**Technisch Ontwerp – Bioscoopketen Website & Ticketsysteem**

***Versie:* 1.0  
*Datum:* 8 september 2025  
*Team:* Groep 3 — Rody (PL & DB), Jasper (CSS), Bas (HTML), Levi (ticketsysteem)**

**1. Samenvatting**

**We bouwen een multi-tenant webplatform voor een bioscoopketen met:**

* **Beheersysteem (HQ-only, inloggen vereist) voor Filmgegevens, Vertoningsadministratie, Gebruikers & Rechten, en optioneel Kaartverkoopadministratie.**
* **Website vestegingen (landing met vestigingen en aanbevolen films).**

**Vestiging-websites met filmagenda, filmpagina’s en kaartbestelsysteem met stoelkeuze en betalingen (iDEAL, creditcard, Bioscoopbon).**

* **Externe bron: TheMovieDB (TMDB) API voor filmdata (posters, cast, info).**

**Via hoofdkantoor**

**2. Architectuur**

**Patroon: 3-laags (Presentation ↔ Application/API ↔ Data) met duidelijke scheiding tussen Admin (HQ) en Publiek (Vestigingen).**

* **Frontend: HTML5, CSS3 (Jasper), JS (ES6). Voor componenten: lichte vanilla JS of Vue 3 (optioneel) voor stoelkaart en filter UI.**
* **Backend: nog niet bekend**
* **Database: MySQL 8 (InnoDB, utf8mb4, strict mode).**
* **Cache/Queue: Redis voor caching (posters, queries) en jobs (TMDB sync, rapporten).**
* **Authenticatie: Laravel Sanctum (sessie-cookie voor web) + policies/roles.**
* **Betalingen: Provider zoals Mollie of Stripe (iDEAL, creditcard). Abstraheren via PaymentService.**
* **Bestanden/Media: S3-compatibele storage (bijv. MinIO in dev, S3 in prod) voor posters/trailers (indien gehost), anders hotlink/URL van TMDB volgens hun policies.**
* **Logging/Monitoring: Monolog → files + centrale aggregator (bijv. Sentry).**
* **Deployment: GitHub → CI (GitHub Actions) → staging → productie via zero-downtime (Laravel Vapor/Forge of Docker + ECS).**

**2.1 Contextdiagram**

* **TMDB API → FilmMetadataService → DB (Films, Genres, Personen)**
* **Payment Provider ↔ PaymentWebhookController ↔ Orders/Tickets**
* **Vestigingen ↔ Public API (Films, Speeltijden, Tickets)**
* **HQ Medewerkers ↔ Admin UI ↔ Admin API**

**3. Security & Privacy**

* **RBAC: Rollen: *Admin*, *Editor*, *Viewer* met fijnmazige permissies per menu-optie (filmgegevens, vertoningsadministratie, gebruikers, rapportage).**
* **Auth: Email+password (bcrypt/argon2id), sessions, CSRF-protectie, brute-force throttling.**
* **Inputvalidatie: server- en client-side.**
* **Gegevensminimalisatie: bewaar alleen noodzakelijke persoonsgegevens (naam, e-mail, aankoopreferentie). Geen kaartgegevens server-side (tokenized via PSP).**
* **AVG/GDPR: privacyverklaring, bewaartermijnen tickets ≤ 24m, logging DPIA-light.**
* **Rate limiting: API 60 req/min per IP/Token.**
* **Transport: overal HTTPS (HSTS).**

**4. Niet-functionele eisen (NFR)**

* **Performance: < 200 ms p95 voor veelgevraagde caches (filmoverzicht), < 1 s voor uncached detail.**
* **Beschikbaarheid: 99,5% (schoolprojectnorm); graceful degradation bij PSP of TMDB-storing.**
* **Schaalbaarheid: horizontale scaling via Docker/containers, cache-first.**
* **Accessibility: WCAG 2.1 AA (contrast, toetsenbord, aria-labels).**
* **SEO: server-side rendered pages, met OpenGraph/Twitter cards.**

**5. Data Model & ERD**

**5.1 Tabellen (hoofdlijnen)**

**venues (vestigingen)**

* **id (PK), name, slug, address, city, lat, lng, phone, email, timezone**

**films**

* **id (PK), tmdb\_id (unique), title, overview, runtime\_min, release\_date, min\_price, poster\_url, backdrop\_url, trailer\_url, rating**

**genres**

* **id, name**

**film\_genre (pivot)**

* **film\_id (FK), genre\_id (FK)**

**people**

* **id, tmdb\_id, name, profile\_url, role\_type (actor/director)**

**film\_person (pivot)**

* **film\_id, person\_id, character (nullable)**

**screens (zalen per vestiging)**

* **id, venue\_id (FK), name, seat\_map\_json (grid/SVG), capacity**

**showtimes (vertoningen)**

* **id, film\_id (FK), venue\_id (FK), screen\_id (FK), starts\_at (datetime), ends\_at (datetime), base\_price, status (planned/on\_sale/cancelled)**

**users**

* **id, name, email (unique), password\_hash, role (admin/editor/viewer), venue\_id (nullable voor HQ)**

**permissions**

* **id, key (e.g. films.read), description**

**user\_permission (pivot)**

* **user\_id, permission\_id**

**orders**

* **id, code (public ref), email, first\_name, last\_name, total\_amount, status (pending/paid/cancelled/refunded), payment\_provider, payment\_ref, created\_at**

**tickets**

* **id, order\_id (FK), showtime\_id (FK), seat\_label, price, qr\_code, status (valid/refunded)**

**sales\_summaries (optioneel dagelijks)**

* **id, venue\_id, film\_id, date, total\_tickets, total\_revenue**

**audit\_logs**

* **id, user\_id, action, entity\_type, entity\_id, meta\_json, created\_at**

**api\_jobs**

* **id, type (tmdb.sync), payload\_json, status, attempts, run\_at, finished\_at**

**ERD (beschrijving): Films ↔ Genres (N:N), Films ↔ People (N:N). Venues hebben Screens; Screens hebben Showtimes (voor Films). Orders bevatten Tickets; Tickets verwijzen naar Showtimes. Users hebben Permissions en zijn óf HQ (venue\_id null) óf gekoppeld aan een Venue.**

**5.2 Indexering**

* **films(tmdb\_id), showtimes(venue\_id, starts\_at), tickets(showtime\_id), orders(code), fulltext op films.title, films.overview.**

**6. Integraties**

**6.1 TMDB API**

* **Endpoints: /movie/{id}, /movie/{id}/credits, /search/movie, images config.**
* **Service: FilmMetadataService met caching (24h) en fallback naar DB.**
* **Flow:**
  1. **Admin zoekt film (autocomplete op TMDB).**
  2. **Selectie → fetch details + credits → persist in DB.**
  3. **Poster/backdrop URL’s en trailers (YouTube key) opslaan.**
* **Rate limiting & keys: .env TMDB\_API\_KEY; gebruik backoff en job-queue.**

**6.2 Payment Provider (Mollie/Stripe)**

* **Checkout: POST /api/public/orders → Payment intent/session aanmaken.**
* **Webhook: POST /api/webhooks/payment → verifieer signatuur, update order/tickets.**
* **Return URL: /bestelling/gelukt?order=CODE.**

**7. API Ontwerp**

**7.1 Public API (vestiging/website)**

* **GET /api/public/venues – lijst vestigingen**
* **GET /api/public/venues/{slug} – details vestiging**
* **GET /api/public/venues/{slug}/films?date=YYYY-MM-DD&filter= – draait deze week**
* **GET /api/public/films/{id} – film detail (met showtimes per venue)**
* **GET /api/public/showtimes/{id} – detail vertoning (prijzen, zaal)**
* **POST /api/public/orders – maak order (validatie data, prijsberekening)**
* **POST /api/public/orders/{code}/reserve-seats – lock seats (optimistic locking, TTL)**
* **POST /api/public/orders/{code}/confirm – start betaling**
* **GET /api/public/orders/{code} – status**

**7.2 Admin API (HQ)**

* **Auth vereist**
* **GET/POST/PUT/DELETE /api/admin/films**
* **GET/POST/PUT/DELETE /api/admin/showtimes**
* **GET/POST/PUT/DELETE /api/admin/users**
* **GET /api/admin/reports/sales?venue=&film=&from=&to=**
* **RBAC via Laravel Policies (bijv. FilmPolicy@update).**

**8. Frontend Ontwerp**

**8.1 Hoofdkantoor-site**

* **Home: vestigingen dropdown, aanbevolen films, nieuwsblokken.**
* **Navigatie: fixed top bar; logo → home; footer met (Werken bij, FAQ, Vestigingen, Contact, Voorwaarden, Privacy, Socials).**

**8.2 Vestiging-site**

* **Home: korte vestiging-info, draaiende films (week vooruit), filters (datum, vandaag, verwacht).**
* **Filmagenda: filter UI (checkbox/radio), categorie dropdown; klik → film detail.**
* **Filmpagina: lijst + doorklik naar informatiepagina.**
* **Informatiepagina film: poster, beschrijving, genre(s), lengte, land, releasedatum, regisseur, acteurs (met foto), kijkwijzer-iconen, trailer (iframe), CTA “Tickets”.**
* **Bestelpagina: 5-staps flow; validaties; dynamische prijsberekening; iDEAL/creditcard/bon.**

**8.3 Stoelkaart (Seat Map)**

* **Representatie: seat\_map\_json met rijen/kolommen en labels; rendering als SVG/Canvas.**
* **Selectieflow: beschikbaar → geselecteerd → bevestigd; disabled voor bezet.**
* **Locking: server lock per stoel (Redis key met TTL, bijv. 5 min).**

**9. Business Logica**

* **Minimale prijs per film is ondergrens voor base\_price in Showtimes; validatie in Admin UI.**
* **Prijsklassen: uitbreiding mogelijk via price\_tiers (optioneel).**
* **Annuleren: tickets alleen voor pending/paid via policy + PSP refund API (indien ondersteund).**
* **Rapportage (nice-to-have): dagelijkse ETL job → sales\_summaries.**

**10. Validatie & Foutenafhandeling**

* **Server-side validation messages (Laravel FormRequests).**
* **Gestandaardiseerde error body {error: code, message, details}.**
* **404’s voor onbekende film/vert.**
* **PSP timeouts → uservriendelijke melding + retry.**

**11. QA & Testplan**

* **Unit tests: Services (FilmMetadataService, PaymentService), Policies.**
* **Feature tests: bestelflow, auth, RBAC.**
* **Browser tests (Dusk/Playwright): stoelselectie, filters, formulieren.**
* **Performance tests: showtimes list (100+), seat-lock concurrency.**
* **Accessibility audits: Lighthouse/axe.**

**12. CI/CD & Omgevingen**

* **Branches: main (prod), develop (staging), feature branches feat/\*.**
* **Pipelines: lint (PHPStan, ESLint), tests, build assets (Vite), migrations, deploy.**
* **Omgevingen:**
  + ***Dev*: lokale Docker (PHP-FPM, Nginx, MySQL, Redis, Mailhog).**
  + ***Staging*: seed data, test-PSP keys, TMDB sandbox.**
  + ***Prod*: HPA, S3, RDS, real PSP keys.**

**13. Configuratie (.env keys)**

* **APP\_ENV, APP\_KEY, APP\_URL**
* **DB\_HOST, DB\_PORT, DB\_DATABASE, DB\_USERNAME, DB\_PASSWORD**
* **REDIS\_HOST, REDIS\_PORT**
* **TMDB\_API\_KEY**
* **PAYMENT\_PROVIDER={mollie|stripe}**
* **MOLLIE\_KEY of STRIPE\_KEY, webhook secrets**
* **FILESYSTEM\_DISK={s3|local}, S3 creds**

**14. Migraties (voorbeeld)**

**15. UX & Componenten (indicatief)**

* **Buttons/Inputs:** focus states, aria, form-errors.
* **Filters:** chips/radio; actieve filter met checkmark in cirkel.
* **Trailers:** YouTube iframe met consent (cookie banner).

**16. Planning & Deliverables (koppeling)**

* **8 sep 14:00**: Technisch ontwerp + repo + backlog opgeleverd.
* **10 sep 12:00**: Database-ontwerp + migraties gereed.
* **24 sep**: Opleverdocument, procesverslag, demo-staging.
* **25 sep**: Eindpresentatie.

**17. Openstaande keuzes**

* **PSP keuze** (Mollie vs Stripe).
* **Frontend framework** (vanilla vs Vue 3 alleen voor seat map).
* **Hosting** (Laravel Forge vs Docker/K8s schoolvoorziening).