**Interview Questions**

Contents

[NodeJS 3](#_Toc172119430)

[What is Nodejs 3](#_Toc172119431)

[What is a prototype chain 3](#_Toc172119432)

[What is the Temporal Dead Zone 3](#_Toc172119433)

[What is Immediately Invoked Function Expression 3](#_Toc172119434)

[What is memoization 3](#_Toc172119435)

[What is Hoisting 3](#_Toc172119436)

[What are closures 3](#_Toc172119437)

[What are server-sent events 3](#_Toc172119438)

[Why do you need strict mode 3](#_Toc172119439)

[What is event bubbling 3](#_Toc172119440)

[How do you generate random integers 3](#_Toc172119441)

[What is the purpose of freeze method 4](#_Toc172119442)

[What is V8 JavaScript engine 4](#_Toc172119443)

[What is destructuring assignment 4](#_Toc172119444)

[What are streams 4](#_Toc172119445)

[What is JWT 4](#_Toc172119446)

[What is the difference between for, foreach, map, filter and reduce 4](#_Toc172119447)

[Event Loop Can you explain in detail the phases of the Node.js event loop? Describe what happens in each phase. 4](#_Toc172119448)

[What is the event loop in Node.js, and why is it important? 4](#_Toc172119449)

[What happens in the timers phase of the event loop? 4](#_Toc172119450)

[What is the difference between setImmediate() and setTimeout in Node.js? 4](#_Toc172119451)

[How can you prevent the event loop from being blocked? 5](#_Toc172119452)

[What are micro and macro task queues? 5](#_Toc172119453)

[Memory Management What is a memory leak and how to detect and diagnose it. 5](#_Toc172119454)

[What is the process.memoryUsage() method, and what information does it provide? 5](#_Toc172119455)

[What is the difference between stack memory and heap memory in Node.js? 5](#_Toc172119456)

[How does the V8 engine handle garbage collection. 5](#_Toc172119457)

[What is buffer in Node.js 5](#_Toc172119458)

[Why is it important to avoid using global variables 5](#_Toc172119459)

[Clustering What is Node.js clustering 5](#_Toc172119460)

[Can you describe a scenario where Node.js clustering might not be the best solution for scaling an application? 5](#_Toc172119461)

[What are the advantages and disadvantages of clustering 5](#_Toc172119462)

[For what purpose we will use OS module for clustering 6](#_Toc172119463)

[What is the default load balancer is being used for clusting 6](#_Toc172119464)

[TypeScript 7](#_Toc172119465)

[What is TypeScript, and how does it differ from JavaScript? 7](#_Toc172119466)

[What are the differences between interfaces and type aliases in TypeScript? 7](#_Toc172119467)

[How do you handle null and undefined in TypeScript? 7](#_Toc172119468)

[Explain the concept of generics in TypeScript 7](#_Toc172119469)

[What are decorators in TypeScript 7](#_Toc172119470)

[Explain the concept of type guards in TypeScript 7](#_Toc172119471)

[What is the never type in TypeScript 7](#_Toc172119472)

[How do you handle enums in TypeScript 7](#_Toc172119473)

[Micro-Service 8](#_Toc172119474)

[How do microservices communicate with each other? Explain various communication protocols and patterns 8](#_Toc172119475)

[Explain the concept of service discovery in microservices architecture. 8](#_Toc172119476)

[What is containerization, and how does it relate to microservices? 8](#_Toc172119477)

[What is the role of API gateways 8](#_Toc172119478)

[Explain the differences between synchronous and asynchronous communication in microservices. 8](#_Toc172119479)

[What are the security measures that can be implemented for an API gateway in a microservices architecture. 8](#_Toc172119480)

[Database 9](#_Toc172119481)

[Relational Explain the concept of normalization in relational databases. 9](#_Toc172119482)

[What is ACID transactions? 9](#_Toc172119483)

[What is the difference between a join and a subquery in SQL? 9](#_Toc172119484)

[What are stored procedures and triggers 9](#_Toc172119485)

[Explain the differences between clustered and non-clustered indexes 9](#_Toc172119486)

[What are some common optimization techniques for improving the performance 9](#_Toc172119487)

[What is database replication 9](#_Toc172119488)

[What is database sharding 9](#_Toc172119489)

[Explain the difference between LEFT OUTER JOIN and RIGHT OUTER JOIN. 9](#_Toc172119490)

[When would you use UNION instead of a join? 9](#_Toc172119491)

[What is a composite index 9](#_Toc172119492)

[How does an index improve query performance 10](#_Toc172119493)

[What is the difference between a unique index and a primary key constraint 10](#_Toc172119494)

[What is connection pooling 10](#_Toc172119495)

[What is table locking, give me some types of locking 10](#_Toc172119496)

[What is query optimization, and why is it important in database systems 10](#_Toc172119497)

[What are query hints 10](#_Toc172119498)

[What is view and when to use it 10](#_Toc172119499)

[Explain the differences between data-at-rest encryption and data-in-transit encryption. 10](#_Toc172119500)

[Explain the concept of database auditing. 10](#_Toc172119501)

[What is SQL injection, and how can you prevent 10](#_Toc172119502)

[Explain the concept of database anomaly detection 10](#_Toc172119503)

[Database - No SQL 11](#_Toc172119504)

[What is denormalization, and why is it commonly used in NoSQL databases? 11](#_Toc172119505)

[What are secondary indexes in NoSQL databases, and how are they used for query optimization? 11](#_Toc172119506)

[What are some common security considerations in NoSQL databases? 11](#_Toc172119507)

[What is horizontal partitioning, and how does it help improve scalability in NoSQL databases? 11](#_Toc172119508)

[Explain how indexing works in MongoDB to optimize query performance. 11](#_Toc172119509)

[Describe best practices for securing and managing data in MongoDB. 11](#_Toc172119510)

[System Design/Pattern 12](#_Toc172119511)

[Explain what SOLID principles are 12](#_Toc172119512)

[What are design patterns, and why are they important in software development? 12](#_Toc172119513)

[Describe the Singleton pattern. 12](#_Toc172119514)

[Explain the Factory Method pattern 12](#_Toc172119515)

[What is the Observer pattern 12](#_Toc172119516)

[Explain the Decorator pattern 12](#_Toc172119517)

[Security 13](#_Toc172119518)

[What are some common security vulnerabilities in Node.js applications 13](#_Toc172119519)

[Explain XSS attacks and how to prevent it. 13](#_Toc172119520)

[What is Cross-Site Request Forgery (CSRF) 13](#_Toc172119521)

[Explain the concept of SQL injection attacks. 13](#_Toc172119522)

[What are some best practices that must be implemented to secure the application 13](#_Toc172119523)

[What is rate limiting and helmet package for securing header 13](#_Toc172119524)

[Error Handling 14](#_Toc172119525)

[What is error handling in Node.js and why is it important? 14](#_Toc172119526)

[How do you handle errors in asynchronous code in Node.js? 14](#_Toc172119527)

[What is the difference between operational errors and programmer errors? 14](#_Toc172119528)

[How does the try...catch block work in Node.js? 14](#_Toc172119529)

[What is the role of the process object in error handling? 14](#_Toc172119530)

[How can you handle uncaught exceptions in Node.js? 14](#_Toc172119531)

[Explain the use of Promise and async/await in error handling. 14](#_Toc172119532)

[How do you handle errors in callback functions? 14](#_Toc172119533)

[What is a global error handler and how do you implement one in Node.js? 14](#_Toc172119534)

[How do you manage error logging in a Node.js application? 14](#_Toc172119535)

[How do you handle errors in Express.js middleware? 14](#_Toc172119536)

[What is the error-first callback pattern? 15](#_Toc172119537)

[Good to have 16](#_Toc172119538)

[Briefly explain the purpose and benefits of using Kubernetes in container orchestration. 16](#_Toc172119539)

[Describe the CI/CD pipeline and its role in automating the software development lifecycle. 16](#_Toc172119540)

[Explain how Docker containers provide isolation and portability for backend applications. 16](#_Toc172119541)

[Differentiate between RESTful APIs and GraphQL and discuss potential use cases for GraphQL. 16](#_Toc172119542)

[Describe the role of Kafka as a distributed streaming platform. 16](#_Toc172119543)

[Explain the components of the ELK Stack (Elasticsearch, Logstash, Kibana) and its use for log management and analytics. 16](#_Toc172119544)

[Discuss how message queues facilitate asynchronous communication between backend services. 16](#_Toc172119545)

# NodeJS

## What is Nodejs

## What is a prototype chain

## What is the Temporal Dead Zone

## What is Immediately Invoked Function Expression

## What is memoization

## What is Hoisting

## What are closures

## What are server-sent events

## Why do you need strict mode

## What is event bubbling

## How do you generate random integers

## What is the purpose of freeze method

## What is V8 JavaScript engine

## What is destructuring assignment

## What are streams

## What is JWT

## What is the difference between for, foreach, map, filter and reduce

## Event Loop Can you explain in detail the phases of the Node.js event loop? Describe what happens in each phase.

## What is the event loop in Node.js, and why is it important?

## What happens in the timers phase of the event loop?

## What is the difference between setImmediate() and setTimeout in Node.js?

## How can you prevent the event loop from being blocked?

## What are micro and macro task queues?

## Memory Management What is a memory leak and how to detect and diagnose it.

## What is the process.memoryUsage() method, and what information does it provide?

## What is the difference between stack memory and heap memory in Node.js?

## How does the V8 engine handle garbage collection.

## What is buffer in Node.js

## Why is it important to avoid using global variables

## Clustering What is Node.js clustering

## Can you describe a scenario where Node.js clustering might not be the best solution for scaling an application?

## What are the advantages and disadvantages of clustering

## For what purpose we will use OS module for clustering

## What is the default load balancer is being used for clusting

# TypeScript

## What is TypeScript, and how does it differ from JavaScript?

## What are the differences between interfaces and type aliases in TypeScript?

## How do you handle null and undefined in TypeScript?

## Explain the concept of generics in TypeScript

## What are decorators in TypeScript

## Explain the concept of type guards in TypeScript

## What is the never type in TypeScript

## How do you handle enums in TypeScript

# Micro-Service

## How do microservices communicate with each other? Explain various communication protocols and patterns

## Explain the concept of service discovery in microservices architecture.

## What is containerization, and how does it relate to microservices?

## What is the role of API gateways

## Explain the differences between synchronous and asynchronous communication in microservices.

## What are the security measures that can be implemented for an API gateway in a microservices architecture.

# Database

## Relational Explain the concept of normalization in relational databases.

## What is ACID transactions?

## What is the difference between a join and a subquery in SQL?

## What are stored procedures and triggers

## Explain the differences between clustered and non-clustered indexes

## What are some common optimization techniques for improving the performance

## What is database replication

## What is database sharding

## Explain the difference between LEFT OUTER JOIN and RIGHT OUTER JOIN.

## When would you use UNION instead of a join?

## What is a composite index

## How does an index improve query performance

## What is the difference between a unique index and a primary key constraint

## What is connection pooling

## What is table locking, give me some types of locking

## What is query optimization, and why is it important in database systems

## What are query hints

## What is view and when to use it

## Explain the differences between data-at-rest encryption and data-in-transit encryption.

## Explain the concept of database auditing.

## What is SQL injection, and how can you prevent

## Explain the concept of database anomaly detection

# Database - No SQL

## What is denormalization, and why is it commonly used in NoSQL databases?

## What are secondary indexes in NoSQL databases, and how are they used for query optimization?

## What are some common security considerations in NoSQL databases?

## What is horizontal partitioning, and how does it help improve scalability in NoSQL databases?

## Explain how indexing works in MongoDB to optimize query performance.

## Describe best practices for securing and managing data in MongoDB.

# System Design/Pattern

## Explain what SOLID principles are

## What are design patterns, and why are they important in software development?

## Describe the Singleton pattern.

## Explain the Factory Method pattern

## What is the Observer pattern

## Explain the Decorator pattern

# Security

## What are some common security vulnerabilities in Node.js applications

## Explain XSS attacks and how to prevent it.

## What is Cross-Site Request Forgery (CSRF)

## Explain the concept of SQL injection attacks.

## What are some best practices that must be implemented to secure the application

## What is rate limiting and helmet package for securing header

# Error Handling

## What is error handling in Node.js and why is it important?

## How do you handle errors in asynchronous code in Node.js?

## What is the difference between operational errors and programmer errors?

## How does the try...catch block work in Node.js?

## What is the role of the process object in error handling?

## How can you handle uncaught exceptions in Node.js?

## Explain the use of Promise and async/await in error handling.

## How do you handle errors in callback functions?

## What is a global error handler and how do you implement one in Node.js?

## How do you manage error logging in a Node.js application?

## How do you handle errors in Express.js middleware?

## What is the error-first callback pattern?

# Good to have

## Briefly explain the purpose and benefits of using Kubernetes in container orchestration.

## Describe the CI/CD pipeline and its role in automating the software development lifecycle.

## Explain how Docker containers provide isolation and portability for backend applications.

## Differentiate between RESTful APIs and GraphQL and discuss potential use cases for GraphQL.

## Describe the role of Kafka as a distributed streaming platform.

## Explain the components of the ELK Stack (Elasticsearch, Logstash, Kibana) and its use for log management and analytics.

## Discuss how message queues facilitate asynchronous communication between backend services.