🚀 Oliver Platform - Admin Panel

Afficher l'image

Afficher l'image

Afficher l'image

Enterprise-grade admin dashboard for the Oliver platform with KYC verification, AI moderation, financial management, and comprehensive user administration.

✨ Features

🎯 Core Features

- **Dashboard Analytics** - Real-time metrics, sales trends, and activity monitoring
- **User Management** - Advanced user administration with bulk actions and detailed profiles
- **KYC Verification** - Automated document verification with AI-powered scoring (Yoti integration)
- **Content Moderation** - AI-assisted moderation queue with Google Vision and AWS Rekognition
- **Reports System** - Comprehensive reporting and ticket management
- **Transaction Management** - Complete financial transaction tracking and refund system
- **Accounting & Exports** - Financial summaries with PDF, CSV, and Excel exports
- **Audit Logging** - Complete activity tracking for compliance
- **Settings Management** - Platform configuration with role-based permissions

🔒 Security Features

- JWT authentication with refresh tokens
- Role-based access control (RBAC)
- 2FA support
- IP whitelisting
- Rate limiting
- Session management
- Audit trail for all admin actions

🎨 UI/UX

- Modern, responsive design with Tailwind CSS
- Dark mode support
- Real-time notifications
- Loading states and skeleton screens
- Accessible components (WCAG AA compliant)
- Mobile-first approach

📐 Architecture

Tech Stack

Backend

- **Framework**: NestJS
- **Database**: PostgreSQL with Prisma ORM
- **Cache**: Redis
- **Storage**: AWS S3
- **Email**: SendGrid
- **AI Services**: Google Vision API, AWS Rekognition
- **KYC**: Yoti

Frontend

- **Framework**: Next.js 14 (App Router)
- **UI Library**: Radix UI + Tailwind CSS
- **State Management**: TanStack Query (React Query)
- **Charts**: Recharts
- **Forms**: Zod validation

DevOps

- **Containerization**: Docker & Docker Compose
- **Monorepo**: Turborepo
- **CI/CD**: GitHub Actions
- **Monitoring**: Sentry, Datadog

📋 Prerequisites

```bash
Node.js >= 18.0.0
npm >= 9.0.0
PostgreSQL >= 15.x
Redis >= 7.x
Docker & Docker Compose (optional)
```

Required Accounts

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

🚀 Quick Start

1. Clone & Install

```bash
git clone https://github.com/your-org/oliver-platform.git
cd oliver-platform
npm install
```

2. Environment Setup

```bash
# Copy environment templates
cp apps/api/.env.example apps/api/.env
cp apps/web/.env.local.example apps/web/.env.local

# Edit the .env files with your credentials
```

3. Database Setup

```bash
# Start PostgreSQL and Redis
docker-compose up -d postgres redis

# Run migrations and seed
npm run prisma:migrate
npm run prisma:seed
```

4. Start Development Servers

```bash
# Start all services
npm run dev

# Or start individually
cd apps/api && npm run dev      # Backend on :4000
cd apps/web && npm run dev      # Frontend on :3000
```

5. Access the Application

- **Frontend**: http://localhost:3000
- **Backend API**: http://localhost:4000
- **API Docs**: http://localhost:4000/api/docs
- **Prisma Studio**: `npm run prisma:studio`

Default Credentials

Admin:
Email: admin@oliver.com
Password: Admin123!

Creator (Test):
Email: creator@test.com
Password: Test123!

Fan (Test):
Email: fan@test.com
Password: Test123!

📦 Project Structure

```
oliver-platform/
├── apps/
│   ├── api/              # NestJS backend
│   │   ├── src/
│   │   │   ├── modules/
│   │   │   │   ├── admin/    # Admin module
│   │   │   │   ├── auth/     # Authentication
│   │   │   │   ├── database/ # Prisma
│   │   │   │   ├── cache/    # Redis
│   │   │   │   ├── storage/  # S3
│   │   │   │   └── email/    # SendGrid
│   │   │   └── main.ts
│   │   └── Dockerfile
│   │
│   └── web/              # Next.js frontend
│       ├── app/
│       │   └── (admin)/  # Admin pages
│       ├── components/   # UI components
│       ├── hooks/        # React hooks
│       ├── lib/          # Utilities
│       └── Dockerfile
│
├── packages/
│   └── database/         # Shared Prisma schema
│
├── docker-compose.yml
├── turbo.json
└── package.json
```

🔧 Available Scripts

Root Commands

bash

```bash
npm run dev          # Start all services in dev mode
npm run build        # Build all apps
npm run start        # Start all apps in production
npm run lint         # Lint all workspaces
npm run test         # Run all tests
npm run format       # Format code with Prettier
npm run setup        # Complete setup (install + migrate + seed)
```

Docker Commands

```bash
npm run docker:dev      # Start dev environment
npm run docker:prod     # Start production environment
npm run docker:down     # Stop all containers
```

Database Commands

```bash
npm run prisma:generate  # Generate Prisma Client
npm run prisma:migrate   # Run migrations
npm run prisma:deploy    # Deploy migrations (prod)
npm run prisma:seed      # Seed database
npm run prisma:studio    # Open Prisma Studio
```

🐳 Docker Deployment

Development

```bash
# Start all services
docker-compose up -d

# View logs
docker-compose logs -f

# Stop services
docker-compose down
```

Production

```bash
# Build images
docker-compose -f docker-compose.prod.yml build

# Start production stack
docker-compose -f docker-compose.prod.yml up -d

# Scale services
docker-compose -f docker-compose.prod.yml up -d --scale api=3
```

🌐 Environment Variables

Backend (apps/api/.env)

bash

```
# Database
DATABASE_URL="postgresql://user:password@localhost:5432/oliver_db"

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

# JWT
JWT_SECRET="your-super-secret-key-min-32-characters"
JWT_EXPIRY="7d"

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

# Google Vision
GOOGLE_VISION_KEY_PATH="/path/to/credentials.json"

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

# SendGrid
SENDGRID_API_KEY="SG.xxxxxxxxxxxxx"
FROM_EMAIL="no-reply@oliver.com"
SUPPORT_EMAIL="support@oliver.com"

# Application
PORT=4000
NODE_ENV="development"
FRONTEND_URL="http://localhost:3000"
```

Frontend (apps/web/.env.local)

```bash
NEXT_PUBLIC_API_URL="http://localhost:4000/api"
NEXT_PUBLIC_APP_URL="http://localhost:3000"
```

📚 API Documentation

Once the backend is running, access the interactive API documentation:

Swagger UI: http://localhost:4000/api/docs

Key Endpoints

```
POST   /api/auth/login              # Admin login
GET    /api/admin/dashboard/metrics # Dashboard metrics
GET    /api/admin/users             # List users
POST   /api/admin/users/:id/suspend # Suspend user
GET    /api/admin/kyc/pending       # KYC queue
POST   /api/admin/kyc/:id/approve   # Approve KYC
GET    /api/admin/transactions      # Transactions list
POST   /api/admin/accounting/export # Export financial data
GET    /api/admin/audit-log         # Audit logs
PATCH  /api/admin/settings          # Update settings
```

🧪 Testing

```bash
# Unit tests
npm run test

# E2E tests
npm run test:e2e

# Coverage
npm run test:cov
```

🚀 Production Deployment

1. Build Docker Images

```bash
# Tag with version
export VERSION=1.0.0

# Build and tag
docker build -t oliver-api:$VERSION -f apps/api/Dockerfile .
docker build -t oliver-web:$VERSION -f apps/web/Dockerfile .

# Push to registry
docker push your-registry.com/oliver-api:$VERSION
docker push your-registry.com/oliver-web:$VERSION
```

2. Deploy to Server

```bash
# SSH to production server
ssh user@production-server

# Pull images
docker pull your-registry.com/oliver-api:$VERSION
docker pull your-registry.com/oliver-web:$VERSION

# Update docker-compose
export VERSION=1.0.0
docker-compose -f docker-compose.prod.yml up -d

# Run migrations
docker exec oliver-api-prod npx prisma migrate deploy
```

3. Health Checks

```bash
# API health
curl http://your-domain.com/api/health

# Database check
docker exec oliver-postgres-prod pg_isready

# Redis check
docker exec oliver-redis-prod redis-cli ping
```

🔐 Security Best Practices

1. **Environment Variables**
   - Never commit `.env` files
   - Use strong, random secrets (min 32 chars)
   - Rotate credentials regularly
2. **Database**
   - Use strong passwords
   - Enable SSL connections in production
   - Regular backups (automated daily)
3. **API**
   - HTTPS only in production
   - Rate limiting enabled
   - CORS properly configured
   - Input validation with Zod
4. **Authentication**
   - JWT with short expiry (7d)
   - Refresh token rotation
   - 2FA for admin users
   - IP whitelisting option

📊 Monitoring & Logging

Sentry Integration

```bash
# Set in .env
SENTRY_DSN="your-sentry-dsn"
```

Datadog Integration

```bash
# Set in .env
DATADOG_API_KEY="your-datadog-key"
```

Logs

```bash
# View API logs
docker logs -f oliver-api

# View Web logs
docker logs -f oliver-web

# Database logs
docker logs -f oliver-postgres
```

🤝 Contributing

We welcome contributions! Please see our Contributing Guide.

Development Workflow

1. Fork the repository
2. Create a feature branch: `git checkout -b feature/amazing-feature`
3. Commit changes: `git commit -m 'Add amazing feature'`
4. Push to branch: `git push origin feature/amazing-feature`
5. Open a Pull Request

Code Standards

- TypeScript strict mode
- ESLint + Prettier
- Conventional Commits
- 80%+ test coverage

📝 License

👥 Team

- **Product Owner**: Your Name
- **Tech Lead**: Your Name
- **Backend**: Your Name
- **Frontend**: Your Name

📞 Support

- **Documentation**: docs.oliver.com
- **Email**: support@oliver.com
- **Slack**: #oliver-support
- **GitHub Issues**: Issues

🗺️ Roadmap

Q1 2025

- [ ] Mobile app admin
- [ ] Advanced analytics dashboard
- [ ] Multi-language support
- [ ] Automated KYC with ML

Q2 2025

- [ ] Real-time notifications
- [ ] Advanced reporting
- [ ] Webhook system
- [ ] GraphQL API

Q3 2025

- [ ] White-label solution
- [ ] Plugin system
- [ ] Advanced permissions
- [ ] Multi-tenant support

📈 Performance

- **API Response Time**: < 100ms (avg)
- **Page Load Time**: < 2s
- **Lighthouse Score**: 95+
- **Uptime**: 99.9%

🙏 Acknowledgments

- NestJS - Backend framework
- Next.js - Frontend framework
- Prisma - Database ORM
- Radix UI - UI components
- Tailwind CSS - Styling

Built with ❤️ by the Oliver Team

🚀 Quick Links

- Installation Guide
- API Documentation
- Architecture Guide
- Deployment Guide
- Security Guide
- Troubleshooting

Last updated: October 2025