PS 1 (SPONSOR)
Jungle Safari Souvenir Shop – Smart Inventory Management System

Objective:
Develop an intelligent Inventory Management System for Jungle Safari's souvenir shop to streamline stock tracking, automate sales processes, and integrate secure payment transactions.

Key Challenges:

1. Multi-Channel Sales Management:
Enable seamless tracking of in-store and online sales.

2. Real-Time Inventory Tracking & Reordering:
Implement real-time stock updates.
Develop an automated low-stock alert and reordering system.
Product Range: Shirts, Bastar Art Products, Bottles, Keyrings, Canvas, Stationery

3. Supplier & Purchase Order Management:
Maintain supplier records and track purchase orders efficiently.

4. Advanced Reporting & Analytics:
Generate insightful sales and inventory reports for data-driven decision-making.

5. User Roles & Permissions:
Define access control for sales and inventory management personnel.

6. Secure Payment & Banking Integration:
Ensure seamless integration with banking systems for smooth transactions.

7. Platform Compatibility:
Build both mobile and desktop versions for accessibility and usability.

8. Enhanced Features for Efficiency & Customer Engagement:
Implement barcode scanning for quick product identification.
Integrate a CRM module for personalized customer service.
Develop a loyalty program to boost customer retention.

Expectation from Participants:

- Build a scalable, secure, and user-friendly Inventory Management System that optimizes shop operations, enhances customer experience, and ensures financial transparency.

- Tech Stack (Recommended but not mandatory):

1. Frontend: React Native / Flutter (Mobile), React / Angular (Web)

2. Backend: Node.js / Django / Spring Boot

3. Database: PostgreSQL / Firebase / MongoDB

4. Payment Integration: Razorpay / Stripe / PayPal

5. Additional: Barcode SDK, CRM APIs, Cloud Services

PS 2 (AIML TRACK)
Building an multimodal Chatbot for Efficient Retrieval-Augmented Generation (RAG)

Objective:

Develop a chatbot capable of efficiently retrieving and processing information from diverse document formats (PDFs with images/links, CSVs, voice files, and web links, including nested links) by converting them into a vector database. The system should support automatic updates upon document modifications and provide transparent insights into retrieval, re-ranking, and final LLM responses.

Key Challenges:

1. Data Ingestion & Preprocessing:
Process various document formats (PDF, CSV, voice files) and extract meaningful content, including handling images, links, and nested links.
Convert extracted data into a structured format suitable for vector database storage.

2. Vector Database & RAG Implementation:
Efficiently store and retrieve information using a vector database (e.g., FAISS, Pinecone, ChromaDB).
Implement RAG (standard, Graph RAG, or Agentic RAG) to enhance the chatbot's contextual understanding.
Automatically update the vector database when documents are edited in the collection.

3. Retrieval Transparency & Explainability:
Display retrieved information, re-ranking results, and final LLM-generated responses to ensure transparency in the chatbot's decision-making.

4. User Interaction & UI/UX Considerations:
Allow users to create multiple collections and query specific collections within the chatbot interface.
Provide a seamless and intuitive UI for managing document ingestion and chatbot interactions.

5. API & Integration Support:
Expose APIs for RAG-based retrieval, enabling external applications to fetch answers programmatically.

6. Security & Guardrails:
Implement guardrails to prevent hallucinations, misinformation, or inappropriate responses.
Ensure secure document handling and retrieval.

Deliverables:

- A working chatbot capable of processing large documents or web links (e.g., LangChain documentation) and providing contextually relevant answers.
- Automatic vector database updates upon document modifications.
- Transparent display of retrieved information, re-ranking results, and final responses.
- An interactive UI allowing users to create and query collections.
- API endpoints for programmatic access to RAG-based answers.
- A live demo showcasing retrieval from large documents or links.

PS 3 (OPEN TRACK)
BRING YOUR OWN IDEA.

Note : The project must be made during hackathon time.