🌿 **Project Report: EcoPackAI**

Intelligent Sustainable Packaging Recommendation System

1. Executive Summary (Abstract)

EcoPackAI is a machine learning-powered decision support system designed to help supply chain managers switch to sustainable packaging. Unlike traditional databases, EcoPackAI uses predictive AI to estimate the cost and environmental impact of materials that do not yet exist in the market. The system features a **FastAPI backend** for high-performance inference and a **Streamlit Business Intelligence (BI) Dashboard** for real-time analytics, enabling businesses to balance sustainability goals with financial viability.

2. System Architecture

The project follows a modular **Microservices Architecture**:

1. **Data Layer:** PostgreSQL Database for storing raw material properties.
2. **ML Pipeline:** Python scripts for synthetic data generation, preprocessing, and model training.
3. **Backend API:** FastAPI server to expose AI models as REST endpoints.
4. **Frontend Dashboard:** Streamlit UI for user interaction, visualization, and reporting.

3. Module-wise Implementation Details

Module 3: Data Engineering & Pipeline

- **Objective:** overcome the limitation of small datasets (20 raw materials) to train robust ML models.
- **Technique Used: Synthetic Data Augmentation**.
    - We took seed data (e.g., Bamboo, Recycled Steel) and generated **1,000 synthetic variations** by introducing controlled statistical noise.
    - This prevents "Overfitting" (where the AI just memorizes the data) and ensures the model learns the underlying physics of the materials.
- **Preprocessing:**

- Used StandardScaler from Scikit-Learn to normalize inputs.
- Converted categorical text (e.g., "Plastic", "Metal") into numerical format (Encoding).

Module 4: AI Model Development

We trained two separate "Brains" to handle distinct prediction tasks:

1. **The Cost Predictor (The Accountant):**
    - **Algorithm: Random Forest Regressor**.
    - **Reasoning:** Random Forest is excellent for tabular data and resists outliers, making it stable for financial predictions.
    - **Accuracy Achieved:** R^2 Score > 0.99 (Error margin < ₹5).
2. **The Impact Predictor (The Ecologist):**
    - **Algorithm: XGBoost Regressor** (Extreme Gradient Boosting).
    - **Reasoning:** XGBoost is state-of-the-art for capturing complex, non-linear patterns in environmental data (CO_2 emissions).
    - **Accuracy Achieved:** R^2 Score > 0.99.

Module 5: Backend API Development

- **Framework: FastAPI** (Modern, High-performance Python framework).
- **Key Endpoints:**
    - POST /recommend: Accepts industry constraints (e.g., "Food", "Electronics") and user priorities (Cost vs. Eco). Returns ranked material list from PostgreSQL.
    - POST /predict_impact: Accepts raw technical specs (Tensile Strength, Weight). Invokes the .pkl ML models to return real-time predictions.
- **Security:** Implemented **CORS Middleware** to allow secure communication with the frontend.

Module 6: Frontend User Interface

- **Framework: Streamlit** (Python-based Web App).
- **Features:**
    - **Interactive Sliders:** Users can weigh "Sustainability" vs "Cost" dynamically.

- **Visualizations:** Used **Plotly** for:
    - **Scatter Plots:** Visualizing the trade-off between Price and CO_2.
    - **Spider/Radar Charts:** Showing the "personality" of a material (Strength vs. Eco vs. Cost).
- **Session State Management:** Ensures data persists while users navigate between tabs.

Module 7: Business Intelligence (BI) Dashboard

- **Objective:** Translate technical data into business value.
- **Features:**
    - **Comparative Analytics:** Side-by-side comparison of "Current Material" vs. "Proposed Green Material."
    - **Financial Projection:** Calculates "Annual Cost Savings" based on user-defined volume (e.g., 50,000 units/year).
    - **Reporting:**
        - **PDF Generation:** Uses fpdf to generate a signed Sustainability Report.
        - **Excel Export:** Uses openpyxl to download raw data for procurement teams.

4. Technologies Used

Category	Tools / Libraries
Languages	Python 3.10+
Machine Learning	Scikit-Learn (Random Forest), XGBoost, Pandas, NumPy
Backend	FastAPI, Uvicorn, SQLAlchemy
Frontend	Streamlit, Plotly (Charts)
Database	PostgreSQL
Reporting	FPDF (PDF generation), OpenPyXL (Excel)

5. How to Run the Project

Step 1: Start the Database

Ensure PostgreSQL is running and the ecopack_db contains the material data.

Step 2: Start the Backend (The Brain)

Open a terminal in the backend/ folder:

Bash

uvicorn main:app --reload

(Verify at http://127.0.0.1:8000/docs)

Step 3: Start the Frontend (The Face)

Open a **new** terminal in the frontend/ folder:

Bash

streamlit run dashboard.py

(The dashboard will launch automatically in your browser)

6. Future Scope

1. **Computer Vision:** Allow users to upload a photo of a material to auto-detect its type.
2. **Live Market Pricing:** Connect to an external API to fetch real-time global material prices (e.g., Aluminium prices).
3. **Supplier Integration:** Directly connect users with suppliers of the recommended materials.

Submitted By: Shriyash Rajendra Joshi