



ANALYSE DE LA STRUCTURE - CALLASTAR

Date d'analyse: 27 décembre 2024

Repository: <https://github.com/StreallyX/callastar>

Répertoire local: `/home/ubuntu/callastar`



1. ARCHITECTURE GÉNÉRALE

1.1 Stack Technique

Framework Principal:

- **Next.js 14.2.32** avec App Router (architecture moderne)
- **TypeScript 5.9.3** pour le typage statique
- **React 18.2.0**

Backend/Base de données:

- **Prisma 6.7.0** (ORM)
- **PostgreSQL** (base de données)
- **NextAuth 4.24.11** (authentification)

Paiements & Transferts:

- **Stripe 20.1.0** (SDK serveur)
- **@stripe/stripe-js 8.6.0** (SDK client)
- **@stripe/react-stripe-js 5.4.1** (composants React)
- **Architecture:** Destination Charges + Stripe Connect

Vidéo:

- **Daily.co** (@daily-co/daily-js 0.85.0)

UI/UX:

- **Tailwind CSS 3.3.3**
- **Radix UI** (composants accessibles)
- **Framer Motion** (animations)
- **React Hook Form 7.53.0** + **Zod 4.2.1** (validation)
- **Lucide React** (icônes)

Services Externes:

- **Resend** (emails)
 - **Google Analytics** (analytics)
 - **AWS S3** (stockage fichiers)
-

2. STRUCTURE DU PROJET

```

callastar/
├── app/
│   ├── api/
│   │   ├── auth/
│   │   ├── book/
│   │   ├── call/
│   │   ├── creators/
│   │   ├── dashboard/
│   │   │   ├── admin/
│   │   │   ├── creator/
│   │   │   └── user/
│   │   ├── globals.css
│   │   ├── layout.tsx
│   │   └── page.tsx
│   ├── components/
│   │   ├── admin/
│   │   ├── ui/
│   │   ├── NotificationBell.tsx
│   │   ├── call-request-dialog.tsx
│   │   ├── creator-card.tsx
│   │   ├── navbar.tsx
│   │   ├── providers.tsx
│   │   └── theme-provider.tsx
│   ├── hooks/
│   │   ├── use-auth.ts
│   │   └── use-toast.ts
│   └── lib/
│       ├── analytics.ts
│       ├── auth.ts
│       ├── calendar.ts
│       ├── creator-debt.ts
│       ├── currency-converter.ts
│       ├── currency-utils.ts
│       ├── daily.ts
│       ├── db.ts
│       ├── email.ts
│       ├── logger.ts
│       ├── nextauth.ts
│       ├── notifications.ts
│       ├── payout-eligibility.ts
│       ├── payout-validation.ts
│       ├── settings.ts
│       ├── stripe-account-validator.ts
│       ├── stripe.ts
│       ├── types.ts
│       └── utils.ts
├── prisma/
│   ├── migrations/
│   ├── schema.prisma
│   └── seed.ts
├── scripts/
├── tests/

```

Next.js App Router
 # Routes API (backend)
 # Pages authentication
 # Réservation d'appels
 # Interface d'appel vidéo
 # Pages publiques créateurs
 # Tableaux de bord
 # Dashboard admin
 # Dashboard créateur
 # Dashboard utilisateur
 # Styles globaux
 # Layout racine
 # Page d'accueil

Composants React
 # Composants admin
 # Composants UI réutilisables
 # Cloche notifications
 # Dialog demandes d'appel
 # Carte créateur
 # Barre de navigation
 # Providers (React Query, etc.)
 # Provider de thème





























Custom React hooks
 # Hook authentication
 # Hook notifications toast

Utilitaires et logique métier
 # Google Analytics
 # Authentification
 # Gestion calendrier
 # Gestion dettes créateurs
 # Conversion devises
 # Utilitaires devises
 # Daily.co (vidéo)
 # Client Prisma
 # Envoi emails
 # Logging
 # Configuration NextAuth
 # Système notifications
 # Éligibilité payouts
 # Validation payouts
 # Paramètres plateforme
 # Validation compte Stripe
 # Configuration Stripe
 # Types TypeScript
 # Utilitaires divers

Prisma ORM
 # Migrations **base** de données
 # Schéma de la **base** de données
 # Script de seed

Scripts utilitaires

Tests

	<code>fee-calculation-test.ts</code>	
	<code>separate-charges-transfers-test.ts</code>	
		
	<code>docs/</code>	 Documentation
	<code>PAYOUT_SYSTEM.md</code>	
	<code>STRIPE_CONNECT_EMBEDDED_ONBOARDING.md</code>	
		
	<code>public/</code>	 Fichiers statiques
	<code>favicon.svg</code>	
	<code>og-image.png</code>	
	<code>robots.txt</code>	
		
	<code>.env.example</code>	 Exemple variables d'environnement
	<code>next.config.js</code>	 Configuration Next.js
	<code>package.json</code>	 Dépendances
	<code>prisma/schema.prisma</code>	 Schéma base de données
	<code>tailwind.config.ts</code>	 Configuration Tailwind
	<code>tsconfig.json</code>	 Configuration TypeScript
	<code>vercel.json</code>	 Configuration déploiement

3. ROUTES API (Backend)

3.1 Authentification (`/api/auth/`)

- `POST /api/auth/login` - Connexion
- `POST /api/auth/logout` - Déconnexion
- `POST /api/auth/signin` - Inscription (OAuth)
- `GET /api/auth/me` - Utilisateur actuel
- `POST /api/auth/change-password` - Changer mot de passe
- `POST /api/auth/update-profile` - Mettre à jour profil
- `GET /api/auth/csrf` - Token CSRF
- `GET /api/auth/providers` - Providers OAuth
- `[...nextauth]` - Routes NextAuth

3.2 Utilisateurs & Inscription (`/api/`)

- `POST /api/signup` - Inscription

3.3 Créateurs (`/api/creators/`)

- `GET /api/creators` - Liste créateurs
- `GET /api/creators/[id]` - Détails créateur
- `GET /api/creators/profile` - Profil créateur actuel
- `PUT /api/creators/profile` - Mettre à jour profil
- `GET /api/creators/balance` - Balance créateur
- `GET /api/creators/payout-settings` - Paramètres payout
- `PUT /api/creators/payout-settings` - Mettre à jour paramètres
- `GET /api/creators/payouts/history` - Historique payouts
- `POST /api/creators/payouts/request` - Demander payout

3.4 Offres d'Appels (`/api/call-offers/`)

- `GET /api/call-offers` - Liste offres

- `POST /api/call-offers` - Créer offre
- `GET /api/call-offers/[id]` - Détails offre
- `PUT /api/call-offers/[id]` - Mettre à jour offre
- `DELETE /api/call-offers/[id]` - Supprimer offre

3.5 Demandes d'Appels (`/api/call-requests/`)

- `GET /api/call-requests` - Liste demandes
- `POST /api/call-requests` - Créer demande
- `GET /api/call-requests/[id]` - Détails demande
- `POST /api/call-requests/[id]/accept` - Accepter demande
- `POST /api/call-requests/[id]/reject` - Rejeter demande

3.6 Réservations (`/api/bookings/`)

- `GET /api/bookings` - Liste réservations (utilisateur)
- `POST /api/bookings` - Créer réservation
- `GET /api/bookings/[id]` - Détails réservation
- `PUT /api/bookings/[id]` - Mettre à jour réservation
- `GET /api/bookings/creator` - Réservations créateur
- `GET /api/bookings/[id]/calendar` - Événement calendrier

3.7 Paiements (`/api/payments/`)

- `POST /api/payments/create-intent` - Créer PaymentIntent Stripe
- `POST /api/payments/webhook` - Webhook Stripe (événements paiements)

3.8 Payouts (`/api/payouts/`)

- `POST /api/payouts/request` - Demander payout (créateur)
- `GET /api/payouts/creator` - Payouts créateur
- `POST /api/payouts/update-status` - Mettre à jour statut

3.9 Notifications (`/api/notifications/`)

- `GET /api/notifications` - Liste notifications
- `POST /api/notifications/[id]/read` - Marquer comme lu
- `POST /api/notifications/mark-all-read` - Tout marquer comme lu
- `DELETE /api/notifications/[id]` - Supprimer notification

3.10 Reviews (`/api/reviews/`)

- `GET /api/reviews` - Liste reviews
- `POST /api/reviews` - Créer review
- `GET /api/reviews/creator/[id]` - Reviews d'un créateur

3.11 Stripe Connect (`/api/stripe/`)

- `POST /api/stripe/connect-onboard` - Onboarding Stripe Connect
- `POST /api/stripe/express-dashboard` - Lien dashboard Express
- `GET /api/stripe/balance/[creatorId]` - Balance créateur

3.12 Daily.co (`/api/daily/`)

- `POST /api/daily/create-room` - Créer room vidéo
- `POST /api/daily/get-token` - Obtenir token vidéo

3.13 Cron Jobs (/api/cron/)

- POST /api/cron/process-payouts - Traiter payouts manuels
- POST /api/cron/process-automatic-payouts - Traiter payouts automatiques

3.14 Admin - Dashboard (/api/admin/)

- GET /api/admin/dashboard - Dashboard admin
- GET /api/admin/users - Liste utilisateurs
- GET /api/admin/bookings - Liste réservations
- GET /api/admin/logs - Logs transactions

3.15 Admin - Paiements (/api/admin/payments/)

- GET /api/admin/payments - Liste paiements

3.16 Admin - Payouts (/api/admin/payouts/)

- GET /api/admin/payouts - Liste payouts
- GET /api/admin/payouts/dashboard - Dashboard payouts
- GET /api/admin/payouts/pending - Payouts en attente
- POST /api/admin/payouts/[id]/approve - Approuver payout
- POST /api/admin/payouts/[id]/reject - Rejeter payout
- POST /api/admin/payouts/trigger - Déclencher payout manuel
- POST /api/admin/payouts/block - Bloquer payouts créateur
- POST /api/admin/payouts/unblock - Débloquer payouts créateur
- POST /api/admin/payouts/test-eligibility - Tester éligibilité

3.17 Admin - Remboursements & Litiges (/api/admin/refunds/ , /api/admin/refunds-disputes/)

- GET /api/admin/refunds - Liste remboursements
- POST /api/admin/refunds - Créer remboursement
- GET /api/admin/refunds/[id] - Détails remboursement
- GET /api/admin/refunds-disputes - Remboursements et litiges

3.18 Admin - Paramètres (/api/admin/settings/)

- GET /api/admin/settings - Paramètres plateforme
- PUT /api/admin/settings - Mettre à jour paramètres
- PUT /api/admin/settings/platform-fee - Mettre à jour commission

3.19 Admin - Créateurs (/api/admin/creators/)

- GET /api/admin/creators/[id]/payout-settings - Paramètres payout créateur
 - PUT /api/admin/creators/[id]/payout-settings - Mettre à jour paramètres
 - POST /api/admin/creators/[id]/block-payout - Bloquer payout créateur
-

4. SCHÉMA DE BASE DE DONNÉES (Prisma)

4.1 Modèles Principaux

User

- Utilisateurs de la plateforme (fans et créateurs)
- Rôles: `USER` , `CREATOR` , `ADMIN`
- Relations: Creator (1-1), Bookings, CallRequests, Reviews, Notifications

Creator

- Profil créateur lié à un User
- Champs: bio, profileImage, stripeAccountId, currency
- Paramètres payout: payoutSchedule, payoutMinimum, isPayoutBlocked
- Relations: CallOffers, CallRequests, Reviews, Payouts

CallOffer

- Offres d'appels créées par les créateurs
- Champs: title, description, price, currency, dateTime, duration
- Statuts: `AVAILABLE` , `BOOKED` , `COMPLETED` , `CANCELLED`
- Relations: Creator, Booking (1-1)

Booking

- Réservations d'appels par les utilisateurs
- Champs: totalPrice, stripePaymentIntentId, dailyRoomUrl
- Statuts: `PENDING` , `CONFIRMED` , `COMPLETED` , `CANCELLED`
- Relations: User, CallOffer, Payment, Review

Payment

- Paiements effectués pour les réservations
- Champs: amount, currency, stripePaymentIntentId, platformFee, creatorAmount
- Statuts: `PENDING` , `SUCCEEDED` , `FAILED`
- Tracking payout: payoutStatus, payoutReleaseDate, transferId
- Relations: Booking, Refunds, Disputes

CallRequest

- Demandes d'appels personnalisées des utilisateurs
- Champs: proposedPrice, proposedDateTime, message
- Statuts: `PENDING` , `ACCEPTED` , `REJECTED`
- Relations: User, Creator

Payout

- Demandes de paiement des créateurs
- Champs: amount, currency, stripePayoutId
- Statuts: `REQUESTED` , `APPROVED` , `PROCESSING` , `PAID` , `FAILED` , `REJECTED` , `CANCELED`
- Dates: requestedAt, approvedAt, paidAt, failedAt, rejectedAt
- Relations: Creator, ApprovedBy (User), AuditLogs

Review

- Avis laissés par les utilisateurs
- Champs: rating (1-5), comment

- Relations: Booking, User, Creator

Notification

- Notifications pour les utilisateurs
- Types: BOOKING_CONFIRMED, PAYMENT_RECEIVED, PAYOUT_COMPLETED, etc.
- Champs: title, message, link, read, metadata
- Relations: User

4.2 Modèles Financiers Avancés

Refund

- Remboursements de paiements
- Champs: amount, currency, reason, stripeRefundId
- Tracking dette: creatorDebt, reconciled, reconciledAt, reversalId
- Statuts: PENDING, SUCCEEDED, FAILED, CANCELLED

Dispute

- Litiges/Chargebacks Stripe
- Champs: amount, currency, reason, stripeDisputeId
- Tracking dette: creatorDebt, reconciled, reconciledAt, reversalId
- Statuts: WARNING_NEEDS_RESPONSE, NEEDS_RESPONSE, UNDER_REVIEW, WON, LOST, etc.

PlatformSettings

- Paramètres globaux de la plateforme (singleton)
- Champs: platformFeePercentage, minimumPayoutAmount, holdingPeriodDays
- Modes: payoutMode (AUTOMATIC/MANUAL)
- Configuration: payoutFrequencyOptions, currency

PayoutScheduleNew

- Calendrier de payout par créateur
- Champs: mode, frequency, nextPayoutDate, isActive

TransactionLog

- Journal de toutes les transactions financières
- Types d'événements: PAYMENT_CREATED, PAYOUT_PAID, TRANSFER_SUCCEEDED, etc.
- Champs: eventType, entityType, entityId, amount, metadata

PayoutAuditLog

- Audit des actions sur les payouts
- Actions: TRIGGERED, BLOCKED, UNBLOCKED, COMPLETED, FAILED
- Champs: action, amount, status, adminId, reason

4.3 Modèles d'Authentification (NextAuth)

Account

- Comptes OAuth liés aux utilisateurs
- Providers: Google, etc.

Session

- Sessions actives des utilisateurs

VerificationToken

- Tokens de vérification email



5. COMPOSANTS REACT PRINCIPAUX

5.1 Composants de Base (/components/)

Navigation & Layout

- `navbar.tsx` - Barre de navigation principale
- `NotificationBell.tsx` - Cloche de notifications avec badge

Créateurs

- `creator-card.tsx` - Carte affichant un créateur
- `call-request-dialog.tsx` - Dialog pour demandes d'appel personnalisées

Providers

- `providers.tsx` - Providers globaux (React Query, etc.)
- `theme-provider.tsx` - Provider de thème (dark/light)

5.2 Composants Admin (/components/admin/)

Affichage Données

- `CurrencyDisplay.tsx` - Affichage devise formatée
- `MultiCurrencyDisplay.tsx` - Affichage multi-devises
- `DateDisplay.tsx` - Affichage dates formatées
- `StatusBadge.tsx` - Badge de statut coloré
- `DataTable.tsx` - Table de données générique

Navigation & État

- `FilterBar.tsx` - Barre de filtres
- `Pagination.tsx` - Pagination
- `EmptyState.tsx` - État vide
- `LoadingSpinner.tsx` - Spinner de chargement

5.3 Composants UI (/components/ui/)

Composants Radix UI:

- `button.tsx`, `card.tsx`, `dialog.tsx`, `dropdown-menu.tsx`
- `input.tsx`, `label.tsx`, `select.tsx`, `textarea.tsx`
- `table.tsx`, `tabs.tsx`, `badge.tsx`, `avatar.tsx`
- `alert.tsx`, `alert-dialog.tsx`, `toast.tsx`, `toaster.tsx`
- `calendar.tsx`, `date-range-picker.tsx`
- `accordion.tsx`, `collapsible.tsx`, `separator.tsx`
- `popover.tsx`, `tooltip.tsx`, `hover-card.tsx`
- `sheet.tsx`, `drawer.tsx`, `scroll-area.tsx`
- Et bien d'autres...

Composants Personnalisés:

- `currency-display.tsx` - Affichage devise
- `task-card.tsx` - Carte de tâche

6. PAGES PRINCIPALES

6.1 Pages Publiques

Racine (/app/)

- `page.tsx` - Page d'accueil (landing page)
- `layout.tsx` - Layout racine avec providers

Authentification (/app/auth/)

- `login/page.tsx` - Page de connexion
- `register/page.tsx` - Page d'inscription

Créateurs (/app/creators/)

- Pages publiques pour découvrir les créateurs

Réservation (/app/book/)

- Pages pour réserver un appel

Appel Vidéo (/app/call/)

- `[bookingId]/page.tsx` - Interface d'appel vidéo (Daily.co)

6.2 Dashboard Utilisateur (/app/dashboard/user/)


- `page.tsx` - Dashboard principal utilisateur
- `settings/page.tsx` - Paramètres utilisateur

6.3 Dashboard Créateur (/app/dashboard/creator/)

Pages Principales

- `page.tsx` - Dashboard principal créateur
- `settings/page.tsx` - Paramètres créateur
- `payment-setup/page.tsx` - Configuration Stripe Connect

Offres & Appels

- `offers/page.tsx` -  Gestion des offres d'appels
- `requests/page.tsx` - Demandes d'appels reçues
- `calls/page.tsx` - Historique des appels

Financier

- `earnings/page.tsx` - 💰 Page Earnings (revenus, graphiques)
- `payments/page.tsx` - 🇪🇺 Historique des paiements reçus
- `payouts/page.tsx` - 🏠 Historique des payouts effectués
- `payouts/request/page.tsx` - Demander un payout
- `payouts/settings/page.tsx` - Paramètres de payout

Autres

- `notifications/page.tsx` - 🔔 Notifications créateur
- `reviews/page.tsx` - Avis reçus

6.4 Dashboard Admin (/app/dashboard/admin/)

Pages Principales

- `page.tsx` - Dashboard principal admin

- `settings/page.tsx` - Paramètres plateforme
- `testing/page.tsx` - Page de tests

Gestion

- `notifications/page.tsx` - Notifications admin
- `logs/page.tsx` - Logs de transactions

Financier

- `payments/page.tsx` - 🇪🇺 Liste de tous les paiements
- `payouts/page.tsx` - 💰 Gestion des payouts
- `payouts/dashboard/page.tsx` - Dashboard payouts détaillé
- `refunds/page.tsx` - Gestion des remboursements
- `refunds-disputes/page.tsx` - 🔥 Remboursements & Litiges

Créateurs

- `creators/[id]/stripe/page.tsx` - Détails Stripe Connect créateur



7. CONFIGURATION STRIPE

7.1 Architecture Stripe

Type: Destination Charges (Stripe Connect)

Flux de Paiement

1. **Client paie** 100 EUR → Compte Plateforme
2. **Stripe déduit ses frais** (ex: 3.20 EUR) de la part créateur
3. **Plateforme garde** 15 EUR (commission via `application_fee_amount`)
4. **Créateur reçoit** 81.80 EUR automatiquement (via `transfer_data`)

Avantages

- ✅ Transfert automatique par Stripe (pas de webhook complexe)
- ✅ Moins de risque d'échec de transfert
- ✅ Fonds immédiatement disponibles au créateur
- ✅ Logic simplifiée

7.2 Fichier Principal (`/lib/stripe.ts`)

Fonctions Principales

`createPaymentIntent()`

- Crée un PaymentIntent Stripe avec Destination Charges
- Paramètres: `amount`, `currency`, `stripeAccountId`, `platformFeePercentage`
- Calcule automatiquement `application_fee_amount`
- Configure `transfer_data.destination`

`createConnectPayout()`

- Crée un payout du compte Stripe Connect vers la banque du créateur
- Utilisé quand le créateur demande un retrait

`getConnectAccountBalance()`

- Récupère le solde disponible sur le compte Stripe Connect

getCreatorCurrency()

- Récupère la devise du compte Stripe Connect du créateur

verifyWebhookSignature()

- Vérifie la signature des webhooks Stripe

getConnectAccountDetails()

- Récupère les détails d'un compte Stripe Connect

retrievePaymentIntent()

- Récupère un PaymentIntent existant

Configuration

- **Holding Period:** 7 jours (PAYOUT_HOLDING_DAYS)
- **API Version:** 2025-12-15.clover

7.3 Variables d'Environnement (.env.example)

```
# Stripe
STRIPE_SECRET_KEY="sk_test_..."      # Clé secrète Stripe
STRIPE_PUBLISHABLE_KEY="pk_test_..."  # Clé publique Stripe
STRIPE_WEBHOOK_SECRET="whsec_..."     # Secret webhook Stripe
```

7.4 Stripe Connect Onboarding

Route API: /api/stripe/connect-onboard

- Génère un lien d'onboarding Stripe Connect
- Embedded onboarding pour une expérience fluide

Documentation: docs/STRIPE_CONNECT_EMBEDDED_ONBOARDING.md

**8. SYSTÈME DE PAIEMENTS & PAYOUTS****8.1 Flux de Paiement**

1. **Utilisateur réserve un appel** → Création Booking
2. **Paiement Stripe** → Payment Intent créé avec Destination Charges
3. **Paiement réussi** →
 - Payment.status = SUCCEEDED
 - Booking.status = CONFIRMED
 - Transfer automatique vers créateur
 - Commission retenue par plateforme
4. **Holding Period** → 7 jours avant éligibilité payout
5. **Payout éligible** → Créateur peut demander retrait
6. **Payout traité** → Fonds transférés vers banque créateur

8.2 Gestion des Payouts**Routes Créateur**

- **Demander payout:** POST /api/creators/payouts/request
- **Historique:** GET /api/creators/payouts/history

- **Paramètres:** GET/PUT /api/creators/payout-settings

Routes Admin

- **Dashboard payouts:** GET /api/admin/payouts/dashboard
- **Approuver:** POST /api/admin/payouts/[id]/approve
- **Rejeter:** POST /api/admin/payouts/[id]/reject
- **Déclencher manuel:** POST /api/admin/payouts/trigger
- **Bloquer/Débloquer:** POST /api/admin/payouts/block|unblock

Automatisation

- **Cron automatique:** /api/cron/process-automatic-payouts
- **Cron manuel:** /api/cron/process-payouts

8.3 Système de Dettes (Refunds & Disputes)

Quand un remboursement ou litige survient:

1. **Dette créateur** calculée (85% du montant)
2. **Reconciliation** via:
 - Reversal de transfert (si possible)
 - Déduction sur prochain payout
 - Action manuelle admin

Tables:

- Refund → Champs: creatorDebt, reconciled, reversalId
- Dispute → Champs: creatorDebt, reconciled, reversalId

Fichier: /lib/creator-debt.ts



9. BIBLIOTHÈQUES UTILITAIRES (/lib/)

9.1 Authentification & Autorisation

- auth.ts - Gestion sessions, tokens
- nextauth.ts - Configuration NextAuth

9.2 Base de Données

- db.ts - Client Prisma singleton

9.3 Paiements & Stripe

- stripe.ts - Configuration et fonctions Stripe
- stripe-account-validator.ts - Validation comptes Connect
- payout-eligibility.ts - Vérification éligibilité payout
- payout-validation.ts - Validation demandes payout
- creator-debt.ts - Gestion dettes créateurs

9.4 Devises

- currency-converter.ts - Conversion entre devises
- currency-utils.ts - Formatage et utilitaires devises

9.5 Notifications & Communications

- `notifications.ts` - Création notifications
- `email.ts` - Envoi emails (Resend)

9.6 Configuration

- `settings.ts` - Paramètres plateforme (cache + DB)

9.7 Vidéo

- `daily.ts` - Daily.co (création rooms, tokens)

9.8 Autres

- `calendar.ts` - Gestion calendrier (événements .ics)
- `analytics.ts` - Google Analytics
- `logger.ts` - Logging
- `utils.ts` - Utilitaires divers (classNames, etc.)



10. PAGES CLÉS MENTIONNÉES

10.1 Pages Offres



Créateur - Gestion Offres

- **Fichier:** `/app/dashboard/creator/offers/page.tsx`
- **Fonctionnalités:**
 - Créer nouvelles offres d'appels
 - Modifier offres existantes
 - Supprimer offres
 - Voir statuts (AVAILABLE, BOOKED, COMPLETED, CANCELLED)

10.2 Pages Notifications



Créateur - Notifications

- **Fichier:** `/app/dashboard/creator/notifications/page.tsx`
- **Fonctionnalités:**
 - Liste notifications (réservations, paiements, payouts)
 - Marquer comme lu
 - Filtrer par type



Admin - Notifications

- **Fichier:** `/app/dashboard/admin/notifications/page.tsx`
- Notifications système pour admin

Composant:

- `NotificationBell.tsx` - Cloche avec badge (navbar)

10.3 Pages Earnings (Revenus)



Créateur - Earnings

- **Fichier:** `/app/dashboard/creator/earnings/page.tsx`
- **Fonctionnalités:**
 - Vue d'ensemble revenus totaux
 - Graphiques de revenus (par jour/semaine/mois)

- Revenus disponibles vs en attente
- Breakdown par offre
- Commission plateforme déduite

10.4 Pages Paiements

Créateur - Paiements Reçus

- **Fichier:** `/app/dashboard/creator/payments/page.tsx`
- **Fonctionnalités:**
 - Historique des paiements reçus
 - Détails: montant, commission, montant créateur
 - Statuts: PENDING, SUCCEEDED, FAILED
 - Lien vers réservation associée

Admin - Tous Paiements

- **Fichier:** `/app/dashboard/admin/payments/page.tsx`
- **Fonctionnalités:**
 - Liste complète de tous les paiements
 - Filtres (créateur, statut, dates)
 - Export données

10.5 Pages Payouts

Créateur - Payouts

- **Fichier:** `/app/dashboard/creator/payouts/page.tsx`
- **Fonctionnalités:**
 - Historique des payouts
 - Statuts détaillés
 - Voir rejections/échecs

Créateur - Demander Payout

- **Fichier:** `/app/dashboard/creator/payouts/request/page.tsx`
- **Fonctionnalités:**
 - Formulaire demande payout
 - Vérification éligibilité
 - Montant disponible affiché

Créateur - Paramètres Payout

- **Fichier:** `/app/dashboard/creator/payouts/settings/page.tsx`
- **Fonctionnalités:**
 - Choisir schedule (DAILY, WEEKLY, MANUAL)
 - Définir minimum payout
 - Voir statut blocage

Admin - Payouts

- **Fichier:** `/app/dashboard/admin/payouts/page.tsx`
- **Fonctionnalités:**
 - Liste tous payouts
 - Approuver/Rejeter
 - Bloquer/Débloquer créateurs

Admin - Dashboard Payouts

- **Fichier:** `/app/dashboard/admin/payouts/dashboard/page.tsx`

- **Fonctionnalités:**

- Vue d'ensemble payouts
- Statistiques
- Payouts en attente
- Montants totaux



11. AUTHENTIFICATION & AUTORISATION

11.1 NextAuth Configuration

Fichier: `/lib/nextauth.ts`

Providers:

- Credentials (email/password)
- Google OAuth

Adapter:

- Prisma Adapter pour stocker sessions dans la base

Callbacks:

- `jwt` - Enrichit token avec role
- `session` - Enrichit session avec user.id, role

11.2 Rôles

Enum Role :

- `USER` - Utilisateur normal (fan)
- `CREATOR` - Créateur (propose des appels)
- `ADMIN` - Administrateur plateforme

11.3 Middleware

Fichier: `/middleware.ts`

- Protection routes dashboard selon rôle
- Redirection si non authentifié



12. SCRIPTS & UTILITAIRES

12.1 Scripts (`/scripts/`)

- `seed.ts` - Seed données de test
- `migrate-payout-release-dates.ts` - Migration payouts
- `test-webhooks.ts` - Test webhooks Stripe
- `update_existing_payments.mjs` - Mise à jour paiements existants

12.2 Scripts Racine

- `check-payout-data.js` - Vérifier données payouts
- `check_bookings.mjs` - Vérifier réservations
- `fix_bookings.mjs` - Corriger réservations

12.3 Tests (/tests/)

- `fee-calculation-test.ts` - Test calcul commissions
- `separate-charges-transfers-test.ts` - Test architecture charges/transferts

13. DOCUMENTATION (/docs/)

13.1 Documents Existants

- `PAYOUT_SYSTEM.md` - Documentation système payouts
- `STRIPE_CONNECT_EMBEDDED_ONBOARDING.md` - Guide onboarding Stripe
- `PAYOUT_SYSTEM.pdf` - Version PDF
- `STRIPE_CONNECT_EMBEDDED_ONBOARDING.pdf` - Version PDF

13.2 Documents Racine

- `CORRECTIONS_TYPESCRIPT.md` - Corrections TypeScript
- `REFACTORING_SUMMARY.md` - Résumé refactoring
- `GIT_PUSH_INSTRUCTIONS.md` - Instructions push Git
- (Versions PDF également disponibles)

14. VARIABLES D'ENVIRONNEMENT

14.1 Base de Données

```
DATABASE_URL="postgresql://..."
```

14.2 Authentification

```
NEXTAUTH_SECRET="..." # Secret NextAuth
NEXTAUTH_URL="http://localhost:3000"
```

14.3 Stripe

```
STRIPE_SECRET_KEY="sk_test_..."
STRIPE_PUBLISHABLE_KEY="pk_test_..."
STRIPE_WEBHOOK_SECRET="whsec_..."
```

14.4 Daily.co (Vidéo)

```
DAILY_API_KEY="..."
```

14.5 Resend (Emails)

```
RESEND_API_KEY="re_..."
EMAIL_FROM="Call a Star <noreply@callstar.com>"
```

14.6 Google

```
# Analytics
NEXT_PUBLIC_GA_MEASUREMENT_ID="G-XXXXXXXXXX"

# OAuth
GOOGLE_CLIENT_ID="...apps.googleusercontent.com"
GOOGLE_CLIENT_SECRET="GOCSPX-..."
```



15. DÉPLOIEMENT

15.1 Configuration Vercel

Fichier: `vercel.json`

15.2 Build

```
npm run build      # Build Next.js
```

15.3 Développement

```
npm run dev        # Serveur dev (http://localhost:3000)
```



16. POINTS D'ATTENTION POUR CORRECTIONS

16.1 Architecture Paiements

- **Destination Charges** utilisé (automatique)
- Vérifier que tous les PaymentIntents utilisent bien `transfer_data` et `application_fee_amount`
- S'assurer que la commission est dynamique (via `PlatformSettings`)

16.2 Système Payouts

- Vérifier logique d'éligibilité (`payout-eligibility.ts`)
- Tester crons automatiques
- Vérifier holding period (7 jours)
- Tester approbation/rejet admin

16.3 Gestion Dettes

- Vérifier réconciliation refunds (`creator-debt.ts`)
- Tester déduction sur payouts futurs
- Vérifier reversals de transferts

16.4 Multi-Devises

- Vérifier que la devise du créateur est bien propagée partout
- Tester conversion devises (`currency-converter.ts`)
- Vérifier affichage multi-devises dans admin

16.5 Webhooks Stripe

- ⚠ Route: `/api/payments/webhook`
- ⚠ Vérifier gestion de tous les événements critiques:
- `payment_intent.succeeded`
- `payment_intent.payment_failed`
- `payout.paid`
- `payout.failed`
- `charge.refunded`
- `charge.dispute.created`

16.6 Notifications

- ⚠ Vérifier que toutes les actions critiques envoient notifications
- ⚠ Tester emails (Resend)
- ⚠ Vérifier NotificationBell en temps réel

16.7 UI/UX

- ⚠ Vérifier toutes les pages mentionnées (offres, earnings, payouts, etc.)
- ⚠ Tester flows complets (réservation → paiement → payout)
- ⚠ Vérifier responsive design
- ⚠ Tester dark mode

16.8 TypeScript

- ⚠ Corriger erreurs TypeScript (`CORRECTIONS_TYPESCRIPT.md`)
- ⚠ Vérifier types Prisma
- ⚠ Valider formulaires avec Zod

16.9 Sécurité

- ⚠ Vérifier authentification sur toutes routes API
- ⚠ Vérifier autorisation par rôle
- ⚠ Valider inputs (XSS, SQL injection)
- ⚠ Rate limiting sur API critiques

16.10 Performance

- ⚠ Vérifier caches (settings, balance)
- ⚠ Optimiser queries Prisma (includes, selects)
- ⚠ Tester avec beaucoup de données



17. STATISTIQUES DU PROJET

- **Total fichiers TypeScript/TSX:** ~233 fichiers
- **Routes API:** ~66 routes
- **Composants React:** ~75+ composants
- **Pages Next.js:** ~30+ pages
- **Modèles Prisma:** 17 modèles principaux
- **Dépendances:** ~100+ packages

18. RESSOURCES & LIENS

- **Repository:** <https://github.com/StreallyX/callastar>
 - **Next.js Docs:** <https://nextjs.org/docs>
 - **Prisma Docs:** <https://www.prisma.io/docs>
 - **Stripe Connect:** <https://stripe.com/docs/connect>
 - **Daily.co:** <https://docs.daily.co>
 - **Radix UI:** <https://www.radix-ui.com>
 - **Tailwind CSS:** <https://tailwindcss.com>
-

19. CHECKLIST POUR CORRECTIONS

Configuration & Setup

- ☐ Vérifier `.env` complet avec toutes les clés
- ☐ Tester connexion base de données
- ☐ Vérifier migrations Prisma à jour
- ☐ Seed données de test si nécessaire

Stripe & Paiements

- ☐ Vérifier Destination Charges partout
- ☐ Tester création PaymentIntent
- ☐ Tester webhook Stripe
- ☐ Vérifier calcul commissions dynamique
- ☐ Tester multi-devises

Payouts

- ☐ Tester demande payout créateur
- ☐ Tester approbation/rejet admin
- ☐ Tester payouts automatiques (cron)
- ☐ Vérifier holding period
- ☐ Tester blocage/déblocage payouts

Refunds & Disputes

- ☐ Tester création refund
- ☐ Vérifier calcul dette créateur
- ☐ Tester réconciliation dette
- ☐ Tester gestion disputes

UI/UX - Pages Créateur

- ☐ Page Offres fonctionnelle
- ☐ Page Notifications fonctionnelle
- ☐ Page Earnings avec graphiques
- ☐ Page Paiements avec historique
- ☐ Page Payouts avec historique

- [] Page Demander Payout fonctionnelle
- [] Page Paramètres Payout fonctionnelle

UI/UX - Pages Admin

- [] Dashboard admin
- [] Page Paiements admin
- [] Page Payouts admin
- [] Dashboard Payouts admin
- [] Page Refunds & Disputes
- [] Page Paramètres plateforme

Authentication & Sécurité

- [] Tester login/logout
- [] Tester OAuth Google
- [] Vérifier protection routes
- [] Vérifier autorisation par rôle
- [] Tester changement mot de passe

Notifications & Emails

- [] Tester notifications in-app
- [] Tester NotificationBell
- [] Tester envoi emails (Resend)
- [] Vérifier tous types notifications

Vidéo & Appels

- [] Tester création room Daily.co
- [] Tester interface appel vidéo
- [] Tester tokens Daily.co

Tests & Qualité

- [] Corriger erreurs TypeScript
- [] Run linter (ESLint)
- [] Tester avec données volumineuses
- [] Vérifier performance queries
- [] Tester responsive mobile

Déploiement








- [] Build réussi (`npm run build`)
- [] Vérifier configuration Vercel
- [] Tester en production
- [] Vérifier logs erreurs








20. NOTES FINALES

Cette analyse complète couvre tous les aspects du projet callastar. Le projet est une plateforme bien structurée utilisant des technologies modernes (Next.js 14, Prisma, Stripe Connect).

Points forts:

-  Architecture claire et modulaire
-  Séparation concerns (routes API, composants, lib)
-  Système de paiements robuste (Destination Charges)
-  Gestion multi-devises
-  Système de notifications complet
-  Admin dashboard complet
-  Documentation existante

Points à améliorer (probablement raison des corrections):

-  Potentielles erreurs TypeScript à corriger
 -  Tests à compléter/fixer
 -  Webhooks Stripe à vérifier
 -  Logique payouts à valider
 -  Gestion dettes créateurs à tester
-

Document créé le: 27 décembre 2024

Analyste: DeepAgent (Abacus.AI)

Version: 1.0