**Agile development & Deployment is difficult in case of \_\_\_\_**

Monolithic

**Software built as microservices can, by definition, be broken down into multiple component services ?**

True

**Separating components with conflicting resource requirements falls under the bucket of \_**

Microservices

**Complexity of developing, testing & deploying distributed system, Handling partial failures account to disadvantages of**

Microservices

**Benefits of Microservices include -**

All the options

**Decomposition of Microservices based on 2 categories namely**

Business capability , Subdomain

**Simple to Develop, Test, Deploy, Scale represents \_\_\_\_**

Monolithic

**Is Microservice is considered as subset of SOA ?**

True

**The 3Cs of Microservices includes all these except**

Control

**Microservice Architecture adapts following concepts**

All the options

**The client expects a timely response from the service and might even block while it waits represents\_\_client service interaction styles**

Synchronous

**The 2 types of Service Discovery only includes Client-side & server-side discovery ?**

True

**Which ensures to insulate the applications by acting as barrier**

API Gateway

**Which acts as database of services**

Service Registry

**Scenarios where client takes onus & are responsible for determining the network locations of available service instances**

Client-side

**AWS Elastic Load Balancer (ELB) is an example of a \_\_\_\_**

Server-side discovery

**Netflix OSS is example**

Client-side

**API Gateway ensures \_**

All the options

**External clients communicate with Microservices using \_\_\_**

API GATEWAY

**\_ helps to control & limit the number of consecutive request failures crosses a threshold,**

Circuit breaker

**The services communicate with each other internally \_**

Both Messaging and Remote Procedure invocation

**The transactions that span across multiple services are handled by**

Event driven Architecture

**The 2 components of CQRS include\_**

Query & command side

**Microservices based architecture prefers \_**

No dependency on any particular DB

**Microservices-based architecture enforces a Modular structure ?**

True

**Amazon EC2 Container Service is an example of \_\_**

Docker

**Microservices supports different kinds of databases and this is called .**

Polyglot persistence

**Efficient Utilization & Fast Deployment represent which pattern**

Multiple Services per Host pattern

**Which of these represent the drawback of Multiple Services per Host pattern**

faulty deployment

**Service Instance per Host pattern provides \_\_**

All the options

**packer.io & Boxful represents type of pattern**

Service Instance per Container / Single Service Instance per Host

**IDL stands for**

Interface definition language

**\_\_ is ability to store something temporarily in order to reduce the loading times and I/O of a system.**

Cache

**In Microservices, the API Gateway takes care of the security aspect by rendering \_\_**

Access tokens

**Microservice Chassis takes care of all except \_\_\_\_**

token generation

**In \_\_, each external request is tagged with unique identifier which is passed to all services involved in handling the request and included in application logging messages**

Distributed tracing

**Caching helps in improving the performance of the system ?**

True

**Which of this does not represent caching types\_**

Server cache

**Which pattern collects & reports all exceptions to a centralized exception tracking service**

Exception tracing

**Limited Efficient Utilization & slow deployment represent \_**

Service Instance per Container

**Monolithic is identified by all these disadvantages except\_\_\_\_**

Quick initial development

**Scaling the application can be challenging in \_ due to conflicting needs of Memory, CPU, IO**

Monolithic