# NodeJS&Express:

1. Why we need express js when we already have node js?
2. Some questions on express js features like Routing, Middleware
3. Explain event loop
4. Explain node\_env
5. How can we use Typescript in Nodejs App.
6. Have you ever created your own npm module or published to npm repo?
7. Differences between promise and async await
8. Difference between asynchronous and non-blocking code
9. What is connect-src header
10. Asked about few http statuses codes like **412**
11. Event life cycle
12. CSS3 advanced features like key frames, transform
13. What happens when you don’t have any error handlers in Nodejs code
14. Explain various timing features in Node JS and their differences
15. Reactor pattern in Node JS
16. How do we use https with localhost?
17. In Nodejs app, can we connect to socket.io server without using client?
18. How do we detect clients when using socket.io with child\_processes?
19. If our application grows exponentially, it is difficult to maintain state in redux based apps, how do we overcome this problem?
20. Few scenario-based questions using design patterns?

* How distributed Node applications work? Ex: cluster and PM2
* How do you design stateful & stateless applications based on scenarios? When to choose which architecture?
* Dealing with huge file (say 10+ MB) uploads in Node.js, taking care of chunks and streams.
* Node.js latest APIS – async hooks and performance API etc
* Designing micro services & communication protocols among multiple micro services, containers and DB.
* Advanced REST standards like Idempotent methods
* Security best practices for APIs built on Node & Express – Attacks like “Man in the middle”, “DOS”, “DDOS”, CSRF, XSS etc

   1. Difference between Promise.all & Promise.race  
    2. How do you serve static files in express  
    3. Handle CPU intensive tasks in Node.js  
    4. Hoisting in JavaScript, var, let, const  
    5. difference between spread and Object.assign  
    6. How do you protect your Routes in Nodejs? i.e jsonwebtoken  
    7. Differnece between for-in & for-each loops  
    8. git rebase between two commits, git pull vs fetch  
    9. Microservices Architecture & Docker  
    10. How to handle CPU Intensive Tasks in Node  
    11. what is the best way to read a huge file in Node.js?- fs.readStream()

//IQVIA

1. How do you implement routing in node app without using express?
2. Difference between various Promise methods
3. Why do we switch to using promises from callbacks apart from callback hell issue?
4. How do we implement idle timeout using jwt?
5. Have you implemented an SSO in Node js?
6. Session sharing in Node js or express
7. What is the internal implementation of PM2?
8. How do you handle multiple CPU intensive tasks in node js?
9. Why jwt is hard to decode without secret key or how secure is jwt?
10. Node js microservices and inter communication between them using messaging brokers etc?
11. Using socket.io in microservices architecture.
12. Have you worked on container orchestration tools such as Kubernetes or cloud orchestration?
13. AWS Lambda and Step functions.
14. What is AWS elastic beanstalk?
15. How do you deal with the access related stuff between various services in AWS? - by creating roles
16. What is serverless computing?

1. Whats the ways to secure the API's in Node?  
2. Whats the implementation needs to be done when the request token is expired ?  
3. How can we execute two Process while calling one API. For example calling one Api needs to update the DB and upload the file to the server. Two things needs to be done parallel.  
    We need to send the response oback only when the two things are done.  
4. Multiple Promises are there like [p1, p2, p3, p4, p5]. Let's say p3 is rejected or error. Then how we can use p4 & p5. Because promise.all() will stop executing if one is rejected.  
5. How to Send 2GB of file to the frontend / If we have huge JSON Object with 100 objects , whats the best method to handle this.  
6. /user/:id how can we make id as an optional ? Best way to deal /user and /user/:id both the endpoints at a time.  
7. Whats the best way to implement Roll based access to API ?  
8. About JWT implementation and Public key implementation ?  
9. Scaling Node JS Applications ?  
10. How can we Manage concurrent DB updates with the multiple process in Node ?  
11. Sticky sessions ?  
12. How we can deploy code to server using AWS ?  
13. Certain API's in s3 bucket needs to access certain DB objects provided by AWS. How can we manage the access to the API's in AWS ?  
14. Object.create() ?  
15. How to addd new method to existing function with out re writing the entire code ?  
16. PM2 tools ?  
17. How can we gracefully handle the application before killed the process in node ?

1) Node.js Event loop with libuv library.

2) SQL & NoSQL database question.

3) unit testing.

4) Lambda function.

5) Design pattern

6) Docker & Kubernetes

Advantage If anyone worked before serverless architecture.

 How you will handle thousands record from a file and store in DB

 Optional routes in express

 Promise.all and Promise.race

 JWT token – how do u handle if user log out from frontend and use token without login

 Default and max timeout of AWS lambda

 IAM role for lambda

 Session handling in case of load balancer and multiple machine

 Type of databases, how do we connect DB from nodejs

 Cloudformation

 JWT role based authorization and payload structure

 How do we merge to JSON array of objects

 Working of lambdas, API gateway

 Best way to pass environment configuration for different environments

 How can we avoid issues come from npm update or unstable version of the npm

 Docker based microservices ECS

1. Explain event loop
2. Explain node\_env
3. How can we use Typescript in Nodejs App.
4. Have you ever created your own npm module or published to npm repo?
5. Differences between promise and async await
6. Difference between asynchronous and non-blocking code
7. What is connect-src header
8. Asked about few http status codes like **412**
9. Event life cycle
10. CSS3 advanced features like key frames, transform
11. What happens when you don’t have any error handlers in Nodejs code
12. Explain various timing features in Node JS and their differences
13. Reactor pattern in Node JS
14. How do we use https with localhost
15. In Nodejs app, can we connect to socket.io server without using client?
16. How do we detect clients when using socket.io with child\_processes?
17. If our application grows exponentially, it is difficult to maintain state in redux based apps, how do we overcome this problem?
18. Few scenario based questions using design patterns?

* How distributed Node applications work? Ex: cluster and PM2
* How do you design stateful & stateless applications based on scenarios? When to choose which architecture?
* Dealing with huge file (say 10+ MB) uploads in Node.js, taking care of chunks and streams.
* Node.js latest APIS – async hooks and performance API etc
* Designing micro services & communication protocols among multiple micro services, containers and DB.
* Advanced REST standards like Idempotent methods
* Security best practices for APIs built on Node & Express – Attacks like “Man in the middle”, “DOS”, “DDOS”, CSRF, XSS etc

//IQVIA

1. How do you implement routing in node app without using express?
2. Difference between various Promise methods
3. Why do we switch to using promises from callbacks apart from callback hell issue?
4. How do we implement idle timeout using jwt?
5. Have you implemented an SSO in Node js?
6. Session sharing in Node js or express
7. What is the internal implementation of PM2?
8. How do you handle multiple CPU intensive tasks in node js?
9. Why jwt is hard to decode without secret key or how secure is jwt?
10. Node js microservices and inter communication between them using messaging brokers etc?
11. Using socket.io in microservices architecture.
12. Have you worked on container orchestration tools such as Kubernetes or cloud orchestration?
13. AWS Lambda and Step functions.
14. What is AWS elastic beanstalk?
15. How do you deal with the access related stuff between various services in AWS?
16. What is serverless computing?
17. ci/cd

# Syngenta

1)What is single threaded  
2)drawbacks of single threaded  
3)drawback of closure  
4)call apply bind  
5)nodejs middleware  
6)realtime example of closure  
7)program - check string is balanced or not - <https://www.geeksforgeeks.org/check-for-balanced-parentheses-in-an-expression/>  
8)clustering in nodejs and how it work  
9)in which phase of event loop promise handled.  
10)Why use nodejs

# Cepheid

1. Past Projects
2. Architecture of past project and why these services are used?
3. API Gateway
4. Lambda function – last time used
5. Other AWS services and from when you are using AWS services
6. How do you provide security to Nodejs applications. (Code and architecture wise)
7. Different challenges you faced in our projects and how you improved it? (e.g.: API request per secs)