Project Title	Building and Deploying a Question Answering System with Hugging Face
Skills take away From This Project	1. **Dataset Selection** **2. Data Preprocessing** **3. Model Selection** **4. Model Fine-Tuning** **5. Model Evaluation** **6. Model Deployment**
Domain	**Based on the chosen dataset**

Problem Statement:

- **Inefficient Information Retrieval:** Locating specific answers within large volumes of text, such as documents and websites, can be time-consuming and frustrating.
- **Limited Search Capabilities:** Traditional search engines often prioritize keyword matching over truly understanding the user's query intent.
- **Lack of Contextual Understanding:** Search tools may struggle to deliver accurate answers, especially when questions are complex or require an understanding of relationships between entities within the text.
- **Accessibility of Information:** Important information can be trapped within specialized documents or formats that aren't easily searchable by the general public.
- **Need for Domain-Specific QA:** Businesses and organizations frequently need rapid access to information within their internal knowledge bases, which may not be indexed by public search engines.

Business Use Cases:

A Question Answering (QA) system functions like an advanced search engine that understands your questions and seeks out the exact answer within a given text. Instead of just providing links, it reads the content and identifies the most likely answer. These systems leverage machine learning to mimic human-like answer-finding behavior, making them ideal for quickly retrieving information from extensive documents, websites, or powering chatbots to answer user queries. Essentially, it's like having a personal research assistant that finds the specific answer you need.

Approach:

1. **Dataset Selection:** Identify a relevant dataset for your QA system based on the domain (e.g., news articles, company reports, product manuals, scientific literature). Common QA datasets include SQuAD, NewsQA, and Natural Questions.
2. **Data Preprocessing:** Clean and prepare your dataset for training, which might include text normalization, tokenization, and organizing the data into question-context-answer triplets.
3. **Model Selection:** Choose an appropriate pre-trained QA model from the Hugging Face Model Hub, such as BERT, DistilBERT, or RoBERTa, all of which are fine-tuned for QA tasks.
4. **Fine-Tuning:** Fine-tune your selected model on your chosen dataset using the Hugging Face Transformers library, adjusting hyperparameters to optimize performance.
5. **Evaluation:** Assess your model's performance using standard QA metrics like Exact Match (EM) and F1 score. Analyze errors to pinpoint areas for improvement.
6. **Deployment:** Deploy your fine-tuned model as a web application using tools like Gradio, Streamlit, or Flask, enabling users to interact with it.

Results:

The QA system should take a question and relevant context as input and provide a concise and accurate answer extracted from the context. You can utilize Gradio to create the UI directly within Jupyter notebooks.

Project Evaluation metrics:

1. Standard QA metrics like Exact Match (EM) and F1 score. Analyze errors to pinpoint areas for improvement.

Technical Tags:

BERT, Transformers, QA systems, Hugging Face

Data Set:

Any Question answer dataset from hugging face ‘datasets’

Data Set Explanation:

- **Context:** "The Amazon rainforest is one of the world's most biodiverse habitats. It covers a vast area of South America, spanning multiple countries. The Amazon plays a critical role in regulating the global climate."
- **Question:** "What role does the Amazon rainforest play in the climate?"
- **Answer:** "The Amazon plays a critical role in regulating the global climate."

Project Deliverables:

Purpose:

This project provides exposure to core NLP concepts and practical applications of QA for knowledge extraction and search enhancement.

Skills:

- Data preparation and understanding
- Fine-tuning Transformer models
- Model deployment
- Familiarity with the Hugging Face ecosystem
- Understanding NLP concepts for Question Answering

Project Guidelines:

- **Modular Code:** Write your code in a modular fashion, with functional blocks.
- **Maintainability:** The code should be maintainable as the codebase grows.
- **Portability:** Ensure the code runs consistently across different environments (operating systems).
- **GitHub Usage:** Maintain your code on a public GitHub repository (Mandatory).
- **Readme File:** Include a detailed readme file on GitHub that outlines the workflow and execution of the project (Mandatory).
- **Coding Standards:** Follow PEP 8 coding standards as outlined at PEP 8 – Style Guide for Python Code (Mandatory).
- **Demo Video:** Create a demo video of your working model and post it on LinkedIn (Mandatory).

Timeline:

The project timeline spans for two weeks.

PROJECT DOUBT CLARIFICATION SESSION (PROJECT AND CLASS DOUBTS)

About Session: The Project Doubt Clarification Session is a helpful resource for resolving questions and concerns about projects and class topics. It provides support in understanding project requirements, addressing code issues, and clarifying class concepts. The session aims to enhance comprehension and provide guidance to overcome challenges effectively.
Note: Book the slot at least before 12:00 Pm on the same day

Timing: Tuesday, Thursday, Saturday (5:00PM to 7:00PM)

Booking link :https://forms.gle/XC553oSbMJ2Gcfug9

LIVE EVALUATION SESSION (CAPSTONE AND FINAL PROJECT)