Following are various concepts we need to focus

1. **Difference between design and architecture** – 1. architecture is for entire system . However design is particularly for a system component.

2. architecture focus on abstraction that can be used by the multiple design. But, Design is the concrete plan to build something. Ex: Apartment

2) **Microservices design patterns (domain based and event based)** –

Event based design works in asynchronous way like Node js where they communicate through event message.

In domain based design, you can solve complex domain model. The solution revolves around the business model by connecting the implementation of core business model.

3) about various microservices concepts

- **API gateway** – is a service that’s entry point into the application from outside world. It’s responsible for “Request routing”, “API composition”, “protocol translation” & “implementing edge services(authorization, authentication, logging, caching, metrices collection, rate limiting)

- **Fault tolerance** – to work the system continuously operating without interruption when one or more components fails.

- **Service discovery**- actual process to know how to connect to a service. Each service updates service registry when server instances start or stop. When client invokes a service, the service discovery mechanism query service registry.

- **Circuit breakers** – to check the continuous failure of rest API, we define a threshold if failure goes beyond that it notifies to stop the sending request for particular time.

- **Scaling of microservices** – [Scaling](https://dzone.com/articles/scaling-microservices-the-challenges-and-solutions#:~:text=Scalability%20is%20determined%20by%20how,able%20to%20process%20these%20tasks.&text=In%20order%20to%20scale%20successfully,part%20of%20a%20larger%20system.) :

When you are running a single monolithic application from a single server, you can handle an increase in demand by allocating more resources to the application.

With microservices, on the other hand, upscaling can involve handling a number of different components and services. This means that either all the components need to upscale at the same time, or you need a means of identifying which individual components to upscale, and a method of ensuring that they can still integrate with the rest of the system.

When scaling microservices, you need to consider both the individual comp onents and the system as a whole.

- **security management of microservices**- [Security](https://techbeacon.com/app-dev-testing/8-best-practices-microservices-app-sec)- Use OAuth for user identity and access control, crypto code - handle encryption and decryption (you should use [NaCl](http://nacl.cr.yp.to/index.html)/[libsodium](https://download.libsodium.org/doc/) for encryption, wildly popular [Bouncy Castle](https://www.bouncycastle.org/) library.), Monitor everything with a tool like Prometheus

- **communication between microservices** –

1. Synchronous communication – In this, a service calls other service exposed api through http/gPRC protocol. This is the synchronous messaging pattern bcoz caller wait for the response.

2. Asynchronous communication – In this, sender sends a message without waiting for a response.

- coordination/communication issues and other issues related to microservices - Teams implementing different microservices must be able “to talk” to each other during a sprint, so a change can be implemented parallel in a coordinated way. If you cannot have short-lived “feature teams” across the microservices teams, at least nominate someone in each team to be responsible for the communication and coordination with the other teams for that specific feature. Ensure that project planning takes into account the extra communication and coordination effort for cross-microservice activities.

- **orchestration and monitoring of microservices** – Prometheus

- example questions

some quick questions about the manageability of a microservices deployment:

• Some quick questions about the manageability of a microservices deployment:

• How do we start and stop a fleet of services? •

How do we aggregate logs/metrics/SLAs across microservices? • - Central log system

How do we discover services in an elastic environment where they can be coming, going, moving, etc.?

• How do we do load balancing? – Load balancing defined as efficient way of distribution of network/application traffic across multiple servers. It sits between client device & backend servers, receiving and distributing incoming requests to any available server to fulfill that. -- **Kubernetes** using kube-proxy which manages virtual IP used by Service

• How do we learn about the health of our cluster or individual services?

• How do we restart services that have fallen over?

• How do we do fine-grained API routing?

• How do we secure our services?

• How do we throttle or disconnect parts of a cluster if it starts to crash or act unexpectedly?

• How do we deploy multiple versions of a service and route to them appropriately?

• How do we make configuration changes across a large fleet of services?

**Microservices: Advantages:**

1. It enables CI/CD approach continuous Integration & continuous Deployment to develop a large & complex application.
2. Services are small & easily maintained
3. Services are independently scalable
4. Services are independently deployable
5. It allows easy experimenting & adopting the new technologies
6. It has better fault tolerance.

**Disadvantages:**

1. Finding a right set of services is challenging
2. Distributed systems are complex which make development, testing & deployment complex
3. Deploying feature which involves multiple services requires careful coordination

**MySQL vs NoSQL :**

1. MySQL is static schema based where you have to follow predefined structure of your data when change in structure will happen then it will be difficult for whole system. But, NoSQL has dynamic schema for unstructured data. Data can be stored in multiple ways document based, column based, graph based, .This flexibility means that documents can be created without having defined structure. Also, each document can have its own unique structure.
2. SQL db’s are vertically scalable. This means you can increase load on a single server by increasing RAM, CPU or SSD. But, on the other side, NoSQL db’s are horizontally scalable. This means that you can handle more traffic by increasing more servers.
3. SQL follow ACID(Atomicity, Consistency, Isolation, Durability) property where as NoSQL follows Brewers CAP theorem.(Consistency, Availability, Partition tolerance)
4. SQL DBs are table based on the other hand NoSQL DBs are key-value pairs, document based or graph based.

4) System designs

a) how we design apache kafka your own

b) microservices design for table booking app

c) microservices design for food app

d) photos uploader app

While explaining the design we need to explain each component, interaction between components/services, where async call / where sync calls, how various events management between the services

5) Various approaches for Migration of project to new teams and its challenges

6) code skills test

1. convert digital watch to binary watch

2. find first duplicate character in the string

3. Find whether 2 strings are anagrams :

Fried --- fired

Gainly – laying

Sadder – dreads

4. Suppose there are 2 strings source =”abcde” target =”ab” . Your program should return true if target is there in source. And also if the reverse of target is there in source

5. An array has numbers from 1 to 20 . some of the numbers are missing if we sort the numbers in asending order. Find out the missing numbers.

6. Find the first non repeating character in a string eg: helloworld h is the answer.

7. Find the palindromic permutations for Malayalam.

8. Give an array of anagrams [“Fried”, “Gainly” ,”fired”, “dreads” ,”laying”, “Sadder” ,”god”, “goat”]

Group them as follows:

[

[“Fried”, ”fired”],

[“Gainly”, ”laying”],

[“dreads”, “Sadder”],

[“god”],

[“goat”]

]

9. Find out all palindromic substrings from a give string eg: aabbaacabcadadad

10. Problem Statement:

Line1 is a dictionary of words. Convert Line 2 to a meaningful sentence by referring to the words in Line1.

Example1:

// String line1 = "I,A,AM,HE,HERE";

// String line2 = "IAMHERE";

Solution :I AM HERE

Example2:

//String line1 = "he,hello,the,there";

//String line2 = "hellothere";

Solution: hello there

11. Given an array [1,2,3,4,5] group them as possible sums like [((1,4), (2,3)), ((4,2),(1,5))]