### Project Overview: Holistic Financial Assistant

This project is a personal finance application built with a **hybrid, microservice-based architecture**. The goal is to evolve it from a foundational backend system into an intelligent, interactive, and visually engaging platform that demonstrates a unique combination of engineering, AI, and creative skills.

### Current State of the Application

Your project is currently a **robust and production-ready backend foundation**.

**Core Backend:** A Java 21 Spring Boot application that serves as the "engineering backbone." It handles all business logic and data persistence.

**Database:** A PostgreSQL database managed by Docker Compose.

**User Profile Service:** A RESTful API that allows for the creation and management of user profiles.

**Testing:** A comprehensive testing pipeline is in place. Unit tests use **Mockito**, while integration tests use **Testcontainers** to spin up a real PostgreSQL database for a reliable and isolated testing environment.

**Code Coverage:** **JaCoCo** is configured to generate code coverage reports, ensuring the quality of your test suite.

**Build Automation:** **Gradle** manages dependencies, builds the project, and runs tests.

### The Future Vision

The planned features will be implemented in a phased approach, building upon the solid backend you have created.

**Interactive UI:** The user interface will be a separate application (e.g., built with React). This UI will not just display data but will be highly interactive, featuring a conversational chat interface for the AI assistant.

**Creative & UX Layer:** This goes beyond a standard app. The UI will include elements like "AI-generated monthly infographics" that turn spending data into a shareable visual story. Future plans include exploring **AR/VR prototypes** to visualize financial data in an immersive way.

**Generative Infographics:** The plan is to create a feature that turns dry financial data into a shareable, stylized visual. For instance, your monthly spending report wouldn't just be a pie chart; it could be a "city skyline" where each building's height represents a spending category. **Playful UX:** The UI will have a modern, minimalist, and engaging feel. The "Explain Like I'm 5" mode for the AI is a perfect example of this. **AR/VR Hooks:** The UI will eventually include advanced, future-facing elements like an AR dashboard prototype that visualizes spending as a 3D environment.

**Structured AI:** The AI will reside in a separate **Python microservice**. It will use frameworks like **LangChain** to orchestrate powerful features, including:

**Retrieval-Augmented Generation (RAG):** The AI will access your financial data to provide accurate, context-aware answers. For example, when a user asks about their spending, the AI will retrieve the relevant transaction data from your backend.

**Generative AI:** Beyond text, the AI will use tools like **Hugging Face Diffusers** to create creative, visual outputs, such as AI-generated infographics of your spending habits.

### Architectural Breakdown

**UI Layer (Frontend):** This is the **planned, interactive user interface**. It will be a separate application (e.g., built with React or another frontend framework) that handles all user-facing elements, including a conversational chat window and data visualizations. This UI will communicate with your backend via its API.

**Backend Layer (Java/Spring Boot):** This is your current project. It's the **foundation of the entire system**, responsible for business logic, data storage (PostgreSQL), and providing the API that the UI will consume.

**AI/Creative Layer (Python/LangChain):** This is a planned, separate microservice that will handle all AI functionality, including natural language processing, orchestrating LLM agents, and generating creative outputs.