TECHNISCH ONTWERP

Project casino

INHOUDSOPGAVE

INLEIDING

Dit technisch ontwerp beschrijft de architectuur en opzet van een web-based casino applicatie. Het doel van de applicatie is om een veilige en schaalbare omgeving te creëren waar spelers diverse casinospellen kunnen spelen. De focus ligt op het intern beheren van data en functionaliteiten, waarbij we in dit document de belangrijkste technische aspecten van de applicatie bespreken.

De volgende onderwerpen komen aan bod:

1. **Databasestructuur**: Hier wordt de opzet van de database behandeld, inclusief de opslag van speldata, gebruikersinformatie, en transacties. We bespreken hoe de relaties tussen de verschillende tabellen zijn opgezet en hoe dit bijdraagt aan een efficiënte verwerking en beheer van de data.

2. **API's**: In dit hoofdstuk wordt ingegaan op de API-laag binnen de applicatie, waarin we de verschillende interne endpoints bespreken die communicatie tussen de frontend en backend faciliteren. Hierbij bespreken we hoe deze API's zijn ontworpen voor prestatieoptimalisatie en beveiliging, ondanks dat we geen gebruik maken van externe API-diensten.

3. **Programmeertaal en Framework**: We geven een toelichting op de gekozen programmeertaal en het webframework voor de applicatie. Er wordt uitgelegd waarom deze technologieën geschikt zijn voor het ontwikkelen van een casino-applicatie en hoe ze bijdragen aan een schaalbare en veilige infrastructuur.

4. **Apparaatspecificaties**: Dit deel geeft een overzicht van de minimum systeemeisen en aanbevolen specificaties voor het draaien van de applicatie. Hierbij worden aspecten zoals processorgebruik, geheugencapaciteit en netwerkvereisten besproken, zodat de applicatie optimaal kan presteren op verschillende apparaten.

5. **Beveiliging**: Beveiliging is een cruciaal aspect van de applicatie, vooral gezien de financiële aard van de diensten. In dit hoofdstuk richten we ons op de implementatie van beveiligingsmaatregelen, zoals de OWASP-richtlijnen. We bespreken hoe de applicatie is ontworpen om spelers te beschermen tegen hackingpogingen en frauduleuze activiteiten, waarbij specifieke aandacht wordt besteed aan de veiligheid van gevoelige data en transacties.

I. DATABASESTRUCTUUR

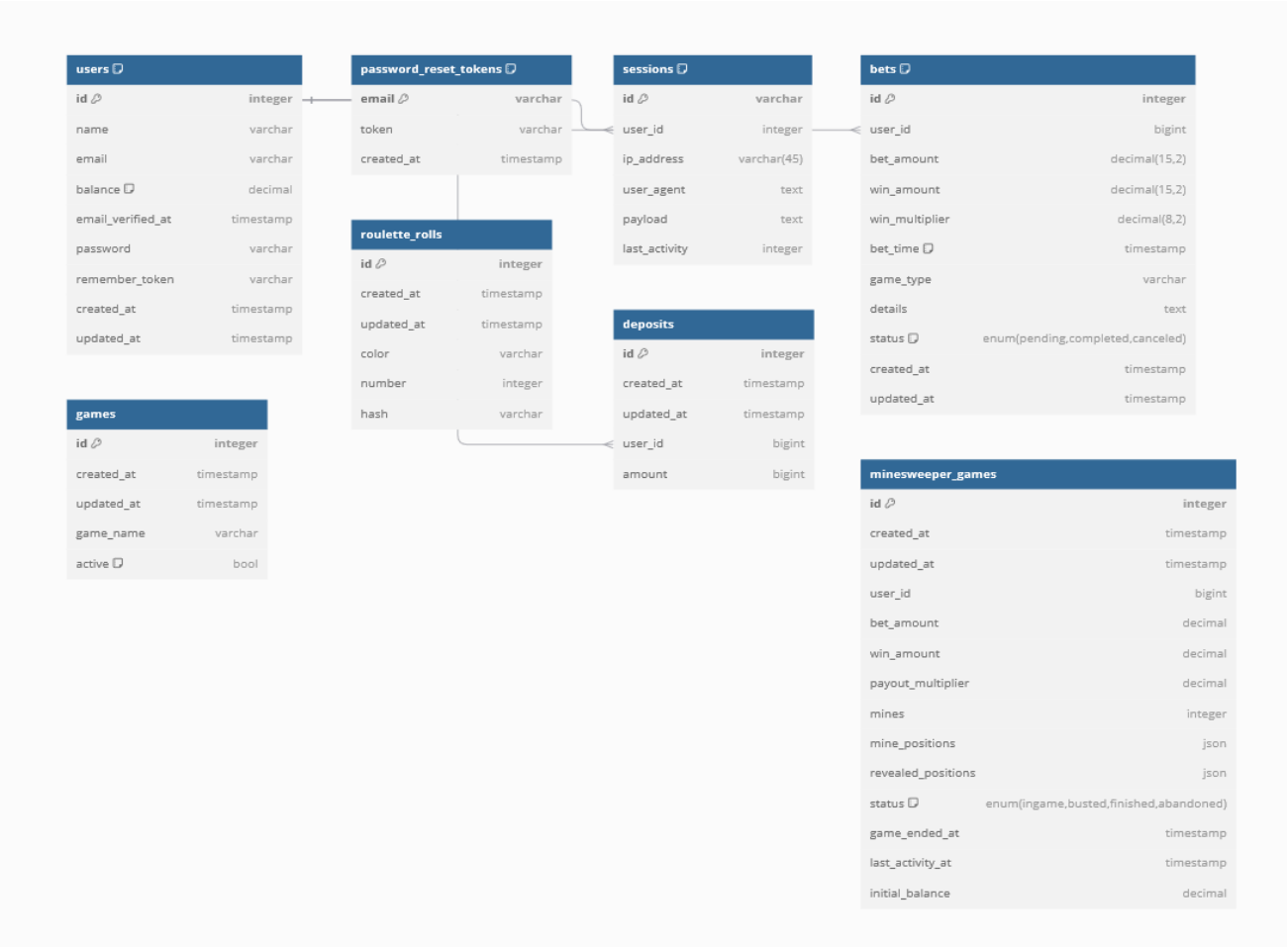
De databasestructuur van onze casino applicatie is ontworpen om efficiënt en schaalbaar te zijn, rekening houdend met de specifieke behoeften van gebruikers en de interacties binnen de applicatie. Deze structuur omvat verschillende tabellen die elk hun eigen unieke rol vervullen, zoals het beheren van gebruikers, bets, en andere essentiële gegevens.

In de diagram weergegeven in onze Entity-Relationship Diagram (ERD), zijn de belangrijkste entiteiten en hun onderlinge relaties in detail vastgelegd. We hebben onder andere de volgende tabellen:

- **Users**: Bevat informatie over geregistreerde gebruikers, waaronder naam, e-mailadres en verificatiestatus.
- **Bets**: Houdt gegevens bij over de inzetten die door gebruikers worden geplaatst, inclusief het bedrag, de status, en de bijbehorende speltype.
- **Games**: Iedere game heeft haar eigen tabel, dit komt omdat iedere game gelogd moet worden, ook zodat gebruikers vorige resultaten kunnen inzien.
- **Password Reset Tokens**: Beheert tokens voor het resetten van wachtwoorden om de veiligheid van gebruikersaccounts te waarborgen.
- **Sessions**: Houdt actieve sessies bij, inclusief gebruikersinformatie en laatste activiteit.

Deze structuur maakt gebruik van geneste relaties en is geoptimaliseerd voor snelle gegevensopvragingen door middel van indexen. Dit stelt ons in staat om gebruikerservaring te verbeteren en een snellere reactietijd te garanderen tijdens het spelen.

https://dbdiagram.io/d/66fe4872fb079c7ebd2fed3a


users
id integer
name varchar
email varchar
balance decimal
email_verified_at timestamp
password varchar
remember_token varchar
created_at timestamp
updated_at timestamp
password_reset_tokens
email varchar
token varchar
created_at timestamp
sessions
id varchar
user_id integer
ip_address varchar(45)
user_agent text
payload text
last_activity integer
bets
id integer
user_id bigint
bet_amount decimal(15,2)
win_amount decimal(15,2)
win_multiplier decimal(8,2)
bet_time timestamp
game_type varchar
details text
status enum(pending,completed,canceled)
created_at timestamp
updated_at timestamp
roulette_rolls
id integer
created_at timestamp
updated_at timestamp
color varchar
number integer
hash varchar
deposits
id integer
created_at timestamp
updated_at timestamp
user_id bigint
amount bigint
games
id integer
created_at timestamp
updated_at timestamp
game_name varchar
active bool
minesweeper_games
id integer
created_at timestamp
updated_at timestamp
user_id bigint
bet_amount decimal
win_amount decimal
payout_multiplier decimal
mines integer
mine_positions json
revealed_positions json
status enum(ingame,busted,finished,abandoned)
game_ended_at timestamp
last_activity_at timestamp
initial_balance decimal

2. API'S

In onze casino applicatie maken we uitsluitend gebruik van interne API's die zijn ontworpen om de functionaliteit en gebruikerservaring te optimaliseren. Deze interne API's zijn verantwoordelijk voor verschillende kritieke taken, zoals het plaatsen van een bet wanneer een gebruiker op een knop klikt. Dit zorgt ervoor dat de communicatie tussen de front-end en back-end soepel en efficiënt verloopt.

Door gebruik te maken van interne API's kunnen we volledige controle houden over de datastromen en logica binnen onze applicatie. Dit draagt bij aan de beveiliging en integriteit van de gegevens, aangezien we geen externe API's integreren die potentiële risico's met zich mee kunnen brengen.

Onze aanpak zorgt ervoor dat alle gegevensverwerking en -communicatie binnen ons eigen systeem blijven, wat ons in staat stelt om optimale prestaties en een veilige omgeving voor onze gebruikers te waarborgen.

3. TAAL EN FRAMEWORKS

Onze casino applicatie is ontwikkeld met behulp van een krachtige combinatie van programmeertalen en frameworks die zijn afgestemd op de behoeften van zowel de back-end als de front-end. De gebruikte programmeertalen zijn:

- **PHP**: Voor de back-end logica en server-side verwerking.
- **HTML**: Voor de structuur van de webpagina's.
- **JavaScript**: Voor dynamische interacties en client-side functionaliteit.

Voor de back-end maken we gebruik van het Laravel framework, dat ons in staat stelt om robuuste, schaalbare en onderhoudbare webapplicaties te ontwikkelen. Laravel biedt een uitgebreide set tools en functies die ons helpen bij het beheren van de applicatie-architectuur en het waarborgen van de veiligheid van gebruikersdata.

Aan de front-end kant gebruiken we React, een populaire JavaScript-bibliotheek die ons helpt bij het bouwen van interactieve gebruikersinterfaces. React maakt het mogelijk om component-gebaseerde applicaties te creëren die efficiënt reageren op gebruikersinvoer.

Om de styling van onze applicatie te stroomlijnen en een responsieve gebruikersinterface te creëren, maken we gebruik van Tailwind CSS. Deze utility-first CSS-bibliotheek stelt ons in staat om snel en effectief te stylen zonder dat we tijd hoeven te besteden aan het schrijven van op maat gemaakte CSS.

Daarnaast implementeren we WebSockets om een live dataverbinding te realiseren tussen de back-end en de front-end. Dit stelt ons in staat om gebruikers in real-time te informeren over hun bets, waardoor de gebruikerservaring wordt verbeterd en gebruikers direct feedback krijgen over hun interacties binnen de applicatie.

4. APPARAATSPECIFICATIES

Om een optimale gebruikerservaring te garanderen en de prestaties van onze casino applicatie te waarborgen, hebben we bepaalde apparaatspecificaties gedefinieerd voor onze tech stack. Hieronder staan de minimumvereisten en aanbevolen specificaties voor de back-end:

Minimale Vereisten

- **Besturingssysteem**: Linux (bijvoorbeeld Ubuntu 20.04 of later) of Windows (Windows 10 of later)
- **Webserver**: Nginx of Apache
- **PHP**: Versie 8.0 of hoger
- **Database**: MySQL of PostgreSQL, versie 5.7 of hoger
- **Geheugen**: Minimaal 2 GB RAM
- **CPU**: Dual-core processor met een kloksnelheid van minimaal 2.0 GHz
- **Schijfruimte**: Minimaal 20 GB vrije schijfruimte

Aanbevolen Specificaties

- **Besturingssysteem**: Linux (bijvoorbeeld Ubuntu 22.04 of later) of Windows (Windows 10 of later)
- **Webserver**: Nginx (met PHP-FPM) of Apache met mod_rewrite
- **PHP**: Versie 8.1 of hoger
- **Database**: MySQL of PostgreSQL, versie 8.0 of hoger
- **Geheugen**: Minimaal 4 GB RAM (bij voorkeur 8 GB of meer voor betere prestaties)
- **CPU**: Quad-core processor met een kloksnelheid van minimaal 2.5 GHz
- **Schijfruimte**: Minimaal 50 GB vrije schijfruimte (afhankelijk van de hoeveelheid data en logbestanden)

Door te voldoen aan deze vereisten kunnen we ervoor zorgen dat de back-end van onze casino applicatie efficiënt draait en dat de gebruikers een soepele ervaring hebben, zelfs tijdens piekbelasting.

5. OWASP

De beveiliging van onze casino applicatie is van het grootste belang, en we volgen de OWASP (Open Web Application Security Project) richtlijnen om ervoor te zorgen dat ons systeem bestand is tegen mogelijke aanvallen en manipulaties. De architectuur van de applicatie is zo ontworpen dat deze op geen enkele manier gemanipuleerd kan worden. Hieronder worden de belangrijkste OWASP-principes beschreven die we in onze applicatie toepassen.

1. Inadequate Authentication
Onze applicatie maakt gebruik van een robuust authenticatiesysteem. Gebruikers moeten sterke wachtwoorden instellen en kunnen twee-factor-authenticatie (2FA) inschakelen om extra beveiliging toe te voegen aan hun accounts. Wachtwoorden worden veilig opgeslagen met hashingmethoden, zoals bcrypt.

2. Inadequate Session Management
We zorgen ervoor dat sessies veilig worden beheerd. Sessies hebben een beperkte levensduur en worden automatisch beëindigd na een periode van inactiviteit. Dit vermindert het risico op sessieovername.

3. Insecure Direct Object References
Toegang tot gevoelige gegevens is strikt gereguleerd. Gebruikers kunnen alleen hun eigen gegevens bekijken en bewerken. Pogingen om gegevens van andere gebruikers te benaderen worden actief gecontroleerd en geblokkeerd.

4. Cross-Site Scripting (XSS)
Alle invoer van gebruikers wordt gevalideerd en ontsmet voordat deze op de client wordt weergegeven. Dit voorkomt dat kwaadwillenden scripts kunnen injecteren die de gegevens van andere gebruikers kunnen stelen of de interface kunnen manipuleren.

5. Cross-Site Request Forgery (CSRF)
We implementeren CSRF-tokens om ervoor te zorgen dat alle kritieke acties die door gebruikers worden uitgevoerd, zoals het plaatsen van inzetten, alleen kunnen plaatsvinden na een geldige verificatie. Dit voorkomt ongewenste acties op gebruikersaccounts.

6. Security Misconfiguration
Onze servers en applicaties zijn zorgvuldig geconfigureerd om beveiligingslekken te minimaliseren. We houden onze software up-to-date met de nieuwste beveiligingspatches en -updates en voeren regelmatig beveiligingsaudits uit.

7. Insecure Cryptographic Storage
Gevoelige gegevens, zoals wachtwoorden en financiële informatie, worden veilig opgeslagen met moderne encryptie-algoritmen. Hierdoor zijn deze gegevens beschermd, zelfs als de database zou worden gecompromitteerd.

8. Insufficient Logging & Monitoring
We implementeren uitgebreide logging en monitoring om verdachte activiteiten in de gaten te houden. Dit stelt ons in staat om snel in te grijpen bij ongebruikelijke of kwaadaardige activiteiten.

9. Unvalidated Redirects and Forwards
Onze applicatie controleert en valideert alle omleidingen en doorstuurverzoeken om te voorkomen dat gebruikers naar onveilige of kwaadaardige locaties worden geleid.

Manipulatie van Spelresultaten

Alle spelresultaten worden vanaf het moment van plaatsing van de inzet op de server bepaald. De client is alleen verantwoordelijk voor het tonen van animaties en het weergeven van informatie. Dit zorgt ervoor dat de integriteit van de spellen gewaarborgd blijft en dat spelers er zeker van kunnen zijn dat de uitkomsten eerlijk zijn.

Beveiliging tegen API-misbruik

In het geval dat iemand probeert de client te misbruiken, zal dit alleen negatieve gevolgen hebben voor die specifieke gebruiker. Ons systeem is zo ontworpen dat het controleert of een gebruiker daadwerkelijk een inzet heeft geplaatst voordat er verdere acties kunnen plaatsvinden. Dit voorkomt misbruik van onze interne API's en beschermt de belangen van alle gebruikers.

Veilige Technologie

Laravel, ons gekozen backend-framework, is community-ondersteund en wordt wereldwijd gebruikt door grote bedrijven. Het framework biedt robuuste beveiligingsmechanismen die ons helpen bij het beschermen van gebruikersgegevens en het voorkomen van veelvoorkomende kwetsbaarheden, zoals SQL-injecties, cross-site scripting (XSS) en cross-site request forgery (CSRF).

Door deze maatregelen te implementeren, zorgen we ervoor dat onze casino applicatie niet