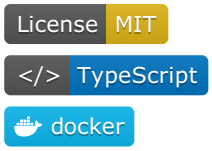
KwachaNow - African Economic & Cultural Platform

> A modern, full-stack web application showcasing African countries' economic data, cultural insights, and interactive features powered by AI.

Build Status



About KwachaNow

KwachaNow is a comprehensive platform dedicated to showcasing the economic potential and rich cultural heritage of African nations. The platform provides real-time economic data, interactive maps, cultural insights, and AI-powered assistance to help users explore and understand the African continent's diverse landscape.

Key Features

- **Interactive Africa Map**: Explore countries with detailed economic and cultural data
- **Real-time Economic Data**: Currency rates, market information, and economic indicators
- **Cultural Insights**: Traditional practices, modern developments, and cultural heritage
- **AI-Powered Chat**: Get instant answers about African countries and cultures
- **Travel Information**: Comprehensive travel guides and recommendations
- **Business Intelligence**: Market opportunities and business insights

- **Community Features**: User profiles and community engagement tools

Quick Start

Prerequisites

- **Node.js** 18+ and npm/pnpm
- **Docker** and Docker Compose (for containerized development)
- **PostgreSQL** (or use Docker)

Installation

1. **Clone the repository**
   `bash git clone https://github.com/kwachanow/kwachanow.git cd kwachanow`
2. **Install dependencies**
   `bash npm install`
3. **Environment setup**
   `bash cp .env.example .env # Edit .env with your configuration`
4. **Database setup**
   `bash npx prisma migrate dev npx prisma db seed`
5. **Start development servers**
   `bash npm run dev`

The application will be available at `http://localhost:5173`

Docker Development

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

# View logs
docker-compose logs -f

# Stop services
docker-compose down
```

Project Structure

```
kwachanow-app/
├── src/                    # Frontend source code
│   ├── components/         # React components
│   ├── pages/              # Page components
│   ├── hooks/              # Custom React hooks
│   ├── utils/              # Utility functions
│   ├── types/              # TypeScript type definitions
│   └── styles/             # Global styles
├── public/                 # Static assets
│   ├── images/             # Image assets
│   ├── css/                # Legacy CSS files
│   └── js/                 # Legacy JavaScript files
├── server/                 # Backend source code
│   ├── routes/             # API route handlers
│   ├── middleware/         # Express middleware
│   ├── utils/              # Server utilities
│   └── types/              # Backend type definitions
├── prisma/                 # Database schema and migrations
│   ├── schema.prisma       # Database schema
│   ├── migrations/         # Database migrations
│   └── seed.ts             # Database seeding script
├── tests/                  # Test files
│   ├── unit/               # Unit tests
│   ├── integration/        # Integration tests
│   └── e2e/                # End-to-end tests
├── docker/                 # Docker configurations
├── .github/                # GitHub Actions workflows
└── docs/                   # Documentation
```

Technology Stack

Frontend

- **React 18** - UI library
- **TypeScript** - Type safety
- **Vite** - Build tool and dev server
- **Tailwind CSS** - Utility-first CSS framework
- **React Router** - Client-side routing

Backend

- **Express.js** - Web framework
- **TypeScript** - Type safety
- **Prisma** - Database ORM
- **PostgreSQL** - Primary database
- **JWT** - Authentication

DevOps & Tools

- **Docker** - Containerization
- **GitHub Actions** - CI/CD
- **ESLint** - Code linting
- **Prettier** - Code formatting
- **Jest/Vitest** - Testing
- **Husky** - Git hooks

API Endpoints

Core APIs

- `GET /api/countries` - List all African countries
- `GET /api/countries/:code` - Get specific country data
- `POST /api/chat` - AI chat functionality
- `GET /api/news?country=:code` - Country-specific news
- `GET /api/exchange-rates` - Current exchange rates

Authentication

- `POST /api/auth/login` - User login
- `POST /api/auth/register` - User registration
- `POST /api/auth/logout` - User logout
- `GET /api/auth/profile` - User profile

User Features

- `GET /api/users/profile` - Get user profile
- `PUT /api/users/profile` - Update user profile
- `GET /api/users/saved` - Get saved content
- `POST /api/users/save` - Save content

For detailed API documentation, visit `/api/docs` when the server is running.

Testing

Running Tests

```
# Run all tests
npm test

# Run tests in watch mode
npm run test:watch

# Run tests with coverage
npm run test:coverage

# Run E2E tests
npm run test:e2e
```

Test Types

- **Unit Tests**: Individual component and function testing
- **Integration Tests**: API endpoint and database testing
- **E2E Tests**: Full user journey testing
- **Visual Regression Tests**: UI consistency testing

Deployment

Production Build

```
# Build frontend and backend
npm run build

# Start production server
npm start
```

Docker Deployment

```
# Build production image
docker build -t kwachanow:latest .

# Run container
docker run -p 3000:3000 --env-file .env kwachanow:latest
```

Platform Deployments

Heroku

```
heroku create kwachanow-app
heroku config:set NODE_ENV=production
git push heroku main
```

Render

```
# Use the Procfile for automatic deployment
# Configure environment variables in Render dashboard
```

Railway

```
railway login
railway new
railway up
```

Environment Variables

Create a `.env` file based on `.env.example`:

```
# Database
DATABASE_URL="postgresql://username:password@localhost:5432/
kwachanow"

# Server
PORT=3000
NODE_ENV=development

# External APIs
OPENAI_API_KEY="your-openai-api-key"
NEWS_API_KEY="your-news-api-key"
CURRENCY_API_KEY="your-currency-api-key"

# Authentication
JWT_SECRET="your-jwt-secret"
JWT_EXPIRES_IN="7d"

# CORS
CORS_ORIGIN="http://localhost:5173"

# Redis (optional)
REDIS_URL="redis://localhost:6379"

# Email (optional)
SMTP_HOST="smtp.gmail.com"
SMTP_PORT=587
SMTP_USER="your-email@gmail.com"
SMTP_PASS="your-app-password"
```

Performance

Optimization Features

- **Code Splitting**: Automatic route-based splitting
- **Image Optimization**: WebP format with fallbacks
- **Caching**: Redis-based API response caching
- **CDN**: Static asset delivery optimization
- **Database**: Query optimization and indexing

Monitoring

- **Health Checks**: `/health` endpoint
- **Metrics**: Performance and usage analytics
- **Error Tracking**: Comprehensive error logging
- **Uptime Monitoring**: Service availability tracking

Contributing

We welcome contributions! Please see our Contributing Guide for details.

Development Workflow

1. Fork the repository
2. Create a feature branch
3. Make your changes
4. Add tests
5. Submit a pull request

Code Quality

```
# Lint code
npm run lint

# Format code
npm run format

# Type check
npm run type-check

# Pre-commit checks
npm run pre-commit
```

License

This project is licensed under the MIT License - see the LICENSE file for details.

Security

For security concerns, please review our Security Policy.

Support

- **Documentation**: docs.kwachanow.com
- **Issues**: GitHub Issues
- **Discussions**: GitHub Discussions
- **Email**: support@kwachanow.com

Acknowledgments

- **African Development Bank** - Economic data sources
- **OpenAI** - AI chat functionality
- **Mapbox** - Interactive mapping services
- **Contributors** - Community development efforts

Built with care for Africa by the KwachaNow Team

Last updated: January 2025