WealthTech Platform - Backend Implementation Documentation

Overview:

The WealthTech Platform is a backend system built using Node.js and Sequelize.

It follows a microservices architecture where each service is responsible for a specific function.

The services implemented are:

- User Service: Manages user profiles and sensitive information.

- Investment Service: Manages investment accounts linked to clients.

- Transaction Service: Handles financial transactions like deposits, withdrawals, and trades.

- Compliance Service: Tracks and logs compliance events and audits.

The platform also implements data encryption and decryption to secure sensitive user and financial data using AES-256 encryption.

System Architecture:

The system is designed with the following architecture:

1. API Gateway: A single entry point to all services.

2. Microservices: Each service (User, Investment, Transaction, Compliance) operates independently and has its own database schema.

3. Encryption: Sensitive data such as user email, phone number, transaction amounts, and investment account references are encrypted before storage and decrypted when retrieved.

4. Sequelize ORM: Used for database interaction, with lifecycle hooks to manage encryption and decryption.

1. Microservices Overview:

User Service:

- Functionality: Manages user profiles (KYC-compliant).
- Key Operations:
  - Creates a new user profile.
  - Retrieves user details (with decryption).
  - Updates user information (with encryption before updating).
- Sensitive Data: email_address, telephone_number, sources_of_wealth, annual_income are encrypted.

Investment Service:
- Functionality: Manages investment accounts linked to clients.
- Key Operations:
  - Creates a new investment account.
  - Retrieves investment account details (with decryption).
  - Updates investment account information (with encryption before updating).
- Sensitive Data: client_reference is encrypted.

Transaction Service:
- Functionality: Handles transactions such as deposits, withdrawals, and trades.
- Key Operations:
  - Creates a new transaction.
  - Retrieves transaction details (with decryption).
  - Updates transaction details (with encryption before updating).
- Sensitive Data: consideration_amount, charges_amount are encrypted.

Compliance Service:
- Functionality: Logs compliance-related events.
- Key Operations:

  - Creates new compliance logs.

  - Retrieves compliance logs (with decryption).

  - Updates compliance logs (with encryption before updating).

- Sensitive Data: log_details, response_id are encrypted.

Encryption/Decryption Workflow:

- Encryption:

  - Encryption is done using AES-256-CBC before storing data in the database (e.g., user email, telephone number, investment client reference, etc.).

  - The encryption process uses a secret key (ENCRYPTION_SECRET_KEY) and an initialization vector (IV).

- Decryption:

  - Decryption happens after fetching the data from the database using Sequelize hooks (afterFind).

  - Decrypted data is returned to the client when retrieving sensitive fields.

Sequelize Lifecycle Hooks:

We use Sequelize Lifecycle Hooks to handle encryption and decryption at the right moments:

1. beforeFind: Prepares data or modifies queries before querying the database.

2. afterFind: Decrypts sensitive fields after retrieving data from the database.

3. beforeUpdate: Encrypts sensitive fields before updating the record in the database.

API Endpoints:

1. User Service:

- Create User: POST /api/user

- Get User: GET /api/user/:id

2. Investment Service:

- Create Investment Account: POST /api/investment

- Get Investment Account: GET /api/investment/:id

3. Transaction Service:

- Create Transaction: POST /api/transaction

- Get Transaction: GET /api/transaction/:id

4. Compliance Service:

- Log Compliance Event: POST /api/compliance

- Get Compliance Log: GET /api/compliance/:id