Project Proposal: "BudgetWise" - A Mobile App for Personal Finance Prediction and Analytics

1. Project Title

BudgetWise: Mobile App for Personal Finance Prediction with Analytics and Web API Integration

2. Project Overview

BudgetWise is a mobile application designed for individuals who want to manage their personal finances smarter by leveraging data analytics. Users can register, log in, input their income and expense data (e.g., monthly spending on food, rent, entertainment), and use analytics tools to predict future savings or identify spending patterns. The app integrates with a Web API hosted on a VPS to process data, run regression models (e.g., predict savings growth), decision trees (e.g., classify overspending risks), and generate visualizations (e.g., spending trends) using Matplotlib.

The purpose is to empower users with actionable financial insights, combining my interest in data-driven decision-making with practical analytics. It will use a public dataset as a baseline (e.g., personal finance or spending data from Kaggle) and allow users to upload their own financial records for personalized analysis.

3. Objectives

- **User Registration & Authentication**: Provide secure sign-up, login, and account management features.
- **Analytics Features**: Enable regression analysis (e.g., predict savings), decision tree classification (e.g., spending habits), and data visualizations.
- **Web API Integration**: Connect the app to a backend API on a VPS for data processing and analytics computation.

- **User Interface**: Design a clean, money-themed UI that’s intuitive for everyday users.
- **Security**: Ensure user financial data is protected with robust encryption and authentication.

4. Scope of Work

4.1 Mobile App Features

Registration & Authentication:

- Sign-up with email/password or third-party login (e.g., Google).
- Secure login/logout and password reset via email.
- Optional: Two-factor authentication for added security.

Analytics:

- **Regression Models**: Users can upload income/expense data to predict future savings or expenses (e.g., linear regression for monthly savings trends).
- **Decision Trees**: Classify financial health (e.g., "safe," "at risk," "overspending") based on spending patterns, with accuracy metrics.
- **Data Visualizations**: Display bar charts (e.g., category-wise spending), line graphs (e.g., savings over time), and decision tree diagrams using Matplotlib.
- **Train & Test**: Split uploaded data into training/testing sets, displaying performance metrics (e.g., R^2 for regression, accuracy for decision trees).

4.2 Web API Integration

Backend API:

- RESTful API developed with CodeIgniter 3 (CI3) to handle requests from the mobile app.
- **Endpoints**:
  - /register, /login: Manage user authentication.
  - /upload: Accept CSV files or manual input of financial data (e.g., income, expenses).
  - /train: Train regression or decision tree models on user data.
  - /predict: Return predictions (e.g., savings in 6 months).
  - /visualize: Generate and return Matplotlib charts (e.g., PNG files).
- **Data Storage**: Use MySQL to store user accounts and financial datasets.
- **Hosting**: Deploy the API on a Hostinger VPS running Windows, ensuring reliability and scalability.

4.3 Security & Privacy

- Implement JWT for secure authentication between app and API.
- Use HTTPS (SSL/TLS certificates) to encrypt data in transit.
- Store sensitive financial data securely in MySQL with hashed passwords and encrypted fields where applicable.

5. Technical Requirements

5.1 Mobile App

- **Platform**: Android Studio
- **Language**: Java
- **Libraries**:
  - MySQL connector (via API calls).
  - Retrofit or OkHttp for HTTP requests to the Web API.
  - MPAndroidChart or similar to display Matplotlib visuals locally, if needed.

5.2 Backend API

- **Framework**: CodeIgniter 3 (CI3)
- **Database**: MySQL
- **Analytics Libraries**:
  - Python integration (e.g., via exec() or a separate microservice) with:
    - **scikit-learn**: For regression (e.g., savings prediction) and decision trees (e.g., spending classification).
    - **Matplotlib**: For visualizations (e.g., charts saved as PNGs).
- **Security**: JWT for authentication, HTTPS for secure communication.

5.3 Hosting

- **VPS Provider**: Hostinger
- **Operating System**: Windows

- **SSL/TLS**: Free certificates via Certbot for HTTPS.

6. Dataset Suggestion

- **Source**: Kaggle - "Personal Finance" or "Personal Finance Data".
- **Details**: These datasets include features like income, expenses by category (e.g., food, transport), and savings—ideal for regression (predicting savings) and decision trees (classifying spending behavior). You can use this as the default dataset and let users upload their own data in a similar format (e.g., CSV).