Project Abstract: FinGuard - A Finance Management Application Using Spring Boot Microservices

The goal of this project is to develop **FinGuard**, a comprehensive personal finance management application, using a **Spring Boot Microservices** architecture. The application will provide users with powerful tools to track their income, manage expenses, set budgets, and make informed financial decisions. It will offer a seamless experience across multiple devices, allowing users to gain control over their financial health.

Key Features:

- **User Authentication & Role-Based Access**: Secure login, registration, and account management. Role-based access will distinguish between regular users, financial advisors, and admin roles.
- **Income & Expense Tracking**: Users can record and categorize income and expenses, and view transaction histories and financial summaries.
- **Budget Planning**: Set monthly or yearly budgets, track spending against limits, and receive alerts when nearing budget limits.
- **Investment Portfolio Management**: Track and analyze investment portfolios including stocks, mutual funds, and other assets.
- **Loan & Debt Management**: Tools to manage loan repayments, calculate EMIs, and view debt summaries.
- **Financial Goal Setting**: Users can define short-term and long-term financial goals, such as saving for a car or home, and monitor progress.
- **Reports & Analytics**: Generate detailed financial reports, charts, and visual insights to better understand spending patterns and trends.
- **Expense Categorization & AI-based Insights**: Auto-categorize transactions using machine learning to provide insights and personalized financial recommendations.
- **Multi-Currency Support**: For users dealing with foreign currencies, the application will provide support for tracking finances in multiple currencies with real-time exchange rates.

Architecture:

The **Spring Boot Microservices** architecture ensures the application is highly scalable, modular, and easy to maintain. Each service is independent, handling specific tasks within the finance ecosystem:

- **User Service**: Manages user registration, authentication, and user profiles.
- **Budget Service**: Allows users to set budgets and track expenses within specified limits.
- **Transaction Service**: Handles all incoming/outgoing transactions, income tracking, and expense categorization.
- **Portfolio Service**: Manages investment portfolios, tracks asset performance, and provides financial analytics.
- **Debt Service**: Manages loans and other debts, providing payment schedules and summaries.
- **Goal Service**: Enables users to set and track personal financial goals.

- **Notification Service**: Sends reminders, alerts, and notifications for payments, goal achievements, and budget thresholds.
- **Report Service**: Generates financial reports and visual analytics for users to review.
- **API Gateway**: A central access point for all external client requests, routing them to the relevant services.

Technology Stack:

- **Spring Boot 3.3.2** for developing independent microservices.
- **Spring Security** for secure authentication and role-based access control.
- **MySQL/PostgreSQL** for transactional data and **MongoDB** for flexible storage of non-relational data.
- **Spring Cloud** for microservice discovery, load balancing, and fault tolerance.
- **React.js/Angular** for front-end (optional), providing a modern user interface.

OJET UI Requirements (per Feature)

1. User Authentication & Role-Based Access

- **OJET Screens**:
  - Login/Signup forms (`oj-form-layout`, `oj-input-text`, `oj-button`).
  - Role-based dashboards (User → finance dashboard, Advisor → client portfolio list, Admin → system analytics).
- **Backend APIs**: User Service → `/auth/register`, `/auth/login`, `/auth/roles`.

---

2. Income & Expense Tracking

- **OJET Screens**:
  - Add/Edit transaction form (oj-input-number, oj-select-single for category).
  - Transaction history in **oj-table** with filters (date, type).
  - Pie/Donut chart (oj-chart) for income vs expenses.
- **Backend APIs**: Transaction Service → `/transactions/add`, `/transactions/history`, `/transactions/categories`.

---

3. Budget Planning

- **OJET Screens**:
  - Budget setup wizard (monthly/yearly).
  - Progress bar and alerts (oj-progress-bar, oj-messages).
  - oj-chart (bar chart) showing actual vs planned spend.
- **Backend APIs**: Budget Service → `/budget/set`, `/budget/status`, `/budget/alerts`.

---

4. Investment Portfolio Management

- **OJET Screens**:
  - Portfolio summary dashboard with line/candlestick charts for stock performance.
  - oj-table for listing holdings (stocks, mutual funds, etc.).
  - oj-chart (area/line) for returns over time.
- **Backend APIs**: Portfolio Service → `/portfolio/add`, `/portfolio/holdings`, `/portfolio/performance`.

---

5. Loan & Debt Management

- **OJET Screens**:
  - Loan summary card layout (principal, interest, EMI).
  - EMI calculator form.
  - oj-chart for loan repayment timeline.
- **Backend APIs**: Debt Service → `/debt/add`, `/debt/schedule`, `/debt/summary`.

---

6. Financial Goal Setting

- **OJET Screens**:
  - Goal creation wizard (e.g., car, house).
  - oj-progress-circle showing % completion toward savings goal.
  - Alerts for missed contributions.
- **Backend APIs**: Goal Service → `/goals/set`, `/goals/progress`, `/goals/alerts`.

---

7. Reports & Analytics

- **OJET Screens**:
  - oj-dashboard with multiple charts: income vs expense, asset allocation, savings vs goals.
  - Downloadable reports (PDF/Excel).
  - oj-data-visualization components (scatter, bubble, treemap for category analysis).
- **Backend APIs**: Report Service → `/reports/generate`, `/reports/analytics`.

---

8. Expense Categorization & AI Insights

- **OJET Screens**:
  - Suggested categorization displayed in **oj-table** with user overrides.
  - oj-chart showing category-based spending trends.

    - Recommendation card layout (tips like “Cut dining out by 15%”).
- **Backend APIs**: Transaction/AI Service → `/transactions/auto-categorize`, `/insights/recommendations`.

---

9. Multi-Currency Support