Esoteric Enterprises Financial Platform - Design Document

1. Executive Summary

1.1 Project Overview

Esoteric Enterprises needs an attractive, secure financial website that enables customers to:

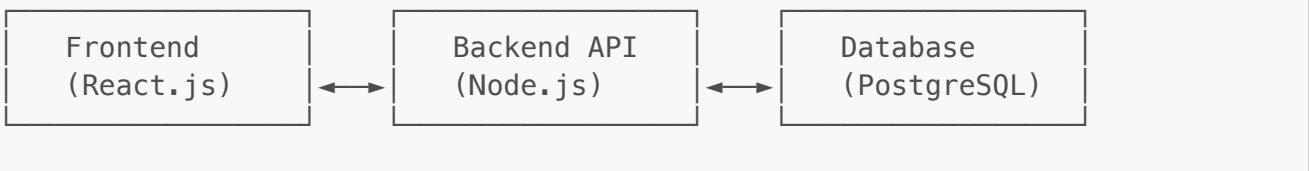
- Create accounts and log in securely with two-factor authentication
- Access a dashboard to view all important data with real-time market updates
- View their financial data in real time (balances, transactions, performance, holdings, allocations)
- Access and download important documents
- Schedule meetings with the investment team

1.2 Key Objectives

- Secure financial data protection
- Modern, intuitive user interface
- Scalable architecture for future growth
- Maintainable, well-documented codebase

2. System Architecture

2.1 System Architecture



2.2 Technology Stack

Frontend

- **Framework**: React.js
- **UI Library**: Material-UI with purple/black theme
- **Routing**: React Router
- **Charts**: Chart.js for financial visualizations
- **HTTP Client**: Axios
- **Deployment**: Netlify/Vercel

Backend

- **Runtime**: Node.js with Express.js

- **Authentication**: JWT tokens + Speakeasy for 2FA
- **Market Data**: WebSocket for real-time feeds
- **Password Security**: Bcrypt hashing
- **File Processing**: Multer
- **Validation**: Input sanitization middleware
- **Deployment**: DigitalOcean

Database

- **Database**: PostgreSQL
- **Hosting**: DigitalOcean Managed Database
- **Features**: Automated backups, connection pooling

3. Core Features

3.1 User Authentication

- Account registration with email verification
- Two-factor authentication (SMS/authenticator apps)
- JWT token authentication
- Password reset functionality
- Session management

3.2 Customer Portal

Dashboard

- Real-time account balances in purple-themed cards
- Live market data feed with stock prices and indices
- Recent transaction history
- Interactive portfolio performance charts
- Market news and alerts

Account Management

- Personal profile management
- Multi-account overview
- Contact information updates

Financial Data

- Transaction history with filtering
- Statement generation and PDF download
- Portfolio performance tracking
- Holdings overview with allocations

Document Center

- Document library with categorization

- Secure document access
- Search functionality
- Download management

3.3 Admin Panel

- Customer management dashboard
- Account administration and editing
- Document management with batch upload
- System analytics and monitoring

4. Security Framework

4.1 Authentication Security

- **Two-Factor Authentication**: TOTP via authenticator apps or SMS
- **Password Protection**: Bcrypt hashing with salt rounds
- **Token Authentication**: JWT with configurable expiration
- **Rate Limiting**: Protection against brute force attacks
- **Password Policy**: Strength requirements and validation

4.2 Data Protection

- **Encryption**: TLS/SSL for all communications
- **Input Validation**: Sanitization of user inputs
- **SQL Injection Defense**: Parameterized queries
- **XSS Prevention**: Content security policies

4.3 Monitoring & Backup

- **Performance Monitoring**: Error tracking and analytics
- **Security Logging**: Audit trails for authentication
- **Automated Backups**: Daily encrypted database backups

5. Database Design

5.1 Core Tables

```sql
-- Users table
CREATE TABLE users (
    id SERIAL PRIMARY KEY,
    email VARCHAR(255) UNIQUE NOT NULL,
    password_hash VARCHAR(255) NOT NULL,
    first_name VARCHAR(100) NOT NULL,
    last_name VARCHAR(100) NOT NULL,
    phone VARCHAR(20),
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    updated_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);
```

```sql
-- Accounts table
CREATE TABLE accounts (
    id SERIAL PRIMARY KEY,
    user_id INTEGER REFERENCES users(id),
    account_number VARCHAR(50) UNIQUE NOT NULL,
    account_type VARCHAR(50) NOT NULL,
    balance DECIMAL(15,2) DEFAULT 0.00,
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);

-- Transactions table
CREATE TABLE transactions (
    id SERIAL PRIMARY KEY,
    account_id INTEGER REFERENCES accounts(id),
    amount DECIMAL(15,2) NOT NULL,
    transaction_type VARCHAR(50) NOT NULL,
    description TEXT,
    transaction_date DATE NOT NULL,
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);

-- Documents table
CREATE TABLE documents (
    id SERIAL PRIMARY KEY,
    user_id INTEGER REFERENCES users(id),
    title VARCHAR(255) NOT NULL,
    file_path VARCHAR(500) NOT NULL,
    file_size INTEGER,
    category VARCHAR(100) NOT NULL,
    upload_date TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);

-- Simple sessions tracking
CREATE TABLE user_sessions (
    id SERIAL PRIMARY KEY,
    user_id INTEGER REFERENCES users(id),
    token_hash VARCHAR(255) NOT NULL,
    expires_at TIMESTAMP NOT NULL,
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);

-- Two-factor authentication
CREATE TABLE user_2fa (
    id SERIAL PRIMARY KEY,
    user_id INTEGER REFERENCES users(id),
    secret VARCHAR(255) NOT NULL,
    is_enabled BOOLEAN DEFAULT false,
    backup_codes TEXT[],
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);

-- Market data cache
CREATE TABLE market_data (
    id SERIAL PRIMARY KEY,
```

```
    symbol VARCHAR(20) NOT NULL,
    price DECIMAL(15,4) NOT NULL,
    change_percent DECIMAL(5,2),
    last_updated TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);
```

6. API Endpoints

6.1 Authentication

```
POST /api/auth/register
POST /api/auth/login
POST /api/auth/logout
POST /api/auth/forgot-password
POST /api/auth/reset-password
POST /api/auth/setup-2fa
POST /api/auth/verify-2fa
POST /api/auth/disable-2fa
```

6.2 User & Account Management

```
GET /api/user/profile
PUT /api/user/profile
GET /api/accounts
GET /api/accounts/:id/transactions
GET /api/accounts/:id/performance
```

6.3 Documents

```
GET /api/documents
GET /api/documents/:id/download
```

6.4 Market Data

```
GET /api/market/prices
GET /api/market/symbol/:symbol
WebSocket /ws/market-updates
```

7. User Interface Design

7.1 Design Approach

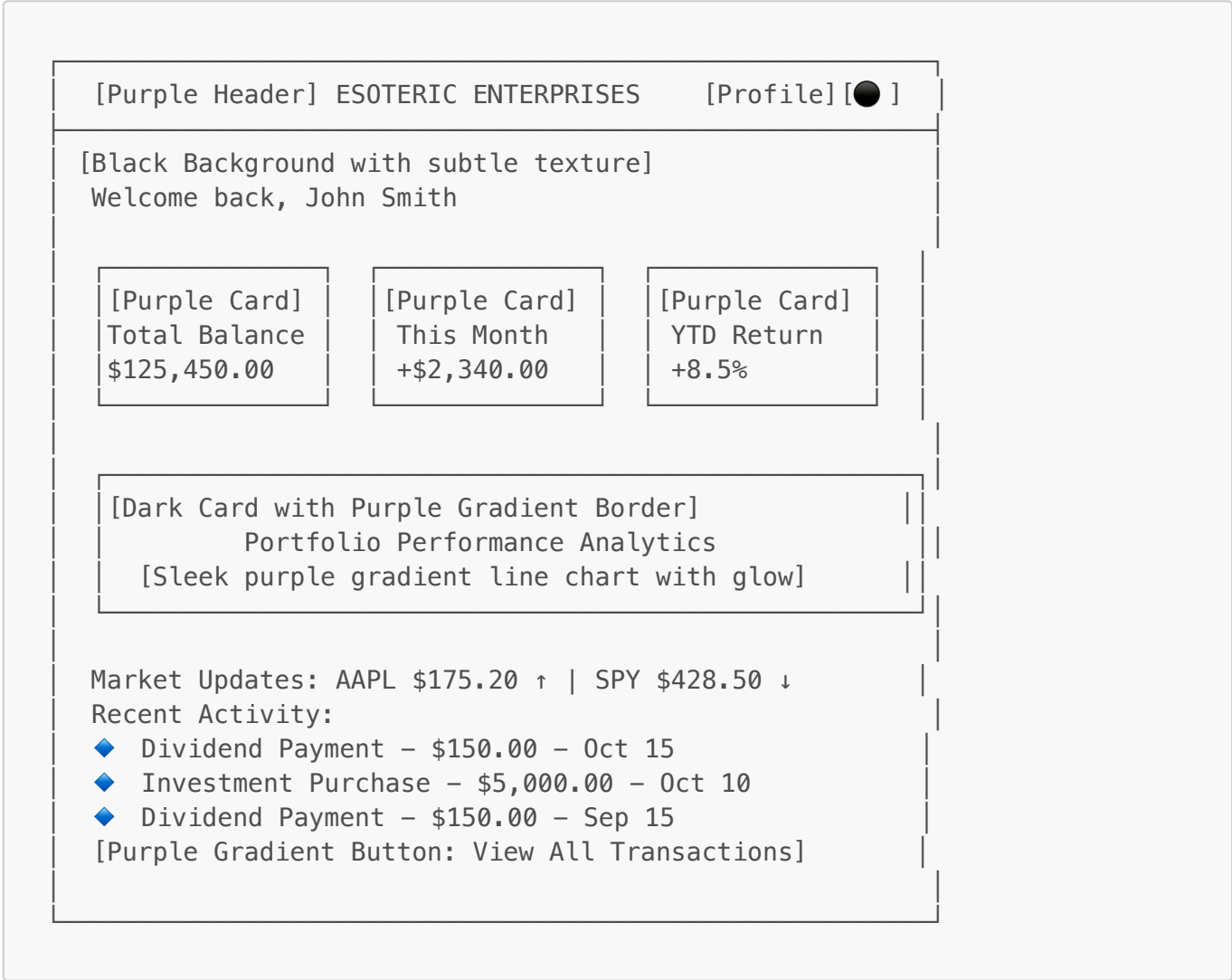
- Modern purple and black color scheme
- Professional design for financial confidence
- Fast loading with sub-second performance

7.2 Key Pages

1. **Login/Register**
2. **Dashboard**
3. **Transactions**
4. **Documents**
5. **Profile**

7.3 Dashboard Design

Color Palette: Deep purple (#6B46C1) with matte black (#1F2937).



Design Features:

- Purple gradient headers with shadows
- Matte black backgrounds
- Purple accents with hover animations
- White typography for contrast
- Card-based layout with rounded corners

- Modern icons and micro-interactions

8. Development Timeline

Phase 1: Foundation (Weeks 1-2)

- Set up development environment
- Create basic React app with routing
- Set up Node.js backend with database connection
- Implement user registration and login

Phase 2: Core Features (Weeks 3-4)

- Build user dashboard
- Create transaction display and filtering
- Implement basic account management
- Add simple charts for financial data

Phase 3: Document Management (Weeks 5-6)

- File upload system for admins
- Document categorization and storage
- Download functionality for users
- Admin panel for customer management

Phase 4: Polish & Deploy (Weeks 7-8)

- Security hardening and testing
- UI/UX improvements
- Performance optimization
- Production deployment and testing

9. Budget Breakdown

Development Costs

- **Frontend Development**: $2,200
- **Backend Development**: $3,200
- **Security Implementation (inc. 2FA)**: $2,000
- **Market Data Integration**: $800
- **Database Setup & Design**: $1,000
- **Testing & Deployment**: $800

Total Development: $10,000

Monthly Operational Costs

- **Backend Hosting** (DigitalOcean): $20/month
- **Database Hosting**: $15/month
- **Frontend Hosting** (Netlify/Vercel): Free

- **SSL Certificate**: Free (Let's Encrypt)
- **Domain**: $15/year

Total Monthly: ~$35-40

10. Future Features

These are some other features that you could consider implementing in the future, but wouldn't be in scope for the current website.

Advanced Features (Future Phases)

- Advanced analytics and reporting
- Mobile applications
- Automated compliance reporting
- Advanced monitoring and alerting
- Load balancing and auto-scaling
- Microservices architecture

Enterprise Security Features

- Penetration testing
- Advanced intrusion detection
- Compliance certifications
- Advanced audit logging
- Encryption at rest

11. Success Metrics

Technical Goals

- Page load times under 3 seconds
- 99% uptime
- Zero security incidents
- All core features working properly

Business Goals

- Easy customer onboarding
- Customers can access their data quickly
- Documents are easily accessible
- Admins can manage customer data efficiently

12. Risk Mitigation

Technical Risks

- **Data Loss**: Regular automated backups
- **Security Breach**: Basic security measures, regular updates
- **Server Downtime**: Choose reliable hosting provider

- **Developer Availability**: Keep code well-documented

Business Risks

- **Scope Creep**: Stick to defined features for v1
- **Budget Overrun**: 10% buffer included in budget
- **Timeline Delays**: Conservative 8-week estimate

13. Conclusion

This design delivers Esoteric Enterprises a secure financial platform with modern UI, two-factor authentication, and real-time market data. The solution provides clients with an excellent digital experience while giving administrators powerful management tools.