System Architecture (Mini Finance Prototype)

1. **System Overview**
The system is designed as a financial transaction platform that enables deposits, withdrawals, transfers, audit logging, user management, and administrative monitoring. The system operates as a client server architecture with a Node.js backend, Angular frontend, and MongoDB persistent storage.

2. **Backend Architecture**
The backend is built using Node.js and Express.js, leveraging MongoDB as the primary database. It implements ACID compliant MongoDB transactions for critical operations, especially deposits, withdrawals, and transfers to ensure data consistency. The backend also. Audit logging is implemented for all financial actions, and each transactional flow is fully recorded for traceability and security.

3. **Frontend Architecture**
The frontend is developed using Angular and is structured into reusable components, services, and routing modules. Components handle UI rendering, while services manage state, handle API communication, and encapsulate business logic. Routing separates user dashboards, admin dashboard, transaction pages, and workflow screens. Angular Material is used for UI elements such as tables, drawers, spinners, and form controls.

4. **API Flow**
The API follows REST principles, with separate routes for user operations, admin operations, transaction flows, and audit logs. Each API request triggers authentication, validation, execution of business logic, and persistence in MongoDB. Transactional operations are wrapped in MongoDB sessions to ensure atomicity. Responses are standardized and include clear success or error messages.

5. **Admin and Client Separation**
The system separates admin and client functionality
However anyone can perform both client and admin functionalities without necessarily being logged in