🚀 Oliver Admin Panel

Panel d'administration complet pour la plateforme Oliver - Gestion complète des utilisateurs, modération, transactions, accounting, et KYC.

📋 Table des Matières

🎯 Vue d'ensemble

Le panel admin Oliver est une application full-stack moderne permettant aux administrateurs de gérer tous les aspects de la plateforme:

- ✅ **9 Pages complètes** - Dashboard, Users, Reports, Moderation, Transactions, Accounting, Audit Log, Settings, KYC
- ✅ **28 Endpoints API REST** - Backend NestJS complet
- ✅ **30+ Composants UI** - Interface moderne avec Shadcn/UI + Radix
- ✅ **Intégrations externes** - Google Vision, AWS Rekognition, Yoti KYC, SendGrid
- ✅ **Sécurité robuste** - JWT, RBAC, Rate Limiting, Audit Log
- ✅ **Performance optimisée** - React Query, Redis caching

🛠️ Stack Technique

Frontend

- **Framework**: Next.js 15 (App Router)
- **Language**: TypeScript 5.3
- **State Management**: React Query (TanStack Query)
- **UI Library**: Shadcn/UI + Radix UI
- **Styling**: Tailwind CSS
- **Charts**: Recharts
- **Icons**: Lucide React
- **Forms**: Zod validation

Backend

- **Framework**: NestJS 10
- **Language**: TypeScript 5.3
- **Database**: PostgreSQL (Prisma ORM)
- **Cache**: Redis (ioredis)
- **Authentication**: JWT (Passport)
- **Validation**: Zod schemas
- **API Docs**: Swagger/OpenAPI

Services Externes

- **Storage**: AWS S3
- **AI Moderation**: Google Vision API, AWS Rekognition
- **KYC Provider**: Yoti
- **Email**: SendGrid
- **Monitoring**: (Sentry, Datadog - à configurer)

DevOps

- **Containerization**: Docker + Docker Compose
- **CI/CD**: GitHub Actions
- **Hosting**: AWS / Vercel (Frontend) + AWS EC2/ECS (Backend)

📁 Structure du Projet

```
oliver-admin/
├── apps/
│   ├── web/                    # Next.js Frontend
│   │   ├── app/
│   │   │   ├── (admin)/        # Admin routes group
│   │   │   │   ├── dashboard/
│   │   │   │   ├── users/
│   │   │   │   ├── reports/
│   │   │   │   ├── moderation/
│   │   │   │   ├── transactions/
│   │   │   │   ├── accounting/
│   │   │   │   ├── audit-log/
│   │   │   │   ├── settings/
│   │   │   │   └── kyc/
│   │   │   └── api/            # API routes (optional)
│   │   ├── components/
│   │   │   ├── admin/          # Admin-specific components
│   │   │   └── ui/             # Reusable UI components
│   │   ├── hooks/
│   │   │   └── admin/          # Custom React hooks
│   │   ├── lib/
│   │   │   ├── admin/
│   │   │   │   └── api-client.ts
│   │   │   └── utils.ts
│   │   └── package.json
│   │
│   └── api/                    # NestJS Backend
│       ├── src/
│       │   ├── modules/
│       │   │   ├── admin/      # Admin module
│       │   │   ├── ai-moderation/
│       │   │   ├── kyc-provider/
│       │   │   ├── storage/
│       │   │   ├── email/
│       │   │   ├── database/
│       │   │   └── cache/
│       │   ├── common/
│       │   │   ├── guards/
│       │   │   ├── pipes/
│       │   │   └── interceptors/
│       │   └── main.ts
│       └── package.json
│
├── packages/
│   ├── database/
│   │   └── prisma/
```

```
│   │   ├── schema.prisma
│   │   ├── migrations/
│   │   └── seed.ts
│   └── shared/
│       └── types/
│
├── docker-compose.yml
├── .github/
│   └── workflows/
│       └── deploy.yml
└── README.md
```

✅ Prérequis

Obligatoire

- **Node.js** >= 18.x
- **npm** >= 9.x ou **pnpm** >= 8.x
- **PostgreSQL** >= 14
- **Redis** >= 7
- **Docker** >= 20.x (optionnel mais recommandé)

Services Externes

- **AWS Account** (S3, Rekognition)
- **Google Cloud** (Vision API)
- **Yoti Account** (KYC)
- **SendGrid Account** (Email)

🚀 Installation

1. Cloner le repository

```bash
git clone https://github.com/votre-org/oliver-admin.git
cd oliver-admin
```

2. Installer les dépendances

```bash
# Root (si monorepo)
npm install

# OU installer individuellement
cd apps/web && npm install
cd ../api && npm install
cd ../../packages/database && npm install
```

3. Setup Docker (Recommandé)

```bash
# Démarrer PostgreSQL + Redis
docker-compose up -d postgres redis
```

OU installer manuellement PostgreSQL et Redis localement.

⚙️ Configuration

1. Backend Environment Variables

Créer `apps/api/.env`:

```env
# Database
DATABASE_URL="postgresql://oliver:oliver_password@localhost:5432/oliver_db"

# Redis
REDIS_HOST="localhost"
REDIS_PORT="6379"
REDIS_PASSWORD=""

# JWT
JWT_SECRET="your-super-secret-jwt-key-change-this-in-production"
JWT_EXPIRY="7d"

# AWS
AWS_ACCESS_KEY_ID="your-aws-access-key"
AWS_SECRET_ACCESS_KEY="your-aws-secret-key"
AWS_REGION="us-east-1"
S3_BUCKET="oliver-storage"
S3_BUCKET_KYC="oliver-kyc-documents"

# Google Cloud
GOOGLE_VISION_KEY_PATH="/path/to/google-vision-credentials.json"

# Yoti KYC
YOTI_CLIENT_SDK_ID="your-yoti-client-sdk-id"
YOTI_KEY_FILE_PATH="/path/to/yoti-key.pem"

# SendGrid
SENDGRID_API_KEY="your-sendgrid-api-key"
FROM_EMAIL="no-reply@oliver.com"

# App
PORT=4000
NODE_ENV="development"

# Monitoring (Optional)
SENTRY_DSN=""
DATADOG_API_KEY=""
```

2. Frontend Environment Variables

Créer `apps/web/.env.local`:

```env
NEXT_PUBLIC_API_URL=http://localhost:4000/api
NODE_ENV=development
```

3. Database Setup

```bash
cd packages/database

# Generate Prisma Client
npx prisma generate

# Run migrations
npx prisma migrate dev

# Seed database with admin user
npx prisma db seed
```

Credentials par défaut après seed:

- Email: `admin@oliver.com`
- Password: `Admin123!`

🎬 Lancement

Mode Développement

Option 1: Docker Compose (Recommandé)

```bash
# Démarrer tous les services
docker-compose up

# Frontend: http://localhost:3000
# Backend API: http://localhost:4000
# API Docs: http://localhost:4000/api/docs
```

Option 2: Manuel

Terminal 1 - Backend:

```bash
cd apps/api
npm run start:dev
# API disponible sur http://localhost:4000
```

Terminal 2 - Frontend:

```bash
cd apps/web
npm run dev
# App disponible sur http://localhost:3000
```

Terminal 3 - Database (si pas Docker):

```bash
# Démarrer PostgreSQL et Redis manuellement
```

Mode Production

```bash
# Build backend
cd apps/api
npm run build
npm run start:prod

# Build frontend
cd apps/web
npm run build
npm run start
```

🎨 Fonctionnalités

1. Dashboard 📊

- **Métriques clés**: Revenue, Users, Conversion, Reports
- **Charts**: Sales overview (revenue + payouts)
- **Transactions récentes**: 10 dernières transactions
- **Quick stats**: Avg transaction, subscriptions, earnings
- **Période sélectionnable**: 7d / 30d / 90d

2. Users Management 👥

- **Liste complète** des utilisateurs avec filtres
- **Recherche** par nom, email, username
- **Filtres**: Role (Admin/Creator/Fan), Status (Verified/Pending/Suspended)
- **Bulk actions**: Suspend, Reset password multiple users
- **User detail modal**: Vue complète de l'utilisateur
- **Export CSV** des utilisateurs

3. Reports & Flags 🚩

- **Queue de reports** avec priorités
- **Filtres**: Type (User/Post/Message), Severity, Status
- **Panel de détail** avec tabs (Content, Context, History)
- **Actions**: Dismiss, Warn, Ban, Approve
- **Assign à moderator**

4. Moderation Queue 🛡️

- **Dark mode UI** pour protection des yeux
- **Content blur** avec unblur button
- **AI Analysis**: Scores violence, adult, hate, spam
- **Batch review**: Review multiple items
- **Actions**: Approve, Reject, Escalate

5. Transactions 💳

- **Liste complète** avec filtres (Date, Type, Status)
- **Charts**: Revenue trend, Payouts trend
- **Transaction details** avec user info
- **Refund capability** pour admins
- **Export CSV**

6. Accounting & Export 📈

- **Financial summary**: Revenue, Fees, Commission, Net Profit
- **Revenue breakdown**: Subscriptions, PPV, Tips
- **Export reports**: CSV, XLSX, PDF
- **Export history**: Track tous les exports avec download links
- **Period selector**: Day, Week, Month, Year

7. Audit Log 📝

- **Timeline view** de toutes les actions admin
- **Filtres**: Actor, Action type, Date range
- **Metadata expansion**: Détails complets de chaque action
- **Technical info**: IP, Device, User Agent
- **Export CSV** pour compliance

8. Settings ⚙️

- **3 Tabs**: General, Security, API
- **General**: Platform name, Contact email, Dark mode, Auto backup
- **Security**: Password length, 2FA requirement, IP whitelist
- **API**: Key management avec copy & reset

9. KYC Validation 🔐

- **User info** complète
- **AI Confidence scores**: Identity match, Liveness, Document authenticity
- **Document viewer**: 4 documents (ID front/back, Selfie, Proof)
- **Verification checklist**: Guide pour review
- **Actions**: Approve, Reject (with reason), Request senior review

📚 API Documentation

Swagger UI

Démarrer le backend et visiter:

```
http://localhost:4000/api/docs
```

Endpoints Principaux

Authentication

- `POST /api/auth/login` - Login admin
- `POST /api/auth/logout` - Logout
- `POST /api/auth/refresh` - Refresh token

Dashboard

- `GET /api/admin/dashboard/metrics?period=30d`
- `GET /api/admin/dashboard/sales?period=30d`
- `GET /api/admin/dashboard/transactions?limit=10`

Users

- `GET /api/admin/users` - List users
- `GET /api/admin/users/:id` - Get user details
- `POST /api/admin/users/:id/suspend` - Suspend user
- `POST /api/admin/users/:id/reset-password` - Reset password
- `POST /api/admin/users/bulk-suspend` - Bulk suspend
- `GET /api/admin/users/export.csv` - Export CSV

Reports

- `GET /api/admin/reports` - List reports
- `PATCH /api/admin/reports/:id` - Update report
- `POST /api/admin/reports/:id/assign` - Assign moderator

Moderation

- `GET /api/admin/moderation/queue` - Get queue
- `POST /api/admin/moderation/decision` - Make decision

Transactions

- `GET /api/admin/transactions` - List transactions
- `GET /api/admin/transactions/trends` - Get trends
- `POST /api/admin/transactions/:id/refund` - Refund

Accounting

- `GET /api/admin/accounting/summary` - Get summary
- `POST /api/admin/accounting/export` - Export data
- `GET /api/admin/accounting/exports/:id/download` - Download export

Audit Log

- `GET /api/admin/audit-log` - Get logs
- `GET /api/admin/audit-log/export.csv` - Export CSV

Settings

- `GET /api/admin/settings` - Get settings
- `PATCH /api/admin/settings` - Update settings
- `POST /api/admin/settings/api-key/reset` - Reset API key

KYC

- `GET /api/admin/kyc/pending` - Get pending KYCs
- `GET /api/admin/kyc/:id` - Get KYC details
- `POST /api/admin/kyc/:id/approve` - Approve KYC
- `POST /api/admin/kyc/:id/reject` - Reject KYC

🚢 Déploiement

Production Checklist

- [ ] Mettre à jour toutes les variables d'environnement
- [ ] Changer JWT_SECRET
- [ ] Configurer SSL/HTTPS
- [ ] Setup monitoring (Sentry, Datadog)
- [ ] Configurer backups automatiques PostgreSQL
- [ ] Setup CDN pour assets statiques
- [ ] Activer rate limiting strict
- [ ] Configurer log aggregation
- [ ] Setup alerting
- [ ] Load testing

Option 1: AWS (Recommandé)

Frontend (Vercel):

```bash
# Connect to Vercel
vercel

# Deploy
vercel --prod
```

Backend (AWS ECS):

```bash
# Build Docker image
docker build -t oliver-api:latest ./apps/api

# Tag & Push to ECR
aws ecr get-login-password --region us-east-1 | docker login --username AWS --password-stdin YOUR_ECR
docker tag oliver-api:latest YOUR_ECR/oliver-api:latest
docker push YOUR_ECR/oliver-api:latest

# Update ECS service
aws ecs update-service --cluster oliver-cluster --service oliver-api --force-new-deployment
```

Option 2: Docker Compose (VPS)

```bash
# Sur le serveur de production
git clone https://github.com/votre-org/oliver-admin.git
cd oliver-admin

# Copier .env files
cp apps/api/.env.example apps/api/.env
cp apps/web/.env.local.example apps/web/.env.local

# Éditer les .env avec les vraies credentials

# Build et démarrer
docker-compose -f docker-compose.prod.yml up -d

# Setup reverse proxy (Nginx)
# Configurer SSL avec Let's Encrypt
```

GitHub Actions CI/CD

Le workflow `.github/workflows/deploy.yml` est déjà configuré pour:

- ✅ Linter & Type checking
- ✅ Run tests
- ✅ Build Docker images
- ✅ Deploy to AWS ECS
- ✅ Notify Slack

🧪 Tests

Unit Tests

```bash
# Backend
cd apps/api
npm run test

# Frontend
cd apps/web
npm run test
```

E2E Tests

```bash
# Backend
cd apps/api
npm run test:e2e

# Frontend (Playwright)
cd apps/web
npx playwright test
```

Coverage

```bash
npm run test:cov
```

🔒 Sécurité

Implémenté ✅

- ✅ **JWT Authentication** avec refresh tokens
- ✅ **RBAC** (Role-Based Access Control)
- ✅ **Permissions granulaires** par feature
- ✅ **Rate Limiting** (100 req/min par user)
- ✅ **Input validation** avec Zod schemas
- ✅ **SQL Injection protection** via Prisma
- ✅ **XSS protection** via React + sanitization
- ✅ **CSRF protection** via Helmet
- ✅ **Audit Log** complet de toutes les actions
- ✅ **Password hashing** avec bcrypt
- ✅ **Secure cookies** httpOnly + secure
- ✅ **CORS** configuré strictement

À Implémenter ⚠️

- ⚠️ **2FA** pour tous les admins
- ⚠️ **IP Whitelisting** enforcement
- ⚠️ **Brute force protection** avancé
- ⚠️ **Session management** avec Redis
- ⚠️ **Secrets rotation** automatique
- ⚠️ **Security headers** additionnels (CSP)

Security Audit

Avant production, exécuter:

```bash
# Scan dépendances
npm audit

# Scan OWASP
npm run security:scan

# Penetration testing
# Engager une société spécialisée
```

📊 Monitoring & Logging

Setup Sentry (Errors)

```typescript
// apps/api/src/main.ts
import * as Sentry from '@sentry/node';

Sentry.init({
  dsn: process.env.SENTRY_DSN,
  environment: process.env.NODE_ENV,
});
```

Setup Datadog (APM)

```typescript
// apps/api/src/main.ts
import tracer from 'dd-trace';

tracer.init({
  hostname: 'datadog-agent',
  port: 8126,
});
```

Logs

Tous les logs sont centralisés:

- **Backend**: Console → CloudWatch / ELK
- **Frontend**: Console → Vercel Logs
- **Audit Log**: PostgreSQL table

🤝 Contributing

Code Style

- **Formatter**: Prettier
- **Linter**: ESLint
- **Pre-commit hooks**: Husky + lint-staged

```bash
# Format code
npm run format

# Lint
npm run lint

# Type check
npm run type-check
```

Pull Request Process

1. Create feature branch: `git checkout -b feature/ma-feature`
2. Commit changes: `git commit -m "feat: ajout feature X"`
3. Push: `git push origin feature/ma-feature`
4. Open Pull Request sur GitHub
5. Wait for review & CI/CD checks
6. Merge après approval

📞 Support

Documentation

- **API Docs**: http://localhost:4000/api/docs
- **Prisma Docs**: https://www.prisma.io/docs
- **Next.js Docs**: https://nextjs.org/docs
- **NestJS Docs**: https://docs.nestjs.com

Issues

Ouvrir une issue sur GitHub: https://github.com/votre-org/oliver-admin/issues

Contact

- **Email**: dev@oliver.com
- **Slack**: #oliver-admin-dev

📄 License

🎉 Crédits

Développé par l'équipe Oliver avec ❤️

Technologies utilisées:

- Next.js, React, TypeScript
- NestJS, Prisma, PostgreSQL
- Shadcn/UI, Tailwind CSS, Radix UI
- AWS, Google Cloud, Yoti, SendGrid

🗺️ Roadmap

Phase 2 (Q2 2025)

- [ ] Advanced Analytics Dashboard
- [ ] Real-time notifications (WebSockets)
- [ ] Multi-language support (i18n)
- [ ] Mobile responsive admin panel
- [ ] Advanced reporting & BI

Phase 3 (Q3 2025)

- [ ] Machine Learning fraud detection
- [ ] Automated content moderation
- [ ] Revenue forecasting
- [ ] Customer segmentation
- [ ] A/B testing framework

Happy Coding! 🚀