Software Design Document

Website Festival Travel System (FTS)

**Naam: Erkan Aslantas**

**Email:** [**s1213580@student.windesheim.nl**](mailto:s1213580@student.windesheim.nl)

Inhoudsopgave

[2. Inleiding 4](#_Toc194518403)

[2. Requirementanalyse 5](#_Toc194518404)

[3.1 Aanpak van de requirementsanalyse 5](#_Toc194518405)

[3.2 Functionele requirements 5](#_Toc194518406)

[3.3 Niet-functionele requirements 6](#_Toc194518407)

[3.4 Traceerbaarheid 6](#_Toc194518408)

[3. Functionele eisen 7](#_Toc194518409)

[4. Niet-functionele Eisen 9](#_Toc194518410)

[5.1 Advies op basis van Niet Functionele Eisen 9](#_Toc194518411)

[5. Technisch ontwerp 10](#_Toc194518412)

[6.1 Development environment 10](#_Toc194518413)

[6.2 Deployment environment 10](#_Toc194518414)

[6.3 ERD & Procesdiagram, Klasse diagram 11](#_Toc194518415)

[7. Functioneel ontwerp 12](#_Toc194518416)

[7.1 Wireframes 12](#_Toc194518417)

[Wireframe:Klantenregistratie / login 13](#_Toc194518418)

[Wireframe: Admin Klantenbeheer 14](#_Toc194518419)

[Wireframe: Reisboeking 16](#_Toc194518420)

[6. Testplan 23](#_Toc194518421)

[8.1 Testscenario's 23](#_Toc194518422)

[8.2 Acceptatiecriteria 24](#_Toc194518423)

[8.3 Test methoden 24](#_Toc194518424)

[8.4 Azure DevOps en Acceptatiecriteria 24](#_Toc194518425)

[7. Deployment en Implementatie 26](#_Toc194518426)

[9.1 Deployment naar Productie 26](#_Toc194518427)

[9.1.1 Stappen voor het Deployment Proces 26](#_Toc194518428)

[10. Evaluatie 27](#_Toc194518429)

[10.1 Samenvatting van het Project 27](#_Toc194518430)

[10.2 Resultaten en Conclusies 27](#_Toc194518431)

[10.3 Sterke en Zwakke Punten 27](#_Toc194518432)

[10.4 Verbeterpunten 27](#_Toc194518433)

[10.5 Reflectie 28](#_Toc194518434)

[10.6 Scrum-ervaring buiten het project 28](#_Toc194518435)

[10.7 Niet Voltooide Functionaliteit 28](#_Toc194518436)

[10.8 Redenen voor Niet-Voltooiing 28](#_Toc194518437)

[10.10 Conclusie 28](#_Toc194518438)

# 2. Inleiding

Dit document beschrijft het ontwerp en de ontwikkeling van het Festival Travel System (FTS), Een op maat gemaakte applicatie voor een gespecialiseerd busbedrijf dat reizen naar muziekfestivals door heel Europa gaat. En heeft als doel om de reiservaring voor festivalgangers te optimaliseren, terwijl de innerlijk werkwijze en hulpmiddelen van de busmaatschappij zorgvuldig en efficiënt worden beheerd.

Het FTS project heeft als volgt de kernfunctionaliteiten:

* **Klantenbeheer:** Registratie en beheer van klantgegevens en inzicht in hun reisgeschiedenis.
* **Reisplanning en boekingen:** Het plannen en boeken van reizen naar festivals, inclusief weergave van beschikbare reizen en festival data.
* **Puntensysteem:** Klanten kunnen punten verdienen bij boekingen en deze inwisselen voor nieuwe boeking betaald met de gespaarde punten.

# Requirementanalyse

In dit hoofdstuk wordt de analyse van de functionele en niet-functionele requirements beschreven die ten grondslag liggen aan de ontwikkeling van het FTS-project. De analyse is uitgevoerd op basis van de businesscase en omvat zowel functionele als technische aspecten om de haalbaarheid en de relevantie van de applicatie te waarborgen.

## 3.1 Aanpak van de requirementsanalyse

De requirementsanalyse is op een gestructureerde en systematische wijze uitgevoerd, waarbij de volgende stappen zijn genomen:

* Het identificeren van de belangrijkste belanghebbenden, waaronder de klanten van de busmaatschappij en de beheerders (administrators).
* User stories zijn opgesteld om de specifieke behoeften van de gebruikers vast te leggen. Deze zijn gegroepeerd in epics om een overzichtelijk en gestructureerd geheel te vormen.
* Voor elke user story zijn meetbare acceptatiecriteria opgesteld, zodat de eisen tijdens de ontwikkelingsfase kunnen worden geverifieerd.
* Een gedetailleerde analyse van de technische vereisten, zoals prestaties, schaalbaarheid, beveiliging en compatibiliteit.

## 3.2 Functionele requirements

* **Reisbeheer**: Administrators moeten reizen kunnen aanmaken, bewerken en verwijderen.
* **Klantbeheer**: Administrators moeten klanten kunnen toevoegen, bewerken en verwijderen, evenals toegang hebben tot reisgeschiedenis en puntensysteembeheer.
* **Reisgeschiedenis**: Administrators moeten inzicht hebben in de reisgeschiedenis van klanten, inclusief boekingsdatum, bestemming en kosten.
* **Boekingssysteem**: Klanten moeten reizen kunnen boeken en hun puntensaldo kunnen inwisselen voor kortingen of een nieuwe reis.
* **Puntensysteembeheer**: Administrators moeten de regels voor het verdienen en inwisselen van punten kunnen aanpassen.

## 3.3 Niet-functionele requirements

**Prestaties**:

* De applicatie moet binnen ~3-5 seconden laden op zowel desktop als mobiel.
* De database moet grote hoeveelheden gegevens efficiënt kunnen verwerken.

**Beveiliging**:

* Inputvalidatie moet worden uitgevoerd om SQL-injecties en XSS-aanvallen te voorkomen.
* De applicatie maakt gebruik van HTTPS en heeft een SSL-certificaat geïmplementeerd.

**Schaalbaarheid**:

* De applicatie moet uitbreidbaar zijn, zodat extra functionaliteiten of data eenvoudig kunnen worden toegevoegd zonder grote wijzigingen aan de code.

**Compatibiliteit**:

* De applicatie moet correct functioneren op moderne browsers, zoals Chrome, Firefox, Safari en Edge.

**Gebruiksvriendelijkheid**:

* Een intuïtieve interface die zowel door klanten als administrators eenvoudig te gebruiken is.

## 3.4 Traceerbaarheid

* Zijn alle user stories gekoppeld aan acceptatiecriteria.
* Wordt de voortgang van de implementatie en testen bijgehouden in Azure DevOps, waarin user stories, epics en taken zijn georganiseerd.
* Zijn de requirements gedocumenteerd in dit Software Design Document (SDD) en gekoppeld aan de businesscase.

# Functionele eisen

**EPIC:** Klanten

**Beschrijving:** Festivalgangers zich kunnen registreren, hun persoonlijke gegevens kunnen beheren en inzicht kunnen krijgen in hun reisgeschiedenis.  
**Userstory:**

1. Als festivalganger wil ik me kunnen registreren zodat ik een account kan aanmaken en toegang krijg tot de applicatie.
2. Als festivalganger wil ik mijn profiel kunnen beheren zodat ik mijn persoonlijke gegevens up-to-date kan houden.
3. Als festivalganger wil ik mijn reisgeschiedenis kunnen bekijken zodat ik inzicht heb in mijn gemaakte reizen.

**EPIC:** Reisplanning en Boekingen

**Beschrijving:** Het plannen en boeken van busreizen naar festivals. Gebruikers kunnen beschikbare reizen bekijken en de details inzien, zoals vertrektijd en opstaplocatie.  
**Userstory:**

1. Als festivalganger wil ik een busrit kunnen boeken zodat ik vervoer naar het festival heb.
2. Als festivalganger wil ik beschikbare reizen en festivaldata kunnen bekijken zodat ik een geschikte reis kan selecteren.
3. Als festivalganger wil ik de vertrektijd en locatie van mijn bus kunnen zien zodat ik op tijd bij de juiste opstapplaats ben.

**EPIC:** Puntensysteem

**Beschrijving:** Gebruikers punten kunnen opbouwen bij elke boeking en deze kunnen inwisselen voor voordelen zoals kortingen of VIP-toegang. Dit verhoogt de loyaliteit en betrokkenheid van gebruikers.  
**Userstory:**

1. Als festivalganger wil ik punten kunnen opbouwen bij elke boeking zodat ik deze kan gebruiken voor toekomstige voordelen.
2. Als festivalganger wil ik mijn puntensaldo kunnen inzien zodat ik weet hoeveel punten ik heb verzameld.
3. Als festivalganger wil ik mijn punten kunnen inwisselen voor kortingen of voordelen zodat ik extra waarde haal uit mijn boekingen.

**EPIC**: Dashboard   
**Beschrijving**: Het dashboard is een centraal overzicht voor de gebruiker met directe toegang tot de belangrijkste functies, zoals het bekijken van boekingen, beschikbare reizen, en een zoekfunctie om snel reizen te vinden.

**Userstory**:

1. Als festivalganger wil ik mijn dashboard kunnen openen zodat ik snel toegang heb tot mijn boekingen en beschikbare reizen.
2. Als festivalganger wil ik een zoekbalk kunnen gebruiken op het dashboard zodat ik snel specifieke reizen kan vinden.
3. Als festivalganger wil ik knoppen zien naar mijn boekingen en beschikbare reizen zodat ik snel naar deze secties kan navigeren.

**EPIC**: Admin

**Beschrijving**: Als administrator wil ik volledige toegang tot een beheerdersdashboard waarmee ik klanten, reisplanningen en het puntensysteem kan beheren, zodat de applicatie efficiënt functioneert.

**Userstory**:

**Klantenbeheer**

1. Als administrator wil ik klanten kunnen toevoegen, zodat ik nieuwe gebruikers in het systeem kan registreren.
2. Als administrator wil ik klanten kunnen bewerken, zodat ik onjuiste of gewijzigde klantinformatie kan aanpassen.
3. Als administrator wil ik klanten kunnen verwijderen, zodat inactieve of ongewenste accounts kunnen worden verwijderd.
4. Als administrator wil ik toegang hebben tot klantgegevens, zodat ik een overzicht heb van al hun belangrijke informatie.
5. Als administrator wil ik toegang hebben tot de reisgeschiedenis van klanten, zodat ik inzicht heb in hun reisgedrag en gemaakte boekingen.

**Reisplanning**

1. Als administrator wil ik nieuwe reizen kunnen toevoegen, zodat er altijd actuele reisopties beschikbaar zijn.
2. Als administrator wil ik reisgegevens kunnen bewerken, zodat ik onjuiste informatie kan corrigeren of aanpassen.
3. Als administrator wil ik reizen kunnen verwijderen, zodat verouderde of niet-meer-beschikbare reizen niet meer zichtbaar zijn.

**Puntensysteembeheer**:

1. Als administrator wil ik de regels voor het verdienen en inwisselen van punten kunnen aanpassen, zodat ik flexibel ben in het beloningssysteem.
2. Als administrator wil ik klantenpunten kunnen beheren en corrigeren, zodat fouten in het systeem kunnen worden opgelost.

**Toegang en Beveiliging:**

1. Als administrator wil ik dat alleen geautoriseerde gebruikers toegang hebben tot het beheerders dashboard, zodat onbevoegden geen wijzigingen kunnen aanbrengen.

# Niet-functionele Eisen

De niet functionele eisen zorgen ervoor dat de website naast functioneel ook optimaal presteert in termen van snelheid, veiligheid, gebruiksvriendelijkheid en compatibiliteit.

## 5.1 Advies op basis van Niet Functionele Eisen

**Hosting en Performance**:  
Om te voldoen aan de prestatie-eisen van de applicatie (laden binnen 3 seconden), wordt geadviseerd om te hosten via een platform met ingebouwde caching, zoals **Cloud86** of een andere Cloud gebaseerde hostingprovider. Deze platforms bieden snelle laadtijden en ondersteuning voor schaalbare webhosting, die essentieel is voor een applicatie met mogelijk veel gebruikers.  
De applicatie moet een Lighthouse Performance Score van minimaal 90 behalen en soepel werken met meerdere gelijktijdige gebruikers.

**Beveiliging**:  
Om de veiligheid van gebruikersgegevens, zoals klantinformatie en boekingsgegevens, te waarborgen:

* Gebruik een **SSL-certificaat** (bijvoorbeeld Let's Encrypt) voor HTTPS.
* Zorg voor input validatie en gebruik beveiligingstechnieken tegen SQL-injecties en XSS-aanvallen.
* Regelmatige back-ups van de database en beveiligingsupdates zijn een must.

**Beheerbaarheid**:  
Het admin dashboard moet een intuïtieve interface bieden waarmee beheerders:

* Klanten, boekingen en reisplanningen eenvoudig kunnen beheren.
* Punten voor klanten kunnen aanpassen. Dit vereenvoudigt het beheer zonder diepgaande technische kennis.

**Compatibiliteit**:  
De applicatie moet gebouwd worden met **Laravel** (PHP framework) en een MySQL-database. Frontend wordt ontwikkeld met moderne web standaarden zoals **HTML5, CSS3, en JavaScript**. Dit zorgt voor compatibiliteit met alle moderne browsers en devices (desktop en mobiel).

**Schaalbaarheid**:

* Gebruik een relationele database zoals **MySQL**, waarbij tabellen zoals customers, bus\_planning, bookings, en buses goed zijn genormaliseerd.
* Het systeem moet flexibel zijn om nieuwe functionaliteiten, zoals extra rapportages of nieuwe reisopties, eenvoudig toe te voegen zonder bestaande features te verstoren.

# Technisch ontwerp

## 6.1 Development environment

De ontwikkeling van de FTS (Festival Travel System) is uitgevoerd in een moderne en efficiënte ontwikkelomgeving om een snelle.

Hierbij zijn de volgende tools en versies gebruikt:

* + **Code editor**: PHPstorm voor het schrijven en beheren van de broncode.
  + **Laravel**: Voor de ontwikkeling van backend-functionaliteiten (versie 11).
  + **HTML5 en CSS3**: Voor de structuur en opmaak van de applicatie.
  + **JavaScript**: Voor interactieve functionaliteiten en frontend-verbeteringen (ES6+).
  + **Git en GitHub**: Voor versiebeheer (Git versie 2.39.0) en opslag van code in een centrale repository.
  + **PHP**: Voor server-side functionaliteiten (versie 8.3.11+).
  + **MySQL**: Voor het aanmaken en beheren van databases en het leggen van relaties tussen tabellen.
  + **Testing**: Lokale tests uitgevoerd in Google Chrome DevTools (versie 116+). En gebouwde worden met laravel PHPunit getest door de system.
  + **Figma**: Voor het ontwerpen van wireframes en mockups voor de user interface.
  + **Azure DevOps**: Voor het bijhouden van taken en het organiseren van de voortgang van het project.

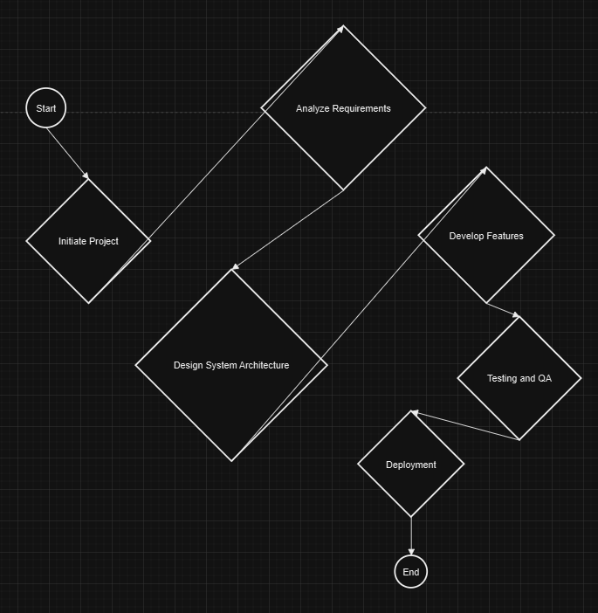
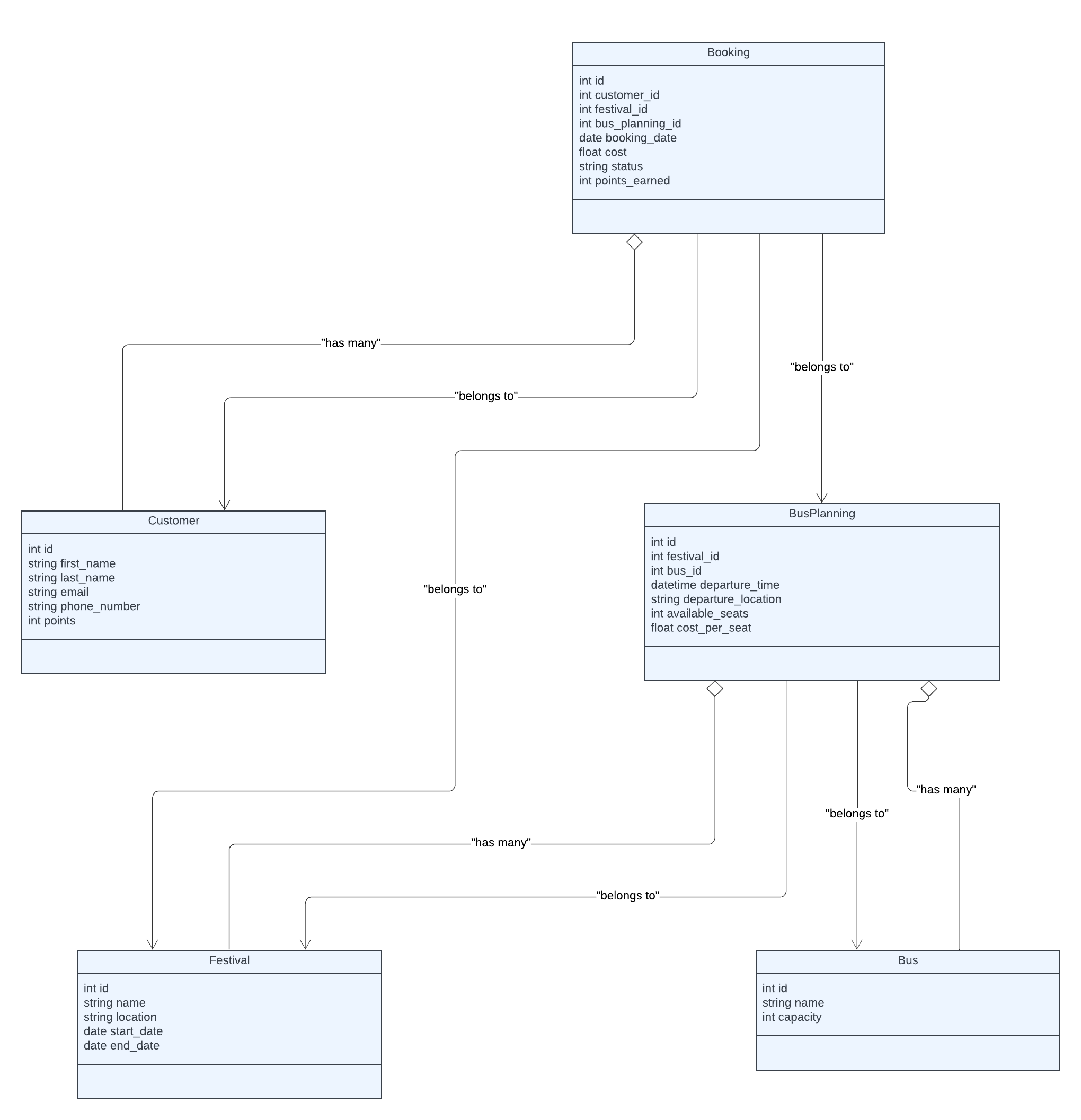
## 6.2 Deployment environment

Voor de implementatie van de website is een makkelijke en schaalbare hosting omgeving gebruikt. Dit zorgt ervoor dat de website snel en veilig kan worden uitgevoerd:

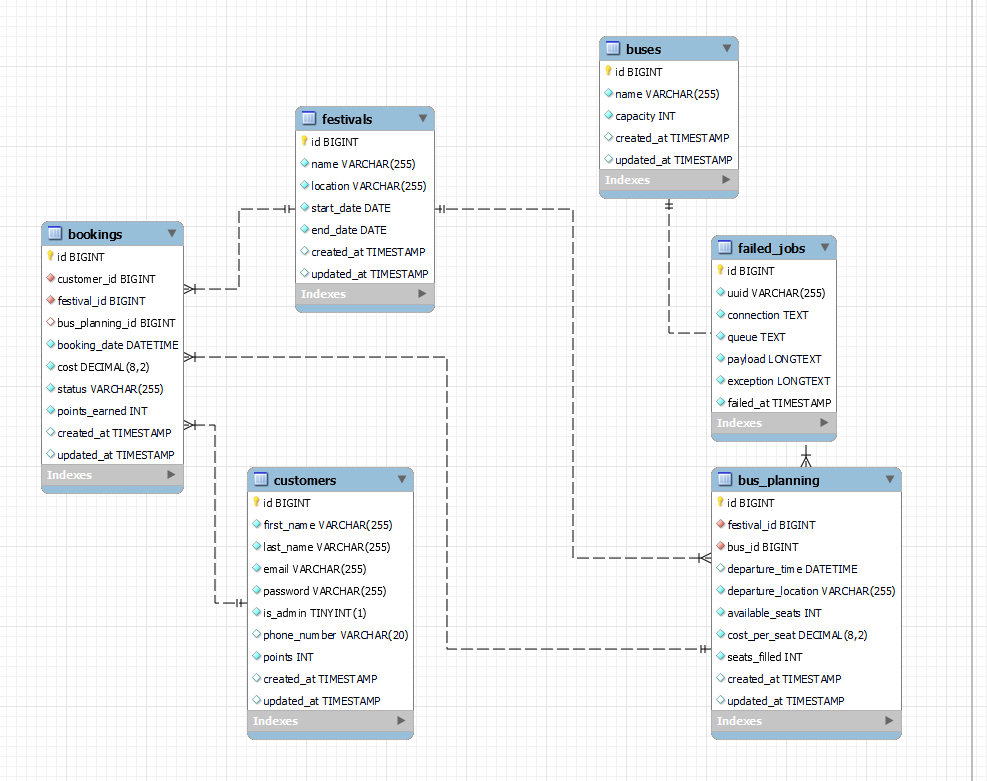
* **Hosting platform**: Cloud86 is gebruikt voor het hosten van de applicatie.
* **Webserver**: De applicatie draait op een server die **Laravel**, **PHP**, **MySQL**, en frontend-bestanden (HTML, CSS, JavaScript) ondersteunt.
* **Domeinnaam**: Een domeinnaam is geregistreerd en gekoppeld aan het hosting platform.
* **SSL-certificaat**: Een SSL-certificaat is geïnstalleerd om veilige HTTPS-verbindingen te garanderen en gebruikersdata te beschermen.
* **Browsersupport**: De applicatie is geoptimaliseerd voor moderne browsers zoals Chrome, Firefox, Safari en Edge.

## 6.3 ERD & Procesdiagram, Klasse diagram

Gemaakt met [draw.io](https://app.diagrams.net/)

[](https://app.diagrams.net/)

ERD gemaakt met MySQL workbench 8.0



# 7. Functioneel ontwerp

Het interface ontwerp is cruciaal voor de gebruikerservaring (UX) en gebruiksvriendelijkheid van de FTS (Festival Travel System) website.

## 7.1 Wireframes

Wireframes dienen als schematische blauwdrukken die de basisstructuur van de website tonen zonder grafische details. Ze helpen bij het plannen van de gebruikersinterface (UI) en zorgen ervoor dat alle functionaliteiten op logische plekken worden geplaatst.

### Wireframe:Klantenregistratie / login

* **Beschrijving**: In deze sectie combineren we twee gerelateerde schermen: het registratieformulier en het inlogscherm.
* **User Story**: "Als festivalganger wil ik me kunnen registreren zodat ik een account kan aanmaken en toegang krijg tot de applicatie."
* **Functionele Requirement (FR):** “Als festivalganger wil ik me kunnen registreren zodat ik een account kan aanmaken en toegang krijg tot de applicatie.”
* **Interacties:**

**Registratie pagina:**

1. Gebruiker vult de verplichte velden in (naam, e-mail, telefoonnummer, wachtwoord) en klikt op “Register.”
2. De gegevens worden via HTTPS naar de server gestuurd. De server valideert de invoer en maakt een nieuw account aan in de database.

**Login pagina:**

1. Gebruiker vult gebruikersnaam en wachtwoord in.
2. De server controleert of de gegevens kloppen (en of het account actief is).
3. Bij correcte inloggegevens krijgt de gebruiker toegang tot zijn persoonlijke dashboard omgeving.

* **Gebruikerservaring (UX) en Design Rationale:**

**Eenduidige labels**: Zowel bij registreren als inloggen krijgen de velden duidelijke namen (zoals ‘Username’, ‘Password’). Dat voorkomt verwarring.

**Consistente stijl:** Beide schermen hebben dezelfde lay-out en knoppen (Register/Login), zodat gebruikers direct begrijpen wat ze kunnen doen.

**Veilige afhandeling:** Door HTTPS en server-side validatie te noemen, toon je dat veiligheid een prioriteit is.

**Directe feedback:** Bij een fout (zoals een al bestaand e-mailadres of onjuist wachtwoord) verschijnt een rode foutmelding, zodat gebruikers weten wat er misging.

* **Wireframes:**

### Wireframe: Admin Klantenbeheer

**Beschrijving:**  
Dit overzichtsscherm toont een lijst met alle bestaande klanten in het systeem, inclusief naam, e-mail en telefoonnummer. Daarnaast zijn er bewerkingsknoppen ("Edit") en een verwijderknop ("Delete"). Als beheerder kun je vanuit dit overzicht ook nieuwe klanten registreren door op “Klant toevoegen” (of “Register”) te klikken.  
  
**Relevante User Storyies:**

* “Als administrator wil ik klanten kunnen toevoegen, zodat ik nieuwe gebruikers in het systeem kan registreren.”
* “Als administrator wil ik klanten kunnen bewerken, zodat ik onjuiste of gewijzigde klantinformatie kan aanpassen.”
* “Als administrator wil ik klanten kunnen verwijderen, zodat inactieve of ongewenste accounts kunnen worden verwijderd.”

**Functionele Requirement:**  
*Klantenbeheer – Administrators moeten klanten kunnen toevoegen, bewerken en verwijderen.*

**Interacties (Client-Server) per Actie:**

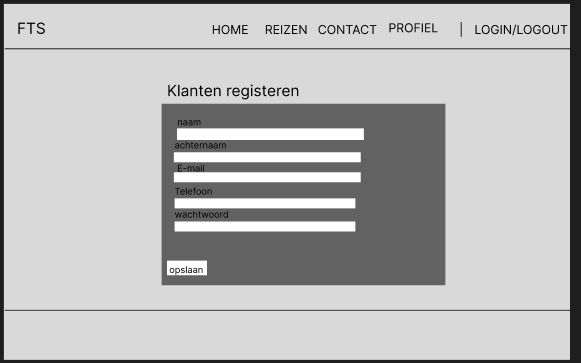
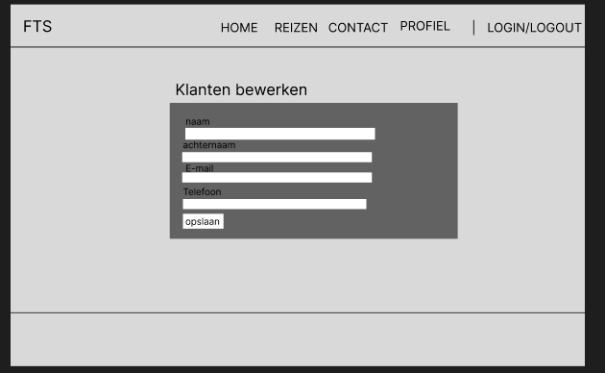
1. **Klant toevoegen:**
   * De administrator klikt op “Klant registreren” en vult het formulier in (naam, e-mail, telefoonnummer, wachtwoord).
   * De gegevens worden via HTTPS naar de server gestuurd, waar ze gevalideerd en in de database opgeslagen worden.
   * Bij geldige invoer ontvangt de admin een bevestiging en ziet hij/zij de nieuwe klant in de lijst.
2. **Klant bewerken:**
   * De administrator klikt op “Edit” naast de klant. Het systeem haalt de bestaande klantgegevens op en toont een formulier.
   * De beheerder past de gegevens aan (bv. nieuw telefoonnummer) en drukt op “Opslaan.”
   * De server valideert de wijzigingen, werkt de database bij en toont een bevestiging.
3. **Klant verwijderen:**
   * De administrator klikt op “Delete” bij een klant.
   * De server vraagt om een bevestiging (bijvoorbeeld een pop-up “Weet u zeker dat u deze klant wilt verwijderen?”).
   * Na bevestiging wordt de klant uit de database verwijderd en verdwijnt uit de lijst in het overzichtsscherm.

**Gebruikerservaring (UX) en Design Rationale:**

* **Overzichtelijkheid:** Door klanten in een tabel weer te geven met kolommen (naam, e-mail, telefoon, acties) kan de beheerder snel zien welke informatie relevant is.
* **Directe knoppen (‘Edit’, ‘Delete’):** Zichtbaar in dezelfde rij als de klantgegevens, zodat de beheerder niet hoeft te zoeken naar bewerkingsopties.
* **Bevestigingsmelding bij verwijderen:** Voorkomt dat de beheerder per ongeluk een klant verwijdert.
* **Formulieren voor invoer:** Eenduidige labels en verplichte velden zorgen voor minder fouten en heldere foutmeldingen bij ongeldige invoer.

**Wireframes:**





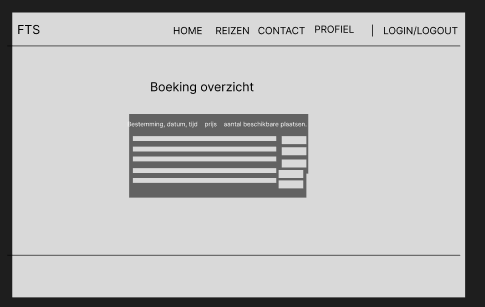
### Wireframe: Reisboeking

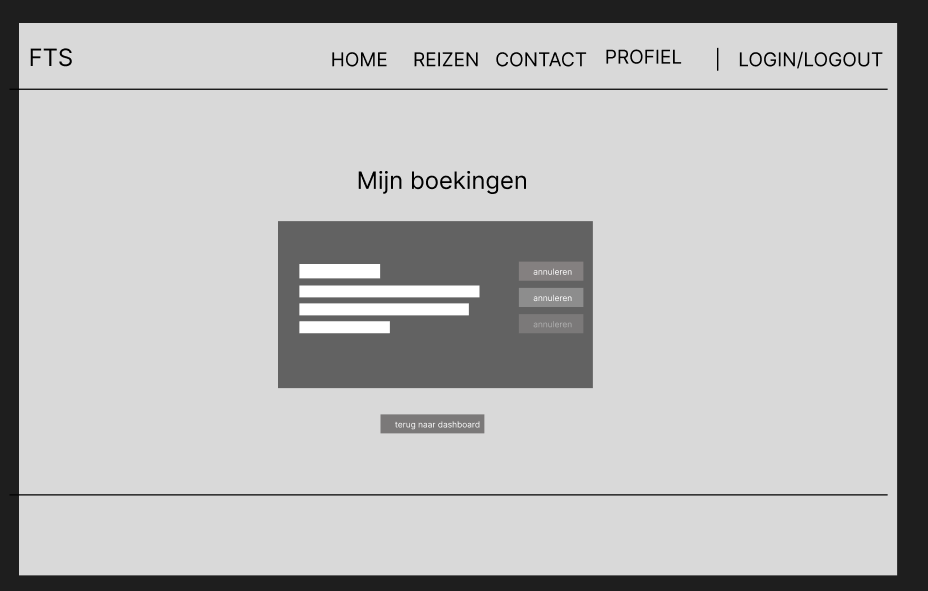
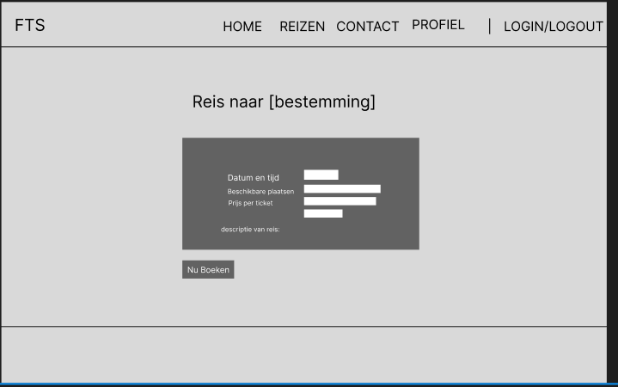
* **Beschrijving:** Deze wireframes tonen de pagina waar gebruikers een reis kunnen boeken. De gebruiker kan een bestemming selecteren, een vertrekdatum kiezen en het aantal stoelen opgeven. Na het voltooien van de boeking wordt hij/zij doorgestuurd naar de pagina “Mijn boekingen,” waar ook de mogelijkheid bestaat om een reis te annuleren.
* **User Story**: “Als festivalganger wil ik een busrit kunnen boeken zodat ik vervoer naar het festival heb.”
* **Functionele Requirement:**  
  *Reisplanning en boekingen – Klanten moeten reizen kunnen boeken en hun puntensaldo kunnen inwisselen voor kortingen of een nieuwe reis.*
* **Interacties:**
* **Bestemming en datum selecteren:** De gebruiker kiest een festivalbestemming uit een lijst en geeft een gewenste vertrekdatum op.
* **Aantal stoelen:** De gebruiker vult in hoeveel stoelen hij/zij wil boeken (1 stoel of meerdere).
* **Boek-knop:** Bij het klikken op “Boeken” stuurt de applicatie de gegevens via een beveiligde verbinding (HTTPS) naar de server.
* **Servervalidatie:** De server controleert of er nog voldoende stoelen beschikbaar zijn en verwerkt de boeking in de database.
* **Bevestiging & Afrekenen (optioneel met punten):**

Als het puntensysteem in gebruik is, kan de gebruiker eventueel een deel betalen met gespaarde punten.

* Na succesvolle afronding verschijnt een bevestiging en wordt de gebruiker doorgestuurd naar “Mijn boekingen.”
* Bij onvoldoende stoelen of andere fouten (bijv. ongeldige datum) toont de pagina een foutmelding.
* **Gebruikerservaring (UX) en Design Rationale:**
* **Eenvoudige invoer:** De belangrijkste gegevens (bestemming, datum, stoelen) staan bovenaan, zodat de gebruiker niet hoeft te scrollen.
* **Beschikbaarheid in real-time:** Eventuele feedback (bijv. “Nog 5 stoelen beschikbaar”) voorkomt frustratie en laat gebruikers weten of ze kunnen boeken.
* **Heldere foutmeldingen:** Indien het aantal stoelen te hoog is, verschijnt direct een foutmelding met uitleg (“Er zijn niet genoeg stoelen beschikbaar”).
* **Overzicht ‘Mijn Boekingen’:** Na het boeken ziet de gebruiker meteen een overzicht van al zijn/haar boekingen. Een “Annuleer”-knop is aanwezig om gemakkelijk een reis te annuleren als de plannen veranderen.

**Wireframes:**





### Wireframe: Admin Reizen & Buskoppeling

**Beschrijving:**Deze set wireframes laat zien hoe de administrator reizen en bussen kan beheren. Zo kan de admin nieuwe reizen toevoegen (met festivalnaam, locatie, start- en einddatum, prijs, beschikbaar aantal stoelen, en de bijbehorende bus), bestaande reizen bewerken of verwijderen, en nieuwe bussen aanmaken. Daarnaast is er een “Reizen overzicht” waarin alle bestaande reizen in een tabel staan, met knoppen om gegevens te wijzigen of te verwijderen.

**User Stories:**

* Als administrator wil ik reizen kunnen verwijderen, zodat verouderde of niet-meer-beschikbare reizen niet meer zichtbaar zijn.
* Als administrator wil ik reisgegevens kunnen bewerken, zodat ik onjuiste informatie kan corrigeren of aanpassen.
* Als administrator wil ik nieuwe reizen kunnen toevoegen, zodat er altijd actuele reisopties beschikbaar zijn.

**Functionele Requirement:***Reisplanning en boekingen – Administrators moeten reizen kunnen aanmaken, bewerken en verwijderen, en daarbij een bus selecteren.*

**Interacties:**

**Nieuwe bus toevoegen:**

* De admin klikt op “Nieuwe Bus Toevoegen” en vult de naam en capaciteit in.
* De gegevens worden via HTTPS naar de server gestuurd. De server valideert (bijvoorbeeld of de bus naam al bestaat) en slaat de nieuwe bus op in de database.
* De admin ziet een bevestiging, en de nieuwe bus wordt zichtbaar in een dropdown-lijst bij het aanmaken van reizen.

**Reis bewerken:**

* In het “Reizen overzicht” ziet de admin alle bestaande reizen in tabelvorm. Door op “Edit” of “Bewerken” te klikken, laadt de applicatie de bestaande gegevens van die reis in een formulier.
* De admin kan nu bv. de datum, bus, of prijs wijzigen. Na het klikken op “Opslaan” valideert de server de aanpassingen en werkt de database bij.
* De admin krijgt een bevestiging van de update; de tabel in “Reizen overzicht” wordt vernieuwd.

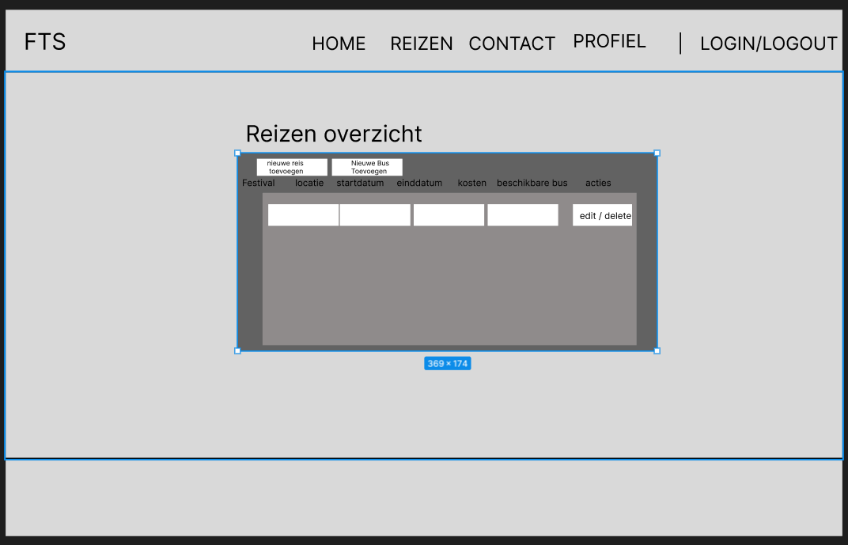
**Reis verwijderen:**

* Met een klik op “Delete” verwijdert de admin een reis uit de lijst. De server vraagt om een bevestiging (“Weet u zeker dat u deze reis wilt verwijderen?”).
* Na bevestiging wordt de reis uit de database gewist en verdwijnt uit het overzicht.

**Gebruikerservaring (UX) en Design Rationale:**

* **Uniforme lay-out:** Elk formulier gebruikt dezelfde stijl (labels, velden, knop “Opslaan”), zodat de admin consistentie ervaart en begrijpt waar de invoervelden voor bedoeld zijn.
* **Overzichtelijke tabellen:** Het “Reizen overzicht” toont alle essentiële kolommen (festivalnaam, startdatum, einddatum, bus, prijs per stoel), zodat de admin snel kan filteren, bewerken of verwijderen.
* **Dropdown-lijst voor bussen:** Dit voorkomt tikfouten en maakt het eenvoudig om een bus aan een reis te koppelen.
* **Validatie en foutmeldingen:** Als een veld leeg is of een datum onjuist is, verschijnt er direct een foutmelding zodat de admin begrijpt welke gegevens niet kloppen.

**Afbeelding met tekst, schermopname, Rechthoek, ontwerp

Door AI gegenereerde inhoud is mogelijk onjuist.Wireframes:**

### **Afbeelding met tekst, schermopname, software, ontwerp Door AI gegenereerde inhoud is mogelijk onjuist.Afbeelding met tekst, schermopname, Rechthoek, ontwerp**Wireframe: Dashboard

**Beschrijving:**  
Het dashboard is een centraal overzicht voor de gebruiker

* **Klant:** De wireframe toont knoppen om direct naar ‘Mijn Boekingen’ of ‘Beschikbare Reizen’ te gaan, plus een zoekbalk om naar specifieke reizen te zoeken. Onder de knoppen staat een beknopt overzicht van de laatste boeking(en) of het puntenaantal.
* **Admin:** In deze variant ziet de administrator knoppen zoals “Klanten Overzicht,” “Reizen Overzicht,” of “Bus Beheren,” zodat hij/zij snel kan navigeren naar de beheermodules.

**User Story:** “Als festivalganger wil ik mijn dashboard kunnen openen zodat ik snel toegang heb tot mijn boekingen en beschikbare reizen.”

**Functionele Requirement:** *Dashboard – Het dashboard is een centraal overzicht voor de gebruiker met directe toegang tot de belangrijkste functies.*

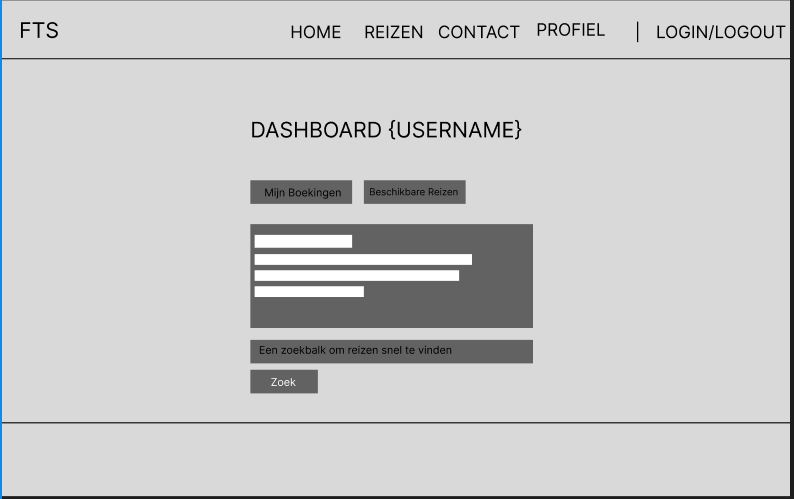
**Interacties:**

* **Dashboard laden:** Zodra de gebruiker inlogt, haalt de server uit de database op of de gebruiker een klant of admin is en toont het bijbehorende dashboard.
* **‘Mijn Boekingen’ knop (klant):**
  + Bij een klik, worden de boekingen van die gebruiker via een serveraanvraag opgehaald en in een tabel op de ‘Mijn Boekingen’-pagina weergegeven.
* **‘Beschikbare Reizen’ knop (klant):**
  + De klant wordt omleidt naar Beschikbare Reizen-pagina en ziet een lijst van aankomende reizen (naam festival, datum, aantal beschikbare stoelen) op basis van data uit de server.
* **Zoekbalk (klant):**
  + De invoer in de zoekbalk wordt verstuurd naar de server, die de query verwerkt en relevante reizen terugstuurt.
* **Knoppen in Admin Dashboard:**
  + Via “Klanten Overzicht” kan de admin de klantenlijst en bijbehorende bewerk-/verwijderfuncties oproepen en dat leidt naar de pagina die dat kan doen.
  + Bij “Reizen Overzicht” hetzelfde principe, maar dan voor reizen en bussen. De server haalt de gegevens op en toont deze in tabelvorm op een nieuwe pagina.

**Gebruikerservaring (UX) en Design Rationale:**

* **Snelle navigatie:** De knoppen staan bovenaan, zodat de gebruiker met één klik bij de meestgebruikte onderdelen is.
* **Rolafhankelijkheid:** Als de gebruiker geen admin is, ziet hij/zij geen admin-opties, wat verwarring voorkomt.
* **Zoekbalk (klantendashboard):** Helpt bij grote aantallen reizen, waardoor de gebruiksvriendelijkheid stijgt.
* **Consistente layout:** Zowel klant als admin krijgen een vergelijkbare structuur (knoppen, overzicht), zodat het systeem herkenbaar blijft.

Afbeelding met tekst, schermopname, scherm, software

Door AI gegenereerde inhoud is mogelijk onjuist.**Wireframes:**

### Wireframe: Admin Puntensysteem

* **Beschrijving:** Deze wireframe toont hoe een administrator het puntensysteem beheert. In een overzicht staan alle klanten vermeld met hun naam, e-mailadres en het huidige puntensaldo. De administrator kan hier nieuwe punten toekennen of punten afschrijven als dat nodig is (bijvoorbeeld bij fouten of speciale promoties).
* **User Story:** "Als festivalganger wil ik mijn punten kunnen inwisselen voor kortingen of voordelen zodat ik extra waarde haal uit mijn boekingen."
* **Functionele Requirement:** Puntensysteembeheer - Administrators moeten de regels voor het verdienen en inwisselen van punten kunnen aanpassen.
* **Interacties:**

**Puntenoverzicht:**

* + - De administrator ziet een tabel met alle klanten (naam, e-mail, huidig puntensaldo).
    - Via een zoekfunctie kan de admin snel een specifieke klant vinden.

**Punten Bewerken:**

* De admin klikt op “Edit” of “Bewerken” naast een klant. Er opent een scherm of pop-up met het huidige puntensaldo.
* Na het invullen van nieuwe waarde punten, drukt de admin op “Opslaan.”
* De invoer wordt via een beveiligde verbinding (HTTPS) naar de server gestuurd.
* De database wordt bijgewerkt en het nieuwe saldo verschijnt direct in het overzicht.

**Gebruikerservaring (UX) en Design Rationale:**

* **Eenvoudig Overzicht:** Een tabel met kolommen voor naam, e-mail en huidig puntensaldo, zodat de admin snel de status van elke klant ziet.
* **Heldere Bewerking:** Een aparte knop of pop-up om punten aan te passen, zodat de admin niet per ongeluk verkeerde gegevens wijzigt.
* **Foutvalidatie:** Bij invoer van ongeldige waarden (bijv. letters in plaats van cijfers of extreem hoge aantallen) geeft de applicatie een duidelijke foutmelding.

**Afbeelding met schermopname, tekst, software

Door AI gegenereerde inhoud is mogelijk onjuist.Wireframes:**

**Wireframe link:**

Link van gemaakte wireframes in Figma: https://www.figma.com/design/AjmjL1H6QdPBNq223J8Uwj/Untitled?node-id=0-1&t=9DL6RiCcXvnvQiBF-1

# Testplan

Het testplan beschrijft de aanpak voor het testen van de FTS (Festival Travel System) applicatie om ervoor te zorgen dat de functionaliteit, prestaties, gebruiksvriendelijkheid en compatibiliteit voldoen aan de gestelde eisen. In deze fase zijn testscenario's en acceptatiecriteria opgesteld om te bepalen wanneer de applicatie klaar is voor lancering. Voor elke feature/userstory is getest met behulp van PHPUnit, inclusief handmatige tests om de functionaliteit te waarborgen.

## 8.1 Testscenario's

Testscenario's zijn situaties waarin de website wordt getest op verschillende functionaliteiten en aspecten. Hier volgen enkele kern scenario’s die zullen worden getest:

**1. Klantenbeheer**

* **Toevoegen van klanten**:
  + Een nieuwe klant kan succesvol worden toegevoegd via het formulier.
  + Validatie zorgt ervoor dat alle verplichte velden correct worden ingevuld.
  + Na toevoeging wordt de klant weergegeven in de klantenlijst.
* **Bewerken van klantgegevens**:
  + Klantgegevens kunnen worden aangepast, inclusief naam, e-mail en telefoonnummer.
  + Validatie voorkomt dat dubbele e-mailadressen worden gebruikt.
* **Verwijderen van klanten**:
  + Een klant kan permanent uit de database worden verwijderd via een veilige actie met bevestiging.

**2. Puntenbeheer**

* **Punten aanpassen**:
  + Administrators kunnen de punten van klanten aanpassen via het puntenbeheer.
  + Validatie voorkomt negatieve nummers.

**3. Reizenbeheer**

* **Toevoegen van reizen**:
  + Nieuwe reizen kunnen worden aangemaakt met correcte gegevens zoals bestemming, kosten en stoelen.
  + Validatie voorkomt invoerfouten, zoals negatieve stoelaantallen of ontbrekende kosten.
* **Bewerken van reizen**:
  + Administrators kunnen bestaande reizen aanpassen om foutieve informatie te corrigeren.
* **Verwijderen van reizen**:
  + Niet meer relevante reizen kunnen permanent worden verwijderd.

**4. Reisgeschiedenis**

* **Overzicht van reisgeschiedenis per klant**:
  + Administrators kunnen de reisgeschiedenis van een specifieke klant inzien, inclusief datum, bestemming en kosten.
  + De lijst is overzichtelijk en gesorteerd op datum.

**5. Beveiliging**

* **Toegangscontrole**:
  + Alleen administrators hebben toegang tot het beheerdersdashboard.
  + Niet-ingelogde gebruikers worden omgeleid naar de loginpagina.

**6. Prestaties**

* **Laadtijd**:
  + Alle pagina's laden binnen ~3-5 seconden, ook bij grotere datasets.
* **Schaalbaarheid**:
  + De applicatie blijft stabiel bij een groeiend aantal klanten, reizen en boekingen.

**7. Compatibiliteit en responsiviteit**

* **Responsieve weergave**:
  + De applicatie is getest op desktop, tablet en mobiel.
  + Knoppen, formulieren en interactieve elementen functioneren correct op touchscreens.
* **Browsercompatibiliteit**:
  + De applicatie werkt correct in moderne browsers zoals Chrome, Firefox, Safari en Edge.

## 8.2 Acceptatiecriteria

De acceptatiecriteria bepalen wanneer een functionaliteit of component als voltooid en goedgekeurd wordt beschouwd. Deze criteria zijn als volgt:

**Functionele eisen:**

* Klanten kunnen worden toegevoegd, bewerkt en verwijderd via een intuïtieve interface.
* Reizen en puntenbeheer zijn volledig functioneel en voorzien van validatieregels.
* De reisgeschiedenis is inzichtelijk en bevat alle vereiste details.

**Prestaties:**

* De laadtijd van elke pagina is maximaal 3 seconden.
* De applicatie kan soepel omgaan met een groeiende database.

**Beveiliging:**

* Alleen geautoriseerde gebruikers hebben toegang tot gevoelige functionaliteiten.
* De applicatie maakt gebruik van HTTPS voor veilige communicatie.

**Gebruiksvriendelijkheid:**

* De interface is intuïtief en geschikt voor alle gebruikers, inclusief administrators.
* Formulieren en knoppen bevatten duidelijke feedback bij interactie

## 8.3 Test methoden

* **Handmatige tests**: Worden uitgevoerd om de functionaliteiten te testen vanuit het perspectief van de eindgebruiker. Hierbij wordt de website op verschillende apparaten en browsers getest.

## 8.4 Azure DevOps en Acceptatiecriteria

Voor de ontwikkeling van de FTS (Festival Travel System) applicatie is Azure DevOps gebruikt om het project te beheren. Binnen Azure DevOps zijn user stories en epics opgesteld, waarbij elk item voorzien is van duidelijke acceptatiecriteria. Deze acceptatiecriteria zijn gebruikt als leidraad tijdens de ontwikkeling en het testen van de applicatie.

Link naar de Azure Devops FTS project: <https://ictwf.visualstudio.com/2024_WDP_Erkan>

Per user story zijn de volgende stappen uitgevoerd in Azure DevOps:

1. **Definiëren van acceptatiecriteria**:  
   Elk item bevat een nauwkeurige beschrijving van wat er moet worden bereikt om te voldoen aan de gestelde eisen.
2. **Planning en toewijzing**:  
   User stories en epics zijn opgedeeld in kleinere taken en toegewezen aan specifieke iteraties.
3. **Testscenario's**:  
   Voor elk acceptatiecriterium zijn relevante testscenario's opgesteld en gekoppeld aan de betreffende user story. Deze scenario's omvatten zowel automatische tests met PHPUnit als handmatige tests.

Door gebruik te maken van Azure DevOps:

* De voortgang van het project effectief monitoren en rapporteren.
* Zorgen voor traceerbaarheid tussen user stories, taken, en hun bijbehorende tests.
* Garanderen dat alle functionaliteiten volledig getest en gevalideerd zijn voordat ze werden geïmplementeerd in de productieomgeving.

Deze aanpak heeft bijgedragen aan een gestructureerd ontwikkelproces waarin de focus lag op kwaliteit en het voldoen aan de behoeften van gebruikers en beheerders.



# Deployment en Implementatie

De deployment van de FTS (Festival Travel System) is een cruciaal onderdeel van het online zetten van de website. In dit hoofdstuk wordt beschreven hoe de website wordt geïmplementeerd in een productieomgeving, welke stappen zijn genomen voor een naadloze overgang van ontwikkeling naar productie, en welke tools, zoals Git voor versiebeheer, zijn gebruikt om de ontwikkeling en implementatie te efficiënt laten verlopen.

## 9.1 Deployment naar Productie

De website wordt gedeployed naar **Cloud86**, een betrouwbaar hostingplatform dat ondersteuning biedt voor statische websites. Ze biedt uitstekende prestaties en eenvoudig te configureren opties voor continue updates en beveiliging. Voor versiebeheer en continue integratie gebruik ik **Git** en **GitHub**. Hieronder volgt een overzicht van het volledige deployment proces:

### 9.1.1 Stappen voor het Deployment Proces

* 1. Tijdens de ontwikkeling wordt gebruikgemaakt van Git voor versiebeheer voor de website. Na elke update of toevoeging aan de website worden de wijzigingen gecommit naar een lokale Git-repository.

De broncode wordt vervolgens gepusht naar een externe repository op GitHub, zodat er een back-up is van alle wijzigingen en de code gemakkelijk kan worden gedeployed naar de productieomgeving.

*Command voorbeeld*:

git add .

git commit -m "Beschrijving van de wijziging"

git push origin main

* 1. **Koppeling met Cloud86 Hosting**

Cloud86 wordt gebruikt als het hostingplatform voor de website. Dit platform ondersteunt statische websites, waardoor de configuratie eenvoudig en betrouwbaar is. Er wordt gebruikgemaakt van FTP/SFTP of een Git-integratie om de bestanden van de GitHub-repository naar de Cloud86-server te uploaden.

* 1. **FTP/SFTP Upload**

Worden de websitebestanden handmatig geüpload via een FTP-client zoals FileZilla of via de webinterface van Cloud86.

* 1. **Configuratie van de Hostingomgeving**

De domeinnaam wordt gekoppeld aan de Cloud86-server.

SSL-certificaten worden geconfigureerd om een beveiligde HTTPS-verbinding te garanderen.

* 1. **Beveiligingsinstellingen**

SSL/TLS-beveiliging wordt ingesteld om ervoor te zorgen dat de verbinding tussen de gebruiker en de website versleuteld is.

* 1. **Continue Updates**

De hostingomgeving kan automatisch worden bijgewerkt door middel van een geautomatiseerd deployment proces of handmatig via de FTP/SFTP-client.

* 1. **Na de deployment**

Na de livegang zal een monitoring tool zoals UptimeRobot of New Relic worden gebruikt om de prestaties en uptime te monitoren.

# 10. Evaluatie

# 10.1 Samenvatting van het Project

Dit project richtte zich op het ontwikkelen van een **Festival Travel System (FTS)**, een gebruiksvriendelijke en responsieve webapplicatie voor het beheren van festivals, klanten en boekingen. Het hoofddoel was om een systeem te creëren dat zowel de klanten als de administrators ondersteunt. Voor klanten biedt het platform een eenvoudige manier om reizen te boeken en hun puntensaldo te beheren. Administrators kunnen klantgegevens, boekingen, reisplanningen en puntensystemen beheren. Het systeem is ontworpen om efficiëntie te verbeteren en het gebruikersgemak te maximaliseren voor zowel klanten als medewerkers van het bedrijf.

# 10.2 Resultaten en Conclusies

De ontwikkeling van FTS heeft geresulteerd in een krachtig en volledig functioneel systeem dat aan de gestelde eisen voldoet. Belangrijke resultaten zijn:

* **Prestaties**: Het systeem biedt snelle responstijden en laadtijden die voldoen aan moderne web standaarden.
* **Functionaliteit**: Alle gedefinieerde user stories en functionaliteiten zijn succesvol geïmplementeerd, inclusief reisgeschiedenisbeheer, puntensystemen, en boekingsbeheer.
* **Gebruikerstevredenheid**: Interne feedback van testers (zoals medestudenten) was positief, met complimenten over het gebruiksgemak en de duidelijke structuur van het admin-paneel.
* **Fout reductie**: Door gebruik te maken van uitgebreide PHPUnit-tests en handmatige testen zijn bugs vroegtijdig geïdentificeerd en opgelost, wat resulteerde in een stabiel systeem.

# 10.3 Sterke en Zwakke Punten

**Sterke punten**:

* Het gebruik van Laravel als framework heeft een gestructureerde en efficiënte ontwikkeling mogelijk gemaakt.
* Een uitgebreide requirementsanalyse heeft ervoor gezorgd dat alle functionele en niet-functionele eisen volledig werden begrepen en geïmplementeerd.
* Het project werd effectief beheerd met behulp van **Azure DevOps**, waarin user stories, epics en taken werden bijgehouden.
* Zowel automatische tests (PHPUnit) als handmatige testen zijn consistent uitgevoerd voor kwaliteitsborging.

**Zwakke punten**:

* In het begin was enige verwarring over het plannen van taken, wat leidde tot een vertraging in het begin van de ontwikkelingsfase.
* Het opstellen van complexe relaties tussen tabellen en modellen, zoals tussen klanten en boekingen, kostte meer tijd dan verwacht.
* Het vragen van externe feedback gebeurde pas later in het proces, wat eerder had kunnen plaatsvinden om mogelijke verbeterpunten eerder te ontdekken.

# 10.4 Verbeterpunten

Als ik dit project opnieuw zou uitvoeren:

* Zou ik meer tijd besteden aan de **initiële planning** om ervoor te zorgen dat alle taken realistisch en haalbaar zijn binnen de tijdslijnen.
* Zou ik meer tussentijdse feedback van gebruikers of stakeholders verzamelen om de gebruikerservaring verder te optimaliseren.
* Zou ik diepgaandere performance-optimalisaties uitvoeren, zoals database-indexering en caching, om het systeem nog efficiënter te maken bij grote datasets.
* Meer aandacht besteden aan **documentatie**, zodat toekomstige ontwikkelaars eenvoudiger kunnen bijdragen aan het project.

# 10.5 Reflectie

Het proces van het bouwen van de FTS-applicatie was zowel uitdagend als leerzaam. Tijdens het ontwikkelen heb ik gewerkt met **Model-View-Controller (MVC)**, relationele databases en URL-routing, wat mijn technische vaardigheden aanzienlijk heeft verbeterd. Bovendien heb ik geleerd hoe belangrijk een duidelijke structuur is voor een applicatie van deze schaal. Het succesvol testen van elke user story gaf me vertrouwen in mijn probleemoplossend vermogen en mijn vermogen om complexe systemen te bouwen.

# 10.6 Scrum-ervaring buiten het project

Tijdens de ontwikkelingsfase van mijn project heb ik deelgenomen aan wekelijkse Scrum-sessies met een groep programmeurs. Hoewel deze sessies niet direct verband hielden met mijn project, hebben ze mij waardevolle inzichten gegeven in agile werkmethoden, en het effectief prioriteren van taken. Deze ervaringen hebben mijn vermogen versterkt om mijn eigen project beter te plannen en te structureren. Door deel te nemen aan dagelijkse stand-ups, sprintplanningen en retrospectives, heb ik geleerd hoe belangrijk het is om kleine, haalbare doelen te stellen en continu feedback te verwerken. Hoewel mijn project individueel is uitgevoerd, hebben deze vaardigheden mijn aanpak en werkwijze positief beïnvloed.

# 10.7 Niet Voltooide Functionaliteit

Hoewel het merendeel van de user stories en functionaliteiten met succes zijn geïmplementeerd, was het helaas niet mogelijk om de volgende user story binnen de gestelde deadline te realiseren:

**Puntensysteembeheer:**  
"Als administrator wil ik de regels voor het verdienen en inwisselen van punten kunnen aanpassen, zodat ik flexibel ben in het beloningssysteem."

Deze functionaliteit vereiste het opzetten van een extra beheersmodule waarin administrators de parameters voor het verdienen en inwisselen van punten konden configureren. Ondanks een goede start was er onvoldoende tijd om de logica, de interface en de benodigde testen volledig af te ronden.

# 10.8 Redenen voor Niet-Voltooiing

* **Complexiteit:** Het flexibel maken van beloningsregels vereiste extra logica in het systeem, zoals het definiëren van dynamische parameters en bijbehorende validaties.
* **Prioriteitstelling:** Tijdens de ontwikkeling zijn andere kernfunctionaliteiten met hogere prioriteit, zoals reisgeschiedenis en klantbeheer, eerst afgerond.
* **Tijdgebrek:** De implementatie van de overige functionaliteiten en het uitgebreide testproces lieten beperkte tijd over voor deze user story.

#### 10.9 Verbeterplan

Als vervolgactie zou ik adviseren om deze functionaliteit als prioriteit te beschouwen in een toekomstige iteratie. Dit zou de flexibiliteit en waarde van het systeem verder vergroten met name voor administrators die een dynamisch beloningssysteem nodig hebben. Door gebruik te maken van de huidige structuur, zoals de admin-panelen en modellen kan de functionaliteit relatief eenvoudig worden uitgebreid en geïmplementeerd.

Deze ervaring benadrukt het belang van een strakke tijdsplanning en het stellen van prioriteiten bij het ontwikkelen van een project binnen een beperkte tijdslijn.

# 10.10 Conclusie

Het project is succesvol afgerond en heeft meeste doelstellingen bereikt. FTS biedt een stabiele, gebruiksvriendelijke en schaalbare oplossing die de administratie van festivals en klantinteracties aanzienlijk verbetert. De opgedane kennis en vaardigheden tijdens dit project vormen een waardevolle basis voor toekomstige softwareontwikkelingsprojecten. Het project heeft niet alleen bijgedragen aan mijn technische vaardigheden, maar ook aan mijn persoonlijke ontwikkeling, met name op het gebied van projectbeheer en samenwerking.