**Real-Time Financial Insights Dashboard - Project Documentation**

**Table of Contents**

1. Project Overview

2. Technology Stack

3. Database Design

4. API Documentation

6. Components

7. Authentication System

8. Real-Time Features

9. Deployment Guide

10. Testing Strategy

11. Performance Optimization

**Project Overview**

**Purpose**

The Real-Time Financial Insights Dashboard is a comprehensive web application designed to provide users with real-time stock market data, portfolio management capabilities, price alerts, and AI-powered market predictions.

**Key Objectives**

- **Real-time Data**: Provide live stock market data using Alpha Vantage API

- **Portfolio Management**: Allow users to track their investments with profit/loss calculations

- **Predictive Analytics**: Implement time-series forecasting for stock price predictions

- **Alert System**: Notify users when stocks reach target prices

- **User Experience**: Modern, responsive interface with intuitive navigation

**Target Users**

- Individual investors and traders

- Financial enthusiasts

- Portfolio managers

- Students learning about financial markets

**Data Flow**

1. **User Authentication**: JWT-based authentication with OAuth support

2. **Market Data**: Real-time stock prices from Alpha Vantage API

3. **Portfolio Calculations**: Server-side computation of P&L and allocations

4. **Predictions**: Time-series analysis using historical data

5. **Alerts**: Background monitoring with real-time notifications

**Technology Stack**

**Frontend Technologies**

|  |  |  |
| --- | --- | --- |
| Technology | Version | Purpose |
| React.js | 18.x | UI Framework |
| Vite | 4.x | Build Tool |
| React Router | 6.x | Client-side Routing |
| Axios | 1.x | HTTP Client |
| CSS3 | - | Styling |

**Backend Technologies**

|  |  |  |
| --- | --- | --- |
| Technology | Version | Purpose |
| Node.js | 18.x | Runtime Environment |
| Express.js | 4.x | Web Framework |
| PostgreSQL | 14.x | Database |
| JWT | 9.x | Authentication |
| Passport.js | 0.6.x | OAuth |
| Bcrypt | 5.x | Password Hashing |

**External Services**

|  |  |  |
| --- | --- | --- |
| Service | Purpose | Rate Limits |
| Alpha Vantage | Stock Market Data | 5 calls/min, 500/day |
| Google OAuth | Social Authentication | - |
| GitHub OAuth | Social Authentication | - |

**Database Design**

**Table Schemas**

**Users Table**

CREATE TABLE users (

id SERIAL PRIMARY KEY,

username VARCHAR(255) NOT NULL,

email VARCHAR(255) UNIQUE NOT NULL,

password VARCHAR(255),

created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP

);

**Portfolio Table**

CREATE TABLE portfolio (

p\_id SERIAL PRIMARY KEY,

user\_id INTEGER REFERENCES users(id),

ticker VARCHAR(10) NOT NULL,

quantity DECIMAL NOT NULL,

buy\_price DECIMAL NOT NULL,

created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP

);

**Alerts Table**

CREATE TABLE alerts (

alert\_id SERIAL PRIMARY KEY,

user\_id INTEGER REFERENCES users(id),

ticker VARCHAR(10) NOT NULL,

comparator VARCHAR(5) NOT NULL,

threshold DECIMAL NOT NULL,

notification\_type VARCHAR(20) DEFAULT 'in-app',

is\_active BOOLEAN DEFAULT TRUE,

created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP

);

**API Documentation**

**Authentication Endpoints**

|  |  |  |  |
| --- | --- | --- | --- |
| Method | Endpoint | Description | Auth Required |
| POST | `/api/auth/signup` | User registration | No |
| POST | `/api/auth/login` | User login | No |
| GET | `/api/auth/me` | Get current user | Yes |
| POST | `/api/auth/logout` | User logout | Yes |

**Portfolio Endpoints**

|  |  |  |  |
| --- | --- | --- | --- |
| Method | Endpoint | Description | Auth Required |
| GET | `/api/portfolio/` | | Get user portfolio | Yes |
| POST | `/api/portfolio/add` | Add new asset | Yes |
| DELETE | `/api/portfolio/:id` | Delete asset | Yes |

**Alerts Endpoints**

|  |  |  |  |
| --- | --- | --- | --- |
| Method | Endpoint | Description | Auth Required |
| GET | `/api/alerts/user/:userId` | Get user alerts | Yes |
| POST | `/api/alerts/add` | Create new alert | Yes |
| DELETE | `/api/alerts/:id` | Delete alert | Yes |
| PATCH | `/api/alerts/:id/toggle` | Toggle alert status | Yes |
| GET | `/api/alerts/check/:userId` | Check triggered alerts | No |

**Market Data Endpoints**

|  |  |  |  |
| --- | --- | --- | --- |
| Method | Endpoint | Description | Auth Required |
| GET | `/api/market/:symbol` | Get stock chart data | No |
| GET | `/api/market/quote/:symbol` | Get current stock price | No |

**Insights Endpoints**

|  |  |  |  |
| --- | --- | --- | --- |
| Method | Endpoint | Description | Auth Required |
| GET | `/api/insights/:userId` | Get portfolio predictions | Yes |

**Key Components**

**Dashboard Component**

- **Purpose**: Main landing page with overview

- **Features**: Stock cards, charts, portfolio summary

- **Data Sources**: Multiple API endpoints

**Portfolio Component**

- **Purpose**: Detailed portfolio management

- **Features**: Add/delete assets, P&L calculations, allocations

- **Real-time Updates**: Live price updates

**Alerts Component**

- **Purpose**: Price alert management

- **Features**: Create/delete alerts, real-time notifications

- **Alert Types**: Price thresholds with various operators

**Insights Component**

- **Purpose**: AI-powered market predictions

- **Features**: Trend analysis, confidence scores, forecasting

- **Algorithms**: Linear regression, volatility analysis

**Authentication System**

**JWT Implementation**

- **Token Generation**: On successful login

- **Token Validation**: Middleware for protected routes

- **Token Refresh**: Automatic renewal

- **Expiration**: 24-hour default

**OAuth Integration**

- **Google OAuth**: Passport.js strategy

- **GitHub OAuth**: Passport.js strategy

- **Callback Handling**: Frontend token storage

- **User Creation**: Automatic account creation

**Security Measures**

- **Password Hashing**: Bcrypt with salt rounds

- **CORS Configuration**: Restricted origins

- **Input Validation**: Server-side validation

- **SQL Injection Prevention**: Parameterized queries

**Real-Time Features**

**Alert System**

- **Background Monitoring**: 30-second intervals

- **Real-time Notifications**: Toast notifications

- **Alert Deactivation**: Automatic after trigger

- **Multiple Conditions**: <, >, = operators

**Live Data Updates**

- **Portfolio Prices**: Real-time market data

- **Dashboard Refresh**: Automatic updates

**Prediction Engine**

- **Time-series Analysis**: Historical data processing

- **Trend Classification**: Bullish/Bearish/Neutral

- **Confidence Scoring**: Volatility-based confidence

- **Multiple Algorithms**: Linear regression, moving averages

**Deployment Guide**

**Production Environment**

**Backend deployment**

cd backend

npm install --production

npm start

**Frontend deployment**

cd frontend

npm run build

Serve dist/ folder with nginx/apache

**Environment Variables**

*Production .env*

NODE\_ENV=production

DATABASE\_URL=postgresql://user:pass@host:5432/db

JWT\_SECRET=secure\_random\_string

ALPHA\_VANTAGE\_KEY=api\_key

FRONTEND\_URL=https://yourdomain.com

**Database Migration**

-- Production database setup

CREATE DATABASE financialdb\_prod;

-- Run all table creation scripts

-- Set up proper indexes

CREATE INDEX idx\_portfolio\_user\_id ON portfolio(user\_id);

CREATE INDEX idx\_alerts\_user\_id ON alerts(user\_id);

**Testing Strategy**

**Unit Testing**

- **Backend**: Jest for API endpoints

- **Frontend**: React Testing Library

- **Database**: Mock database connections

- **Services**: Mock external API calls

**Integration Testing**

- **API Integration**: Full request/response cycle

- **Database Integration**: Real database operations

- **Authentication Flow**: Complete auth process

**Performance Testing**

- **Load Testing**: Multiple concurrent users

- **API Rate Limits**: Alpha Vantage limits

- **Database Performance**: Query optimization

**Performance Optimization**

**Frontend Optimization**

- **Code Splitting**: Route-based splitting

- **Lazy Loading**: Component lazy loading

- **Caching**: API response caching

- **Minification**: Production builds

**Backend Optimization**

- **Database Indexing**: Query optimization

- **Connection Pooling**: PostgreSQL pools

- **Caching**: Redis for frequent data

- **Rate Limiting**: API call optimization

**API Optimization**

- **Batch Requests**: Multiple tickers per call

- **Fallback Data**: Cached prices when API fails

- **Error Handling**: Graceful degradation

- **Retry Logic**: Automatic retry mechanisms

**Conclusion**

The Real-Time Financial Insights Dashboard represents a comprehensive solution for modern portfolio management and market analysis. The system successfully integrates real-time data, predictive analytics, and user-friendly interfaces to provide valuable insights for financial decision-making.

**Key Achievements**

- Real-time market data integration

- Secure user authentication system

- Comprehensive portfolio management

- AI-powered market predictions

- Responsive and modern UI

- Scalable architecture

**Future Enhancements**

- WebSocket implementation for real-time updates

- Mobile application development

- Advanced charting capabilities

- Social trading features

- Machine learning model improvements

- Multi-currency support