**Glossary**

Core Concepts

* System Architecture – The high-level structure of software components and how they interact
* Compute Environment – The complete set of resources and configurations that provide the processing power, runtime conditions, and infrastructure needed to run software applications or perform computational tasks
* API (Application Programming Interface) – a set of rules for how software components communicate
* Microservices – a software design pattern where components are independently deployable services
* CI/CD (Continuous Integration & Continuous Deployment) – Automated process for testing and releasing code changes
* DevOps – A culture and set of practices for combining software development and IT operations
* IAM (Identity & Access Management) – Framework for managing user identities and access permissions

AI/Machine Learning

* AGI (Artificial General Intelligence) – Hypothetical AI with human-level intelligence across all domains
* LLM (Large Language Model) – An AI model trained on massive corpora of text to understand/generate language
* Inference – The process of running a trained model to make predictions or generate output
* Prompt Injection – An attach where malicious input is designed to alter LLM behavior
* Retrieval-Augmented Generation (RAG) – Enhancing LLM responses with external data retrieval
* RLHF (Reinforcement Learning from Human Feedback) – Training an AI to align with human values/preferences
* Tokenization – Breaking text into smaller units (tokens) for model input
* Embeddings – Numeric representations of data (text, images, etc.) used in model understandings

Cybersecurity

* Zero Trust Architecture – a security model assuming no implicit trust, even inside the perimeter
* Redaction – The process of removing sensitive data (e.g. PII) before storage or display
* Data Loss Prevention – Technology that detects/prevents sensitive data from leaving an organization
* Threat Model – a structured presentation of potential security threats
* Encryption at rest/ in transit – Securing data when stored or being transmitted
* Secure Enclave – Isolated compute environments with enhanced security (e.g. Intel SGX)
* SOC (Security Operations Center) – Central team/system that monitors and responds to security incidents

Cloud & Infrastructure

* Terraform – Infrastructure as Code (IaC) tool for provisioning cloud services
* Azure/AWS/GCP – Major cloud platforms (MSFT, AMZ, GOOGL) offering compute/storage
* VPC (Virtual Private Cloud) – Isolated Network environment within a public cloud
* Kubernetes (K8s) – Container orchestration platform for automating deployment and scaling
* Service Mesh – Infrastructure layer for managing service-to-service communication securely
* Load Balancer – Distributes network traffic across servers for reliability and performance
* Containerization (e.g. Docker) – Encapsulating applications in portable containers

Tools, Libraries, Languages:

* Python – High-level language used for scripting, ML, backend systems
* FastAPI – Lightweight Python web framework for building APIs
* Pandas – Python library for data analysis and manipulation
* PyTorch/TensorFlow – Popular ML libraries for model building/training
* Pytest – Framework for writing and running unit test in Python
* Git – version control system used to manage code history
* Docker – Platform for containerizing applications
* PostgreSQL – relational databases used for structured data
* Bash – Command-line shell used for scripting tasks

Clarifying Q&a:

1. What are typical examples of compute environments?
   1. Local Machine, Virtual Machine (VM), Container (e.g. Docker), Cloud compute environment, Supercomputing cluster

Q. What is included in a compute environment?

* 1. 1) Hardware layer: CPU, GPU, RAM, storage

2) OS layer: Linux, Windows, macOS

3) Runtime/dependencies: Python, Java, libraries like NumPy, PyTorch

4) Execution context: Local/remote, single node/distributed, containerized or not

5) Security controls: Access permissions, network isolation, encryption

1. What is involved in the provisioning of a cloud service?
   1. To provision a cloud service means to set up and allocate the necessary cloud-based resources, configurations, and permissions required to make that service available and ready for use
   2. Put simply, provisioning is the process of turning a cloud service “on” – with all the settings, resources, and access it needs.
   3. Typically includes: 1)Resource allocation; 2)Service configuration; 3)Authentication and access controls; 4)Monitoring/logging setup; 5)Tagging
   4. Types include: 1)Manual provisioning; 2)Automated provisioning; 3)Self-service provisioning;
   5. Provisioning matters because of 1)Scalability, 2)Speed, 3)Security, and 4)Cost control