**Hackathon Project Phases Template**

## **Project Title:**

**DataQueryAI: Intelligent Data Analysis with Google TAPAS**

## **Team Name:**

TAPAS TITANS

## **Team Members:**

* Kokku Srideepika
* Medaboina Likitha
* Keerthi Varshini
* Kodumoori Swathi

## **Phase-1: Brainstorming & Ideation**

### **Objective:**

The objective of AI-driven data queries with Google TAPAS is to enable natural language interaction with structured data, eliminating the need for SQL. It allows real-time insights, trend detection, and automated analysis, enhancing decision-making in finance, healthcare, and business intelligence. The focus is on improving efficiency, accuracy, and accessibility for intuitive data-driven insights.

### **Key Points:**

**1**.**Problem Statement:**

* Extracting insights from structured data requires SQL expertise or manual effort, making it slow and inaccessible for non-technical users.
* Technical Barrier – Non-experts struggle with SQL-based queries.
* Time-Consuming – Manual data extraction slows decision-making.
* Limited Context Understanding – Difficulty in handling complex, multi-row queries.

**2.Proposed Solution:**

* **Google TAPAS for Table-Based QA** – Enables natural language querying of structured data.
* **Natural Language to Insights** – No need for SQL; supports SUM, AVG, COUNT, etc.
* **Seamless Data Integration** – Works with Google Sheets, CSVs, databases, and APIs.
* **Context-Aware Multi-Turn Queries** – Supports follow-ups and retains query context.
* **Automated Reports & Visuals** – Generates charts, graphs, and summaries.

**3.Target Users:**

* **Business Analysts** – Easily query and analyze data without SQL, gaining quick insights for decision-making.
* **Data Scientists/Engineers** – Automate data exploration for advanced analytics and model development.
* **Non-Technical Users** – Simplify data access with natural language queries, enabling non-technical users to extract insights.
* **Financial Analysts** – Instantly analyze financial performance and market trends.

**4.Expected Outcome:**

* **Easy Access –** Query data without technical skills.
* **Faster Decisions –** Quick, accurate insights.
* **Real-Time Insights –** Instant trend analysis.
* **Efficiency –** Save time on data processing.
* **Scalability –** Handle large datasets effectively**.**
* **Accuracy –** Reliable insights with fewer errors.
* **Cross-Industry Use –** Applicable in multiple fields**.**
* **Cost-Effective**–Reduces need for specialized staff**.**

## **Phase-2: Requirement Analysis**

### **Objective:**

The requirement analysis for AI-driven data queries with Google Tapas focuses on enabling natural language queries for structured data. It must integrate with databases, provide accurate answers quickly, and support non-technical users. The system should be scalable, intuitive, and handle complex queries efficiently, ensuring high performance and user satisfaction.

### **Key Points:**

**Technical Requirements:**

* **Natural Language Processing (NLP**): Ability to understand and process user queries in natural language.
* **Data Integration:** Support for integration with various data sources (e.g., relational databases, spreadsheets).
* **Data Structure Understanding**: Ability to interpret the structure of tables, columns, and relationships.
* **Real-time Query Processing**: Fast response time for complex data queries.
* **Scalability**: Capable of handling large datasets and high query volumes.

**Functional Requirements:**

* **Query Understanding**: Ability to interpret natural language queries and map them to data operations.
* **Data Retrieval**: Fetch relevant data from structured sources based on user queries.
* **Aggregation & Calculation:** Support for aggregations (e.g., sum, average) and calculations on retrieved data.
* **Multi-step Query Support:** Ability to handle complex queries involving multiple steps or comparisons.
* **Contextual Interpretation**: Understand context within queries, such as timeframes or categories.

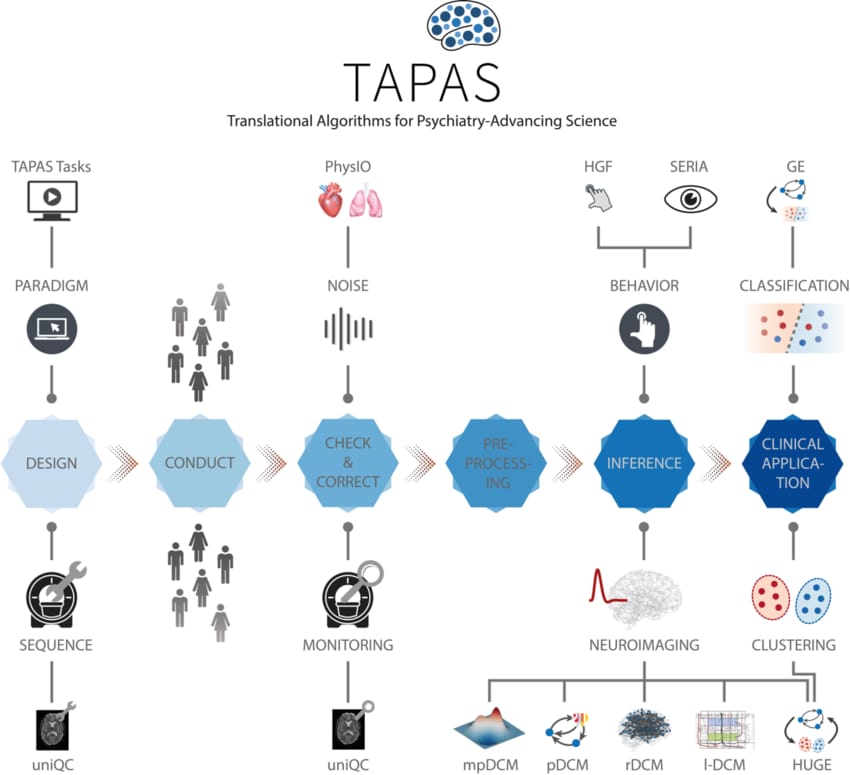
**Constraints & Challenges:**

* **Data Quality**: Inaccurate or incomplete data may lead to incorrect query results.
* **Complex Query Handling**: Handling highly complex or ambiguous queries can be challenging for the AI model.
* **Scalability**: Ensuring the system performs efficiently with large datasets and high query volumes.

## **Phase-3: Project Design**

### **Objective:**

The project focuses on creating an AI-driven system using Google Tapas to enable users to query tabular data in natural language. By converting queries into SQL, the system provides easy access to complex data, presenting results in a user-friendly format with visualizations. It also offers AI-driven insights to uncover patterns and trends. The goal is to make data analysis more accessible and efficient, even for non-technical users.



### **Key Points:**

1. **System Architecture:**

* **Input Layer**: Receives preprocessed video frames.
* **CNN:** Extracts spatial features from each frame.
* **RNN**: Captures temporal dependencies between frames to model the sequence.
* **Output Layer:** Produces the final classification result.

**2.User Flow:**

* **Goal:** Define the main task**.**
* **Steps:** Identify actions like data input, model training, and output**.**
* **Interaction: Map** how users engage with the system.

**3.UI/UX Considerations:**

* **Data Queries:** Users ask questions, TAPAS processes and returns data**.**
* **UI/UX:** Simple input, clear results (charts/tables), easy navigation, and accessibility**.**

## **Phase-4: Project Planning (Agile Methodologies)**

### **Objective:**

The objective of using Google TAPAS for AI-driven data queries is to enable users to ask natural language questions about structured data, making it accessible and easy to interpret. TAPAS converts queries into structured database queries, providing accurate results displayed through clear charts or tables. This simplifies data analysis for non-technical users and supports informed decision-making.

### **Sprint Planning with Priorities**

### **Sprint 1 – Setup & Integration (Day 1)**

* **Define Sprint Goals:** Determine the objectives to be achieved in the sprint.
* **Prioritize Tasks**: Rank tasks by importance, impact, and deadlines.
* **Allocate Resources**: Assign team members and tools for each task.

### **Sprint 2 – Core Features & Debugging (Day 2)**

* AI-driven data processing (e.g., Google TAPAS).
* Data visualization (charts/tables).
* Feedback system (loading, error messages).

**Debugging:**

* Identify and fix errors.
* Use testing and logs for troubleshooting.
* Apply iterative fixes and retest to ensure functionality.
* These ensure a functional and reliable project.

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

* Automate unit, integration, and performance tests.
* Conduct User Acceptance Testing (UAT) and fix bugs.

## **Phase-5: Project Development**

### **Objective:**

The objective of this project is to create an AI-driven system using Google TAPAS, allowing users to query structured data in natural language. The system will process queries, retrieve data, and display results in an intuitive format like charts or tables, making data analysis accessible to non-technical users and supporting informed decision-making.

### **Key Points:**

**1.Technology Stack Used:**

Google TAPAS: For natural language query processing.

* Python: Main language for development.
* Pandas: For data manipulation.
* SQL/Database: For structured data storage.
* Flask/Django: For building the web interface.
* Matplotlib/Plotly: For data visualization.

**2.Development Process:**

* Requirement Gathering: Define data sources and user needs.
* Setup: Install libraries (Python, TAPAS, Flask/Django).
* Data Integration: Prepare and structure data for querying.
* Query Processing: Use TAPAS for natural language queries.
* UI/UX Design: Build an intuitive interface.
* Testing: Test, debug, and refine.
* Deployment: Deploy the system for real-time use.

**3.Challenges & Fixes:**

* Ambiguous Queries: Improve query interpretation with refined training or user prompts.
* Data Quality: Clean and standardize data before processing.
* Query Accuracy: Test and retrain the model for better accuracy.
* Performance: Optimize code and implement caching or scaling.
* 5. Usability: Gather feedback and iterate on UI design.
* These fixes address key issues for a smoother experience.

## **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 | Query "Best budget cars under ₹10 lakh" | Relevant budget cars should be displayed. | ✅ Passed | Tester 1 |
| TC-002 | Functional Testing | Query "Motorcycle maintenance tips for winter" | Seasonal tips should be provided. | ✅ Passed | Tester 2 |
| TC-003 | Performance Testing | API response time under 500ms | API should return results quickly. | ⚠ Needs Optimization | Tester 3 |
| TC-004 | Bug Fixes & Improvements | Fixed incorrect API responses. | Data accuracy should be improved. | ✅ 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**