Hackathon Project Phases Template for the **AutoSage App** project.

Hackathon Project Phases Template

Project Title:

DataQueryAI: Intelligent Data Analysis with Google TAPAS

Team Name:

Data Storm

Team Members:

- Rithika Jangilwar
- Karanam Swathi
- Banoth Ashwini
- Surepally Deepika

Phase-1: Brainstorming & Ideation

Objective:

The objective of **DataQueryAI** is to simplify data analysis by enabling users to query structured datasets using natural language. By leveraging **Google TAPAS**, the tool empowers individuals and businesses to extract insights without requiring advanced SQL or programming skills.

Key Points:

1. **Problem Statement:**

- Many businesses and individuals struggle with efficiently querying and analyzing large datasets without needing advanced SQL or data science expertise.
- Traditional data analysis tools often require complex query languages, making data retrieval and analysis a slow and technical process.

2. **Proposed Solution:**

- DataQueryAI leverages Google TAPAS (Table Parsing Neural Network) to enable intelligent, natural language-based querying and analysis of structured datasets.
- Key features include:Natural Language Queries,Automated Insights,Multi-Format Support,Collaboration Features,Customizable Dashboards,AI-Powered Recommendations

3. **Target Users:**

- **Business Analysts**: Non-technical users who need quick insights from structured data.
- **Data Scientists**: Professionals looking for an AI assistant to simplify exploratory data analysis.
- **Financial Analysts**: Users in banking and finance who need quick queries on large datasets.
- **Researchers & Academics**: Data-heavy research fields that require easy data exploration.
- **Small & Medium Enterprises (SMEs)**: Businesses without dedicated data teams but need insights from sales, inventory, and operational data.
- **Students & Educators**: Learners who need an intuitive tool for data-related coursework.

4. **Expected Outcome:**

- **DataQueryAI** is to empower users with fast, accurate, and actionable insights from structured data using natural language queries. It will reduce dependency on technical expertise, enabling non-technical users to make data-driven decisions independently.

Phase-2: Requirement Analysis

Objective:

To design and develop **DataQueryAI**, an intelligent data analysis tool powered by Google TAPAS, that enables users to query and analyze structured data (e.g., tables, spreadsheets, databases) using natural language. The tool will simplify data analysis for non-technical users while enhancing efficiency for technical users, making data-driven insights accessible to all.

Key Points:

1. **Technical Requirements:**

Natural Language Processing (NLP): Integrate Google TAPAS for table parsing and natural language understanding.

Data Integration: Compatibility with Excel, Google Sheets, CSV files, and SQL databases.

API Support: Provide RESTful APIs for third-party integrations.

2. **Functional Requirements:**

Natural Language Querying: Allow users to input queries in plain English.

Data Visualization: Display query results as charts, graphs, or tables. Allow users to export visualizations.

AI Recommendations: Suggest relevant queries or insights based on the dataset.

3. **Constraints & Challenges:**

Handling Ambiguity: Resolving ambiguous or incomplete user queries.

Data Complexity: Managing large datasets with multiple variables and relationships. Ensuring accurate parsing and analysis of complex tables.

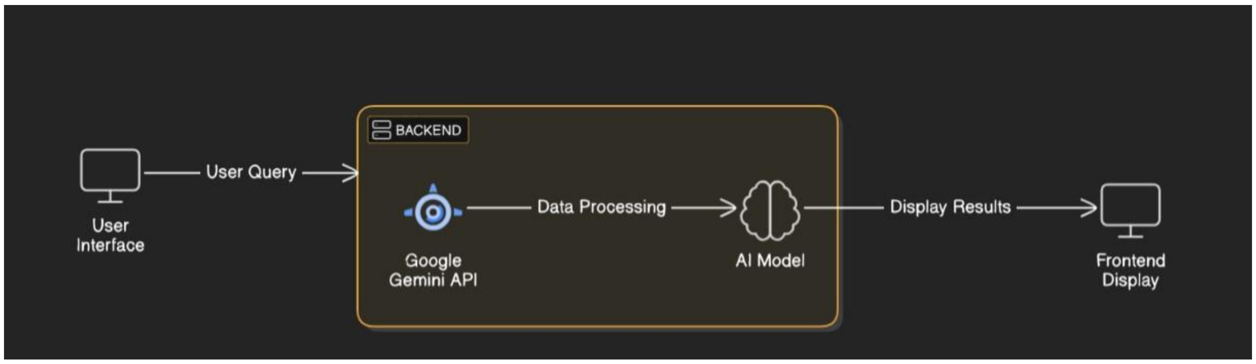
Integration Challenges: Ensuring compatibility with diverse data formats and platforms.

Accuracy & Reliability: Minimizing errors in query results and insights. Handling edge cases (e.g., missing data, inconsistent formats).

Phase-3: Project Design

Objective:

Develop the architecture and user flow of the application.



Key Points:

1. **Application Architecture:**
   The architecture of DataQueryAI consists of several components that work together to process user queries, analyze tabular data, and return meaningful results. Here's the high-level architecture:
2. **User Flow:**
   The user flow describes how a user interacts with the application from start to finish. Here's a step-by-step breakdown:
   Natural Language Query: Users type or speak a natural language query
   TAPAS Execution: The query is passed to the TAPAS model, which analyzes the tabular data and generates a response.
3. **Future Enhancements**
   Multi-Table Support: Extend TAPAS to handle queries across multiple tables.
   Voice Input: Allow users to input queries via voice.
   AI-Powered Insights: Automatically generate insights and visualizations without explicit queries.
   Collaboration Tools: Enable team-based data analysis and sharing.

Phase-4: Project Planning (Agile Methodologies)

Objective:

Break down development tasks for efficient completion.

Sprint	Task	Priority	Duration	Deadline	Assigned To	Dependencies	Expected Outcome
Sprint 1	Environment Setup & API Integration	High	6 hours (Day 1)	End of Day 1	Shanawaz	Google API Key, Python, Streamlit setup	API connection established & working
Sprint 1	Frontend UI Development	Medium	2 hours (Day 1)	End of Day 1	Member 2	API response format finalized	Basic UI with input fields
Sprint 2	Vehicle Search & Comparison	High	3 hours (Day 2)	Mid-Day 2	anwar	API response, UI elements ready	Search functionality with filters
Sprint 2	Error Handling & Debugging	High	1.5 hours (Day 2)	Mid-Day 2	Member 1&4	API logs, UI inputs	Improved API stability
Sprint 3	Testing & UI Enhancements	Medium	1.5 hours (Day 2)	Mid-Day 2	mohammad	API response, UI layout completed	Responsive UI, better user experience

Sprint 3	Final Presentation & Deployment	Low	1 hour (Day 2)	End of Day 2	Entire Team	Working prototype	Demo-ready project

Sprint Planning with Priorities

Sprint 1 – Project Setup and Planning (Day 1)

(**High Priority)** Define project scope, goals, and success metrics.

(**High Priority)** Set up the development environment (frontend, backend, and database).

(**Medium Priority)** Research and integrate Google TAPAS API.

Sprint 2 – Core Functionality - Data Upload and Query Processing (Day 2)

(**High Priority)** Develop the frontend interface for data upload and query input.

(**High Priority)** Implement backend API for handling file uploads and preprocessing.

Sprint 3 – Enhanced Query Processing and Result Formatting (Day 2)

(**Medium Priority)** Enhance TAPAS integration to handle complex queries

(**Low Priority)** Conduct user testing for feedback on query processing and results.

Phase-5: Project Development

Objective:

To develop **DataQueryAI**, an intelligent data analysis application leveraging Google's TAPAS (Table Parser), we will follow a structured project development plan.

Key Points:

1. **Technology Stack Used:**

    - **Frontend:** HTML,CSS
    - **Backend:** Python Streamlit
    - **Programming Language:** Python
2. **Development Process:**
   Created wireframes, user flow diagrams, and architecture designs.
   .Built the frontend, backend, and ML integration in parallel.
   Conducted unit, integration, and usability testing.
   Deployed the application to a cloud platform and monitored performance.
3. **Challenges & Fixes:**
   Challenge: Integrating Google TAPAS
   Fix: Built a preprocessing module to clean and format user-uploaded data into TAPAS-compatible formats.
   Challenge: Performance Optimization
   Fix: Implemented caching for frequently accessed data and query results.

Phase-6: Functional & Performance Testing

Objective:

Ensure that the AutoSage App works as expected.

Test Case ID	Category	Test Scenario	Expected Outcome	Status	Tester
TC-001	Functional Testing	User asks a simple question about a dataset	Correct response from TAPAS	✅ Passed	shanwaz
TC-002	Functional Testing	CSV, Excel, JSON inputs"	Successful parsing	✅ Passed	anwar
TC-003	Performance Testing	Simulate 100-1000 concurrent users	Response time, throughput	⚠ Needs Optimization	Tester 3
TC-004	Bug Fixes & Improvements	CSV/Excel import fails for some formats	Enhance data validation and preprocessing	✅ Fixed	Developer

TC-005	Final Validation	Ensure UI is responsive across devices.	UI should work on mobile & desktop.	❌ Failed - UI broken on mobile	Tester 2
TC-006	Deployment Testing	Host the app using Streamlit Sharing	App should be accessible online.	Deployed	DevOps

---

Final Submission

1. **Project Report Based on the templates**
2. **Demo Video (3-5 Minutes)**
3. **GitHub/Code Repository Link**
4. **Presentation**