uitbreidde naar verschillende ziekenhuizen, werd de vroegere werking met Excelbestanden minder praktisch. Daarom ontstond de nood om deze werking overzichtelijker en eenvoudiger te maken via een digitale oplossing.

In dit document wordt eerst de context van het project uitgelegd. Daarbij wordt besproken wat Appies Legobib is, welke gebruikers er zijn en aan welke vereisten de applicatie moest voldoen. Daarna wordt kort toegelicht welke technologieën gebruikt werden en waarom deze binnen het project passen.

Vervolgens wordt de opbouw van de applicatie besproken. Hierbij komen de werkwijze, het datamodel en de technische structuur aan bod. Daarna volgt de realisatie van de drie portalen: het managementportaal, het bibliotheekportaal en het reservatieportaal. Bij deze onderdelen wordt ook beschreven aan welke functionaliteiten ik zelf heb meegewerkt.

Tot slot wordt de oorspronkelijke planning uit het projectplan vergeleken met de werkelijke uitvoering van de stage. Ook wordt kort vooruitgekeken naar de verdere toekomst van de applicatie. Het document eindigt met een besluit waarin de belangrijkste resultaten en leerpunten worden samengevat.

2. Analyse

2.1. Onderzoekscontext

Appies Legobib is ontstaan vanuit het verhaal van Lowieke, een kind dat tijdens zijn ziekte vaak met lego speelde in het ziekenhuis van Geel. Na zijn overlijden wilden zijn ouders dat zijn lego-sets een tweede leven kregen. Door de sets beschikbaar te stellen aan andere kinderen in het ziekenhuis, kunnen ook zij tijdens hun opname afleiding en ontspanning vinden (Appies vzw, z.d.).

Het initiatief bestond dus al voordat de digitale toepassing werd ontwikkeld. In Ziekenhuis Geel wordt Appies Legobib bijvoorbeeld gebruikt op Planeet Goudgeel, de kinderafdeling van het ziekenhuis. Kinderen kunnen daar een lego-set kiezen, bouwen en nadien terugbrengen (Ziekenhuis Geel, 2025).

Omdat Appies Legobib verder uitbreidt naar meerdere ziekenhuizen, zoals Gent, Leuven, Herentals en Hasselt, werd het beheer via Excelbestanden steeds minder praktisch. Bricks & More, de organisatie die de lego-sets beheert en verdeelt over de ziekenhuizen, moet kunnen opvolgen welke sets beschikbaar zijn, waar ze zich bevinden en welke problemen er gemeld zijn. Met losse Excelbestanden wordt dat moeilijker naarmate het aantal ziekenhuizen, afdelingen en lego-sets toeneemt.

Daarom werd beslist om een digitale toepassing te ontwikkelen. Deze toepassing centraliseert het beheer van lego-sets, bibliotheken, bedden en reservaties. Hierdoor wordt alle informatie op één plaats bijgehouden en kan de werking eenvoudiger opgeschaald worden naar meerdere ziekenhuizen en afdelingen.

Tijdens mijn stage bij dotNET lab werkte ik mee aan deze toepassing. In de eerste weken lag de nadruk vooral op het begrijpen van de bestaande code, het datamodel en de noden van de gebruikers. Daarna werkte ik stap voor stap mee aan verschillende onderdelen van de applicatie.

De applicatie ondersteunt drie gebruikersgroepen. Bricks & More beheert de globale lego-inventaris en de verdeling over de ziekenhuizen. Het ziekenhuispersoneel volgt de dagelijkse werking op, zoals reservaties, bedden en teruggebrachte sets. Patiënten, ouders en bezoekers gebruiken de toepassing om eenvoudig lego-sets te bekijken en te reserveren.

Om deze doelgroepen goed te ondersteunen, bestaat de applicatie uit drie portalen: het managementportaal voor Bricks & More, het bibliotheekportaal voor het ziekenhuispersoneel en het reservatieportaal voor patiënten, ouders en bezoekers.

2.2. Functionele en niet-functionele requirements

De requirements beschrijven waaraan de toepassing moet voldoen. De functionele requirements gaan over wat de applicatie moet kunnen. De niet-functionele requirements gaan over de kwaliteit van de applicatie, zoals gebruiksvriendelijkheid, veiligheid, performantie en betrouwbaarheid.

2.2.1. Functionele requirements

De functionele requirements zijn verdeeld over de drie portalen van de applicatie. Elk portaal is afgestemd op een andere doelgroep en ondersteunt andere taken.

Voor Bricks & More ligt de focus op het algemene beheer van de legobibliotheken. Via het managementportaal moeten zij lego-sets kunnen beheren, importeren, aanpassen en verdelen over verschillende bibliotheken. Daarnaast moeten ziekenhuizen, bibliotheken, collecties en verantwoordelijken beheerd kunnen worden. Ook het opvolgen van problemen, zoals ontbrekende stukjes of beschadigde sets, behoort tot deze functionaliteiten. Het managementportaal bevat ook statistieken die Bricks & More een duidelijker beeld geven van de werking van de legobibliotheken. Zo kan de organisatie zien welke sets vaak gereserveerd worden, waar de voorraad beperkt is en welke sets mogelijk extra aangekocht moeten worden.

Voor het ziekenhuispersoneel ligt de nadruk op de dagelijkse werking van een afdeling. Via het bibliotheekportaal moeten medewerkers de inventaris van hun bibliotheek kunnen raadplegen, reservaties kunnen opvolgen en indien nodig zelf een reservatie kunnen aanmaken. Daarnaast moeten zij bedden kunnen beheren, QR-codes kunnen afdrukken en problemen met lego-sets kunnen melden of oplossen.

Voor patiënten, ouders en bezoekers moet de applicatie zo eenvoudig mogelijk blijven. Via het reservatieportaal krijgen zij toegang tot de juiste bibliotheek door een QR-code aan het bed te scannen. Daarna kunnen zij beschikbare lego-sets bekijken, een set reserveren, detailinformatie raadplegen en aangeven wanneer een set teruggebracht mag worden. Ook het raadplegen van een digitale handleiding en informatie over de oorsprong van Appies Legobib maakt deel uit van dit portaal.

Door de functionaliteiten per doelgroep te scheiden, krijgt elke gebruiker enkel toegang tot de onderdelen die voor hem of haar relevant zijn. Dit houdt de toepassing overzichtelijk en beperkt de kans op foutief gebruik.

2.2.2. Niet-functionele requirements

Naast de functionele vereisten zijn er ook niet-functionele requirements. Deze bepalen niet wat de applicatie precies doet, maar wel aan welke kwaliteitsvoorwaarden ze moet voldoen.

Gebruiksvriendelijkheid is belangrijk omdat de toepassing door verschillende doelgroepen wordt gebruikt. Kinderen, ouders, bezoekers en ziekenhuispersoneel moeten zonder technische kennis met de applicatie kunnen werken. Daarom moeten de schermen duidelijk zijn en moeten belangrijke acties snel gevonden worden.

Ook mobiel gebruik speelt een grote rol. Patiënten gebruiken het reservatieportaal vooral via een smartphone of tablet. Ziekenhuispersoneel werkt vaak met tablets tijdens de dagelijkse werking. Daarom moet de interface goed bruikbaar zijn op kleinere schermen, met duidelijke knoppen en overzichtelijke informatie.

Veiligheid en privacy spelen ook een belangrijke rol binnen de applicatie. Voor het managementportaal en het bibliotheekportaal werken gebruikers met een persoonlijk account. Via dat account kan gecontroleerd worden wie toegang heeft tot welke onderdelen van de applicatie. Zo krijgt elke gebruiker enkel toegang tot de gegevens die nodig zijn voor zijn of haar rol. Voor het reservatieportaal is bewust gekozen om niet met accounts te werken. Patiënten krijgen toegang door een QR-code aan het bed te scannen. Daardoor hoeven zij geen persoonlijk account aan te maken en moeten er minder persoonsgegevens opgeslagen worden. De applicatie gebruikt enkel de gegevens die nodig zijn om een reservatie te kunnen maken of terug te vinden. Dit is beter voor de privacy en sluit aan bij het GDPR-principe om niet meer persoonsgegevens te verzamelen dan nodig is (Intersoft Consulting, z.d.).

Betrouwbaarheid is nodig omdat meerdere gebruikers met dezelfde voorraad werken. Wanneer een set gereserveerd of teruggebracht wordt, moet de status correct aangepast worden. Anders kan een set foutief als beschikbaar of onbeschikbaar getoond worden.

Ook performantie speelt een rol. Voor gewone acties, zoals het openen van een overzicht, het aanpassen van een status of het ophalen van reservaties, wordt gestreefd naar responstijden onder de 200 milliseconden aan back-endzijde. Zwaardere acties, zoals het importeren van een Excelbestand of het genereren van meerdere QR-codes, mogen langer duren, maar moeten duidelijke feedback geven aan de gebruiker.

Tot slot moet de applicatie de administratieve druk verlagen. De toepassing mag vooral geen extra werk creëren, maar moet helpen om reservaties en lego-sets eenvoudiger op te volgen. Het doel is dus niet alleen om de oude Excelwerking te vervangen, maar ook om minder manueel werk en minder kans op fouten te hebben.

2.3. Technologiekeuze

Binnen dit project lagen de meeste technologieën al vast. Toch is het belangrijk om kort toe te lichten waarom deze keuzes passen bij Appies Legobib. De toepassing bestaat uit meerdere portalen, verschillende gebruikersrollen en een databank met onder andere lego-sets, bibliotheken, bedden en reservaties. Daarom waren vooral structuur, onderhoudbaarheid, performantie en integratie met de bestaande projectomgeving belangrijk.

Voor de back-end werd gewerkt met ASP.NET Core en de front-end werd ontwikkeld in Angular. Voor het ontwerp van schermen werd Figma gebruikt en het datamodel werd uitgewerkt met StarUML. Tijdens het project werd de databank omgezet van SQL Server naar PostgreSQL. Daarnaast werden Docker en Dokploy gebruikt om de applicatie lokaal en online beschikbaar te maken.

GitHub werd gebruikt voor versiebeheer en samenwerking. Via GitHub Actions werd ook een CI/CD-flow voorzien. Wanneer een pull request wordt aangemaakt, controleert de pipeline of de back-end en front-end correct bouwen, of de testen slagen en of de codekwaliteit in orde is. Wanneer wijzigingen op de main-branch terechtkomen en alle controles slagen, wordt de applicatie via Docker, GitHub Container Registry en Dokploy gedeployed naar de testomgeving.

De vergelijkingen in dit onderdeel zijn geen volledige technologiekeuze vanaf nul. Ze dienen vooral om te controleren of de gebruikte technologieën logisch aansluiten bij de noden van Appies Legobib. De scores zijn daarom projectgericht en gebaseerd op de vereisten van deze applicatie en op geraadpleegde bronnen.

2.3.1. Criteria

De technologieën werden vergeleken op basis van vijf criteria. Structuur en onderhoudbaarheid kregen het hoogste gewicht, omdat de toepassing uit meerdere portalen bestaat en later verder uitgebreid moet kunnen worden. Ook schaalbaarheid, performantie, ontwikkelsnelheid en integratie met de bestaande projectomgeving werden meegenomen.

Criteria	Gewicht
Structuur en onderhoudbaarheid	30%
Geschiktheid voor grotere applicaties	25%
Performantie	20%
Ontwikkelsnelheid	15%
Integratie met bestaande projectomgeving	10%

Tabel 1: Criteria voor de technologievergelijking

2.3.2. Back-end vergelijking

Voor de back-end werden ASP.NET Core en Node.js met elkaar vergeleken. ASP.NET Core wordt omschreven als een open-source en cross-platform framework voor moderne webapplicaties. Node.js is geen framework, maar een JavaScript runtime die sterk is in event-driven toepassingen en real-time communicatie (Fishchuk, 2025).

Technologie	Structuur	Grote app	Performantie	Snelheid	Integratie	Totaal
ASP.NET Core	8	8	8	7	9	8.0
Node.js	7	7	7	8	7	7.1

Tabel 2: Vergelijking tussen ASP.NET Core en Node.js

ASP.NET Core scoort hoger op structuur, onderhoudbaarheid en integratie met de projectomgeving. Dit is belangrijk omdat Appies Legobib bestaat uit meerdere portalen en verschillende gebruikersrollen.

Daarnaast bevat de toepassing veel domeinlogica rond reservaties, statussen, bibliotheken en bedden. Binnen dotNET lab sloot ASP.NET Core ook beter aan bij de bestaande kennis en werkwijze.

Node.js scoort hoger op ontwikkelsnelheid en flexibiliteit. Voor dit project waren structuur, onderhoudbaarheid en integratie echter belangrijker dan maximale flexibiliteit. Daarom paste ASP.NET Core beter binnen Appies Legobib.

2.3.3. Front-end vergelijking

Voor de front-end werden Angular en React met elkaar vergeleken. Angular biedt een vaste structuur en bevat ingebouwde functies zoals routing, dependency injection en formvalidatie. React is flexibeler en lichter, maar vraagt bij grotere projecten vaker extra bibliotheken en duidelijke afspraken rond structuur (Gommers, 2025).

Technologie	Structuur	Grote app	Performantie	Snelheid	Integratie	Totaal
Angular	8	8	7	7	9	7.8
React	7	7	8	8	7	7.3

Tabel 3: Vergelijking tussen Angular en React

React scoort hoger op flexibiliteit en ontwikkelsnelheid. Dit kan een voordeel zijn bij kleinere projecten of projecten waar veel vrijheid in structuur gewenst is.

Angular scoort sterker op structuur en geschiktheid voor grotere applicaties. Voor Appies Legobib was die vaste structuur belangrijk, omdat de applicatie uit drie portalen bestaat en door meerdere ontwikkelaars werd opgebouwd. Daarnaast was Angular de standaard binnen dotNET lab, waardoor deze keuze goed aansloot bij de bestaande werkwijze van het team.

2.3.4. Databank vergelijking

Voor de databank werden PostgreSQL en SQL Server met elkaar vergeleken. Beide databanksystemen zijn geschikt voor relationele data, maar verschillen in kostprijs, flexibiliteit en integratie. PostgreSQL is open-source en uitbreidbaar. SQL Server is sterk geïntegreerd binnen het Microsoft-ecosysteem en biedt uitgebreide beheertools (Ravoof, 2023).

Technologie	Structuur	Grote app	Performantie	Snelheid	Integratie	Totaal
PostgreSQL	8	8	8	7	8	7.9
SQL Server	8	8	8	7	7	7.9

Tabel 4: Vergelijking tussen PostgreSQL en SQL Server

De totaalscore van PostgreSQL en SQL Server ligt dicht bij elkaar, omdat beide databanken technisch geschikt waren voor dit project. SQL Server biedt sterke beheertools en integreert goed met Microsoft-producten, terwijl PostgreSQL open-source is en daardoor goedkoper gebruikt kan worden. Tijdens het project werd daarom overgeschakeld van SQL Server naar PostgreSQL. Vooral de lagere kostprijs en de goede samenwerking met Docker en Dokploy maakten PostgreSQL een logischere keuze voor de uiteindelijke opzet van Appies Legobib.

3. Ontwerp & Architectuur

3.1. Aanpak en werkwijze

Bij de start van de stage lag de nadruk op het begrijpen van het project en de bestaande werkwijze binnen het team. Omdat Appies Legobib voor mij een nieuw project was, werd eerst de bestaande code en structuur geanalyseerd. Daarna werd het project lokaal opgezet, zodat functionaliteiten getest en aangepast konden worden.

De ontwikkeling verliep via een backlog. Taken werden besproken, verdeeld en opgevolgd tijdens overlegmomenten. Nieuwe functionaliteiten werden lokaal ontwikkeld en getest. Daarna werden de aanpassingen via GitHub in een aparte branch geplaatst en via een pull request nagekeken. Feedback van medestudenten en de stagebegeleider werd verwerkt voordat de code werd samengevoegd met de hoofdbranch.

GitHub werd niet alleen gebruikt voor versiebeheer en samenwerking, maar ook als onderdeel van de automatische CI/CD-flow. Wanneer code werd gemerged naar de juiste branch, werd de testomgeving automatisch bijgewerkt. Daardoor was er steeds een recente versie van de applicatie beschikbaar voor demo's, tests en feedbackmomenten.

De voortgang werd elke dag besproken tijdens stand-ups. Daarnaast waren er ongeveer wekelijks uitgebreidere vergaderingen waarin de applicatie samen werd overlopen. Tijdens die momenten werd bekeken wat al werkte, welke problemen er nog waren en welke nieuwe taken aan de backlog toegevoegd moesten worden.

Om feedback te verzamelen, werden ook demo's georganiseerd. Een eerste demo werd gegeven aan mensen binnen dotNET lab. Ongeveer twee weken voor het einde van de stage werd ook een demo gegeven aan Bricks & More, de ouders van Lowieke en betrokken ziekenhuispersoneel. Die feedback hielp om de applicatie beter af te stemmen op de echte werking van Appies Legobib.

3.2. Structuur en datamodel

De applicatie bestaat uit een front-end, een back-end en een databank. Deze onderdelen werken samen, maar hebben elk een andere taak. De front-end is het deel waarmee de gebruiker werkt. Hierin bevinden zich de schermen van de verschillende portalen.

De back-end vormt de tussenlaag tussen de front-end en de databank. Wanneer een gebruiker een actie uitvoert, zoals een reservatie maken, een status aanpassen of een bed toevoegen, wordt deze actie eerst verwerkt door de back-end. Daar wordt gecontroleerd of de actie geldig is en of de gebruiker hiervoor de juiste rechten heeft. Daarna haalt de back-end gegevens op uit de databank of slaat aangepaste gegevens opnieuw op.

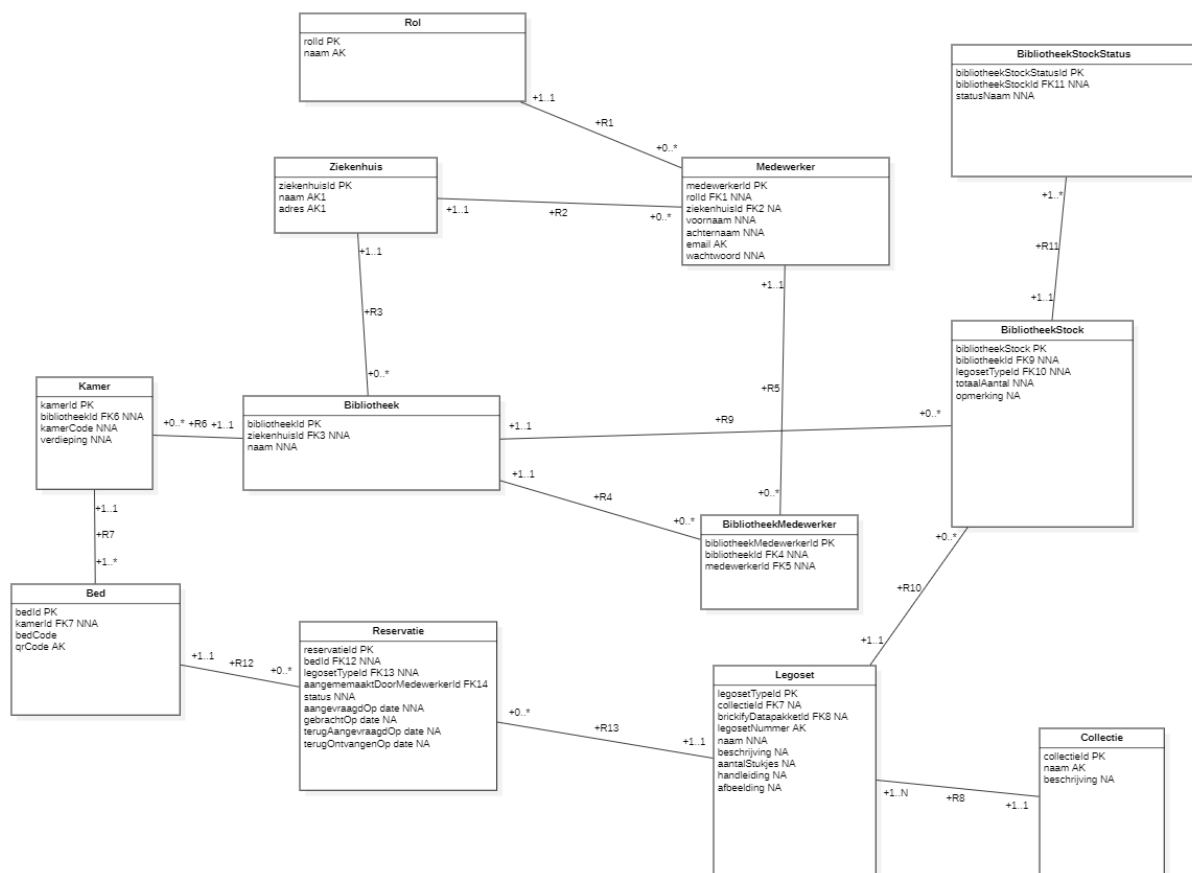
De databank bewaart de gegevens van de toepassing. In het datamodel staan de belangrijkste onderdelen van Appies Legobib, zoals ziekenhuizen, bibliotheken, lego-sets, bedden, reservaties, medewerkers en collecties. Door deze gegevens centraal op te slaan, werken de verschillende portalen met dezelfde informatie.

Een ziekenhuis kan meerdere bibliotheken of afdelingen bevatten. Binnen een bibliotheek worden kamers en bedden bijgehouden. Een bed heeft een eigen QR-code, zodat een patiënt via die QR-code automatisch in de juiste context terechtkomt. Reservaties worden gekoppeld aan een bed en aan een lego-set. Zo kan het systeem bijhouden welke set gereserveerd is, waar deze gebruikt wordt en wat de huidige status is.

De lego-sets worden centraal bijgehouden in de entiteit Legoset. Via BibliotheekStock wordt bepaald welke lego-sets aanwezig zijn in een bepaalde bibliotheek. Daar wordt ook informatie bijgehouden zoals het bibliotheeknummer, het aantal exemplaren en eventuele opmerkingen. De status van een set binnen een bibliotheek wordt gekoppeld aan BibliotheekStockStatus, bijvoorbeeld om aan te geven of een set beschikbaar, gereserveerd of niet beschikbaar is.

Ook gebruikers en rechten zijn opgenomen in het datamodel. Medewerkers zijn gekoppeld aan een rol en eventueel aan een ziekenhuis. Via BibliotheekMedewerker kunnen medewerkers aan één of meerdere bibliotheken gekoppeld worden. Daardoor kan de applicatie bepalen welke gegevens een gebruiker mag bekijken of beheren.

Daarnaast kunnen lego-sets aan een collectie gekoppeld worden. Een collectie groepeerd meerdere sets rond een bepaald thema of verhaal, zoals Lowie's collectie. Op die manier kunnen sets niet alleen praktisch beheerd worden, maar ook inhoudelijk gegroepeerd worden.



Afbeelding 1: ERD van Appies Legobib

3.3. Technische opbouw

Bij de start van de stage moest het project eerst lokaal opgezet worden. Hiervoor werd de bestaande code opgehaald via GitHub en werd de applicatie lokaal gestart met de nodige services. Omdat de toepassing uit een front-end, back-end en databank bestaat, was het belangrijk om eerst te begrijpen hoe deze onderdelen met elkaar samenwerken. Pas daarna kon ik gericht functionaliteiten aanpassen en testen.

De back-end werd ontwikkeld in ASP.NET Core. Binnen de back-end werd gewerkt op een manier die aansluit bij Domain-Driven Design. Om deze manier van werken beter te begrijpen, kregen we toegang tot een cursus op Pluralsight. Deze cursus moesten we afronden om meer inzicht te krijgen in hoe Domain-Driven Design wordt toegepast binnen een softwareproject.

Domain-Driven Design betekent dat de code zoveel mogelijk vertrekt vanuit de echte werking van de toepassing. Begrippen zoals bed, bibliotheek, reservatie, collectie en lego-set komen daardoor ook duidelijk terug in de code. Dit maakte het voor mij eenvoudiger om de logica van Appies Legobib te begrijpen en nieuwe onderdelen op de juiste plaats toe te voegen (Pluralsight, z.d.).

Deze manier van werken kwam ook terug in de structuur van de back-end. Er werd namelijk gewerkt met een vertical slice-architectuur, waarbij functionaliteiten zoveel mogelijk als één geheel worden opgebouwd. Bij beddenbeheer staan bijvoorbeeld de request, response, validatie en endpointlogica dicht bij elkaar. Daardoor is het makkelijker om later terug te vinden waar bepaalde code staat en om aanpassingen te doen. Deze manier van werken sluit aan bij het idee dat code beter gegroepeerd wordt per feature dan per technische laag (Dudycz, 2025).

Om deze structuur technisch uit te werken, werd voor de API-endpoints FastEndpoints gebruikt. Hiermee konden endpoints duidelijk per functionaliteit worden opgebouwd. Elke actie heeft zo een eigen request en response. Dit zorgde ervoor dat de back-end overzichtelijk bleef, ook wanneer er nieuwe functionaliteiten bijkwamen (FastEndpoints, z.d.).

Voor de databank werd PostgreSQL gebruikt. In Appies Legobib zijn veel gegevens met elkaar verbonden, zoals ziekenhuizen, bibliotheken, bedden, reservaties en lego-sets. PostgreSQL was hiervoor geschikt omdat het relationele gegevens gestructureerd kan opslaan (PostgreSQL Global Development Group, z.d.).

De communicatie tussen de back-end en de databank verliep via Entity Framework Core. Hierdoor kon er gewerkt worden met .NET-objecten in plaats van rechtstreeks met SQL-query's. Dit maakte het eenvoudiger om tabellen, relaties en migraties te beheren. Zeker bij aanpassingen aan het datamodel was dit handig, omdat wijzigingen op een gestructureerde manier konden worden doorgevoerd (Microsoft, 2024).

Voor de front-end werd Angular gebruikt. De applicatie werd opgebouwd met componenten, services en routing. Componenten vormen de herbruikbare onderdelen van de interface, services verzorgen de communicatie met de back-end en routing bepaalt welke pagina wordt getoond. Deze structuur maakte het mogelijk om de verschillende portalen overzichtelijk op te bouwen (Angular, z.d.).

Voor de opmaak werd Tailwind CSS gebruikt. Hiermee konden schermen snel en consistent opgebouwd worden. Dit was handig omdat de portalen visueel bij elkaar moesten passen, maar toch niet volledig hetzelfde zijn. Het reservatieportaal moest bijvoorbeeld eenvoudiger en mobielvriendelijker zijn dan het managementportaal (Tailwind CSS, z.d.).

Ook authenticatie en autorisatie waren belangrijk binnen de applicatie. Voor het managementportaal en het bibliotheekportaal werd gewerkt met Zitadel. Dit is een identityplatform dat gebruikt wordt voor authenticatie en autorisatie. Via Zitadel kunnen gebruikers inloggen en worden rollen meegegeven, zoals beheerder of bibliotheekverantwoordelijke. Op basis van die rollen bepaalt de applicatie welke onderdelen een gebruiker mag bekijken of beheren (Zitadel, z.d.).

De login verloopt via de back-end. Na een succesvolle login wordt een sessie bijgehouden met een cookie. De Angular front-end vraagt daarna aan de back-end wie de huidige gebruiker is en welke rollen hij heeft. Daardoor hoeft Angular zelf geen tokens van Zitadel te beheren en blijft de authenticatie vooral aan de serverkant.

Voor het reservatieportaal werd bewust niet met gewone gebruikersaccounts gewerkt. Patiënten krijgen toegang via een QR-code aan het bed. Zo kunnen zij eenvoudig een lego-set reserveren zonder eerst een account te moeten aanmaken. Dit houdt de drempel laag en beperkt ook de hoeveelheid persoonsgegevens die opgeslagen moeten worden.

Voor het genereren en aanpassen van QR-codes werd SkiaSharp gebruikt. Hiermee konden QR-code-afbeeldingen aangemaakt worden, bijvoorbeeld met het logo van een ziekenhuis. De QR-codes vormen binnen Appies Legobib de koppeling tussen de fysieke ziekenhuisomgeving en de digitale applicatie. Een QR-code aan een bed verwijst de gebruiker meteen door naar het juiste reservatieportaal (SkiaSharp, z.d.).

Voor automatische controles en deployment werd GitHub Actions gebruikt. Bij een pull request controleerde de pipeline eerst of de back-end en front-end nog correct builden. Aan de back-endzijde werden integratietesten uitgevoerd. Daarbij werd de applicatie getest in combinatie met een PostgreSQL-testdatabase, zodat niet alleen losse stukken code, maar ook de werking van endpoints, validaties en databankacties gecontroleerd werd. Daarnaast werd de test coverage berekend en werd nagekeken of de code correct geformatteerd was.

Aan de front-endzijde werden eerst de dependencies geïnstalleerd. Daarna voerde de pipeline de Angular-build, linting en testen uit. De front-endtesten bestonden uit unit tests en component tests. Daarnaast werd Playwright gebruikt om deze testen in een echte browseromgeving te laten draaien. Hierdoor werd niet alleen gecontroleerd of de TypeScript-code correct was, maar ook of componenten en schermen zich correct gedroegen in een browser.

Pas wanneer deze controles slaagden en de wijzigingen op de main-branch terechtkwamen, werd de testomgeving gedeployed. Daarbij werden de containerimages aangemaakt en doorgestuurd naar GitHub Container Registry. Daarna werd Dokploy gebruikt om de testomgeving bij te werken. Zo was er telkens een recente versie van de applicatie beschikbaar voor demo's, testen en feedbackmomenten (GitHub, 2026).

4. Realisatie

4.1. Managementportaal

Het managementportaal ondersteunt Bricks & More bij het centrale beheer van Appies Legobib. Binnen dit portaal worden de globale inventaris, ziekenhuizen, bibliotheken, collecties en statistieken beheerd. De functionaliteiten zijn opgedeeld in vier onderdelen: lego-sets, bibliotheekbeheer, collecties en statistieken.

Naast het beheer biedt het portaal ook inzicht in voorraad, reservaties en gemelde problemen. Daardoor kan Bricks & More beter opvolgen waar sets ontbreken, welke sets vaak gebruikt worden en waar extra actie nodig is.

4.1.1. Lego-sets

Het onderdeel lego-sets bevat de globale inventaris van Appies Legobib. Bricks & More krijgt hier een overzicht van alle lego-sets die binnen het project beschikbaar zijn. Deze inventaris vormt de basis voor de verdeling van sets over de verschillende ziekenhuizen en bibliotheken.

Omdat de oude werking gebaseerd was op Excelbestanden, kan Bricks & More een Excelbestand uploaden. De lego-sets uit dit bestand worden automatisch geïmporteerd in de applicatie. Hierdoor moeten bestaande sets niet één voor één handmatig ingevoerd worden. Na de import blijven de gegevens aanpasbaar, zodat fouten gecorrigeerd kunnen worden of ontbrekende informatie later aangevuld kan worden. Daarnaast kunnen lego-sets ook handmatig toegevoegd worden. Bij het toevoegen van een set wordt gebruikgemaakt van de Rebrickable API om informatie over officiële LEGO-sets op te halen (Rebrickable, z.d.)

Niet alle gegevens kunnen automatisch opgehaald worden. De Nederlandstalige naam van een set moet handmatig worden ingevuld, omdat deze niet via Rebrickable beschikbaar is. Daarom wordt op basis van het setnummer een URL voorzien naar de juiste LEGO-pagina, zodat de gebruiker de Nederlandstalige naam kan opzoeken en overnemen.

In het overzicht kan gefilterd worden op naam, thema, bibliotheek en status. Daarnaast kan per lego-set via een dropdown snel aangepast worden tot welke bibliotheek de set behoort. Hierdoor kan Bricks & More de globale inventaris sneller beheren en sets eenvoudiger toewijzen of herverdelen.

Ook problemen bij lego-sets worden in dit onderdeel opgevolgd. Wanneer er bijvoorbeeld stukjes ontbreken of een set beschadigd is, wordt dit zichtbaar in het overzicht. Zodra het probleem opgelost is, kan de melding als afgehandeld worden aangeduid.

Legosets
Een overzicht van alle legosets.

Excel uploaden Legoset toevoegen

Zoek op titel of setnummer: Big Ben of 60316

Bibliotheek: Alle bibliotheken

Thema: Alle thema's

Filters wissen

< 1 - 15 >

AFBEELDING	TITEL	LEEFTIJD	STENEN	THEMA	STATUS
	4x4 Terreinambulance redding Setnummer: 40582 Geel - Goudgeel	6+	150	City	Meer details Probleem gemeld
	Ijswagen Setnummer: 60253 Geel - Goudgeel	5+	200	City	Meer details Probleem gemeld
	Familiehuis Setnummer: 10467 UZG - UZG	2+	84	DUPLO Peppa Pig	Meer details Is in orde

Stan Dens
Uitloggen

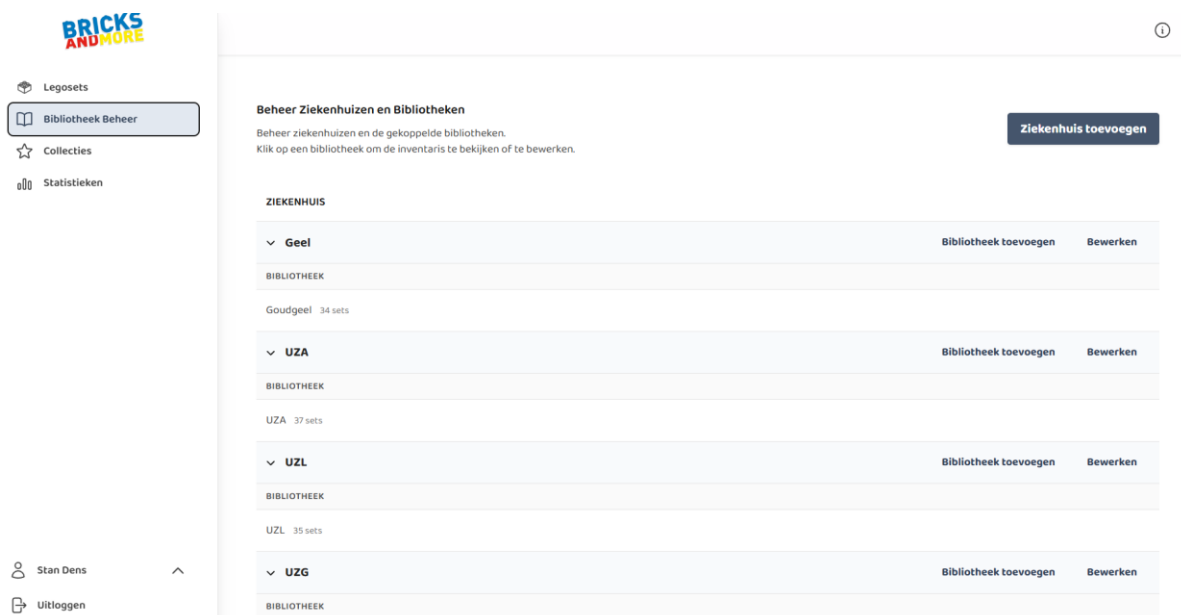
Afbeelding 2: Overzicht legosets managementportaal

4.1.2. Bibliotheekbeheer

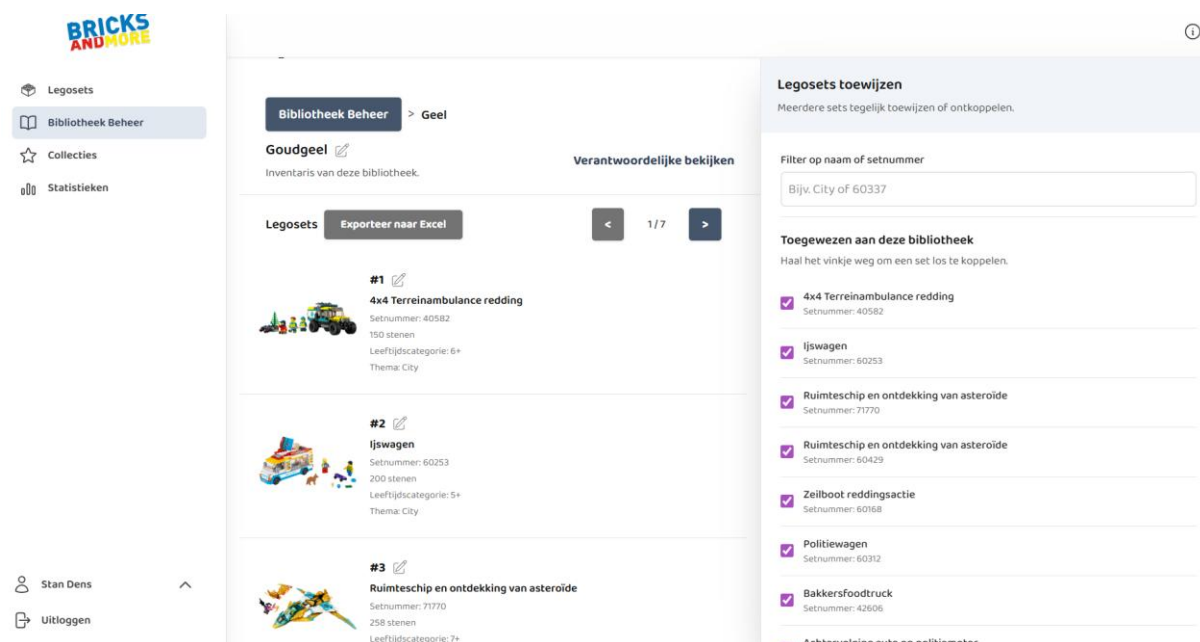
Via bibliotheekbeheer beheert Bricks & More de structuur van de legobibliotheken. Hier kunnen ziekenhuizen toegevoegd worden en kunnen binnen die ziekenhuizen bibliotheken of afdelingen aangemaakt worden. Dit is nodig omdat één ziekenhuis meerdere afdelingen kan hebben waar lego-sets worden uitgeleend.

Per bibliotheek kunnen lego-sets uit de globale inventaris worden toegewezen. Daardoor blijft er een duidelijk onderscheid tussen de volledige voorraad en de sets die effectief in een bepaalde afdeling aanwezig zijn. Dit maakt het eenvoudiger om op te volgen waar sets zich bevinden en welke bibliotheken nog aangevuld moeten worden.

Binnen bibliotheekbeheer kunnen ook bibliotheekverantwoordelijken toegevoegd of verwijderd worden. Wanneer een verantwoordelijke wordt toegevoegd, krijgt deze automatisch een account met een tijdelijk wachtwoord. Bij de eerste aanmelding moet dit wachtwoord gewijzigd worden. Zo krijgt de juiste persoon toegang tot het bibliotheekportaal en blijft de toegang beheersbaar.



Afbeelding 3: Overzicht ziekenhuizen en bibliotheken



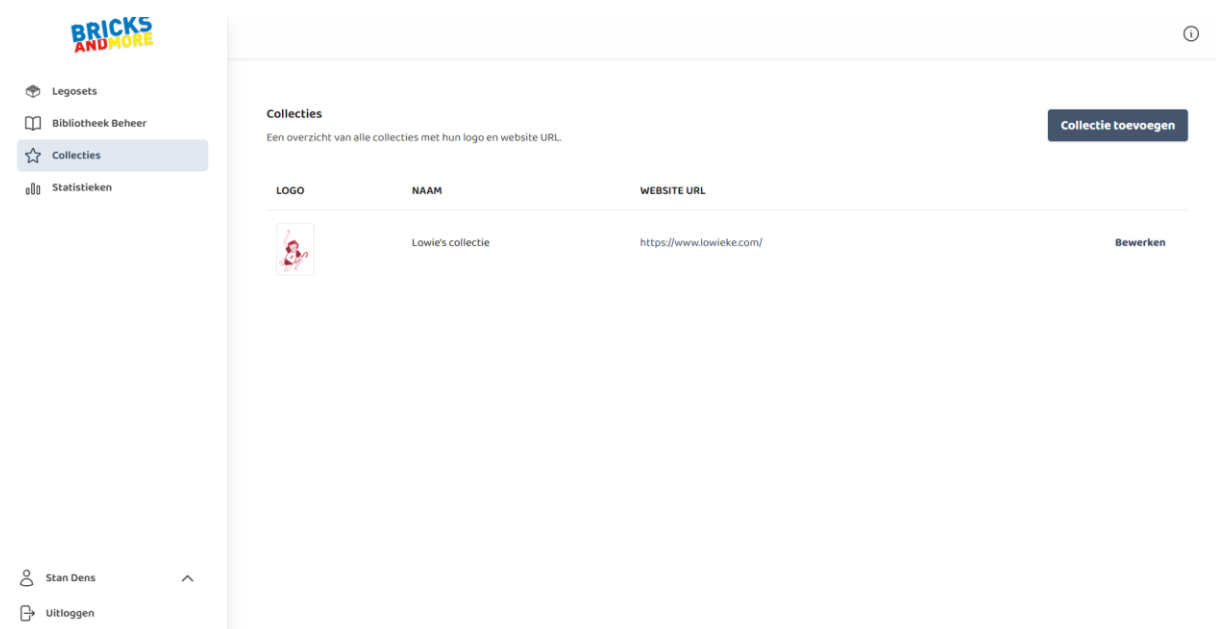
Afbeelding 4: Bibliotheek van een ziekenhuis en zijn legosets

4.1.3. Collecties

Het onderdeel collecties wordt gebruikt om verzamelingen van lego-sets te beheren. Een collectie groepeert meerdere sets rond een bepaald thema, verhaal of doel. Hierdoor kunnen bepaalde sets herkenbaar samengebracht worden.

Een belangrijk voorbeeld is Lowie's collectie. Deze collectie verwijst naar het kind waarvoor het initiatief oorspronkelijk begon. Door deze collectie in de applicatie op te nemen, blijft het verhaal achter Appies Legobib ook zichtbaar binnen de digitale werking.

Binnen dit onderdeel kunnen collecties aangemaakt en aangepast worden. Bij een collectie kan ook een logo of afbeelding geüpload worden. Dit logo wordt later in het reservatieportaal naast de lego-set getoond wanneer die set tot de collectie behoort. Hierdoor wordt de collectie ook voor patiënten, ouders en bezoekers visueel herkenbaar. Het koppelen van lego-sets aan een collectie gebeurt vanuit de inventaris. Op die manier kan Bricks & More tijdens het beheren van de lego-sets meteen aangeven of een set tot een bepaalde collectie behoort.



Afbeelding 5: Overzicht legosets

4.1.4. Statistieken

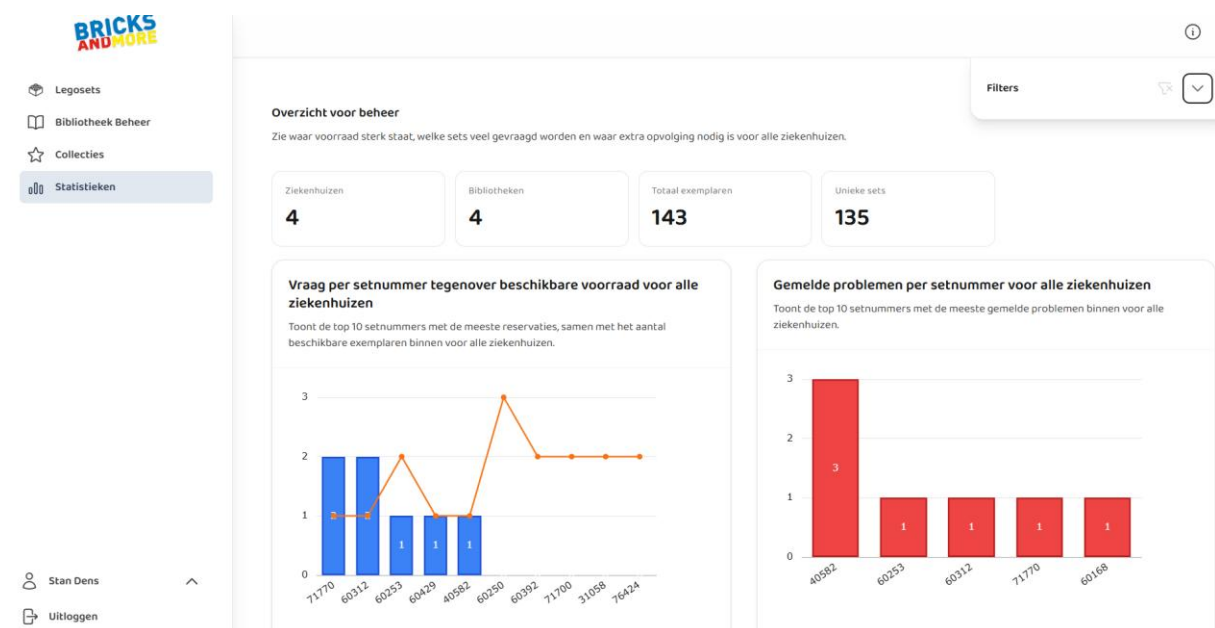
Het onderdeel statistieken geeft Bricks & More inzicht in de voorraad en het gebruik van de lego-sets. Met filters kan gekozen worden om de gegevens voor alle ziekenhuizen, één specifiek ziekenhuis of één bibliotheek te bekijken.

Bovenaan het scherm wordt een algemeen overzicht getoond met het aantal ziekenhuizen, bibliotheken, exemplaren en unieke sets. Daarnaast toont het portaal de vraag per setnummer tegenover de beschikbare voorraad. Hierbij worden de setnummers met de meeste reservaties vergeleken met het aantal beschikbare exemplaren. Zo kan Bricks & More zien welke sets vaak gevraagd worden en of er voldoende voorraad beschikbaar is.

Ook gemelde problemen per setnummer worden weergegeven. Hierdoor kan opgevolgd worden welke sets het vaakst problemen veroorzaken, zoals ontbrekende stukjes of beschadigingen. Daarnaast toont het portaal de verdeling van reservaties per thema, zodat zichtbaar wordt welke thema's het populairst zijn.

Wanneer er voldoende gegevens beschikbaar zijn, kan het systeem ook tonen welke set interessant is om bij te kopen. Tot slot bevat het scherm een detailoverzicht per ziekenhuis, met per ziekenhuis het aantal bibliotheken, exemplaren en unieke sets.

Deze statistieken helpen Bricks & More om beter onderbouwde beslissingen te nemen over voorraad, herverdeling en mogelijke nieuwe aankopen.



Afbeelding 6: Statistieken managementportaal

4.1.5. Mijn realisaties

Binnen het managementportaal werkte ik aan verschillende onderdelen van lego-sets, bibliotheken en collecties.

Bij de lego-sets werkte ik mee aan de basis om sets weer te geven en handmatig toe te voegen. Daarnaast maakte ik het overzicht beter door filters toe te voegen. Hierdoor kan er gezocht worden op naam, thema, bibliotheek en status.

Binnen bibliotheekbeheer werkte ik mee aan de lay-out van de pagina. Ook voegde ik pagination toe aan het overzicht van lego-sets, zodat lange lijsten niet volledig onder elkaar getoond worden. Daarnaast werkte ik aan de mogelijkheid om de naam van een bibliotheek of afdeling te bewerken.

Voor collecties werkte ik de volledige beheerpagina uit. Op deze pagina kunnen collecties toegevoegd en aangepast worden. Ook maakte ik het mogelijk om een logo of afbeelding aan een collectie te koppelen.

Verder voegde ik doorheen de applicatie laadiconen toe. Deze verschijnen wanneer gegevens worden opgehaald of wanneer een actie nog bezig is. Zo ziet de gebruiker dat de applicatie aan het laden is en niet vastloopt.

Mijn bijdrage binnen het managementportaal lag vooral bij het verbeteren van het overzicht en beheer van lego-sets, het uitbreiden van bibliotheekbeheer en het volledig uitwerken van het collectiebeheer.

4.2. Bibliotheekportaal

Het bibliotheekportaal ondersteunt het ziekenhuispersoneel bij de dagelijkse werking van een legobibliotheek. Medewerkers kunnen hier de inventaris van hun afdeling opvolgen, bedden beheren en statistieken raadplegen.

Ook Bricks & More kan dit portaal gebruiken om een specifieke bibliotheek te bekijken. In dat geval kiest de gebruiker eerst een bibliotheek, waarna enkel de gegevens van die bibliotheek getoond worden. Bovenaan het portaal wordt duidelijk weergegeven welke bibliotheek actief is, samen met het bijhorende logo.

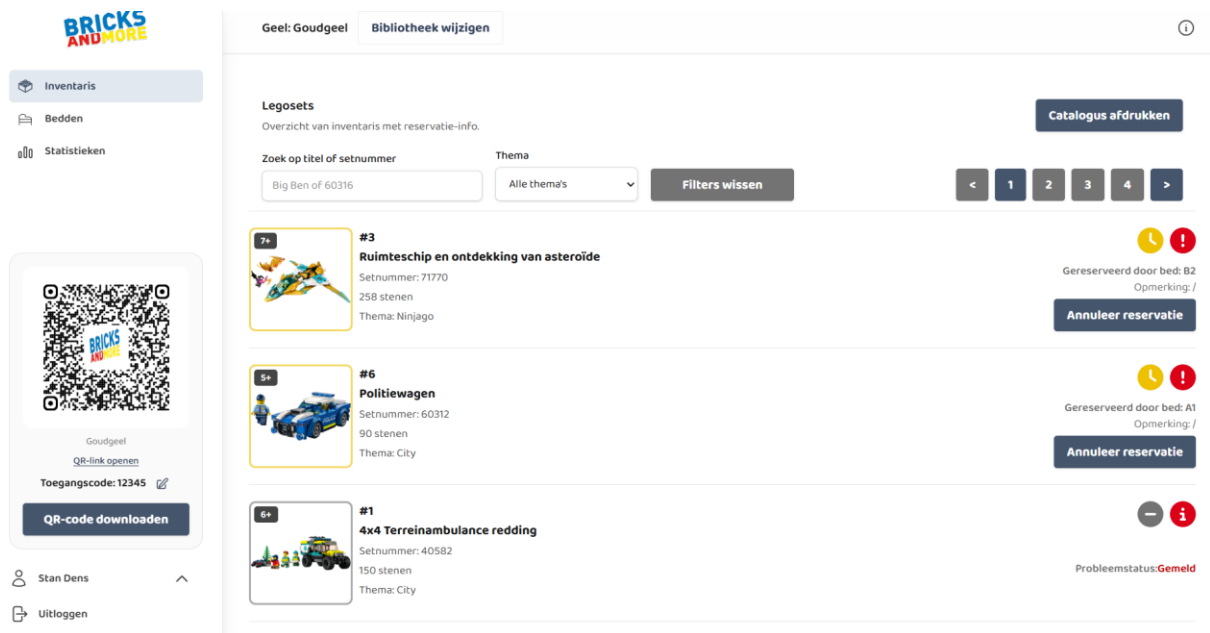
4.2.1. Inventaris

Het onderdeel inventaris geeft een overzicht van alle lego-sets die aanwezig zijn binnen de geselecteerde bibliotheek of afdeling. Per lego-set is zichtbaar wat de huidige status is, bijvoorbeeld beschikbaar, gereserveerd of niet beschikbaar. Hierdoor kunnen medewerkers snel zien welke sets klaar zijn om uitgeleend te worden en welke sets tijdelijk niet gebruikt kunnen worden.

Binnen dit overzicht kan gefilterd worden op naam, thema en status. Vanuit de inventaris kunnen ook reservaties aangemaakt worden. Dit is nuttig wanneer een patiënt, ouder of bezoeker zelf geen reservatie kan maken via het reservatieportaal. Het ziekenhuispersoneel kan de reservatie dan toch correct registreren, zodat de voorraad en status van de set blijven kloppen.

Daarnaast kunnen medewerkers problemen melden bij een lego-set. Dit kan bijvoorbeeld gaan over ontbrekende stukjes, beschadigingen of andere opmerkingen. Deze meldingen blijven zichtbaar in het portaal, zodat medewerkers weten welke sets aandacht nodig hebben. Wanneer het probleem opgelost is, kan de melding als afgehandeld aangeduid worden.

Ook kan vanuit de inventaris een catalogus worden afgedrukt. Deze catalogus bevat alle lego-sets van de gekozen bibliotheek. De afgedrukte versie kan fysiek gebruikt worden op de afdeling, zodat kinderen door de beschikbare sets kunnen bladeren zonder dat ze meteen een digitaal toestel nodig hebben.

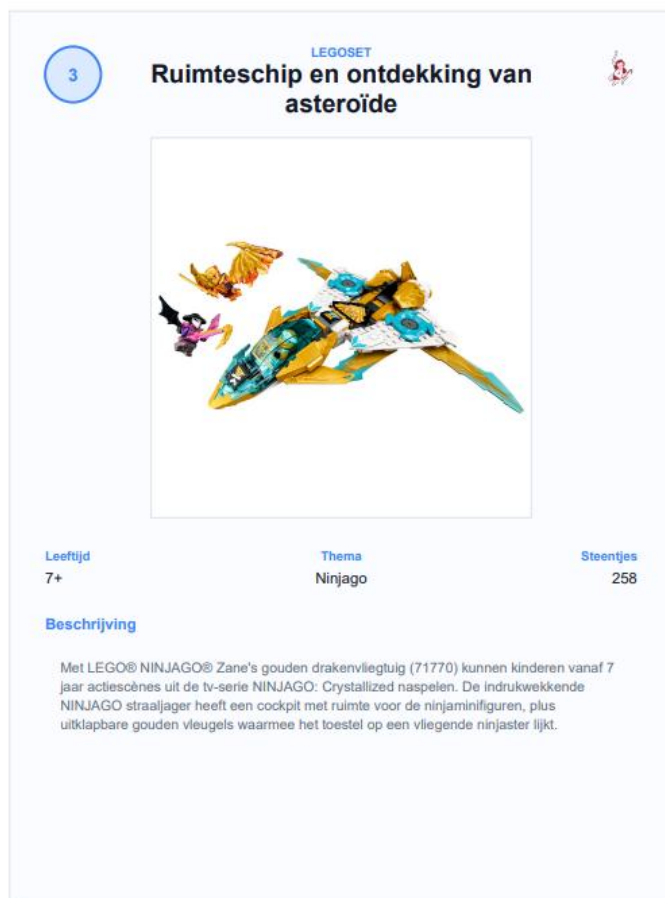


Afbeelding 7: Overzicht legosets bibliotheekportaal

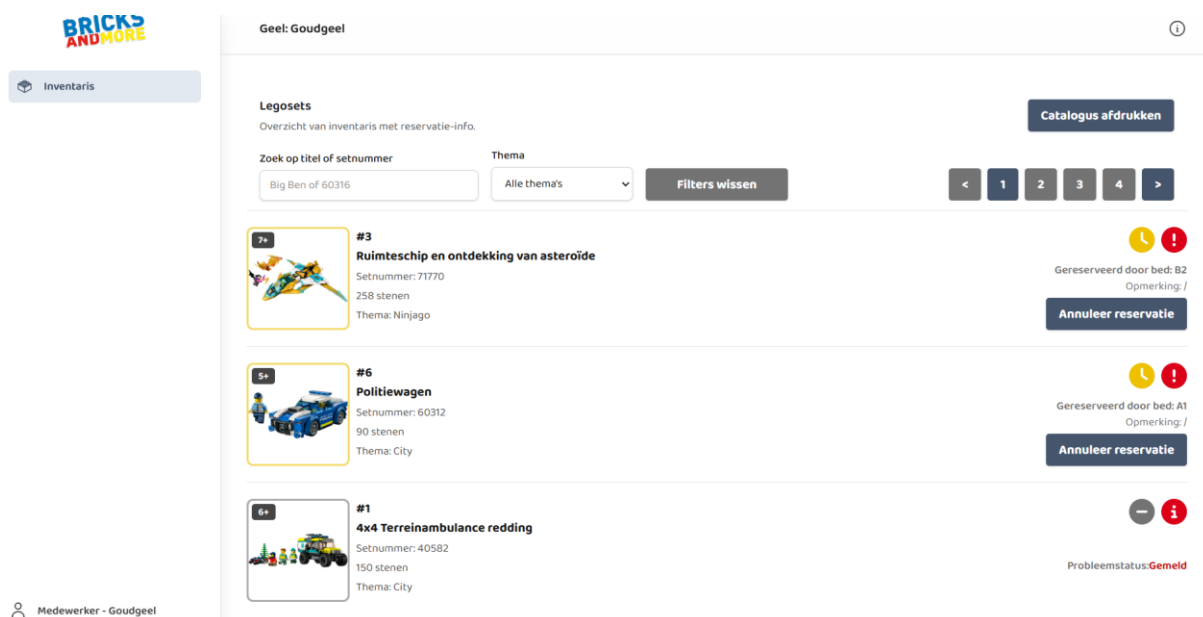
Catalogus Goudgeel

Geel

Pagina 1/34



Afbeelding 8: Catalogus bibliotheek



Afbeelding 10: Pagina na toegangscode bibliotheekportaal

4.2.3. Statistieken

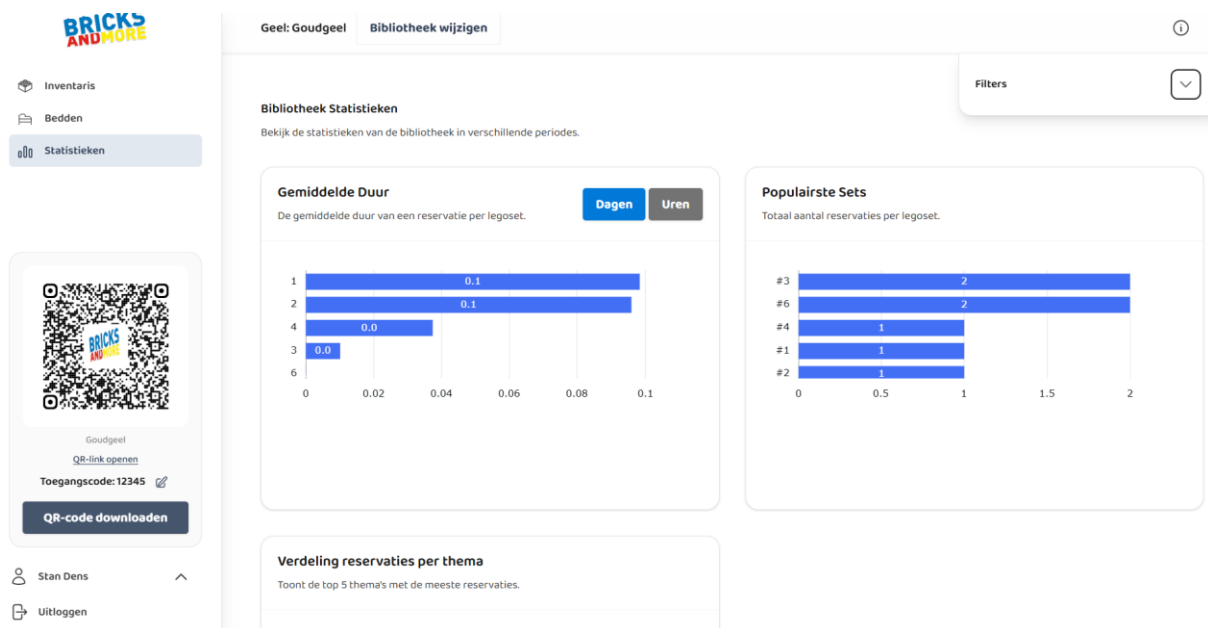
Het onderdeel statistieken geeft een overzicht van de werking van één specifieke bibliotheek of afdeling. Medewerkers kunnen hier bekijken hoe de lego-sets gebruikt worden binnen hun bibliotheek.

Met de filters kan gekozen worden voor welke periode de gegevens bekeken worden: totaal, per maand of per week. Daarnaast kan gekozen worden of alle resultaten getoond worden, of enkel een selectie van de belangrijkste resultaten.

In dit onderdeel wordt onder andere de gemiddelde duur van reservaties per lego-set weergegeven. Hierdoor krijgt het ziekenhuispersoneel een beter beeld van hoe lang sets gemiddeld bij patiënten blijven.

Daarnaast toont het portaal welke sets het vaakst gereserveerd worden. Zo wordt duidelijk welke lego-sets populair zijn binnen de bibliotheek. Ook wordt weergegeven welke thema's het meest gekozen worden. Dit kan helpen om beter te begrijpen welke soorten sets aansluiten bij de interesses van de kinderen.

Onderaan wordt een legende getoond met het volgnummer, de naam en het setnummer van de betrokken lego-sets. Dit helpt om de grafieken en cijfers correct te interpreteren. Wanneer er nog onvoldoende gegevens beschikbaar zijn, wordt aangegeven dat er geen data beschikbaar is.



Afbeelding 11: Statistieken bibliotheekportaal

4.2.4. Mijn realisaties

Het bibliotheekportaal is het onderdeel waaraan ik tijdens de stage het meest heb gewerkt.

Bij de inventaris voegde ik filters toe op naam, thema en status, zodat ziekenhuispersoneel sneller kan zoeken binnen de sets van de eigen bibliotheek. Daarnaast werkte ik mee aan het melden en opvolgen van problemen bij lego-sets, bijvoorbeeld wanneer stukjes ontbreken of een set beschadigd is.

Ook zorgde ik ervoor dat de catalogus van een bibliotheek afgedrukt kan worden. Deze catalogus bevat alle lego-sets van de gekozen bibliotheek en kan fysiek gebruikt worden op de afdeling.

Daarnaast werkte ik aan het beddenbeheer. Ik zorgde ervoor dat bedden toegevoegd en aangepast kunnen worden en dat elk bed automatisch een unieke QR-code krijgt. Deze QR-codes kunnen gegenereerd en afgedrukt worden, met het logo van het ziekenhuis erin verwerkt.

Naast de QR-codes voor bedden werkte ik ook de QR-code voor de bibliotheek uit. Na het scannen moet eerst een toegangscode ingegeven worden. Daarna wordt een pagina geopend waarop de reservaties van die bibliotheek geraadpleegd kunnen worden. De bibliotheekverantwoordelijke kan deze toegangscode zelf aanpassen.

Verder zorgde ik ervoor dat de actieve bibliotheek duidelijk zichtbaar is in de interface. Ook werkte ik het keuzeschermbij de bibliotheek eerst een bibliotheek kiest voordat het bibliotheekportaal geopend wordt.

4.3. Reservatieportaal

Het reservatieportaal is bedoeld voor patiënten, ouders en bezoekers. Via dit portaal kunnen zij lego-sets bekijken, reserveren en terugbrengen. De gebruiker komt in het portaal terecht door een QR-code aan het bed te scannen. Daardoor worden automatisch de juiste bibliotheek en het juiste bed gekoppeld. De patiënt hoeft dus zelf geen afdeling of bed te kiezen.

Het portaal is bewust eenvoudig gehouden. De belangrijkste acties moeten snel duidelijk zijn, omdat de toepassing gebruikt wordt door kinderen, ouders en bezoekers in een ziekenhuisomgeving.

4.3.1. Overzicht lego-sets

In het overzicht ziet de patiënt de lego-sets die tot de eigen bibliotheek of afdeling behoren. Beschikbare sets kunnen meteen gereserveerd worden via de knop bij de set. Sets die al gereserveerd of tijdelijk niet beschikbaar zijn, kunnen niet opnieuw gereserveerd worden.

De patiënt kan ook de taal van de applicatie wijzigen. Omdat Appies Legobib in verschillende ziekenhuizen gebruikt wordt, is het mogelijk dat niet elke gebruiker Nederlands spreekt. Daarom kan de applicatie ook in het Engels gebruikt worden.

Wanneer een patiënt een set reserveert, wordt deze gekoppeld aan het bed waarvan de QR-code werd gescand. Hierdoor kan per bed opgevolgd worden welke set actief gereserveerd is. Dit voorkomt dat dezelfde set dubbel wordt uitgeleend en zorgt ervoor dat het ziekenhuispersoneel de reservaties correct kan opvolgen.

Via het overzicht kan de gebruiker ook naar de eigen reservatie gaan. Daar ziet de patiënt welke set op dat moment gereserveerd is. Wanneer de set teruggebracht mag worden, kan dit via dezelfde omgeving aangeduid worden. De set wordt daarna niet meteen opnieuw beschikbaar, omdat een medewerker ze eerst moet controleren of desinfecteren. Pas daarna kan de set opnieuw gereserveerd worden.



Afbeelding 12: Overzicht legosets reservatieportaal

4.3.2. Detailpagina lego-set

Wanneer een gebruiker op een lego-set klikt, wordt een detailpagina geopend. Op deze pagina staat extra informatie over de set, zoals de naam, het setnummer, het thema, het aantal steentjes, een afbeelding en een korte beschrijving. Ook kan de digitale handleiding van de lego-set geraadpleegd worden.

De detailpagina helpt patiënten en bezoekers om een betere keuze te maken. Ze kunnen eerst bekijken welke set het is, hoeveel onderdelen de set bevat en welk setnummer erbij hoort. Dat setnummer is ook nuttig wanneer de gebruiker de set wil opzoeken in de LEGO Builder-app. Wanneer de gebruiker een actieve reservatie heeft, kan op deze pagina ook de actie om de set terug te brengen beschikbaar zijn.



Afbeelding 13: Detailpagina legoset reservatieportaal

4.3.3. Informatiepagina Appies Legobib

Het reservatieportaal bevat ook een informatiepagina over Appies Legobib. Op deze pagina wordt kort uitgelegd waar het initiatief vandaan komt en hoe het reservatieportaal gebruikt kan worden.

De pagina vertelt dat Appies Legobib ontstond ter nagedachtenis van Lowieke, die graag met LEGO bouwde. Zijn LEGO-sets kregen een warme tweede bestemming in het kinderziekenhuis, zodat kinderen tijdens hun verblijf kunnen bouwen, spelen en even afleiding vinden. Vanuit deze pagina kan de gebruiker ook doorklikken naar de externe pagina met het volledige verhaal van Appies Legobib.

Daarnaast legt de pagina op een kindvriendelijke manier uit hoe het systeem werkt. De gebruiker krijgt stap voor stap uitleg over het kiezen van een set, het reserveren, het bekijken van details en het gebruiken van de handleiding. Ook wordt uitgelegd dat de set na het spelen opnieuw netjes terug in de doos moet worden gestopt en via de applicatie teruggebracht kan worden.



Afbeelding 14: Infopagina Appies legobib reservatieportaal

4.3.4. Mijn realisaties

Binnen het reservatieportaal was mijn bijdrage beperkter dan bij het bibliotheekportaal.

Een eerste bijdrage was het mee beheren van de statussen achter de schermen. Wanneer een lego-set gereserveerd wordt, moet de status correct aangepast worden zodat de set niet opnieuw door iemand anders gereserveerd kan worden. Wanneer een patiënt aangeeft dat een set teruggebracht mag worden, wordt de set niet meteen opnieuw beschikbaar gemaakt. De status wordt dan aangepast naar niet beschikbaar, zodat een medewerker de set eerst kan controleren of desinfecteren. Pas daarna kan de medewerker de set opnieuw beschikbaar maken.

Daarnaast werkte ik mee aan het design van het reservatieportaal. Het uiteindelijke portaal bevat verschillende elementen uit deze ontwerpen, zoals de cards voor lego-sets, de pop-ups en de algemene lay-out.



Afbeelding 15: Figma-ontwerp van het reservatieportaal.

5. Voortgang en planning

5.1. Initiële planning

Bij de start van de stage werd een initiële planning opgesteld om de belangrijkste taken over de stageperiode te verdelen. Deze planning gaf een eerste beeld van de volgorde waarin onderdelen onderzocht, ontwikkeld en getest zouden worden. De volledige initiële planning is opgenomen in bijlage A.

In de eerste weken lag de nadruk vooral op het leren kennen van Angular, het lokaal opzetten van het project en het begrijpen van de bestaande structuur. Daarna was er ruimte voorzien voor de eerste CRUD-schermen, de QR-code-integratie, het onderzoek naar API-integratie, de migratie naar PostgreSQL en het verder toepassen van Domain-Driven Design.

Vanaf het midden van de stage verschoof de geplande focus naar de concrete functionaliteiten van de drie portalen. In de laatste weken zouden vooral UX-verbeteringen, testen en het verwerken van feedback centraal staan.

De planning werd niet opgesteld als een vast schema dat exact gevolgd moest worden, maar als een hulpmiddel om de voortgang op te volgen. Tijdens de stage konden taken dus verschuiven of uitbreiden door nieuwe inzichten, technische problemen of feedbackmomenten.

5.2. Werkelijke uitvoering

De werkelijke uitvoering week op verschillende punten af van de initiële planning. Een overzicht hiervan is opgenomen in bijlage B. De eerste weken verliepen grotendeels zoals verwacht: ik leerde Angular beter kennen, zette het project lokaal op en verdiepte mij in de bestaande code en structuur. Daarna volgden de eerste CRUD-schermen, de QR-codefunctionaliteit en de migratie naar PostgreSQL.

Het grootste verschil zat in de manier waarop taken werden afgewerkt. Functionaliteiten waren niet altijd volledig klaar zodra ze technisch werkten. Vaak volgde er nadien nog feedback uit pull requests, overlegmomenten of demo's. Daardoor moesten onderdelen opnieuw aangepast worden, bijvoorbeeld door extra validaties toe te voegen, de lay-out te verbeteren of code eenvoudiger en consistenter te maken.

Daarnaast kwamen er tijdens de stage regelmatig nieuwe taken bij. Sommige onderdelen moesten beter aansluiten bij de Domain-Driven Design-structuur, waardoor refactoring nodig was. Ook uit meetings en demo's kwamen nieuwe inzichten naar voren, bijvoorbeeld wanneer een flow niet logisch genoeg bleek of een scherm gebruiksvriendelijker kon.

Hierdoor liep het project minder rechtlijnig dan oorspronkelijk voorzien. Ontwikkelen, testen, feedback verwerken en bijsturen liepen vaak door elkaar. Dat maakte de uitvoering minder voorspelbaar, maar zorgde er wel voor dat de applicatie beter aansloot bij de noden van Bricks & More en het ziekenhuispersoneel.

5.3. Toekomst van de applicatie

Na mijn stage wordt de applicatie verder klaargemaakt voor echt gebruik. Tijdens de ontwikkeling werd vooral gewerkt met testgegevens. Daardoor konden de schermen, reservaties en QR-codes al uitgewerkt en getest worden zonder dat er meteen echte gegevens van alle ziekenhuizen nodig waren.

Een eerste belangrijke test buiten de gewone testomgeving is het gebruik van de applicatie op Nerdland. In de laatste week van de stage werd hiervoor de juiste Excelimport uitgevoerd, zodat de applicatie klaarstond met de correcte gegevens. Door de toepassing daar in een echte context te gebruiken, kan waardevolle feedback verzameld worden over de werking, gebruiksvriendelijkheid en stabiliteit.

Na Nerdland kan de applicatie verder klaargemaakt worden voor gebruik in meerdere ziekenhuizen. Op dat moment zal de toepassing niet meer alleen met testgegevens werken, maar met echte bibliotheken, lego-sets en reservaties van verschillende locaties. Het blijft daarom belangrijk om de werking goed op te volgen en feedback te verzamelen van Bricks & More, ziekenhuispersoneel en andere betrokkenen.

Daarnaast moet de applicatie nog definitief online beschikbaar gemaakt worden. De testomgeving werd al gebruikt voor demo's en feedbackmomenten, maar voor het effectieve gebruik moet nog beslist worden welke URL gebruikt zal worden. Daarbij moet rekening gehouden worden met het correcte gebruik van de lego-naam en het lego-merk.

6. Besluit

Appies Legobib was voor mij een waardevolle stageopdracht omdat ik aan een echte applicatie heb gewerkt, met functionaliteiten die ook effectief gebruikt zullen worden. Dat zorgde ervoor dat ik niet alleen bezig was met code schrijven, maar ook met de vraag of mijn werk bruikbaar, duidelijk en betrouwbaar genoeg was voor de mensen die ermee moeten werken.

Het onderdeel waar ik het meest trots op ben, zijn de QR-codes. Ik werkte met SkiaSharp aan het genereren, afdrukken en personaliseren ervan. Dit was technisch interessant en tegelijk belangrijk voor de werking van de applicatie. Daarnaast werkte ik ook aan andere onderdelen, zoals beddenbeheer, collecties, lego-sets, PostgreSQL en het beschikbaar maken van de applicatie op de testomgeving via Dokploy.

Tijdens dit project heb ik veel bijgeleerd over .NET, Angular en PostgreSQL, maar vooral over hoe deze onderdelen samen één applicatie vormen. Ook Domain-Driven Design was hierbij belangrijk. Door dezelfde begrippen te gebruiken als aan de businesskant, zoals bed, bibliotheek, reservatie en lego-set, werd duidelijker waar bepaalde logica thuishoort en hoe de code beter aansluit bij de echte werking van Appies Legobib.

Een belangrijk leerpunt was dat een functionaliteit niet klaar is zodra ze technisch werkt. Ze moet ook getest worden, passen binnen de bestaande structuur en duidelijk zijn voor de gebruiker. Feedback uit pull requests, overlegmomenten en demo's heeft mij geholpen om kritischer naar mijn eigen werk te kijken en mijn oplossingen beter af te stemmen op het volledige project.

Ik sluit deze stage af met meer technische kennis en een beter beeld van hoe er professioneel aan een grotere applicatie wordt gewerkt. Ik heb geleerd om gestructureerder te werken, keuzes beter af te stemmen met het team en feedback bewuster te verwerken. Omdat Appies Legobib ook echt gebruikt zal worden en een duidelijk doel heeft, voelde mijn bijdrage extra waardevol aan.

LITERATUURLIJST

- Angular. (z.d.). *Angular*. Opgehaald van Angular: <https://angular.dev/>
- Appies vzw. (z.d.). *Appie's Legobib*. Opgehaald van Lowieke onze superheld: <https://www.lowieke.com/doelen/appieslegobib>
- Aspire. (z.d.). *Aspire documentation*. Opgehaald van <https://aspire.dev/>
- Beyens, J. (2026). Stagebegeleiding en feedback vanuit Thomas More.
- Deschryver, T. (2026). Begeleiding, technische feedback en codefeedback tijdens de stage bij dotNET lab.
- Domain Language. (z.d.). *Domain-Driven Design Reference*. Opgehaald van <https://www.domainlanguage.com/ddd/>
- Dudycz, O. (2025, Augustus 25). *My thoughts on Vertical Slices, CQRS, Semantic Diffusion and other fancy words*. Opgehaald van Architecture Weekly: <https://www.architecture-weekly.com/p/my-thoughts-on-vertical-slices-cqrs>
- Eccelpoel, B. v. (2026). Overleg en samenwerking tijdens de stage bij dotNET lab.
- FastEndpoints. (z.d.). *FastEndpoints*. Opgehaald van FastEndpoints: <https://fast-endpoints.com/>
- Fishchuk, I. (2025, oktober 2). *ASP.NET Core vs Node.js: Finding the best fit for your app development*. Opgehaald van Leobit: <https://leobit.com/blog/asp-dot-net-core-vs-node-js/>
- FluentValidation. (z.d.). *FluentValidation documentation*. Opgehaald van <https://docs.fluentvalidation.net/>
- GitHub. (2026). *GitHub Actions documentation*. Opgehaald van GitHub Docs: <https://docs.github.com/en/actions>
- Gommers, B. (2025, mei 14). *React, Angular of Vue.js: Welk front-end framework kies je?* Opgehaald van DataLeaf: <https://www.dataleaf.nl/blog/react-angular-of-vue-js-welk-front-end-framework-kies-je/>
- Infiniot. (z.d.). *Angular VS React*. Opgehaald van Infiniot: <https://infiniot.nl/blog/angular-vs-react/>
- Intersoft Consulting. (z.d.). *General Data Protection Regulation (GDPR)*. Opgehaald van GDPR-info.eu: <https://gdpr-info.eu/>
- Microsoft. (2024, November 12). *Overview of Entity Framework Core*. Opgehaald van Microsoft Learn: <https://learn.microsoft.com/en-us/ef/core/>
- Microsoft. (2025, Augustus 8). *Introduction to authorization in ASP.NET Core*. Opgehaald van Microsoft Learn: <https://learn.microsoft.com/en-us/aspnet/core/security/authorization/introduction?view=aspnetcore-10.0>
- More, B. &. (2026). Feedback tijdens de demo over de werking en verdere uitwerking van Appie's Legobib.
- Peeters, Y. (2026). Overleg en samenwerking tijdens de stage bij dotNET lab.
- Pluralsight. (z.d.). *Domain-Driven Design*. Opgehaald van Pluralsight: <https://www.pluralsight.com/paths/domain-driven-design>
- PostgreSQL Global Development Group. (z.d.). *About*. Opgehaald van PostgreSQL: <https://www.postgresql.org/>
- Ravoof, S. (2023, mei 22). *PostgreSQL vs SQL Server: 16 kritische verschillen*. Opgehaald van Kinsta: <https://kinsta.com/nl/blog/postgresql-vs-sql-server/>
- Rebrickable. (z.d.). *Find Lego Sets*. Opgehaald van Rebrickable: <https://rebrickable.com/sets/>
- Skiasharp. (z.d.). *SkiaSharp – Cross-Platform .NET Graphics Library*. Opgehaald van Skiasharp: <https://skiasharp.com/>
- Stigum, C. (2024, juni 10). *PostgreSQL Vs. MySQL: 7 key differences you need to know to choose the best option*. Opgehaald van CData Software: <https://www.cdata.com/blog/postgresql-vs-mysql>
- Sven, & Ellen. (2026). Feedback tijdens de demo over Appie's Legobib en de achtergrond van het project.
- Tailwind CSS. (z.d.). *Tailwind CSS*. Opgehaald van Tailwind CSS: <https://tailwindcss.com/>
- Ziekenhuis Geel. (2025, juli 8). *Appie's Legobib brengt troost en speelplezier op Planeet Goudgeel*. Opgehaald van Ziekenhuis Geel: <https://www.ziekenhuisgeel.be/nieuws/appies-legobib-brengt-troost-en-speelplezier-op-planeet-goudgeel>
- Zitadel. (z.d.). *Zitadel Documentation*. Opgehaald van <https://zitadel.com/docs>

BIJLAGEN

Legende:

Ontwikkelingsfase



Testen en feedback toepassen



Bijlage A: Initiële projectplanning

Taak	1	2	3	4	5	6	7	8	9	10	11	12	13
Angular leren & project leren kennen	Green	Blue	Light Orange	Light Orange	Light Orange	Light Orange	Light Orange	Light Orange	Light Orange	Light Orange	Light Orange	Light Orange	Light Orange
CRUD schermen ontwikkelen	Green	Green	Blue	Light Orange	Light Orange	Light Orange	Light Orange	Light Orange	Light Orange	Light Orange	Light Orange	Light Orange	Light Orange
QR-code integratie	Light Orange	Green	Blue	Blue	Light Orange	Light Orange	Light Orange	Light Orange	Light Orange	Light Orange	Light Orange	Light Orange	Light Orange
Onderzoek API integratie	Light Orange	Green	Blue	Light Orange	Light Orange	Light Orange	Light Orange	Light Orange	Light Orange	Light Orange	Light Orange	Light Orange	Light Orange
Migratie naar PostgreSQL	Light Orange	Light Orange	Green	Blue	Light Orange	Light Orange	Light Orange	Light Orange	Light Orange	Light Orange	Light Orange	Light Orange	Light Orange
Implementatie Domain Driven Design (DDD)	Light Orange	Light Orange	Green	Blue	Light Orange	Light Orange	Light Orange	Light Orange	Light Orange	Light Orange	Light Orange	Light Orange	Light Orange
Managementportaal functionaliteiten	Light Orange	Light Orange	Light Orange	Green	Green	Green	Blue	Blue	Light Orange	Light Orange	Light Orange	Light Orange	Light Orange
Bibliotheekportaal functionaliteiten	Light Orange	Light Orange	Light Orange	Light Orange	Green	Green	Green	Blue	Blue	Light Orange	Light Orange	Light Orange	Light Orange
Reservatieportaal functionaliteiten	Light Orange	Light Orange	Light Orange	Light Orange	Light Orange	Green	Green	Green	Blue	Blue	Light Orange	Light Orange	Light Orange
UX ontwikkeling	Light Orange	Light Orange	Light Orange	Light Orange	Light Orange	Light Orange	Green	Green	Green	Blue	Blue	Light Orange	Light Orange
Testen & feedback toepassen	Light Orange	Light Orange	Light Orange	Light Orange	Light Orange	Green	Green	Green	Green	Green	Green	Blue	Blue

Bijlage B: Werkelijke uitvoering van de projectplanning

Taak	1	2	3	4	5	6	7	8	9	10	11	12	13
Angular leren & project leren kennen													
CRUD schermen ontwikkelen													
QR-code integratie													
Onderzoek API integratie													
Migratie naar PostgreSQL													
Implementatie Domain Driven Design (DDD)													
Managementportaal functionaliteiten													
Bibliotheekportaal functionaliteiten													
Reservatieportaal functionaliteiten													
UX ontwikkeling													
Testen & feedback toepassen													