AviataChain - Unified Airline Loyalty Platform

A decentralized airline loyalty platform built on the Cardano blockchain, leveraging Blockfrost API, Mesh SDK, and MeshJS to create a unified, interoperable rewards ecosystem for airlines and travelers.

Please visit our video https://youtu.be/1F-MGQkMqLo?si=6FDOQrSBiyk1r3al

☀️ Overview

The Unified Airline Loyalty Platform revolutionizes traditional airline reward programs by creating a blockchain-based ecosystem where loyalty points are tokenized, transferable, and usable across multiple airline partners. Built on Cardano's sustainable blockchain infrastructure, the platform ensures security, transparency, and true ownership of rewards.

✈️ Key Features

- **Multi-Airline Integration**: Seamlessly earn and redeem points across partner airlines

- **Tokenized Loyalty Points**: Blockchain-based tokens representing loyalty rewards

- **Smart Contract Automation**: Automated reward distribution and redemption

- **Real-time Tracking**: Live balance updates and transaction history

- **Cross-Platform Compatibility**: Web and mobile-friendly interface

- **Secure Wallet Integration**: Connect with Cardano wallets for secure transactions

- **Partnership Rewards**: Bonus points for cross-airline bookings and partnerships
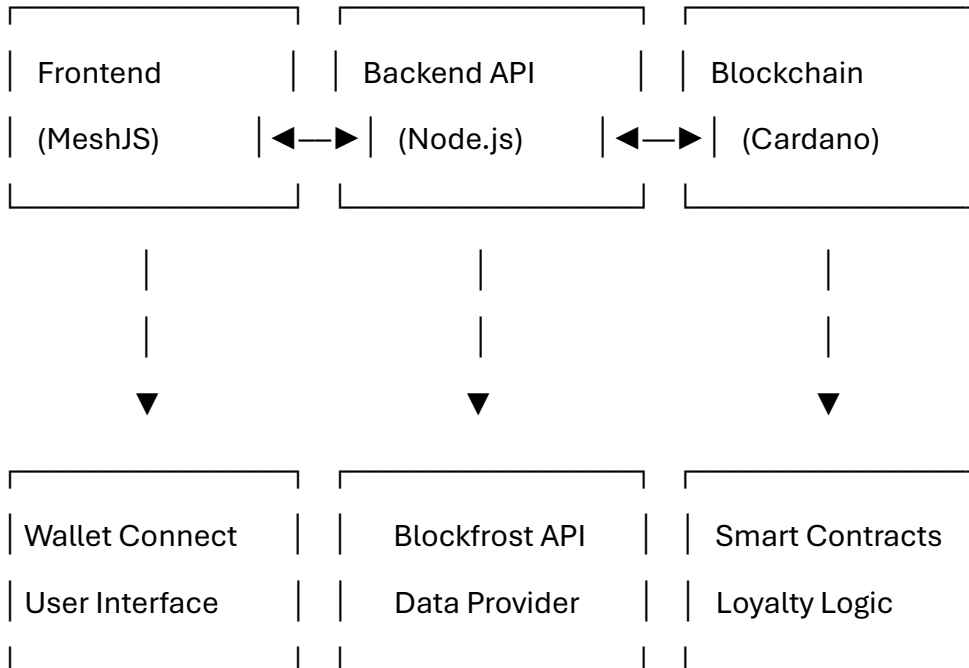
🏛️ Architecture

📁 Project Structure

AviataChain (Unified airline loyalty platform)

```
├── smart-contracts/
│   ├── loyalty-token/
│   │   ├── LoyaltyToken.hs
│   │   └── TokenPolicy.hs
```

```
|   ├── airline-registry/
|   |   └── AirlineRegistry.hs
|   ├── point-exchange/
|   |   └── PointExchange.hs
|   └── redemption/
|       └── Redemption.hs
├── typescript - frontend/
|   ├── src/
|   |   ├── components/
|   |   |   ├── LaceWalletConnector.tsx
|   |   |   ├── MultiWalletConnector.tsx
|   |   |   ├── SendAda.tsx
|   |   |   ├── WalletConnector.tsx
|   |   |   ├── nft/
|   |   |   |   ├── NFTCard.tsx
|   |   |   |   ├── NFTRewardsGrid.tsx
|   |   |   |   ├── NFTRewardsHeader.tsx
|   |   |   |   └── UserNFTCollection.tsx
|   |   ├── views/
|   |   |   ├── FlightsView.tsx
|   |   |   ├── NFTMintView.tsx
|   |   |   ├── PartnersView.tsx
|   |   |   ├── ProfileView.tsx
|   |   |   ├── RewardsView.tsx
|   |   |   ├── TransferView.tsx
```

```
│  │  │  └── WalletView.tsx
│  │  ├── data/
│  │  │  ├── interfaces.ts
│  │  │  ├── Prototypes.ts
│  │  │  └── types.ts
│  │  ├── contexts/
│  │  │  └── MeshProvider.tsx
│  │  ├── App.tsx
│  │  ├── App.css
│  │  ├── Index.tsx
│  │  ├── Index.css
│  │  └── App.test.tsx
│  ├── package.json
│  ├── tsconfig.json
│  ├── craco.config.js
│  └── README.md
│  │  │  └── Transaction.ts
│  │  └── app.ts
│  ├── package.json
│  └── README.md
```

```
┌─────────────┐   ┌─────────────┐   ┌─────────────┐
│ Frontend    │   │ Backend API │   │ Blockchain  │
│ (MeshJS)    │◄─►│ (Node.js)   │◄─►│ (Cardano)   │
└─────────────┘   └─────────────┘   └─────────────┘
      │                 │                 │
      │                 │                 │
      ▼                 ▼                 ▼
┌─────────────┐   ┌─────────────┐   ┌─────────────┐
│Wallet Connect│  │Blockfrost API│  │Smart Contracts│
│User Interface│  │Data Provider │  │Loyalty Logic │
└─────────────┘   └─────────────┘   └─────────────┘
```

🛠️ Technology Stack

- **Blockchain**: Cardano

- **Smart Contracts**: Plutus (Haskell)

- **API Integration**: Blockfrost API

- **Frontend Framework**: MeshJS + React/Next.js

- **Wallet Integration**: Mesh SDK

- **Backend**: Node.js + Express

- **Database**: MongoDB/PostgreSQL

- **Authentication**: JWT + Wallet signatures

📋 Prerequisites

Before running this project, ensure you have:

- Node.js (v16 or higher)

- npm or yarn package manager

- Cardano wallet (Nami, Eternl, Flint, etc.)

- Blockfrost API key

- Git

🚀 Installation

1. **Clone the repository**

bash

git clone https://github.com/your-username/unified-airline-loyalty.git

cd unified-airline-loyalty

2. **Install dependencies**

bash

npm install

 # or

yarn install

npm install @meshsdk/core @meshsdk/react @meshsdk/wallet npm install @blockfrost/blockfrost-js npm install next react react-dom typescript @types/react @types/node npm install lucid-cardano

npm install -D @types/react-dom eslint prettier

npm install @meshsdk/core

npm install @meshsdk/react

3. **Environment setup**

bash

cp .env.example .env

4. **Configure environment variables**

env

Blockfrost Configuration

BLOCKFROST_PROJECT_ID=your_blockfrost_project_id

BLOCKFROST_NETWORK=testnet # or mainnet

Database Configuration

DATABASE_URL=your_database_connection_string

JWT Configuration

JWT_SECRET=your_jwt_secret_key

Application Configuration

PORT=3000

NODE_ENV=development

npx create-react-app aviatachain-loyalty-framework --template typescript

Smart Contract Addresses

LOYALTY_TOKEN_POLICY_ID=your_token_policy_id

REWARDS_CONTRACT_ADDRESS=your_contract_address

5. **Initialize the database**

bash

npm run db:migrate

npm run db:seed

6. **Start the development server**

bash

npm run dev

Visit http://localhost:3000 to access the application.

🔧 **Key Dependencies**

json

```json
{
  "dependencies": {
    "@meshsdk/core": "^1.5.0",
    "@meshsdk/react": "^1.1.0",
    "@meshsdk/wallet": "^1.3.0",
    "next": "^13.0.0",
    "react": "^18.0.0",
    "axios": "^1.4.0",
    "cardano-serialization-lib": "^11.0.0"
  },
  "devDependencies": {
    "@types/node": "^18.0.0",
    "@types/react": "^18.0.0",
    "typescript": "^5.0.0",
    "tailwindcss": "^3.3.0"
  }
}
```

🎯 Core Functionality

1. Wallet Connection

2. Loyalty Token Operations

3. Cross-Airline Redemption

Contract Deployment

bash

 # Deploy smart contracts to testnet

npm run deploy:contracts:testnet

 # Deploy to mainnet

npm run deploy:contracts:mainnet

🔐 Security Features

- **Multi-signature Wallets**: Enhanced security for airline treasury management

- **Time-locked Contracts**: Prevent unauthorized point manipulation

- **Audit Trail**: Complete transaction history on blockchain

- **Rate Limiting**: API protection against abuse

- **Encrypted Communication**: All API communications are encrypted

🧪 Testing

bash

 # Run unit tests

npm run test

 # Run integration tests

npm run test:integration

 # Run end-to-end tests

npm run test:e2e

 # Test smart contracts

npm run test:contracts

🌐 Deployment

Development Deployment

bash

npm run build

npm run start

Production Deployment

bash

 # Build for production

npm run build:production

 # Deploy to cloud provider

npm run deploy:production

Environment-Specific Configurations

- **Testnet**: Uses Cardano testnet for development and testing

- **Mainnet**: Production environment with real ADA transactions

🤝 Contributing

1. Fork the repository

2. Create a feature branch (git checkout -b feature/amazing-feature)

3. Commit your changes (git commit -m 'Add amazing feature')

4. Push to the branch (git push origin feature/amazing-feature)

5. Open a Pull Request

Development Guidelines

- Follow TypeScript best practices

- Write comprehensive tests for new features

- Update documentation for API changes

- Ensure smart contract security audits

📄 License

This project is licensed under the MIT License - see the [LICENSE](#) file for details.

🗺️ Roadmap

Phase 1 (Current)

- Basic wallet integration
- Loyalty token minting
- Simple redemption system
- Multi-airline partnership integration

Phase 2 (Q3 2025)

- Mobile application
- Advanced smart contract features
- Cross-chain compatibility
- Enterprise airline onboarding

Phase 3 (Q4 2025)

- AI-powered personalized rewards
- NFT-based premium memberships
- DeFi integration for staking rewards
- Global airline network expansion

🏆 Acknowledgments

- **Cardano Foundation** - For the robust blockchain infrastructure
- **Blockfrost** - For reliable API services
- **MeshJS Team** - For excellent developer tools
- **Partner Airlines** - For collaboration and integration support

---

Built with ❤️ on Cardano blockchain