# Hackathon Project Phases Template

## Project Title:

DocuQuery: AI-Powered PDF Knowledge Assistant Using Google PALM

## Team Name:

Query Cub

## Team Members:

- Saipa.Pooja
- K.Sri Harshini
- Ch.Keerthana
- G.Navya

## Phase-1: Brainstorming & Ideation

### Objective:

Develop an AI-powered document assistant that leverages Google PALM and NLP techniques to streamline document comprehension, retrieval, and summarization.

### Key Points:

1. **Problem Statement:**

- Manual document analysis is time-consuming and inefficient.
- Keyword-based search tools lack deep contextual understanding.
- Businesses need automated solutions for document insights and summarization.

**2.Proposed Solution:**

- AI-powered PDF assistant that extracts insights, searches documents contextually, and provides summaries.

- Utilizes PyMuPDF, PDFPlumber for PDF parsing, and Google PALM for intelligent querying.
- Interactive UI built with Streamlit for seamless user interaction.

**3.Target Users:**

- AI engineers, data analysts, research professionals.
- Businesses handling compliance, contracts, and regulatory documents.

**4.Expected Outcome:**

- A functional AI assistant that improves knowledge accessibility and decision-making efficiency.

---

# Phase-2: Requirement Analysis

## Objective:

- Define the technical and functional requirements for DocuQuery.

## Key Points:

**1.Technical Requirements:**

- Programming Language: Python
- Backend: Google PALM API, FastAPI/Flask
- Frontend: Streamlit

**2.Functional Requirements:**

- PDF text extraction and OCR support.
- Contextual search using Google PALM.
- AI-generated summaries and structured output.
- Interactive chat-based document query system.

**Constraints & Challenges** :

- Handling large PDFs efficiently.

- Optimizing API usage to reduce costs.

- Ensuring AI-generated responses are accurate and traceable.

---

# Phase-3: Project Design

## Objective:

Develop the architecture and user flow of DocuQuery.

### Key Points:

**1.System Architecture Diagram:**
Data Ingestion: Users upload PDFs for processing.
Text Processing: NLP techniques clean and structure document data.
AI-Powered Query Handling:Google PALM interprets queries for contextual responses.
UI Presentation: Streamlit displays insights in an interactive interface.
**2.User Flow:**
Step 1: User uploads a PDF.
Step 2: AI processes the document and indexes content.
Step 3: User queries the document; AI provides responses and summaries.
**3.UI/UX Considerations:** (If applicable, wireframe or basic layout)
Simple, user-friendly interface for non-technical users.

Highlighted search results within document view.

---

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

### Objective:

Break down development tasks for efficient completion.

### Key Points:

**1.Sprint Planning:**
( High) Setup environment and dependencies.
( High) Implement PDF parsing and OCR support.
( Medium) Develop initial UI for document upload.
**2.Task Allocation:**
( High) Integrate Google PALM API for contextual search.
(High) Implement hybrid search (keyword + semantic search).
( Medium) Optimize NLP preprocessing.

**3.Timeline & Milestones:**
( Medium) Enhance UI for interactive chat-based querying.

( Low) Implement caching mechanisms for optimized performance.

# Phase-5: Project Development

## Objective:

Implement core features of DocuQuery.

## Key Points:

**1.Technology Stack Used:**
Frontend: Streamlit
Backend: FastAPI/Flask
AI Model: Google PALM
Database: PostgreSQL/MongoDB
**2.Development Process:**
Implement AI-driven document search and summarization.
Build an interactive document query system.
Optimize indexing and query response times.
**Challenges & Fixes:**
Challenge: Large PDF processing is slow.
Fix:Implement pagination and batch processing.
Challenge:Ensuring AI accuracy.
Fix: Implement citation tracking for extracted information.

# Phase-6: Functional & Performance Testing

## Objective:

Ensure DocuQuery meets functional and performance standards.

## Key Points:

**1.Technology Stack Used:**

Frontend: Streamlit

Backend: FastAPI/Flask

AI Model: Google PALM

Database: PostgreSQL/MongoDB

**2.Development Process:**

Implement AI-driven document search and summarization.

Build an interactive document query system.

Optimize indexing and query response times.

**3.Challenges & Fixes:**

Challenge: Large PDF processing is slow.Fix: Implement pagination and batch processing.

Challenge: Ensuring AI accuracy.Fix: Implement citation tracking for extracted information.

---

## Final Submission

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

---