Hackathon Project Phases Template

Project Title:

DataQuery AI: Intelligent data Analysis with google TAPAS

Team Name:

DataQuery Minds

Team Members:

- M. Namishna
- N. Hanshitha
- N. Lakshmi Mrudula

Phase-1: Brainstorming & Ideation

Objective:

The objective of the Brainstorm & Ideathon for the DataQueryAI project is to generate innovative ideas and solutions on how to effectively implement Google TAPAS for natural language-based querying of tabular data, explore new use cases, identify potential challenges in data interaction, and design intuitive, user-friendly interfaces to empower both technical and non-technical users to seamlessly extract actionable insights from structured datasets.

Key Points:

1.Problem Statement:

Organizations and individuals often struggle to extract meaningful insights from large, complex tabular datasets due to the need for specialized skills in querying and analyzing data. Existing tools require users to have proficiency in SQL or other programming languages, creating a barrier to data accessibility. There is a growing need for an intelligent, user-friendly solution that allows non-technical users to interact with tabular data using natural language queries, enabling them to effortlessly perform data analysis, make informed decisions, and gain valuable insights without requiring deep technical knowledge.

2.Proposed Solution:

Natural Language Queries & Easy Interface: Enable users to ask questions in plain language and automatically process them with a simple, user-friendly interface, requiring no technical expertise.

Real-Time Data & Automation: Integrate with live databases for up-to-date insights and automate common data analysis tasks like filtering and aggregation.

Security & Adaptation: Ensure secure access to data and continuously improve accuracy through machine learning, while supporting multiple data formats and languages.

3.Target Users:

Non-Technical Users & Business Professionals: Individuals like managers, marketers, and analysts who need easy access to data insights without technical expertise.

Data Analysts & SMEs: Professionals and small businesses seeking efficient, user-friendly tools for quick data analysis and decision-making.

4.Expected Outcome:

The expected outcome of **DataQuery AI** is to empower users of all technical backgrounds to easily query and analyze tabular data using natural language, enabling faster, data-driven decision-making and insights.

Phase-2: Requirement Analysis

Objective:

The objective of **DataQueryAI** is to develop a natural language-based data analysis tool that meets the needs of both technical and non-technical users by providing an intuitive interface, real-time data querying, automated data processing, and accurate insights from tabular data, all while ensuring scalability, security, and ease of integration with existing data systems.

Key Points:

1.Technical Requirements:
Integration with Tabular Data Sources: The system should be able to connect with various data formats (CSV, Excel, SQL databases) and real-time data sources to query and retrieve relevant information efficiently.

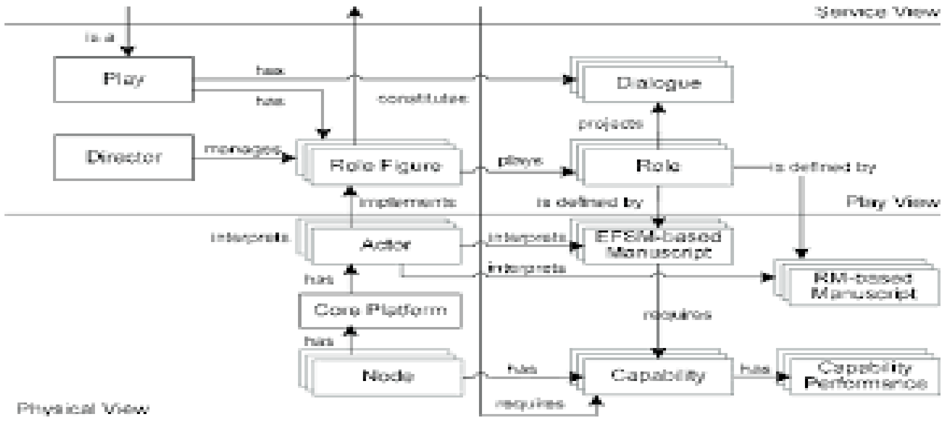
Natural Language Processing (NLP) Capabilities: Utilize Google TAPAS or a similar NLP model to understand and process natural language queries and map them to the corresponding data in tables.

Security and Scalability: Implement secure access controls, ensuring data privacy and compliance, while ensuring the system can scale to handle large datasets and multiple users without performance degradation.
2.Functional Requirements:
Natural Language Querying & Data Processing:
The system should allow users to input natural language queries, process them to perform tasks like filtering, aggregating, and visualizing data, and provide real-time insights from various data formats.
User-Friendly Interface & Security:
The system should feature an intuitive interface for non-technical users, with role-based access control and data security to ensure proper handling of sensitive information.
3.Constraints & Challenges:
Accuracy & Performance:
Ensuring precise interpretation of natural language queries and maintaining system performance when handling large datasets in real-time across various data sources.
Security & Usability:
Implementing strong data privacy and security measures while designing an intuitive interface that caters to both non-technical users and advanced users with customizable features.

Phase-3: Project Design

Objective:

Develop the architecture and user flow of the application.



Key Points:

1. **System Architecture:**

1.Data Ingestion

CSV Loader: Loads tabular data from CSV files.

Data Preprocessing: Cleans and preprocesses the data for analysis.

2.Data Analysis

Summary Generation: Generates summaries and statistical insights from the data.

Question Answering: Uses Google TAPAS to answer questions based on the tabular Data

3.Model Management

TAPAS Model: Pre-trained TAPAS model for semantic analysis and question answering.

Model Inference: Handles inference requests and processes model outputs.

4.API Layer

REST API: Provides endpoints for data ingestion, analysis, and question answering.

Authentication and Authorization: Manages user access and permissions

5.User Interface

Web Application: Interface for uploading data, querying, and viewing results.

Dashboard: Displays summaries, insights, and answers to user querie

6.Storage

Database: Stores metadata, user queries, and analysis results.

File Storage: Stores uploaded CSV files and processed data.

7.Monitoring and Logging

Logging: Captures logs for debugging and monitoring.

Metrics: Collects performance metrics and usage statistics.

2. **User Flow:**

1.User uploads a CSV file via the web application.
2.CSV Loader reads and stores the file.
3.Data Preprocessing cleans the data.
4.Summary Generation creates statistical summaries.
5.User submits a question via the web application.
6.Question Answering module uses TAPAS to generate an answer.
7.REST API provides endpoints for all actions.
8.Authentication and Authorization manage user access.
9.Logs and metrics are captured for monitoring.

3. **UI/UX Considerations:**

- **Natural Language Input**: Simple query bar with autocompletion and smart suggestions.

- **Interactive Table**: Clear, sortable, filterable tables with clickable data for deeper insights.

Phase-4: Project Planning (Agile Methodologies)

Objective:

 Utilize Google Tapas to automate and scale data analysis for faster insights and improved decision-making.

 Integrate AI models to enhance data interpretation and deliver predictive analytics for better business strategies.

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	Namishna	Google API Key, Python, Streamlit setup	API connection established & working
Sprint 1	Frontend UI Development	🟡 Medium	2 hours (Day 1)	End of Day 1	Hanshitha	API response format finalized	Basic UI with input fields
Sprint 2	Code generation & Explanation	🔴 High	3 hours (Day 2)	Mid-Day 2	Mrudula	API response, UI elements ready	Search functionality with filters
Sprint 2	Error Handling & Debugging	🔴 High	1.5 hours (Day 2)	Mid-Day 2	Namishna & Mrudula	API logs, UI inputs	Improved API stability
Sprint 3	Testing & UI Enhancements	🟡 Medium	1.5 hours (Day 2)	Mid-Day 2	Hanshitha & Mrudula	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 – Setup & Integration (Day 1)

(🔴 **High Priority)** Set up the **environment** & install dependencies.
(🔴 **High Priority)** Integrate **Google Gemini API**.
(🟡 **Medium Priority)** Build a **basic UI with input fields**.

Sprint 2 – Core Features & Debugging (Day 2)

(🔴 **High Priority)** Implement **search & comparison functionalities**.
(🔴 **High Priority)** Debug API issues & handle **errors in queries**.

Sprint 3 – Testing, Enhancements & Submission (Day 2)

(🟡 **Medium Priority)** Test API responses, refine UI, & fix UI bugs.
(🟢 **Low Priority)** Final **demo preparation & deployment**.

Phase-5: Project Development

Objective:

Develop a web app that allows users to search, compare, and analyze vehicles with a responsive UI and seamless API integration.

Key Points:

1. **Technology Stack Used:**
    - **Frontend:** Streamlit
    - **Backend:** Google Gemini Flash API
    - **Programming Language:** Python
2. **Development Process:**

- **Planning & Requirement Gathering**: Define project scope, features, and user needs.
- **Environment Setup**: Set up development tools, APIs, and frameworks.
- **Front-End Development**: Design UI/UX, implement input forms, and display results.
- **Back-End Development**: Integrate Google Tapas API, handle data processing, and set

  up server.
- **Testing**: Perform functionality and user experience testing, fix bugs.
- **Deployment**: Deploy the app for demo or production use.
- **Final Review & Presentation**: Prepare for final demo and feedback.

3.Challenges & Fixes:

• **API Connection Issues**
Fix: Double-check API key and request format to ensure proper connection.

• **UI Responsiveness**
Fix: Use responsive design techniques to make the UI work on all devices.

• **Data Accuracy**
Fix: Add error handling and data validation to improve reliability.

Phase-6: Functional & Performance Testing

Objective:

Ensure all features function correctly and the app performs efficiently under varying loads.

Test Case ID	Category	Test Scenario	Expected Outcome	Status	Tester
TC-001	Functional Testing	Query" Show me the value of(column name)"	Relevant answers should be displayed.	✅ Passed	Namishna
TC-002	Functional Testing	Query "What is the maximum value in [column]"	Relevant answer is displayed	✅ Passed	Hanshitha
TC-003	Performance Testing	Streamlit response time under 0.04s	Streamlit should return results quickly.	⚠ Needs Optimization	Mrudula
TC-004	Bug Fixes & Improvements	Fixed incorrect Streamlit responses.	Data accuracy should be improved.	✅ Fixed	Mrudula
TC-005	Final Validation	Ensure UI is responsive across devices.	UI should work on mobile & desktop.	❌ Failed - UI broken on mobile	Hanshitha
TC-006	Deployment Testing	Host the app using Streamlit Sharing	App should be accessible online.	🚀 Deployed	Namishna

---

Final Submission

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