Full Stack Development – Advanced Interview Questions

Technical interview question bank covering five full stack technologies.

JAVA FULL STACK – INTERVIEW QUESTIONS

Part 1 – Core Java and Advanced Programming

1. Explain the internal working of HashMap. How are hash collisions handled?

2. Compare HashMap, ConcurrentHashMap and Hashtable. When would you use each?

3. Explain JVM architecture, memory areas and garbage collection.

4. What causes memory leaks in Java, and how would you identify them?

5. Compare synchronized, volatile, Lock and atomic classes in multithreaded applications.

6. Explain Java Stream API, intermediate operations, terminal operations and lazy evaluation.

7. What are Functional Interfaces and Lambda Expressions? Give practical use cases.

8. Explain CompletableFuture and how asynchronous programming works in Java.

Part 2 – Spring Boot, Hibernate and REST APIs

9. Explain Inversion of Control and Dependency Injection in Spring.

10. How does Spring Boot auto-configuration work?

11. Explain the differences between @Component, @Service, @Repository and @Controller.

12. Explain transaction management in Spring, including propagation and isolation levels.

13. How would you implement JWT authentication, refresh tokens and role-based authorization?

Part 3 – Microservices, Database and AI Integration

14. Explain Hibernate lazy loading, eager loading and the N+1 query problem.

15. How would you design communication and fault tolerance between microservices?

16. How would you troubleshoot and optimize a slow Java REST API in production?

CREENS TECHNOLOGY

PYTHON FULL STACK – INTERVIEW QUESTIONS

Part 1 – Advanced Python Programming

1. Explain Python memory management, reference counting and cyclic garbage collection.

2. What is the Global Interpreter Lock (GIL)? How does it affect CPU-bound and I/O-bound workloads?

3. Compare threading, multiprocessing and asyncio. When is each suitable?

4. Explain decorators, closures, generators and iterators with practical use cases.

5. What are context managers? How would you create a custom context manager?

Part 2 – Django, Flask, FastAPI and REST APIs

6. Compare Django, Flask and FastAPI in terms of architecture and use cases.

7. Explain Django ORM query optimization using select_related() and prefetch_related().

8. Explain the difference between WSGI and ASGI.

9. How would you implement secure authentication and authorization using JWT and refresh tokens?

10. How would you implement request validation and centralized exception handling in FastAPI or Django?

Part 3 – Database, Deployment and AI Integration

11. How would you build asynchronous endpoints and handle background tasks in a Python application?

12. Explain database transactions and isolation levels. How do they maintain data consistency?

13. How would you implement background jobs using Celery and Redis?

14. How would you profile and optimize a slow Python API, and deploy it using Docker?

15. How would you build a RAG-based application using embeddings, a vector database and an LLM?

GREENS TECHNOLOGY

.NET FULL STACK – INTERVIEW QUESTIONS

Part 1 – C# and Advanced .NET

1. Compare .NET Framework, .NET Core and modern .NET. What are their key differences?

2. Explain value types, reference types, boxing and unboxing in C#.

3. How do async and await work? What issues can arise from blocking asynchronous code?

4. Compare Singleton, Scoped and Transient dependency injection lifetimes.

5. Compare IEnumerable, IQueryable and IAsyncEnumerable. When would you use each?

Part 2 – ASP.NET Core, Web API and Entity Framework

6. Explain the ASP.NET Core middleware pipeline. Why does middleware ordering matter?

7. Explain deferred execution in LINQ and how it affects database queries.

8. How does Entity Framework Core track changes? Compare tracking and no-tracking queries.

9. Explain lazy loading, eager loading and explicit loading in Entity Framework Core.

10. How would you implement JWT authentication, refresh tokens and role-based authorization?

Part 3 – SQL Server, Microservices and AI Integration

11. Explain global exception handling and request validation in ASP.NET Core Web API.

12. Explain optimistic concurrency and how you would handle conflicting database updates.

13. How would you design resilient communication between .NET microservices?

14. How would you diagnose and optimize memory, CPU and latency issues in a production .NET application?

15. How would you integrate Azure OpenAI or another LLM service into a .NET Full Stack application?

GREENS TECHNOLOGY

MERN STACK – INTERVIEW QUESTIONS

Part 1 – JavaScript, React and Frontend Development

1. Explain the Node.js event loop and the roles of the call stack, task queue and microtask queue.

2. Compare process.nextTick(), Promises, setImmediate() and setTimeout() in Node.js.

3. Explain React reconciliation, the Virtual DOM and how keys affect rendering.

4. Compare useMemo, useCallback, React.memo and useEffect. When can they negatively affect performance?

5. How would you manage complex global state using Context API, Redux Toolkit or React Query?

Part 2 – Node.js, Express.js and REST APIs

6. Explain Express.js middleware execution and centralized error handling.

7. How would you structure a large Express application using routes, controllers and services?

8. How would you design a scalable REST API with request validation, pagination and consistent error responses?

9. Explain asynchronous error handling in Node.js. How would you handle unhandled promise rejections?

Part 3 – MongoDB, Security and AI Integration

10. Compare MongoDB embedding and referencing. How would you choose a schema design?

11. Explain MongoDB aggregation pipelines, indexing and query optimization using explain().

12. How would you implement JWT authentication, refresh-token rotation and role-based access control?

13. How would you prevent XSS, injection, CSRF and broken access-control vulnerabilities in a MERN application?

14. How would you build an AI chatbot using React, Express, MongoDB and OpenAI or Gemini APIs, including conversation history and secure API-key handling?

GREENS TECHNOLOGY

MEAN STACK – INTERVIEW QUESTIONS

Part 1 – Angular and Frontend Development

1. Explain Angular dependency injection and provider scopes.

2. Compare Angular components, directives, pipes and services.

3. Explain Angular change detection. Compare Default and OnPush strategies.

4. Compare Observables and Promises. Explain switchMap, mergeMap and concatMap with use cases.

5. How would you prevent memory leaks caused by subscriptions in Angular?

Part 2 – Node.js, Express.js and REST APIs

6. Explain lazy loading, route guards and role-based route protection in Angular.

7. Explain the Node.js event loop and how asynchronous I/O is handled.

8. Explain Express.js middleware, centralized error handling and request validation.

9. How would you design a scalable REST API using Node.js and Express?

10. How would you implement secure authentication between an Angular frontend and a Node.js API using JWT and refresh tokens?

Part 3 – MongoDB, Testing and Production Readiness

11. Explain MongoDB schema design, indexing and aggregation pipelines.

12. How would you handle transactions and concurrent updates in MongoDB?

13. How would you test Angular components, services and API integrations?

14. How would you deploy and monitor a MEAN application using Docker, CI/CD and a cloud platform?

15. How would you troubleshoot performance issues across Angular, Node.js and MongoDB?

GREENS TECHNOLOGY