ReactJS

What & Why:
ReactJS is a JavaScript library for building user interfaces, especially single-page applications. It uses a virt

Key Concepts:
- JSX
- Virtual DOM
- Props vs State
- Lifecycle methods
- Hooks (useState, useEffect)
- Context API
- Redux
- Routing with React Router

Top 20 Interview Questions:
- What is JSX?
- Explain the virtual DOM and how React uses it.
- Difference between props and state.
- What are React lifecycle methods?
- What are hooks in React?
- How does useEffect work?
- What is Context API?
- How does Redux work?
- What is React Router?
- How do you optimize performance in React?
- What is reconciliation in React?
- Explain controlled vs uncontrolled components.
- What is the purpose of keys in React?
- How do you handle forms in React?
- What is the difference between class and functional components?
- What is useMemo and useCallback?
- How do you manage side effects in React?
- What is HOC (Higher Order Component)?
- What is lazy loading in React?
- How do you test React components?

ASP.NET Core

What & Why:
ASP.NET Core is a cross-platform, high-performance framework for building modern web apps and APIs. It

Key Concepts:
- Middleware pipeline
- Dependency Injection
- MVC pattern
- Razor Pages
- Entity Framework Core
- Configuration & appsettings.json
- Hosting & Kestrel
- Authentication & Authorization

Top 20 Interview Questions:
- What is ASP.NET Core?
- Explain the middleware pipeline.
- What is dependency injection?
- Difference between MVC and Razor Pages.
- What is Entity Framework Core?
- How do you configure settings in ASP.NET Core?
- What is Kestrel?
- Explain authentication and authorization.
- What are tag helpers?
- How do you handle exceptions?
- What is routing in ASP.NET Core?
- How do you create REST APIs?
- What is model binding?
- What is the Startup.cs file?
- How do you use logging?
- What is the role of Program.cs?
- How do you deploy ASP.NET Core apps?
- What is the difference between .NET Core and .NET Framework?
- How do you use sessions?
- What is the role of appsettings.json?

MySQL / SQL Server

What & Why:
MySQL and SQL Server are relational database systems used to store structured data. SQL is the language

Key Concepts:
- Joins
- Normalization
- Indexing
- Transactions & ACID
- Stored Procedures
- Views
- Triggers

Top 20 Interview Questions:
- What is a primary key?
- Explain different types of joins.
- What is normalization?
- What is a foreign key?
- What is indexing?
- Explain ACID properties.
- What is a stored procedure?
- What is a view?
- What is a trigger?
- How do you optimize SQL queries?
- What is a subquery?
- What is a composite key?
- What is the difference between DELETE and TRUNCATE?
- What is a clustered index?
- What is a non-clustered index?
- What is a transaction?
- How do you handle concurrency?
- What is the difference between MySQL and SQL Server?
- What is a schema?
- How do you backup and restore a database?

JavaScript

What & Why:
JavaScript is the core language of the web. It enables dynamic behavior on websites and is used both on th

Key Concepts:
- Hoisting
- Closures
- Promises & async/await
- Event Loop
- ES6+ features
- DOM Manipulation
- Prototypes & Inheritance

Top 20 Interview Questions:
- What is hoisting?
- Explain closures.
- What is the event loop?
- Difference between var, let, and const.
- What are arrow functions?
- What is a promise?
- How does async/await work?
- What is the difference between == and ===?
- What is a callback function?
- What is the DOM?
- How do you manipulate the DOM?
- What is prototype inheritance?
- What is the difference between null and undefined?
- What is a higher-order function?
- What is the use of 'this' keyword?
- What is the difference between call, apply, and bind?
- What is event delegation?
- What is a module in JavaScript?
- What is the use of setTimeout and setInterval?
- How do you handle errors in JavaScript?

NodeJS

What & Why:
Node.js is a runtime environment that allows JavaScript to run on the server. It is event-driven and non-bloc

Key Concepts:
- Event Loop
- Callbacks vs Promises
- Streams & Buffers
- Express.js
- Middleware
- NPM
- REST APIs

Top 20 Interview Questions:
- What is Node.js?
- Explain the event loop.
- What is the difference between synchronous and asynchronous?
- What are streams in Node.js?
- What is a buffer?
- What is Express.js?
- How do you create a REST API?
- What is middleware?
- What is NPM?
- How do you handle errors in Node.js?
- What is the difference between require and import?
- What is a package.json file?
- What is the use of process object?
- How do you read and write files?
- What is the role of callbacks?
- What is the difference between setImmediate and setTimeout?
- How do you handle authentication?
- What is CORS?
- How do you deploy a Node.js app?
- What is the use of environment variables?

MongoDB

What & Why:
MongoDB is a NoSQL database that stores data in JSON-like documents. It is schema-less and ideal for ap

Key Concepts:
- Documents & Collections
- CRUD Operations
- Aggregation Pipeline
- Indexing
- Replication & Sharding
- BSON
- Mongoose

Top 20 Interview Questions:
- What is MongoDB?
- Difference between SQL and NoSQL.
- What is a document in MongoDB?
- What is a collection?
- What is BSON?
- What is Mongoose?
- How do you perform CRUD operations?
- What is the aggregation pipeline?
- What is indexing in MongoDB?
- What is replication?
- What is sharding?
- How do you query documents?
- What is the use of ObjectId?
- How do you update documents?
- What is the difference between find and findOne?
- How do you delete documents?
- What is the use of $match and $group?
- How do you handle relationships in MongoDB?
- What is the use of populate in Mongoose?
- How do you secure MongoDB?

SQL

What & Why:
SQL is the language used to interact with relational databases. It allows querying, updating, and managing d

Key Concepts:
- Joins
- Subqueries
- Window Functions
- Group By & Having
- Indexing
- Normalization
- Stored Procedures

Top 20 Interview Questions:
- What is SQL?
- What are different types of joins?
- What is a subquery?
- What are window functions?
- What is GROUP BY?
- What is HAVING clause?
- What is indexing?
- What is normalization?
- What is a stored procedure?
- What is a trigger?
- What is a view?
- What is a transaction?
- What is the difference between DELETE and TRUNCATE?
- What is a primary key?
- What is a foreign key?
- What is a composite key?
- What is a schema?
- What is the difference between UNION and UNION ALL?
- What is the use of DISTINCT?
- How do you optimize SQL queries?

Git & GitHub

What & Why:
Git is a version control system. GitHub is a platform to host Git repositories. Together, they help manage co

Key Concepts:
- git init, clone, add, commit, push, pull
- Branching & Merging
- Rebase vs Merge
- git stash
- git log, reflog
- Conflict resolution
- GitHub Pull Requests

Top 20 Interview Questions:
- What is Git?
- What is GitHub?
- How do you initialize a Git repository?
- What is the difference between git pull and git fetch?
- What is branching in Git?
- How do you merge branches?
- What is rebase?
- What is git stash?
- How do you resolve merge conflicts?
- What is a commit?
- What is a push?
- What is a pull request?
- What is git log?
- What is git reflog?
- How do you undo a commit?
- What is the difference between HEAD and master?
- How do you clone a repository?
- What is the use of .gitignore?
- How do you fork a repository?
- How do you contribute to open source using GitHub?

AWS

What & Why:
AWS is a cloud platform offering services like compute (EC2), storage (S3), and databases (RDS). It enable

Key Concepts:
- EC2
- S3
- Lambda
- IAM
- VPC
- RDS
- CloudWatch
- Elastic Beanstalk

Top 20 Interview Questions:
- What is AWS?
- What is EC2?
- What is S3?
- What is Lambda?
- What is IAM?
- What is VPC?
- What is RDS?
- What is CloudWatch?
- What is Elastic Beanstalk?
- What is the difference between EC2 and Lambda?
- What is the use of security groups?
- What is an AMI?
- What is auto-scaling?
- What is CloudFormation?
- What is Route 53?
- What is the difference between public and private subnets?
- What is EBS?
- What is the use of S3 lifecycle policies?
- How do you monitor AWS resources?
- How do you secure AWS resources?

NextJS

What & Why:
Next.js is a React framework for building server-side rendered (SSR) and statically generated (SSG) web ap

Key Concepts:
- SSR vs SSG
- getStaticProps
- getServerSideProps
- API Routes
- File-based Routing
- Image Optimization
- Middleware

Top 20 Interview Questions:
- What is Next.js?
- Difference between SSR and SSG.
- What is getStaticProps?
- What is getServerSideProps?
- How do you create API routes?
- What is file-based routing?
- How do you optimize images?
- What is middleware in Next.js?
- How do you deploy a Next.js app?
- What is dynamic routing?
- What is the use of _app.js?
- What is the use of _document.js?
- How do you handle environment variables?
- What is ISR (Incremental Static Regeneration)?
- How do you use CSS in Next.js?
- What is the use of Head component?
- How do you fetch data in Next.js?
- What is the difference between client-side and server-side rendering?
- How do you handle authentication?
- How do you integrate with a backend API?