Q1.Explain 3-Tier Architecture

A three-tier architecture is a client-server architecture in which the functional process logic, data access, computer data storage and user interface are developed and maintained as independent modules on separate platforms.

Presentation Tier

* + Occupies the top level and displays information related to services commonly available on a web browser or web-based application in the form of a graphical user interface (GUI).
  + It constitutes the front-end layer of the application and the interface with which end-users will interact directly.
  + This tier is usually built on web development frameworks, such as CSS or JavaScript, and communicates with other tiers by sending results to the browser and other tiers in the network through API calls.

### Application Tier

* + This tier — also called the middle tier, logic tier, business logic or logic tier — is pulled from the presentation tier.
  + It controls the application’s core functionality by performing detailed processing and is usually coded in programming languages, such as Python, Java, C++, .NET, etc.

### Data Tier

* + Houses database servers where information is stored and retrieved.
  + Data in this tier is kept independent of application servers or business logic, and is managed and accessed with programs, such as MongoDB, Oracle, MySQL, and Microsoft SQL Server.

Q2. Explain Basic Terminologies

1. Replication:

Replication is the continuous copying of data changes from one database (publisher) to another database (subscriber). The two databases are generally located on a different physical servers, resulting in a load balancing framework by distributing assorted database queries and providing failover capability. The server for the subscriber database may be configured as a backup in the event of failure of the server for the publisher database.

1. Consistency

Consistency, in the context of databases, states that data cannot be written that would violate the database’s own rules for valid data. If a certain transaction occurs that attempts to introduce inconsistent data, the entire transaction is rolled back and an error returned to the user.

1. Eventual consistency :

Eventual Consistency is a guarantee that when an update is made in a distributed database, that update will eventually be reflected in all nodes that store the data, resulting in the same response every time the data is queried

1. Availability :

Availability, in the context of a computer system, refers to the ability of a user to access information or resources in a specified location and in the correct format.

1. Sharding :

Sharding refers to a specific type of database setup where multiple partitions create many pieces of a database that are then referred to as shards. This practice can help with server hosting and other aspects of database maintenance, and can also contribute to faster query times by diversifying the responsibilities of a database structure.

Q3.What is CAP Theorem

The CAP theorem states that a [distributed system](https://www.educative.io/blog/distributed-systems-considerations-tradeoffs) can only provide two of three properties simultaneously: **consistency**, **availability**, and **partition tolerance**. The theorem formalizes the tradeoff between consistency and availability when there’s a partition.

A distributed system is a collection of computers that work together to form a single computer for end users. All of the distributed machines have one shared state and operate concurrently. With distributed systems, users must be able to communicate with any of the distributed machines without knowing it’s only one machine. The distributed system network stores its data on more than just a single node, using multiple physical or virtual machines at the same time.

Q4. What are Public and Private Keys? How are they used in Cryptography?

[Public-key cryptography (PKC)](https://www.gemini.com/cryptopedia/glossary#public-key-cryptography-pkc) is a technology often used to validate the authenticity of data using [asymmetric encryption](https://www.gemini.com/cryptopedia/symmetric-vs-asymmetric-encryption). PKC was first used primarily to encrypt and decrypt messages in traditional computing. Cryptocurrencies now use this technology to encrypt and decrypt transactions. Without PKC, the technology underpinning cryptocurrencies would be practically impossible.

The key to PKC is “[trapdoor functions](https://www.gemini.com/cryptopedia/glossary#trapdoor-function),” one-way mathematical functions that are easy to solve in one way, but nearly impossible to crack in the reverse. While it might be possible, it would likely take a supercomputer — and thousands of years — to reverse engineer these functions.