1️⃣ Angular – Basics & Fundamentals

1. What is Angular and how is it different from AngularJS?
2. Explain the architecture of an Angular application.
3. What are components and modules in Angular?
4. What is a decorator? Explain @Component, @NgModule.
5. Difference between template-driven and reactive forms.
6. What is data binding? Explain all types.
7. What are directives? Structural vs Attribute directives.
8. Explain Angular CLI and its advantages.
9. What is dependency injection in Angular?
10. Explain the Angular application bootstrap process.

2️⃣ Angular – Algorithm Solving & DSA (UI / Logic-Oriented)

These are **logic-based Angular questions**, very common in interviews.

1. Reverse a string using Angular component logic.
2. Remove duplicate items from an array in Angular.
3. Implement a search filter for a list without using pipes.
4. Sort a table dynamically on column click.
5. Implement pagination logic in Angular.
6. Debounce a search input without external libraries.
7. Find max/min value from API response data.
8. Group array of objects by a key (e.g., category).
9. Implement a simple autocomplete logic.
10. Optimize rendering of large lists in Angular.

3️⃣ Angular – Intermediate (Frequently Asked Interview Questions)

1. What is change detection in Angular?
2. Difference between Default and OnPush change detection.
3. What are lifecycle hooks? Explain order.
4. Difference between Subject, BehaviorSubject, ReplaySubject.
5. What is an Observable? How is it different from Promise?

6. What are pipes? Pure vs impure pipes.
7. What is ViewChild and ContentChild?
8. Explain async pipe and its benefits.
9. What is HttpClient and how does it work?
10. How do you share data between components?

4️⃣ Angular – Advanced (Architecture & Ecosystem)

1. Explain Angular module loading (eager vs lazy).
2. What is Angular Ivy and its advantages?
3. Explain Zone.js and its role in Angular.
4. How does Angular handle state management?
5. What is Signal in Angular (Angular 16+)?
6. Explain SSR and Angular Universal.
7. How does Angular support micro-frontend architecture?
8. What is standalone component architecture?
9. How do you optimize Angular application performance?
10. How does Angular handle memory leaks?

5️⃣ Angular Forms & Validation (Enterprise Focus)

1. Difference between Reactive and Template-driven forms.
2. How do you implement custom validators?
3. Explain FormGroup, FormControl, FormArray.
4. How do you implement dynamic forms?
5. How do you handle async validation?
6. How do you manage form state efficiently?
7. What is cross-field validation?
8. How do you handle large forms performance?
9. How do you implement reusable form components?
10. How do you handle form errors globally?

6️⃣ Angular Routing & Security

1. How does Angular routing work internally?
2. Difference between CanActivate, CanLoad, CanDeactivate.
3. How do you secure routes in Angular?
4. What is lazy loading and why is it important?
5. Explain route resolvers.
6. How do you handle 404 pages?
7. How do you manage role-based access?
8. How do you protect against XSS in Angular?
9. What is CSRF and how does Angular handle it?
10. How do you manage authentication tokens securely?

7️⃣ Design Patterns in Angular

1. What design patterns are commonly used in Angular?
2. Explain Singleton pattern in Angular services.
3. How is Observer pattern used with RxJS?
4. Explain Facade pattern in Angular state management.
5. How is Strategy pattern used in Angular?
6. Explain Container–Presenter pattern.
7. What is Smart vs Dumb component pattern?
8. How is Dependency Injection a design pattern?
9. Explain Module Federation pattern.
10. How does Angular enforce separation of concerns?

8️⃣ Angular Capstone Project – Interview Questions

A. Architecture & Design

1. Explain your Angular project architecture.
2. Why did you choose this folder structure?
3. How did you manage shared modules?
4. How did you handle scalability?
5. How did you integrate Angular with backend APIs?
6. How did you manage environment configurations?

7. How did you handle authentication & authorization?
8. How did you implement state management?
9. How did you handle internationalization (i18n)?
10. What would you refactor in your project?

B. Coding & Implementation

1. Explain a complex component you developed.
2. How did you optimize performance?
3. How did you handle API errors globally?
4. How did you manage subscriptions?
5. How did you handle large data rendering?
6. How did you implement reusable components?
7. How did you manage forms at scale?
8. How did you integrate third-party libraries?
9. How did you write unit tests?
10. Explain a difficult bug you fixed.
