PART 1: THEORETICAL UNDERSTANDING
Mastering the AI Toolkit

Q1: Explain the primary differences between TensorFlow and PyTorch. When would you choose one over the other?

TensorFlow and PyTorch are both powerful deep learning frameworks, but they differ in architecture, usability, and community preferences.

- TensorFlow uses a static computation graph, meaning you define the model structure first and run it later. This allows for easy deployment and performance optimization, especially in production settings.
- PyTorch, on the other hand, uses a dynamic computation graph, which is more intuitive and easier to debug because operations are executed immediately. This is great for research and rapid prototyping.

Use TensorFlow when:

- You need scalability and deployment.
- You're building models for mobile or embedded systems.

Use PyTorch when:

- You value flexibility and readability during experimentation.
- You are doing research or academic work, where dynamic graphs are preferred.

Q2: Describe two use cases for Jupyter Notebooks in AI development.

1. Interactive Model Prototyping: Jupyter Notebooks allow developers to write and test AI models in small code blocks, making it easy to debug, visualize, and adjust parameters. This is especially helpful when experimenting with new architectures or data pipelines.
2. Data Exploration & Visualization: Data scientists use notebooks to load datasets, perform exploratory data analysis (EDA), and generate graphs (e.g., using Matplotlib or Seaborn). This helps in understanding patterns, outliers, and relationships before training AI models.

Q3: How does spaCy enhance NLP tasks compared to basic Python string operations?

While basic Python string operations (like .split(), .replace(), or regex) handle simple text manipulation, spaCyoffers powerful, pre-trained NLP tools designed for real-world language understanding.

- spaCy provides:
    - Tokenization (splitting text into words and punctuation)
    - Named Entity Recognition (NER)
    - Part-of-speech tagging
    - Dependency parsing

Unlike basic string ops, spaCy understands context and grammar, making it ideal for complex tasks like extracting names, identifying relationships, or recognizing sentiment. It’s also faster and more accurate thanks to pre-trained pipelines.

Q4: Compare Scikit-learn and TensorFlow in terms of:

Feature	Scikit-learn	TensorFlow	
Type of ML	Traditional ML (e.g., Decision Trees, SVM)	Deep Learning (Neural Networks)	
Ease of Use	Very beginner-friendly, clean API	More complex, suited for advanced users	
Use Case	Small datasets, quick models	Big data, complex models (e.g., image recognition)	
Community Support	Strong in education and data science	Large global backing, popular in tech industry	