**Object-Oriented Programming (OOP)**

1. **Q:** How would you design a system where multiple inheritance is required, but the language (e.g., C#) doesn’t support it?  
   **A:** Use interfaces and composition to simulate multiple inheritance.
2. **Q:** Explain the difference between composition and inheritance. When would you prefer composition over inheritance?  
   **A:** Composition builds objects by combining others, while inheritance derives from a base class. Prefer composition for flexibility and to avoid tight coupling.
3. **Q:** How would you implement a deep copy of an object in C# without using serialization?  
   **A:** Implement the ICloneable interface and manually copy all fields and nested objects.
4. **Q:** What is the diamond problem in OOP, and how would you resolve it in a language like C++?  
   **A:** The diamond problem occurs in multiple inheritance when a class inherits from two classes that have a common base. Resolve it using virtual inheritance.
5. **Q:** How do you ensure thread safety in a singleton class without using locks?  
   **A:** Use the Lazy<T> class or static initialization for thread-safe singleton creation.
6. **Q:** What is the difference between early binding and late binding? Provide a real-world example of each.  
   **A:** Early binding occurs at compile time (e.g., method overloading), while late binding occurs at runtime (e.g., dynamic method invocation).
7. **Q:** How would you design a system to handle multiple types of notifications (email, SMS, push) using OOP principles?  
   **A:** Use the Strategy pattern to encapsulate each notification type and switch between them dynamically.
8. **Q:** What is the Liskov Substitution Principle, and how would you handle a scenario where it’s violated?  
   **A:** LSP states that derived classes should be substitutable for their base classes. Refactor the design to ensure derived classes don’t alter base behavior.
9. **Q:** How would you refactor a tightly coupled system into a loosely coupled one using OOP?  
   **A:** Use dependency injection, interfaces, and design patterns like Observer or Mediator.
10. **Q:** What is the difference between an abstract class and an interface? When would you use one over the other?  
    **A:** Abstract classes can have implementations, while interfaces cannot. Use interfaces for multiple inheritance-like behavior and abstract classes for shared logic.
11. **Q:** How would you implement a state machine using OOP principles?  
    **A:** Use the State pattern, where each state is a separate class implementing a common interface.
12. **Q:** What is the difference between polymorphism and method overloading?  
    **A:** Polymorphism allows a method to behave differently based on the object type, while overloading allows multiple methods with the same name but different parameters.
13. **Q:** How would you design a system to handle undo/redo functionality using OOP?  
    **A:** Use the Command pattern to encapsulate actions as objects and maintain a history stack.
14. **Q:** What is the difference between aggregation and association in OOP?  
    **A:** Aggregation implies a "has-a" relationship with independent lifetimes, while association implies a weaker relationship.
15. **Q:** How would you implement a custom iterator for a complex data structure?  
    **A:** Implement the IEnumerable and IEnumerator interfaces to define custom iteration logic.
16. **Q:** What is the role of the sealed keyword in C#, and when would you use it?  
    **A:** sealed prevents a class from being inherited or a method from being overridden. Use it to enforce immutability or security.
17. **Q:** How would you design a system to handle multiple payment gateways using OOP?  
    **A:** Use the Strategy pattern to encapsulate each gateway’s logic and switch between them dynamically.
18. **Q:** What is the difference between a static class and a singleton class?  
    **A:** A static class cannot be instantiated, while a singleton class allows a single instance with lazy initialization.
19. **Q:** How would you implement a generic repository pattern in C#?  
    **A:** Define a base repository interface with CRUD methods and implement it for specific entities.
20. **Q:** What is the role of the yield keyword in C#, and how does it work under the hood?  
    **A:** yield enables deferred execution in iterators by generating a state machine behind the scenes.

**Entity Framework (EF Core)**

1. **Q:** How would you implement a custom migration in EF Core to handle complex schema changes?  
   **A:** Use MigrationBuilder.Sql() to execute raw SQL commands during migrations.
2. **Q:** What is the difference between IQueryable and IEnumerable in EF Core, and how does it impact performance?  
   **A:** IQueryable builds queries that are executed on the database, while IEnumerable executes queries in memory. Using IQueryable is more efficient for database operations.
3. **Q:** How would you implement a soft delete mechanism in EF Core?  
   **A:** Override the SaveChanges method to filter out entities marked as deleted (e.g., using an IsDeleted flag).
4. **Q:** What is the difference between Add, Attach, and Update in EF Core?  
   **A:** Add marks an entity as new, Attach marks it as existing but unchanged, and Update marks it as modified.
5. **Q:** How would you handle concurrency conflicts in EF Core?  
   **A:** Use a RowVersion column and configure it with the IsConcurrencyToken method.
6. **Q:** What is the difference between Eager Loading, Lazy Loading, and Explicit Loading in EF Core?  
   **A:** Eager loading loads related entities upfront, lazy loading loads them on demand, and explicit loading loads them manually using Load().
7. **Q:** How would you implement a custom value converter in EF Core?  
   **A:** Use HasConversion to define a custom conversion between database and application types.
8. **Q:** What is the difference between DbContext and DbSet in EF Core?  
   **A:** DbContext represents a session with the database, while DbSet represents a collection of entities.
9. **Q:** How would you implement a global query filter in EF Core?  
   **A:** Use HasQueryFilter in the OnModelCreating method to apply a filter to all queries.
10. **Q:** What is the difference between FromSqlRaw and FromSqlInterpolated in EF Core?  
    **A:** FromSqlRaw executes raw SQL, while FromSqlInterpolated uses string interpolation for safer queries.
11. **Q:** How would you implement a transaction in EF Core?  
    **A:** Use DbContext.Database.BeginTransaction to start a transaction and commit or rollback as needed.
12. **Q:** What is the difference between Find and FirstOrDefault in EF Core?  
    **A:** Find uses the primary key to locate an entity, while FirstOrDefault uses a query.
13. **Q:** How would you implement a custom database provider in EF Core?  
    **A:** Implement IDatabaseProvider and register it in the DbContextOptions.
14. **Q:** What is the difference between HasOne and WithOne in EF Core?  
    **A:** HasOne defines the principal entity, while WithOne defines the dependent entity.
15. **Q:** How would you implement a custom function in EF Core for use in LINQ queries?  
    **A:** Use DbFunction to map a database function to a C# method.
16. **Q:** What is the difference between ExecuteSqlRaw and ExecuteSqlInterpolated in EF Core?  
    **A:** ExecuteSqlRaw executes raw SQL, while ExecuteSqlInterpolated uses string interpolation for safer queries.
17. **Q:** How would you implement a custom migration strategy in EF Core?  
    **A:** Override the Migrate method in IMigrationsAssembly to customize migration behavior.
18. **Q:** What is the difference between OnConfiguring and OnModelCreating in EF Core?  
    **A:** OnConfiguring configures the database provider, while OnModelCreating configures the model.
19. **Q:** How would you implement a custom database initializer in EF Core?  
    **A:** Override the Initialize method in IDatabaseInitializer.
20. **Q:** What is the difference between AddDbContext and AddDbContextPool in EF Core?  
    **A:** AddDbContext creates a new instance each time, while AddDbContextPool reuses instances from a pool.

**C#**

1. **Q:** How would you implement a thread-safe singleton in C# without using locks?  
   **A:** Use Lazy<T> or static initialization for thread-safe singleton creation.
2. **Q:** What is the difference between Task and ValueTask in C#?  
   **A:** Task is a reference type, while ValueTask is a value type optimized for performance.
3. **Q:** How would you implement a custom exception in C#?  
   **A:** Create a class that inherits from Exception and add custom properties or methods.
4. **Q:** What is the difference between ref and out parameters in C#?  
   **A:** ref passes a reference to a variable, while out is used for output parameters.
5. **Q:** How would you implement a custom attribute in C#?  
   **A:** Create a class that inherits from Attribute and define its properties or methods.
6. **Q:** What is the difference between IEnumerable and IQueryable in C#?  
   **A:** IEnumerable executes queries in memory, while IQueryable builds queries that are executed on the database.
7. **Q:** How would you implement a custom collection in C#?  
   **A:** Create a class that implements IEnumerable<T> and define its methods.
8. **Q:** What is the difference between StringBuilder and string in C#?  
   **A:** StringBuilder is mutable and optimized for concatenation, while string is immutable.
9. **Q:** How would you implement a custom delegate in C#?  
   **A:** Use the delegate keyword to define a custom delegate type.
10. **Q:** What is the difference between IComparable and IComparer in C#?  
    **A:** IComparable defines natural ordering, while IComparer defines custom ordering.
11. **Q:** How would you implement a custom LINQ operator in C#?  
    **A:** Create an extension method that operates on IEnumerable<T> or IQueryable<T>.
12. **Q:** What is the difference between yield return and return in C#?  
    **A:** yield return enables deferred execution, while return immediately exits the method.
13. **Q:** How would you implement a custom event in C#?  
    **A:** Use the event keyword to define an event and EventHandler to handle it.
14. **Q:** What is the difference between struct and class in C#?  
    **A:** struct is a value type, while class is a reference type.
15. **Q:** How would you implement a custom indexer in C#?  
    **A:** Use the this keyword to define an indexer property.
16. **Q:** What is the difference between Task.Run and Task.Factory.StartNew in C#?  
    **A:** Task.Run is a simplified API for starting tasks, while Task.Factory.StartNew provides more options.
17. **Q:** How would you implement a custom serializer in C#?  
    **A:** Create a class that implements ISerializer and define its methods.
18. **Q:** What is the difference between async void and async Task in C#?  
    **A:** async void is for event handlers and cannot be awaited, while async Task can be awaited.
19. **Q:** How would you implement a custom comparer in C#?  
    **A:** Create a class that implements IComparer<T> and define its Compare method.
20. **Q:** What is the difference between Expression and Func in C#?  
    **A:** Expression represents code as a tree, while Func is a delegate that represents a method.

**ASP.NET Core**

1. **Q:** How would you implement custom middleware in ASP.NET Core?  
   **A:** Create a class with Invoke or InvokeAsync method and register it in Startup.cs.
2. **Q:** What is the difference between AddTransient, AddScoped, and AddSingleton in dependency injection?  
   **A:** AddTransient creates a new instance each time, AddScoped creates one per request, and AddSingleton creates one for the application lifetime.
3. **Q:** How would you implement custom model validation in ASP.NET Core?  
   **A:** Create a custom validation attribute by inheriting from ValidationAttribute.
4. **Q:** What is the difference between IApplicationBuilder.Use and IApplicationBuilder.Run?  
   **A:** Use allows the next middleware to execute, while Run terminates the pipeline.
5. **Q:** How would you implement JWT authentication in ASP.NET Core?  
   **A:** Use AddAuthentication with JwtBearerDefaults.AuthenticationScheme and configure the token validation parameters.
6. **Q:** What is the difference between ActionResult and IActionResult?  
   **A:** ActionResult is a concrete type, while IActionResult is an interface representing various HTTP responses.
7. **Q:** How would you implement rate limiting in ASP.NET Core?  
   **A:** Use middleware like AspNetCoreRateLimit to limit requests based on IP or other criteria.
8. **Q:** What is the difference between appsettings.json and environment variables in ASP.NET Core?  
   **A:** appsettings.json is a configuration file, while environment variables are used for environment-specific settings.
9. **Q:** How would you implement a custom route constraint in ASP.NET Core?  
   **A:** Create a class that implements IRouteConstraint and register it in Startup.cs.
10. **Q:** What is the difference between FromServices and FromBody in ASP.NET Core?  
    **A:** FromServices injects a service, while FromBody binds data from the request body.
11. **Q:** How would you implement a custom exception filter in ASP.NET Core?  
    **A:** Create a class that implements IExceptionFilter and register it in Startup.cs.
12. **Q:** What is the difference between Map and MapWhen in middleware?  
    **A:** Map branches the pipeline based on the path, while MapWhen branches based on a condition.
13. **Q:** How would you implement a custom output formatter in ASP.NET Core?  
    **A:** Create a class that implements OutputFormatter and register it in Startup.cs.
14. **Q:** What is the difference between AddMvc and AddControllersWithViews?  
    **A:** AddMvc registers all MVC services, while AddControllersWithViews registers only controllers and views.
15. **Q:** How would you implement a custom authorization policy in ASP.NET Core?  
    **A:** Use AuthorizationPolicyBuilder to define the policy and register it in Startup.cs.
16. **Q:** What is the difference between IHostedService and BackgroundService?  
    **A:** IHostedService is the base interface, while BackgroundService is an abstract class for background tasks.
17. **Q:** How would you implement a custom view component in ASP.NET Core?  
    **A:** Create a class that inherits from ViewComponent and define its Invoke method.
18. **Q:** What is the difference between UseRouting and UseEndpoints?  
    **A:** UseRouting sets up routing, while UseEndpoints defines the endpoints.
19. **Q:** How would you implement a custom tag helper in ASP.NET Core?  
    **A:** Create a class that inherits from TagHelper and define its Process method.
20. **Q:** What is the difference between AddDbContext and AddDbContextPool?  
    **A:** AddDbContext creates a new instance each time, while AddDbContextPool reuses instances from a pool.

**Angular 13+**

1. **Q:** How does Angular’s change detection work, and how can you optimize it?  
   **A:** Angular uses Zone.js to detect changes. Optimize it with OnPush strategy and immutable data.
2. **Q:** What is the difference between ngOnInit and ngAfterViewInit?  
   **A:** ngOnInit runs after the component is initialized, while ngAfterViewInit runs after the view is rendered.
3. **Q:** How would you implement lazy loading in Angular?  
   **A:** Use loadChildren in the route configuration to load modules on demand.
4. **Q:** What is the difference between @Input and @Output?  
   **A:** @Input passes data into a component, while @Output emits events from a component.
5. **Q:** How would you implement a custom pipe in Angular?  
   **A:** Create a class that implements PipeTransform and define its transform method.
6. **Q:** What is the difference between ngIf and ngSwitch?  
   **A:** ngIf conditionally renders content, while ngSwitch renders content based on a value.
7. **Q:** How would you implement a custom directive in Angular?  
   **A:** Create a class with the @Directive decorator and define its behavior.
8. **Q:** What is the difference between ngModel and formControl?  
   **A:** ngModel is for two-way binding, while formControl is for reactive forms.
9. **Q:** How would you implement a custom validator in Angular?  
   **A:** Create a function that returns a ValidatorFn and use it in reactive forms.
10. **Q:** What is the difference between ViewChild and ContentChild?  
    **A:** ViewChild queries the component’s view, while ContentChild queries projected content.
11. **Q:** How would you implement a custom route guard in Angular?  
    **A:** Create a class that implements CanActivate and define its canActivate method.
12. **Q:** What is the difference between BehaviorSubject and ReplaySubject?  
    **A:** BehaviorSubject emits the latest value, while ReplaySubject emits a specified number of previous values.
13. **Q:** How would you implement a custom interceptor in Angular?  
    **A:** Create a class that implements HttpInterceptor and define its intercept method.
14. **Q:** What is the difference between ngOnChanges and ngDoCheck?  
    **A:** ngOnChanges runs when input properties change, while ngDoCheck runs during every change detection cycle.
15. **Q:** How would you implement a custom state management solution in Angular?  
    **A:** Use services with BehaviorSubject to manage state and provide it to components.
16. **Q:** What is the difference between async pipe and subscribe?  
    **A:** async pipe automatically subscribes and unsubscribes, while subscribe requires manual management.
17. **Q:** How would you implement a custom decorator in Angular?  
    **A:** Create a function that returns a decorator and apply it to classes or properties.
18. **Q:** What is the difference between ngTemplateOutlet and ngComponentOutlet?  
    **A:** ngTemplateOutlet renders templates, while ngComponentOutlet renders components.
19. **Q:** How would you implement a custom animation in Angular?  
    **A:** Use the trigger function to define animations and apply them with @Component.
20. **Q:** What is the difference between ngAfterContentInit and ngAfterViewInit?  
    **A:** ngAfterContentInit runs after content projection, while ngAfterViewInit runs after the view is rendered.

**Medical Billing EDI Development**

1. **Q:** What is the difference between EDI 837 and EDI 835?  
   **A:** EDI 837 is for healthcare claims, while EDI 835 is for payment remittance.
2. **Q:** How would you validate an EDI file?  
   **A:** Use EDI parsers and validators to ensure compliance with HIPAA standards.
3. **Q:** What is the difference between X12 and HL7?  
   **A:** X12 is for healthcare transactions, while HL7 is for clinical data exchange.
4. **Q:** How would you implement an EDI 837 parser?  
   **A:** Use libraries or write custom code to parse and validate claim data.
5. **Q:** What is the difference between EDI 270 and EDI 271?  
   **A:** EDI 270 is for eligibility inquiries, while EDI 271 is for eligibility responses.
6. **Q:** How would you handle EDI file encryption?  
   **A:** Use protocols like SFTP or AS2 for secure file transfer.
7. **Q:** What is the difference between EDI 834 and EDI 820?  
   **A:** EDI 834 is for enrollment data, while EDI 820 is for payment orders.
8. **Q:** How would you implement an EDI 835 generator?  
   **A:** Use templates or libraries to create payment remittance files.
9. **Q:** What is the difference between EDI 276 and EDI 277?  
   **A:** EDI 276 is for claim status inquiries, while EDI 277 is for claim status responses.
10. **Q:** How would you handle EDI file compression?  
    **A:** Use formats like ZIP or GZIP to compress EDI files for transfer.
11. **Q:** What is the difference between EDI 278 and EDI 275?  
    **A:** EDI 278 is for prior authorization, while EDI 275 is for patient information.
12. **Q:** How would you implement an EDI 270/271 transaction?  
    **A:** Use EDI standards to create and parse eligibility inquiry and response files.

**Technical Documentation**

1. **Q:** How would you structure technical documentation for a software project?  
   **A:** Use sections like introduction, installation, usage, API reference, and troubleshooting.
2. **Q:** What is the difference between user documentation and developer documentation?  
   **A:** User documentation explains how to use the software, while developer documentation explains how to extend or contribute to it.
3. **Q:** How would you ensure technical documentation is up-to-date?  
   **A:** Integrate documentation updates into the development workflow and use automated tools.
4. **Q:** What is the difference between Markdown and reStructuredText?  
   **A:** Markdown is simpler and widely used, while reStructuredText is more powerful and used in Python projects.
5. **Q:** How would you document an API?  
   **A:** Use tools like Swagger or Postman to generate interactive API documentation.
6. **Q:** What is the difference between README and wiki documentation?  
   **A:** A README provides an overview, while a wiki provides detailed documentation.
7. **Q:** How would you document a complex algorithm?  
   **A:** Use pseudocode, flowcharts, and detailed explanations of each step.
8. **Q:** What is the difference between inline comments and documentation comments?  
   **A:** Inline comments explain code logic, while documentation comments describe functions and classes.
9. **Q:** How would you document a software architecture?  
   **A:** Use diagrams (e.g., UML) and describe components, interactions, and design decisions.
10. **Q:** What is the difference between versioned documentation and live documentation?  
    **A:** Versioned documentation is tied to specific releases, while live documentation is continuously updated.
11. **Q:** How would you document a database schema?  
    **A:** Use ER diagrams and describe tables, columns, and relationships.
12. **Q:** What is the difference between API documentation and SDK documentation?  
    **A:** API documentation explains endpoints, while SDK documentation explains how to use libraries.
13. **Q:** How would you document a deployment process?  
    **A:** Provide step-by-step instructions, prerequisites, and troubleshooting tips.
14. **Q:** What is the difference between technical writing and creative writing?  
    **A:** Technical writing is precise and factual, while creative writing is imaginative and expressive.
15. **Q:** How would you document a security policy?  
    **A:** Describe security measures, access controls, and incident response procedures.
16. **Q:** What is the difference between user stories and use cases?  
    **A:** User stories describe features from a user’s perspective, while use cases describe interactions with the system.
17. **Q:** How would you document a testing strategy?  
    **A:** Describe test types, tools, and processes for ensuring software quality.
18. **Q:** What is the difference between release notes and changelogs?  
    **A:** Release notes summarize changes in a release, while changelogs track all changes over time.
19. **Q:** How would you document a code review process?  
    **A:** Describe guidelines, tools, and best practices for conducting code reviews.
20. **Q:** What is the difference between static documentation and dynamic documentation?  
    **A:** Static documentation is fixed, while dynamic documentation is generated from code or data.

**Performance Optimization**

1. **Q:** How would you optimize the loading time of a web page?  
   **A:** Minimize HTTP requests, compress assets, use lazy loading, and enable browser caching.
2. **Q:** What is the difference between debounce and throttle in JavaScript?  
   **A:** Debounce delays execution until after a pause, while throttle limits execution to a fixed rate.
3. **Q:** How would you optimize database queries for performance?  
   **A:** Use indexes, avoid SELECT \*, and optimize joins.
4. **Q:** What is the difference between gzip and brotli compression?  
   **A:** gzip is widely supported, while brotli offers better compression ratios.
5. **Q:** How would you optimize images for web performance?  
   **A:** Use modern formats (e.g., WebP), compress images, and use responsive images with srcset.
6. **Q:** What is the difference between async and defer in script loading?  
   **A:** async loads scripts asynchronously, while defer delays execution until the DOM is ready.
7. **Q:** How would you optimize CSS for performance?  
   **A:** Minify CSS, remove unused styles, and use critical CSS for above-the-fold content.
8. **Q:** What is the difference between CDN and local hosting?  
   **A:** CDNs distribute content globally for faster delivery, while local hosting serves content from a single server.
9. **Q:** How would you optimize JavaScript for performance?  
   **A:** Minify and bundle JavaScript, use lazy loading, and avoid blocking the main thread.
10. **Q:** What is the difference between client-side rendering and server-side rendering?  
    **A:** Client-side rendering renders content in the browser, while server-side rendering renders content on the server.
11. **Q:** How would you optimize API calls for performance?  
    **A:** Use caching, batch requests, and pagination to reduce the number of API calls.
12. **Q:** What is the difference between HTTP/1.1 and HTTP/2?  
    **A:** HTTP/2 introduces multiplexing, header compression, and server push for faster performance.
13. **Q:** How would you optimize a single-page application (SPA) for performance?  
    **A:** Use code splitting, lazy loading, and server-side rendering for faster initial load.
14. **Q:** What is the difference between preload and prefetch in web performance?  
    **A:** preload loads critical resources early, while prefetch loads resources for future navigation.
15. **Q:** How would you optimize a database schema for performance?  
    **A:** Normalize the schema, use indexes, and avoid redundant data.
16. **Q:** What is the difference between CPU-bound and I/O-bound performance issues?  
    **A:** CPU-bound issues are caused by high CPU usage, while I/O-bound issues are caused by slow input/output operations.
17. **Q:** How would you optimize a web application for mobile devices?  
    **A:** Use responsive design, optimize images, and minimize JavaScript.
18. **Q:** What is the difference between synchronous and asynchronous code execution?  
    **A:** Synchronous code blocks execution, while asynchronous code allows non-blocking operations.
19. **Q:** How would you optimize a web application for SEO?  
    **A:** Use semantic HTML, optimize page speed, and ensure mobile-friendliness.
20. **Q:** What is the difference between latency and throughput in performance optimization?  
    **A:** Latency is the time taken for a single operation, while throughput is the number of operations per unit time.

**jQuery**

**Q:** How would you implement a custom plugin in jQuery?  
**A:** Extend $.fn with a new method that operates on the selected elements.

1. **Q:** What is the difference between $.ajax() and $.get() in jQuery?  
   **A:** $.ajax() is a low-level method with more options, while $.get() is a shorthand for GET requests.
2. **Q:** How would you implement event delegation in jQuery?  
   **A:** Use on() to attach an event handler to a parent element and filter events for child elements.
3. **Q:** What is the difference between $(document).ready() and $(window).load() in jQuery?  
   **A:** $(document).ready() runs when the DOM is ready, while $(window).load() runs after all assets are loaded.
4. **Q:** How would you implement a custom animation in jQuery?  
   **A:** Use animate() to create custom CSS animations or chain jQuery effects.
5. **Q:** What is the difference between $.each() and $(selector).each() in jQuery?  
   **A:** $.each() iterates over arrays and objects, while $(selector).each() iterates over jQuery objects.
6. **Q:** How would you implement a custom event in jQuery?  
   **A:** Use trigger() to fire custom events and on() to handle them.
7. **Q:** What is the difference between $.extend() and $.fn.extend() in jQuery?  
   **A:** $.extend() merges objects, while $.fn.extend() adds methods to jQuery objects.
8. **Q:** How would you implement a deferred object in jQuery?  
   **A:** Use $.Deferred() to create a deferred object and resolve or reject it as needed.
9. **Q:** What is the difference between $.when() and $.then() in jQuery?  
   **A:** $.when() waits for multiple deferred objects, while $.then() handles the result of a single deferred object.
10. **Q:** How would you implement a custom selector in jQuery?  
    **A:** Use $.expr[':'] to define a custom pseudo-selector.
11. **Q:** What is the difference between $.data() and $.attr() in jQuery?  
    **A:** $.data() stores data in memory, while $.attr() manipulates HTML attributes.
12. **Q:** How would you implement a custom AJAX prefilter in jQuery?  
    **A:** Use $.ajaxPrefilter() to modify AJAX requests globally.
13. **Q:** What is the difference between $.trim() and String.prototype.trim() in jQuery?  
    **A:** $.trim() is a cross-browser utility, while String.prototype.trim() is a native method.
14. **Q:** How would you implement a custom easing function in jQuery?  
    **A:** Use $.easing to define a custom easing function for animations.
15. **Q:** What is the difference between $.parseJSON() and JSON.parse() in jQuery?  
    **A:** $.parseJSON() is a jQuery utility, while JSON.parse() is a native method.
16. **Q:** How would you implement a custom error handler for AJAX requests in jQuery?  
    **A:** Use $.ajaxSetup() to define a global error handler or handle errors in individual requests.
17. **Q:** What is the difference between $.proxy() and Function.prototype.bind() in jQuery?  
    **A:** $.proxy() is a jQuery utility for binding context, while bind() is a native method.
18. **Q:** How would you implement a custom queue in jQuery?  
    **A:** Use $.queue() to manage a custom queue of functions.
19. **Q:** What is the difference between $.noConflict() and $.noop() in jQuery?  
    **A:** $.noConflict() releases the $ alias, while $.noop() is a no-operation function.

**JavaScript**

**Q:** How does the JavaScript event loop work, and what is the role of the call stack, task queue, and microtask queue?  
**A:** The event loop processes tasks from the call stack, task queue (macrotasks), and microtask queue (e.g., Promises). It prioritizes microtasks over macrotasks.

1. **Q:** What is the difference between let, const, and var in terms of scope, hoisting, and reassignment?  
   **A:** let and const are block-scoped and not hoisted, while var is function-scoped and hoisted. const cannot be reassigned.
2. **Q:** How would you implement a deep clone of an object in JavaScript without using libraries?  
   **A:** Use JSON.parse(JSON.stringify(obj)) for simple objects or write a recursive function to handle complex objects.
3. **Q:** What is the difference between == and ===, and how does type coercion work in JavaScript?  
   **A:** == performs type coercion (e.g., "5" == 5 is true), while === checks both value and type ("5" === 5 is false).
4. **Q:** How does JavaScript handle asynchronous operations, and what are the differences between callbacks, Promises, and async/await?  
   **A:** Callbacks are functions passed as arguments, Promises handle asynchronous results with .then and .catch, and async/await provides syntactic sugar for Promises.
5. **Q:** What is the difference between null, undefined, and undeclared in JavaScript?  
   **A:** null is an intentional absence of value, undefined means a variable is declared but not assigned, and undeclared means a variable is not declared.
6. **Q:** How would you implement a debounce function in JavaScript?  
   **A:** Use setTimeout to delay the execution of a function and clear the timeout on subsequent calls.
7. **Q:** What is the difference between Object.create() and new in JavaScript?  
   **A:** Object.create() creates an object with a specified prototype, while new creates an instance of a constructor function.
8. **Q:** How would you implement a custom iterator in JavaScript?  
   **A:** Define a Symbol.iterator method that returns an object with a next() method.
9. **Q:** What is the difference between map(), filter(), and reduce() in JavaScript?  
   **A:** map() transforms elements, filter() selects elements based on a condition, and reduce() accumulates elements into a single value.
10. **Q:** How would you implement a singleton pattern in JavaScript?  
    **A:** Use a closure or a module pattern to ensure only one instance of an object is created.
11. **Q:** What is the difference between bind(), call(), and apply() in JavaScript?  
    **A:** bind() creates a new function with a fixed this, call() invokes a function with a specified this and arguments, and apply() does the same but accepts arguments as an array.
12. **Q:** How would you implement a memoization function in JavaScript?  
    **A:** Use a closure to cache results of expensive function calls.
13. **Q:** What is the difference between prototype and \_\_proto\_\_ in JavaScript?  
    **A:** prototype is a property of constructor functions, while \_\_proto\_\_ is a property of instances that points to the prototype.
14. **Q:** How would you implement a custom event emitter in JavaScript?  
    **A:** Create a class with methods to register, emit, and remove event listeners.
15. **Q:** What is the difference between async and defer in script loading?  
    **A:** async loads scripts asynchronously, while defer delays execution until the DOM is ready.
16. **Q:** How would you implement a polyfill for Array.prototype.flat() in JavaScript?  
    **A:** Use recursion or reduce() to flatten nested arrays.
17. **Q:** What is the difference between WeakMap and Map in JavaScript?  
    **A:** WeakMap holds weak references to keys, allowing garbage collection, while Map holds strong references.
18. **Q:** How would you implement a custom Promise in JavaScript?  
    **A:** Create a class with resolve, reject, and then methods to handle asynchronous operations.
19. **Q:** What is the difference between Generator functions and regular functions in JavaScript?  
    **A:** Generator functions use yield to pause execution and return an iterator, while regular functions run to completion.

**Unit Testing**

1. **Q:** How would you mock dependencies in a unit test?  
   **A:** Use mocking frameworks like Moq (C#) or Jest (JavaScript) to simulate dependencies.
2. **Q:** What is the difference between unit testing and integration testing?  
   **A:** Unit tests isolate individual components, while integration tests verify interactions between components.
3. **Q:** How would you test a private method in a class?  
   **A:** Use reflection or test the public methods that call the private method.
4. **Q:** What is the difference between stub and mock?  
   **A:** A stub provides predefined responses, while a mock verifies interactions.
5. **Q:** How would you implement a parameterized test?  
   **A:** Use test frameworks like NUnit or xUnit to pass multiple inputs to a test method.
6. **Q:** What is the difference between TDD and BDD?  
   **A:** TDD focuses on writing tests before code, while BDD focuses on behavior and collaboration.
7. **Q:** How would you test asynchronous code?  
   **A:** Use async/await in test methods and frameworks that support asynchronous testing.
8. **Q:** What is the difference between assert and verify?  
   **A:** assert checks conditions, while verify ensures methods were called as expected.
9. **Q:** How would you test edge cases in a unit test?  
   **A:** Include inputs at the boundaries of valid and invalid ranges.
10. **Q:** What is the difference between unit test and regression test?  
    **A:** Unit tests verify individual components, while regression tests ensure existing functionality remains intact.
11. **Q:** How would you test a method that throws an exception?  
    **A:** Use test frameworks to assert that the expected exception is thrown.
12. **Q:** What is the difference between test fixture and test suite?  
    **A:** A test fixture sets up the environment for tests, while a test suite groups related tests.
13. **Q:** How would you test a method with side effects?  
    **A:** Use mocks to verify the side effects or isolate the method for testing.
14. **Q:** What is the difference between white-box and black-box testing?  
    **A:** White-box testing uses internal knowledge, while black-box testing treats the system as a black box.
15. **Q:** How would you test a method that depends on time?  
    **A:** Use a mock or fake clock to control time in tests.
16. **Q:** What is the difference between test coverage and `code coverage

**Cloud Platforms (Azure, AWS)**

1. **Q:** How would you implement auto-scaling in AWS?  
   **A:** Use AWS Auto Scaling Groups to adjust the number of EC2 instances based on demand.
2. **Q:** What is the difference between S3 and EBS in AWS?  
   **A:** S3 is object storage, while EBS is block storage for EC2 instances.
3. **Q:** How would you implement a serverless function in Azure?  
   **A:** Use Azure Functions to run code without managing infrastructure.
4. **Q:** What is the difference between Azure Blob Storage and Azure Table Storage?  
   **A:** Blob Storage is for unstructured data, while Table Storage is for NoSQL key-value data.
5. **Q:** How would you implement a load balancer in AWS?  
   **A:** Use AWS Elastic Load Balancer (ELB) to distribute traffic across instances.
6. **Q:** What is the difference between AWS Lambda and Azure Functions?  
   **A:** Both are serverless, but Lambda is AWS-specific, while Azure Functions are for Azure.
7. **Q:** How would you implement a virtual network in Azure?  
   **A:** Use Azure Virtual Network (VNet) to create isolated networks.
8. **Q:** What is the difference between AWS EC2 and Azure VMs?  
   **A:** Both are virtual machines, but EC2 is AWS-specific, while Azure VMs are for Azure.
9. **Q:** How would you implement a database in AWS?  
   **A:** Use AWS RDS for relational databases or DynamoDB for NoSQL.
10. **Q:** What is the difference between AWS S3 and Azure Blob Storage?  
    **A:** Both are object storage, but S3 is AWS-specific, while Blob Storage is for Azure.
11. **Q:** How would you implement a CDN in Azure?  
    **A:** Use Azure CDN to distribute content globally.
12. **Q:** What is the difference between AWS CloudFormation and Azure Resource Manager?  
    **A:** Both are infrastructure-as-code tools, but CloudFormation is AWS-specific, while ARM is for Azure.
13. **Q:** How would you implement a message queue in AWS?  
    **A:** Use AWS SQS to decouple and scale microservices.
14. **Q:** What is the difference between AWS IAM and Azure AD?  
    **A:** IAM is for AWS resource access control, while Azure AD is for identity and access management in Azure.
15. **Q:** How would you implement a container in AWS?  
    **A:** Use AWS ECS or EKS to run Docker containers.
16. **Q:** What is the difference between AWS Glacier and Azure Archive Storage?  
    **A:** Both are for long-term storage, but Glacier is AWS-specific, while Archive Storage is for Azure.
17. **Q:** How would you implement a monitoring solution in Azure?  
    **A:** Use Azure Monitor to collect and analyze metrics and logs.
18. **Q:** What is the difference between AWS RDS and Azure SQL Database?  
    **A:** Both are managed relational databases, but RDS is AWS-specific, while Azure SQL Database is for Azure.
19. **Q:** How would you implement a serverless API in AWS?  
    **A:** Use AWS API Gateway with Lambda to create a serverless API.
20. **Q:** What is the difference between AWS CloudWatch and Azure Monitor?  
    **A:** Both are monitoring tools, but CloudWatch is AWS-specific, while Azure Monitor is for Azure.

**Web Services (REST)**

1. **Q:** How would you version REST APIs, and what are the pros and cons of each approach?  
   **A:** Use URL paths (/v1/resource) or headers (Accept: application/vnd.example.v1+json). URL versioning is simpler but less clean, while header versioning is cleaner but harder to debug.
2. **Q:** What is the difference between PUT and PATCH?  
   **A:** PUT replaces the entire resource, while PATCH updates only specified fields.
3. **Q:** How would you implement rate limiting in a REST API?  
   **A:** Use middleware to track request counts and throttle excessive requests.
4. **Q:** What is the difference between 401 Unauthorized and 403 Forbidden?  
   **A:** 401 indicates missing or invalid authentication, while 403 indicates the user is authenticated but not authorized.
5. **Q:** How would you implement pagination in a REST API?  
   **A:** Use query parameters like page and limit and include metadata in the response.
6. **Q:** What is the difference between SOAP and REST?  
   **A:** SOAP is a protocol with strict standards, while REST is an architectural style using HTTP.
7. **Q:** How would you implement HATEOAS in a REST API?  
   **A:** Include hypermedia links in the response to guide clients through the API.
8. **Q:** What is the difference between JWT and OAuth?  
   **A:** JWT is a token format, while OAuth is an authorization framework.
9. **Q:** How would you implement caching in a REST API?  
   **A:** Use HTTP headers like Cache-Control and ETag to control caching behavior.
10. **Q:** What is the difference between 200 OK and 204 No Content?  
    **A:** 200 includes a response body, while 204 indicates success with no content.
11. **Q:** How would you implement authentication in a REST API?  
    **A:** Use JWT or OAuth tokens and validate them in middleware.
12. **Q:** What is the difference between POST and PUT?  
    **A:** POST creates a new resource, while PUT updates or replaces an existing resource.
13. **Q:** How would you implement error handling in a REST API?  
    **A:** Use consistent error responses with status codes and descriptive messages.
14. **Q:** What is the difference between REST and GraphQL?  
    **A:** REST uses fixed endpoints, while GraphQL allows clients to query for specific data.
15. **Q:** How would you implement versioning in a REST API?  
    **A:** Use URL paths (/v1/resource) or headers (Accept: application/vnd.example.v1+json).
16. **Q:** What is the difference between GET and HEAD?  
    **A:** GET retrieves the resource, while HEAD retrieves only the headers.
17. **Q:** How would you implement file uploads in a REST API?  
    **A:** Use multipart/form-data and handle the file in the backend.
18. **Q:** What is the difference between REST and gRPC?  
    **A:** REST uses HTTP/JSON, while gRPC uses HTTP/2 and Protocol Buffers.
19. **Q:** How would you implement rate limiting in a REST API?  
    **A:** Use middleware to track request counts and throttle excessive requests.
20. **Q:** What is the difference between REST and WebSockets?  
    **A:** REST is request-response, while WebSockets enable real-time, bidirectional communication.

**SQL**

1. **Q:** How would you optimize a slow-running SQL query?  
   **A:** Analyze the execution plan, add indexes, and avoid SELECT \*.
2. **Q:** What is the difference between INNER JOIN and LEFT JOIN?  
   **A:** INNER JOIN returns matching rows, while LEFT JOIN returns all rows from the left table and matching rows from the right.
3. **Q:** How would you implement a recursive query in SQL?  
   **A:** Use a Common Table Expression (CTE) with a recursive anchor and recursive member.
4. **Q:** What is the difference between WHERE and HAVING?  
   **A:** WHERE filters rows before grouping, while HAVING filters after grouping.
5. **Q:** How would you implement a pivot table in SQL?  
   **A:** Use the PIVOT operator to transform rows into columns.
6. **Q:** What is the difference between DELETE and TRUNCATE?  
   **A:** DELETE removes rows one by one and can be rolled back, while TRUNCATE removes all rows at once and cannot be rolled back.
7. **Q:** How would you implement a full-text search in SQL?  
   **A:** Use CONTAINS or FREETEXT with a full-text index.
8. **Q:** What is the difference between ROW\_NUMBER(), RANK(), and DENSE\_RANK()?  
   **A:** ROW\_NUMBER() assigns unique numbers, RANK() leaves gaps for ties, and DENSE\_RANK() does not leave gaps.
9. **Q:** How would you implement a transaction in SQL?  
   **A:** Use BEGIN TRANSACTION, COMMIT, and ROLLBACK to ensure atomicity.
10. **Q:** What is the difference between a clustered and non-clustered index?  
    **A:** A clustered index determines the physical order of data, while a non-clustered index is a separate structure.
11. **Q:** How would you implement a stored procedure with parameters?  
    **A:** Define the procedure with CREATE PROCEDURE and use @parameter for input.
12. **Q:** What is the difference between UNION and UNION ALL?  
    **A:** UNION removes duplicates, while UNION ALL includes all rows.
13. **Q:** How would you implement a trigger in SQL?  
    **A:** Use CREATE TRIGGER to define logic that executes on specific events (e.g., INSERT, UPDATE).
14. **Q:** What is the difference between CHAR and VARCHAR?  
    **A:** CHAR is fixed-length, while VARCHAR is variable-length.
15. **Q:** How would you implement a backup and restore strategy in SQL?  
    **A:** Use BACKUP DATABASE and RESTORE DATABASE commands with a schedule.
16. **Q:** What is the difference between CROSS JOIN and INNER JOIN?  
    **A:** CROSS JOIN returns the Cartesian product, while INNER JOIN returns matching rows.
17. **Q:** How would you implement a view in SQL?  
    **A:** Use CREATE VIEW to define a virtual table based on a query.
18. **Q:** What is the difference between GROUP BY and DISTINCT?  
    **A:** GROUP BY groups rows for aggregation, while DISTINCT removes duplicates.
19. **Q:** How would you implement a foreign key constraint?  
    **A:** Use ALTER TABLE with ADD CONSTRAINT to enforce referential integrity.
20. **Q:** What is the difference between ISNULL and COALESCE?  
    **A:** ISNULL replaces NULL with a single value, while COALESCE returns the first non-null value in a list.

**HTML, CSS, JavaScript, jQuery, and APIs**

1. **Q:** How would you implement a responsive layout without using media queries?  
   **A:** Use CSS Grid or Flexbox with relative units (e.g., %, em, rem) to create a fluid layout.
2. **Q:** What is the difference between display: none and visibility: hidden?  
   **A:** display: none removes the element from the layout, while visibility: hidden hides it but retains its space.
3. **Q:** How would you implement a dark mode toggle using CSS variables?  
   **A:** Define CSS variables for light and dark themes and toggle them using JavaScript.
4. **Q:** What is the difference between == and === in JavaScript?  
   **A:** == performs type coercion, while === checks both value and type.
5. **Q:** How would you implement a debounce function in JavaScript?  
   **A:** Use setTimeout to delay the execution of a function and clear the timeout on subsequent calls.
6. **Q:** What is the difference between let, const, and var in JavaScript?  
   **A:** let and const are block-scoped, while var is function-scoped. const cannot be reassigned.
7. **Q:** How would you implement a custom scrollbar using CSS?  
   **A:** Use ::-webkit-scrollbar pseudo-elements to style the scrollbar.
8. **Q:** What is the difference between event.preventDefault() and event.stopPropagation()?  
   **A:** preventDefault() stops the default action, while stopPropagation() stops the event from bubbling up.
9. **Q:** How would you implement a sticky header in CSS?  
   **A:** Use position: sticky and define the top offset.
10. **Q:** What is the difference between async and defer in script loading?  
    **A:** async loads scripts asynchronously, while defer delays execution until the DOM is ready.
11. **Q:** How would you implement a custom dropdown menu using CSS and JavaScript?  
    **A:** Use :hover or :focus to toggle visibility and style the dropdown.
12. **Q:** What is the difference between flexbox and grid?  
    **A:** Flexbox is for one-dimensional layouts, while Grid is for two-dimensional layouts.
13. **Q:** How would you implement a responsive image gallery using CSS Grid?  
    **A:** Use grid-template-columns with repeat and minmax to create a flexible grid layout.
14. **Q:** What is the difference between margin and padding?  
    **A:** margin creates space outside an element, while padding creates space inside.
15. **Q:** How would you implement a custom checkbox using CSS?  
    **A:** Use :checked and ::before or ::after pseudo-elements to style the checkbox.
16. **Q:** What is the difference between localStorage and sessionStorage?  
    **A:** localStorage persists data until manually cleared, while sessionStorage clears data when the session ends.
17. **Q:** How would you implement a custom tooltip using CSS?  
    **A:** Use ::before or ::after pseudo-elements and :hover to display the tooltip.
18. **Q:** What is the difference between GET and POST requests?  
    **A:** GET retrieves data, while POST submits data to be processed.
19. **Q:** How would you implement a custom modal using JavaScript?  
    **A:** Use display: none to hide the modal and toggle it with JavaScript.
20. **Q:** What is the difference between JSON.parse and JSON.stringify?  
    **A:** JSON.parse converts a JSON string to an object, while JSON.stringify converts an object to a JSON string.

**TypeScript**

1. **Q:** What is the difference between interface and type in TypeScript?  
   **A:** interface can be extended, while type can define unions and intersections.
2. **Q:** How would you implement a generic function in TypeScript?  
   **A:** Use <T> to define a generic type and use it in the function signature.
3. **Q:** What is the difference between any and unknown?  
   **A:** any disables type checking, while unknown requires type assertion before use.
4. **Q:** How would you implement a decorator in TypeScript?  
   **A:** Create a function that takes a target and modifies it.
5. **Q:** What is the difference between readonly and const?  
   **A:** readonly is for properties, while const is for variables.
6. **Q:** How would you implement a mapped type in TypeScript?  
   **A:** Use keyof and in to create a new type based on an existing one.
7. **Q:** What is the difference between namespace and module?  
   **A:** namespace is for internal modules, while module is for external modules.
8. **Q:** How would you implement a conditional type in TypeScript?  
   **A:** Use extends and ? to define a type based on a condition.
9. **Q:** What is the difference between never and void?  
   **A:** never represents a function that never returns, while void represents a function that returns nothing.
10. **Q:** How would you implement a type guard in TypeScript?  
    **A:** Create a function that checks the type and returns a type predicate.
11. **Q:** What is the difference between abstract and interface?  
    **A:** abstract can have implementations, while interface cannot.
12. **Q:** How would you implement a utility type in TypeScript?  
    **A:** Use built-in utility types like Partial, Pick, or Record.
13. **Q:** What is the difference between as and ! in TypeScript?  
    **A:** as is for type assertion, while ! is for non-null assertion.
14. **Q:** How would you implement a mixin in TypeScript?  
    **A:** Use a function that extends a class with additional behavior.
15. **Q:** What is the difference between enum and const enum?  
    **A:** enum generates runtime code, while const enum is inlined at compile time.
16. **Q:** How would you implement a discriminated union in TypeScript?  
    **A:** Use a common property to distinguish between types in a union.
17. **Q:** What is the difference between keyof and typeof?  
    **A:** keyof gets the keys of a type, while typeof gets the type of a value.
18. **Q:** How would you implement a recursive type in TypeScript?  
    **A:** Define a type that references itself, e.g., type Tree<T> = { value: T; children: Tree<T>[] }.
19. **Q:** What is the difference between infer and extends?  
    **A:** infer extracts a type within a conditional type, while extends checks if a type satisfies a condition.
20. **Q:** How would you implement a type-safe dictionary in TypeScript?  
    **A:** Use Record<string, T> or { [key: string]: T }.

**ASP.NET Core**

1. **Q:** How would you implement middleware in ASP.NET Core?  
   **A:** Create a class with Invoke or InvokeAsync method and register it in Startup.cs.
2. **Q:** What is the difference between AddTransient, AddScoped, and AddSingleton in dependency injection?  
   **A:** AddTransient creates a new instance each time, AddScoped creates one per request, and AddSingleton creates one for the application lifetime.
3. **Q:** How would you implement custom model validation in ASP.NET Core?  
   **A:** Create a custom validation attribute by inheriting from ValidationAttribute.
4. **Q:** What is the difference between IApplicationBuilder.Use and IApplicationBuilder.Run?  
   **A:** Use allows the next middleware to execute, while Run terminates the pipeline.
5. **Q:** How would you implement JWT authentication in ASP.NET Core?  
   **A:** Use AddAuthentication with JwtBearerDefaults.AuthenticationScheme and configure the token validation parameters.
6. **Q:** What is the difference between ActionResult and IActionResult?  
   **A:** ActionResult is a concrete type, while IActionResult is an interface representing various HTTP responses.
7. **Q:** How would you implement rate limiting in ASP.NET Core?  
   **A:** Use middleware like AspNetCoreRateLimit to limit requests based on IP or other criteria.
8. **Q:** What is the difference between appsettings.json and environment variables in ASP.NET Core?  
   **A:** appsettings.json is a configuration file, while environment variables are used for environment-specific settings.
9. **Q:** How would you implement a custom route constraint in ASP.NET Core?  
   **A:** Create a class that implements IRouteConstraint and register it in Startup.cs.
10. **Q:** What is the difference between FromServices and FromBody in ASP.NET Core?  
    **A:** FromServices injects a service, while FromBody binds data from the request body.
11. **Q:** How would you implement a custom exception filter in ASP.NET Core?  
    **A:** Create a class that implements IExceptionFilter and register it in Startup.cs.
12. **Q:** What is the difference between Map and MapWhen in middleware?  
    **A:** Map branches the pipeline based on the path, while MapWhen branches based on a condition.
13. **Q:** How would you implement a custom output formatter in ASP.NET Core?  
    **A:** Create a class that implements OutputFormatter and register it in Startup.cs.
14. **Q:** What is the difference between AddMvc and AddControllersWithViews?  
    **A:** AddMvc registers all MVC services, while AddControllersWithViews registers only controllers and views.
15. **Q:** How would you implement a custom authorization policy in ASP.NET Core?  
    **A:** Use AuthorizationPolicyBuilder to define the policy and register it in Startup.cs.
16. **Q:** What is the difference between IHostedService and BackgroundService?  
    **A:** IHostedService is the base interface, while BackgroundService is an abstract class for background tasks.
17. **Q:** How would you implement a custom view component in ASP.NET Core?  
    **A:** Create a class that inherits from ViewComponent and define its Invoke method.
18. **Q:** What is the difference between UseRouting and UseEndpoints?  
    **A:** UseRouting sets up routing, while UseEndpoints defines the endpoints.
19. **Q:** How would you implement a custom tag helper in ASP.NET Core?  
    **A:** Create a class that inherits from TagHelper and define its Process method.
20. **Q:** What is the difference between AddDbContext and AddDbContextPool?  
    **A:** AddDbContext creates a new instance each time, while AddDbContextPool reuses instances from a pool.

**C#**

1. **Q:** What is the difference between IEnumerable and IQueryable?  
   **A:** IEnumerable executes queries in memory, while IQueryable builds queries that are executed on the database.
2. **Q:** How would you implement a thread-safe singleton in C#?  
   **A:** Use Lazy<T> or double-check locking for thread-safe initialization.
3. **Q:** What is the difference between async and await in C#?  
   **A:** async marks a method as asynchronous, while await pauses execution until the task completes.
4. **Q:** How would you implement a custom exception in C#?  
   **A:** Create a class that inherits from Exception and add custom properties or methods.
5. **Q:** What is the difference between ref and out parameters in C#?  
   **A:** ref passes a reference to a variable, while out is used for output parameters.
6. **Q:** How would you implement a custom attribute in C#?  
   **A:** Create a class that inherits from Attribute and define its properties or methods.
7. **Q:** What is the difference between Task and ValueTask in C#?  
   **A:** Task is a reference type, while ValueTask is a value type optimized for performance.
8. **Q:** How would you implement a custom collection in C#?  
   **A:** Create a class that implements IEnumerable<T> and define its methods.
9. **Q:** What is the difference between StringBuilder and string in C#?  
   **A:** StringBuilder is mutable and optimized for concatenation, while string is immutable.
10. **Q:** How would you implement a custom delegate in C#?  
    **A:** Use the delegate keyword to define a custom delegate type.
11. **Q:** What is the difference between IComparable and IComparer in C#?  
    **A:** IComparable defines natural ordering, while IComparer defines custom ordering.
12. **Q:** How would you implement a custom LINQ operator in C#?  
    **A:** Create an extension method that operates on IEnumerable<T> or IQueryable<T>.
13. **Q:** What is the difference between yield return and return in C#?  
    **A:** yield return enables deferred execution, while return immediately exits the method.
14. **Q:** How would you implement a custom event in C#?  
    **A:** Use the event keyword to define an event and EventHandler to handle it.
15. **Q:** What is the difference between struct and class in C#?  
    **A:** struct is a value type, while class is a reference type.
16. **Q:** How would you implement a custom indexer in C#?  
    **A:** Use the this keyword to define an indexer property.
17. **Q:** What is the difference between Task.Run and Task.Factory.StartNew in C#?  
    **A:** Task.Run is a simplified API for starting tasks, while Task.Factory.StartNew provides more options.
18. **Q:** How would you implement a custom serializer in C#?  
    **A:** Create a class that implements ISerializer and define its methods.
19. **Q:** What is the difference between async void and async Task in C#?  
    **A:** async void is for event handlers and cannot be awaited, while async Task can be awaited.
20. **Q:** How would you implement a custom comparer in C#?  
    **A:** Create a class that implements IComparer<T> and define its Compare method.

**Entity Framework (EF Core)**

1. **Q:** How would you optimize EF Core performance for a high-traffic application?  
   **A:** Use eager loading, disable change tracking for read-only queries, and optimize database indexes.
2. **Q:** What is the difference between eager loading and lazy loading in EF Core?  
   **A:** Eager loading loads related entities upfront, while lazy loading loads them on demand.
3. **Q:** How would you handle database migrations in EF Core?  
   **A:** Use Add-Migration and Update-Database commands to manage schema changes.
4. **Q:** What is the difference between DbContext and DbSet?  
   **A:** DbContext represents a session with the database, while DbSet represents a collection of entities.
5. **Q:** How would you implement a many-to-many relationship in EF Core?  
   **A:** Use a join entity to represent the relationship and configure it in the OnModelCreating method.
6. **Q:** What is the difference between AsNoTracking and AsTracking in EF Core?  
   **A:** AsNoTracking disables change tracking for read-only queries, while AsTracking enables it.
7. **Q:** How would you implement a custom query in EF Core?  
   **A:** Use FromSqlRaw or FromSqlInterpolated to execute raw SQL queries.
8. **Q:** What is the difference between IQueryable and IEnumerable in EF Core?  
   **A:** IQueryable builds queries that are executed on the database, while IEnumerable executes queries in memory.
9. **Q:** How would you handle concurrency conflicts in EF Core?  
   **A:** Use optimistic concurrency control with a RowVersion column.
10. **Q:** What is the difference between Add and Attach in EF Core?  
    **A:** Add marks an entity as new, while Attach marks it as existing but unchanged.
11. **Q:** How would you implement a transaction in EF Core?  
    **A:** Use DbContext.Database.BeginTransaction to start a transaction and commit or rollback as needed.
12. **Q:** What is the difference between Find and FirstOrDefault in EF Core?  
    **A:** Find uses the primary key to locate an entity, while FirstOrDefault uses a query.
13. **Q:** How would you implement a custom migration in EF Core?  
    **A:** Use MigrationBuilder to define custom SQL commands in the Up and Down methods.
14. **Q:** What is the difference between Include and ThenInclude in EF Core?  
    **A:** Include loads related entities, while ThenInclude loads nested related entities.
15. **Q:** How would you handle soft deletes in EF Core?  
    **A:** Use a IsDeleted flag and override the SaveChanges method to filter deleted entities.
16. **Q:** What is the difference between HasOne and WithOne in EF Core?  
    **A:** HasOne defines the principal entity, while WithOne defines the dependent entity.
17. **Q:** How would you implement a custom value converter in EF Core?  
    **A:** Use HasConversion to define a custom conversion between database and application types.
18. **Q:** What is the difference between OnConfiguring and OnModelCreating in EF Core?  
    **A:** OnConfiguring configures the database provider, while OnModelCreating configures the model.
19. **Q:** How would you implement a global query filter in EF Core?  
    **A:** Use HasQueryFilter in the OnModelCreating method to apply a filter to all queries.
20. **Q:** What is the difference between ExecuteSqlRaw and ExecuteSqlInterpolated in EF Core?  
    **A:** ExecuteSqlRaw executes raw SQL, while ExecuteSqlInterpolated uses string interpolation for safer queries.

**SSO (Single Sign-On) and SLO (Single Logout)**

1. **Q:** What is the difference between SAML and OAuth in SSO implementations?  
   **A:** SAML is XML-based and used for authentication, while OAuth is token-based and used for authorization.
2. **Q:** How would you implement SSO in a microservices architecture?  
   **A:** Use OAuth 2.0 with a centralized identity provider (e.g., Keycloak, Auth0) to issue tokens for each service.
3. **Q:** What is the difference between IDP-initiated and SP-initiated SSO?  
   **A:** IDP-initiated SSO starts at the identity provider, while SP-initiated SSO starts at the service provider.
4. **Q:** How would you handle token expiration in an SSO system?  
   **A:** Use refresh tokens to obtain new access tokens without requiring user reauthentication.
5. **Q:** What is the difference between JWT and SAML tokens?  
   **A:** JWT is JSON-based and lightweight, while SAML is XML-based and more complex.
6. **Q:** How would you implement Single Logout (SLO) in a distributed system?  
   **A:** Use a centralized logout endpoint to invalidate sessions across all services.
7. **Q:** What is the difference between OpenID Connect and OAuth 2.0?  
   **A:** OpenID Connect adds an authentication layer on top of OAuth 2.0’s authorization framework.
8. **Q:** How would you secure an SSO implementation against replay attacks?  
   **A:** Use nonce values and short-lived tokens to prevent token reuse.
9. **Q:** What is the difference between federated SSO and enterprise SSO?  
   **A:** Federated SSO connects multiple organizations, while enterprise SSO connects internal systems.
10. **Q:** How would you implement SSO for a mobile application?  
    **A:** Use OAuth 2.0 with PKCE (Proof Key for Code Exchange) for secure token exchange.
11. **Q:** What is the difference between session cookies and tokens in SSO?  
    **A:** Session cookies are stored in the browser, while tokens are passed in HTTP headers.
12. **Q:** How would you handle user role mapping in an SSO system?  
    **A:** Use claims in the token to map user roles to application permissions.
13. **Q:** What is the difference between implicit and authorization code flows in OAuth 2.0?  
    **A:** Implicit flow returns tokens directly, while authorization code flow uses an intermediate code.
14. **Q:** How would you implement SSO for a legacy application?  
    **A:** Use a reverse proxy or SAML bridge to integrate the legacy app with the SSO system.
15. **Q:** What is the difference between SP-initiated and IDP-initiated SLO?  
    **A:** SP-initiated SLO starts at the service provider, while IDP-initiated SLO starts at the identity provider.
16. **Q:** How would you handle token revocation in an SSO system?  
    **A:** Use a token revocation list or short-lived tokens to minimize the impact of compromised tokens.
17. **Q:** What is the difference between bearer tokens and proof-of-possession tokens?  
    **A:** Bearer tokens can be used by anyone, while proof-of-possession tokens require additional proof (e.g., a private key).
18. **Q:** How would you implement SSO for a cloud-based application?  
    **A:** Use a cloud identity provider (e.g., Azure AD, AWS Cognito) to manage authentication.
19. **Q:** What is the difference between SAML 1.1 and SAML 2.0?  
    **A:** SAML 2.0 introduces new features like metadata exchange and enhanced security.
20. **Q:** How would you handle user consent in an SSO system?  
    **A:** Use OAuth 2.0’s consent mechanism to prompt users for permission to access resources.

**Microsoft SQL Server**

1. **Q:** How would you optimize a slow-running SQL query?  
   **A:** Analyze the execution plan, add indexes, and avoid SELECT \*.
2. **Q:** What is the difference between INNER JOIN and LEFT JOIN?  
   **A:** INNER JOIN returns matching rows, while LEFT JOIN returns all rows from the left table and matching rows from the right.
3. **Q:** How would you implement a recursive query in SQL Server?  
   **A:** Use a Common Table Expression (CTE) with a recursive anchor and recursive member.
4. **Q:** What is the difference between WHERE and HAVING?  
   **A:** WHERE filters rows before grouping, while HAVING filters after grouping.
5. **Q:** How would you implement a pivot table in SQL Server?  
   **A:** Use the PIVOT operator to transform rows into columns.
6. **Q:** What is the difference between DELETE and TRUNCATE?  
   **A:** DELETE removes rows one by one and can be rolled back, while TRUNCATE removes all rows at once and cannot be rolled back.
7. **Q:** How would you implement a full-text search in SQL Server?  
   **A:** Use CONTAINS or FREETEXT with a full-text index.
8. **Q:** What is the difference between ROW\_NUMBER(), RANK(), and DENSE\_RANK()?  
   **A:** ROW\_NUMBER() assigns unique numbers, RANK() leaves gaps for ties, and DENSE\_RANK() does not leave gaps.
9. **Q:** How would you implement a transaction in SQL Server?  
   **A:** Use BEGIN TRANSACTION, COMMIT, and ROLLBACK to ensure atomicity.
10. **Q:** What is the difference between a clustered and non-clustered index?  
    **A:** A clustered index determines the physical order of data, while a non-clustered index is a separate structure.
11. **Q:** How would you implement a stored procedure with parameters?  
    **A:** Define the procedure with CREATE PROCEDURE and use @parameter for input.
12. **Q:** What is the difference between UNION and UNION ALL?  
    **A:** UNION removes duplicates, while UNION ALL includes all rows.
13. **Q:** How would you implement a trigger in SQL Server?  
    **A:** Use CREATE TRIGGER to define logic that executes on specific events (e.g., INSERT, UPDATE).
14. **Q:** What is the difference between CHAR and VARCHAR?  
    **A:** CHAR is fixed-length, while VARCHAR is variable-length.
15. **Q:** How would you implement a backup and restore strategy in SQL Server?  
    **A:** Use BACKUP DATABASE and RESTORE DATABASE commands with a schedule.
16. **Q:** What is the difference between CROSS JOIN and INNER JOIN?  
    **A:** CROSS JOIN returns the Cartesian product, while INNER JOIN returns matching rows.
17. **Q:** How would you implement a view in SQL Server?  
    **A:** Use CREATE VIEW to define a virtual table based on a query.
18. **Q:** What is the difference between GROUP BY and DISTINCT?  
    **A:** GROUP BY groups rows for aggregation, while DISTINCT removes duplicates.
19. **Q:** How would you implement a foreign key constraint?  
    **A:** Use ALTER TABLE with ADD CONSTRAINT to enforce referential integrity.
20. **Q:** What is the difference between ISNULL and COALESCE?  
    **A:** ISNULL replaces NULL with a single value, while COALESCE returns the first non-null value in a list.

**Web Development**

1. **Q:** How do you ensure cross-browser compatibility in web development?  
   **A:** Use tools like BrowserStack, write standards-compliant code, and leverage polyfills for unsupported features.
2. **Q:** How would you optimize a website for mobile devices?  
   **A:** Use responsive design, optimize images, and minimize HTTP requests.
3. **Q:** What is the difference between HTTP/1.1 and HTTP/2?  
   **A:** HTTP/2 introduces multiplexing, header compression, and server push for faster performance.
4. **Q:** How would you implement lazy loading for images?  
   **A:** Use the loading="lazy" attribute or JavaScript to load images only when they enter the viewport.
5. **Q:** What is the difference between localStorage and sessionStorage?  
   **A:** localStorage persists data until manually cleared, while sessionStorage clears data when the session ends.
6. **Q:** How would you implement a dark mode toggle in a web application?  
   **A:** Use CSS variables and JavaScript to switch between light and dark themes.
7. **Q:** What is the difference between GET and POST requests?  
   **A:** GET retrieves data, while POST submits data to be processed.
8. **Q:** How would you implement a progressive web app (PWA)?  
   **A:** Use a service worker, manifest file, and HTTPS to enable offline functionality and app-like behavior.
9. **Q:** What is the difference between async and defer in script loading?  
   **A:** async loads scripts asynchronously, while defer delays execution until the DOM is ready.
10. **Q:** How would you implement a custom scrollbar in CSS?  
    **A:** Use ::-webkit-scrollbar pseudo-elements to style the scrollbar.
11. **Q:** What is the difference between margin and padding?  
    **A:** margin creates space outside an element, while padding creates space inside.
12. **Q:** How would you implement a sticky header in CSS?  
    **A:** Use position: sticky and define the top offset.
13. **Q:** What is the difference between flexbox and grid?  
    **A:** Flexbox is for one-dimensional layouts, while Grid is for two-dimensional layouts.
14. **Q:** How would you implement a responsive grid layout?  
    **A:** Use CSS Grid with media queries to adjust the layout for different screen sizes.
15. **Q:** What is the difference between em and rem units?  
    **A:** em is relative to the parent’s font size, while rem is relative to the root font size.
16. **Q:** How would you implement a CSS animation?  
    **A:** Use @keyframes to define the animation and apply it with animation properties.
17. **Q:** What is the difference between inline and block elements?  
    **A:** Inline elements flow within text, while block elements create a new line and take full width.
18. **Q:** How would you implement a custom dropdown menu in CSS?  
    **A:** Use :hover or :focus to toggle visibility and style the dropdown.
19. **Q:** What is the difference between position: absolute and position: relative?  
    **A:** absolute positions an element relative to its nearest positioned ancestor, while relative positions it relative to its normal position.
20. **Q:** How would you implement a responsive image gallery?  
    **A:** Use CSS Grid or Flexbox with media queries to adjust the layout for different screen sizes.

**Issue Tracking (Jira, ClickUp)**

1. **Q:** How would you implement a custom workflow in Jira for a complex project?  
   **A:** Use Jira’s workflow editor to define statuses, transitions, and conditions tailored to the project’s needs.
2. **Q:** What is the difference between a Jira epic and a story?  
   **A:** An epic is a large body of work broken into smaller stories, which are individual tasks.
3. **Q:** How would you implement a custom field in Jira to track technical debt?  
   **A:** Create a custom field (e.g., numeric or dropdown) and use it to quantify and prioritize technical debt.
4. **Q:** What is the difference between a Jira board and a project?  
   **A:** A board visualizes issues, while a project is a collection of issues and configurations.
5. **Q:** How would you implement a Jira automation rule to notify stakeholders of a blocked issue?  
   **A:** Create a rule that triggers an email or Slack notification when an issue’s status changes to "Blocked."
6. **Q:** What is the difference between a Jira filter and a dashboard?  
   **A:** A filter is a saved search, while a dashboard is a visual display of widgets based on filters.
7. **Q:** How would you implement a Jira workflow for a distributed team?  
   **A:** Define clear statuses (e.g., "In Progress," "Code Review") and transitions to reflect remote collaboration.
8. **Q:** What is the difference between a Jira issue and a task?  
   **A:** An issue is a generic term for any work item, while a task is a specific type of issue.
9. **Q:** How would you implement a Jira automation rule to escalate overdue issues?  
   **A:** Create a rule that reassigns or notifies stakeholders when an issue’s due date passes.
10. **Q:** What is the difference between a Jira sprint and a release?  
    **A:** A sprint is a time-boxed iteration, while a release is a deliverable set of features.
11. **Q:** How would you implement a Jira workflow for a cross-functional team?  
    **A:** Define statuses (e.g., "Design," "Development," "Testing") to reflect cross-functional collaboration.
12. **Q:** What is the difference between a Jira epic and a milestone?  
    **A:** An epic is a large body of work, while a milestone is a significant point in the project timeline.
13. **Q:** How would you implement a Jira automation rule to assign issues based on priority?  
    **A:** Create a rule that assigns high-priority issues to senior team members.
14. **Q:** What is the difference between a Jira board and a backlog?  
    **A:** A board visualizes active work, while a backlog lists all pending issues.
15. **Q:** How would you implement a Jira workflow for a remote team?  
    **A:** Define clear statuses and transitions to reflect asynchronous collaboration.
16. **Q:** What is the difference between a Jira issue and a bug?  
    **A:** An issue is a generic term, while a bug is a specific type of issue representing a defect.
17. **Q:** How would you implement a Jira automation rule to notify stakeholders of a high-priority issue?  
    **A:** Create a rule that sends notifications when an issue’s priority is set to "High."
18. **Q:** What is the difference between a Jira sprint and an iteration?  
    **A:** A sprint is a time-boxed iteration in Scrum, while an iteration is a generic term for a development cycle.
19. **Q:** How would you implement a Jira workflow for a large team?  
    **A:** Define statuses and transitions to reflect parallel workstreams and handoffs.
20. **Q:** What is the difference between a Jira epic and a feature?  
    **A:** An epic is a large body of work, while a feature is a specific functionality or deliverable.

**Code Versioning Tools (GIT, TFS, SVN, Azure DevOps)**

1. **Q:** How would you recover a deleted branch in Git without using the reflog?  
   **A:** Use git fsck to find the dangling commit and recreate the branch.
2. **Q:** What is the difference between a shallow clone and a deep clone in Git?  
   **A:** Shallow clone fetches only the latest commit, while deep clone fetches the entire history.
3. **Q:** How would you resolve a merge conflict in a large binary file?  
   **A:** Use a custom merge tool or manually choose the correct version.
4. **Q:** What is the difference between git rebase and git merge? When would you use one over the other?  
   **A:** Rebase rewrites commit history, while merge preserves it. Use rebase for cleaner history and merge for collaborative workflows.
5. **Q:** How would you implement a Git hook to enforce coding standards?  
   **A:** Use a pre-commit hook to run linting tools and reject non-compliant code.
6. **Q:** What is the difference between git reset and git revert?  
   **A:** Reset moves the branch pointer, while revert creates a new commit to undo changes.
7. **Q:** How would you squash multiple commits into a single commit without using interactive rebase?  
   **A:** Use git merge --squash to combine commits into a single one.
8. **Q:** What is the difference between git fetch and git pull?  
   **A:** Fetch retrieves remote changes, while pull fetches and merges them.
9. **Q:** How would you implement a Git workflow for a large team with multiple feature branches?  
   **A:** Use GitFlow or a similar branching model with develop, feature, and release branches.
10. **Q:** What is the difference between git stash and git cherry-pick?  
    **A:** Stash temporarily saves changes, while cherry-pick applies specific commits to another branch.
11. **Q:** How would you implement a Git bisect to find a bug in a large codebase?  
    **A:** Use git bisect to perform a binary search between a known good and bad commit.
12. **Q:** What is the difference between git submodule and git subtree?  
    **A:** Submodule links external repos, while subtree merges them into the main repo.
13. **Q:** How would you implement a Git hook to prevent pushing to the main branch?  
    **A:** Use a pre-push hook to check the branch name and reject pushes to main.
14. **Q:** What is the difference between git reflog and git log?  
    **A:** Reflog tracks all reference changes, while log shows commit history.
15. **Q:** How would you implement a Git workflow for a monorepo?  
    **A:** Use tools like lerna or nx to manage multiple projects in a single repo.
16. **Q:** What is the difference between git blame and git log?  
    **A:** Blame shows who last modified each line, while log shows commit history.
17. **Q:** How would you implement a Git hook to enforce commit message conventions?  
    **A:** Use a commit-msg hook to validate message format.
18. **Q:** What is the difference between git cherry-pick and git rebase?  
    **A:** Cherry-pick applies specific commits, while rebase rewrites commit history.
19. **Q:** How would you implement a Git workflow for a distributed team?  
    **A:** Use feature branches, pull requests, and regular syncs with the main branch.
20. **Q:** What is the difference between git tag and git branch?  
    **A:** Tag marks a specific point in history, while branch is a movable pointer to commits.

**Design Patterns**

1. **Q:** How would you implement a thread-safe singleton pattern without using locks?  
   **A:** Use static initialization or Lazy<T> for thread-safe singleton creation.
2. **Q:** What is the difference between the Proxy and Decorator patterns? Provide a real-world example of each.  
   **A:** Proxy controls access to an object, while Decorator adds behavior. Example: Proxy for lazy loading, Decorator for adding logging.
3. **Q:** How would you implement a chain of responsibility pattern for handling multiple levels of user authentication?  
   **A:** Create a chain of handlers where each handler processes the request or passes it to the next.
4. **Q:** What is the difference between the Observer and Pub/Sub patterns?  
   **A:** Observer is a direct relationship between subject and observer, while Pub/Sub uses a message broker for decoupling.
5. **Q:** How would you implement a caching mechanism using the Proxy pattern?  
   **A:** Create a proxy that checks the cache before delegating to the real object.
6. **Q:** What is the difference between the Strategy and State patterns?  
   **A:** Strategy encapsulates interchangeable algorithms, while State changes behavior based on internal state.
7. **Q:** How would you implement a distributed logging system using the Observer pattern?  
   **A:** Use observers to send logs to multiple destinations (e.g., file, database, cloud).
8. **Q:** What is the difference between the Adapter and Facade patterns?  
   **A:** Adapter makes incompatible interfaces work together, while Facade simplifies a complex subsystem.
9. **Q:** How would you implement a retry mechanism using the Command pattern?  
   **A:** Encapsulate the operation in a command object and retry it on failure.
10. **Q:** What is the difference between the Template Method and Strategy patterns?  
    **A:** Template Method defines a skeleton in a base class, while Strategy encapsulates interchangeable algorithms.
11. **Q:** How would you implement a load balancer using the Proxy pattern?  
    **A:** Create a proxy that distributes requests across multiple servers.
12. **Q:** What is the difference between the Composite and Decorator patterns?  
    **A:** Composite treats individual and group objects uniformly, while Decorator adds behavior dynamically.
13. **Q:** How would you implement a distributed transaction system using the Saga pattern?  
    **A:** Break the transaction into smaller, compensatable steps and handle failures with compensating actions.
14. **Q:** What is the difference between the Factory and Abstract Factory patterns?  
    **A:** Factory creates a single type of object, while Abstract Factory creates families of related objects.
15. **Q:** How would you implement a rate-limiting mechanism using the Proxy pattern?  
    **A:** Create a proxy that tracks request counts and throttles excessive requests.
16. **Q:** What is the difference between the Flyweight and Singleton patterns?  
    **A:** Flyweight shares state to reduce memory usage, while Singleton ensures a single instance.
17. **Q:** How would you implement a distributed locking mechanism using the Proxy pattern?  
    **A:** Create a proxy that coordinates locks across distributed systems.
18. **Q:** What is the difference between the Builder and Factory patterns?  
    **A:** Builder constructs complex objects step-by-step, while Factory creates objects in one go.
19. **Q:** How would you implement a circuit breaker pattern in a microservices architecture?  
    **A:** Use a circuit breaker to stop requests to a failing service and retry after a timeout.
20. **Q:** What is the difference between the Mediator and Observer patterns?  
    **A:** Mediator centralizes communication between objects, while Observer decouples subjects and observers.