নিচে একটি **Lead Software Engineer (Angular)** পদের জন্য সম্ভাব্য ইন্টারভিউ প্রশ্ন ও উত্তর বাংলায় দেওয়া হলো। এই প্রশ্নগুলো মূলত Angular, নেতৃত্বদানের দক্ষতা এবং Software Engineering-এর গভীর জ্ঞানের উপর ভিত্তি করে তৈরি।

**🔹 ১. Angular এর কোন কোন ফিচার আপনি সবচেয়ে বেশি ব্যবহার করেছেন এবং কেন?**

**উত্তর:**  
আমি সবচেয়ে বেশি ব্যবহার করেছি:

* **Components & Modules:** অ্যাপ্লিকেশনকে ছোট ছোট অংশে ভাগ করার জন্য।
* **Reactive Forms:** Validation ও Dynamic Form Control এর জন্য।
* **RxJS (Observables, Operators):** Asynchronous data handle করার জন্য।
* **HttpClient:** API call করার জন্য।
* **Route Guards:** Authentication ও Authorization নিশ্চিত করার জন্য।
* **Change Detection Strategy (OnPush):** Performance optimization এর জন্য।

**🔹 ২. আপনি একটি বড় Angular প্রজেক্ট কীভাবে structure করবেন?**

**উত্তর:**  
আমি feature-based folder structure ফলো করি। যেমন:

/src/app/

├── core/

├── shared/

├── features/

├── dashboard/

├── user/

├── settings/

├── services/

├── guards/

├── interceptors/

* **Core module**: App-wide singleton services।
* **Shared module**: Common components, pipes, directives।
* **Feature modules**: Lazy loading এবং Scalability নিশ্চিত করে।

**🔹 ৩. আপনি কীভাবে Angular এ performance optimization করেন?**

**উত্তর:**

* OnPush Change Detection Strategy ব্যবহার করি।
* trackBy ব্যবহার করি \*ngFor এ।
* Lazy Loading implement করি।
* Unsubscribe করি Observables থেকে takeUntil() দিয়ে।
* Pure pipes ব্যবহার করি।
* Heavy computation হলে Web Workers ব্যবহার করি।

**🔹 ৪. RxJS এর সবচেয়ে ব্যবহৃত Operators গুলো কী কী এবং আপনি কীভাবে ব্যবহার করেন?**

**উত্তর:**

* **map**: ডেটা ট্রান্সফর্ম করার জন্য।
* **switchMap**: Nested Observable এর মধ্যে subscription handle করার জন্য।
* **mergeMap**: Parallel calls এর জন্য।
* **debounceTime**: Type-ahead input optimization এর জন্য।
* **catchError**: Error handle করার জন্য।
* **takeUntil**: Component destroy হলে subscription বন্ধ করার জন্য।

**🔹 ৫. আপনি কীভাবে Junior Developer দের মেন্টর করেন?**

**উত্তর:**

* Code Review করি এবং improvement এর পরামর্শ দেই।
* Pair Programming করি জটিল টাস্কে।
* Task breakdown করে ছোট ছোট অংশে দেই।
* Best Practices, SOLID, এবং Clean Code শেখাই।
* Weekly Knowledge Sharing সেশন আয়োজন করি।

**🔹 ৬. আপনি কীভাবে একটি Angular App এর সিকিউরিটি নিশ্চিত করেন?**

**উত্তর:**

* Route Guards দিয়ে route protection।
* JWT দিয়ে token-based authentication।
* HttpInterceptor দিয়ে request/response এ secure header add করি।
* XSS protection এর জন্য Angular এর DomSanitizer ব্যবহার করি।
* Server side validation নিশ্চিত করি।
* API তে CORS এবং Rate Limiting implement করি।

**🔹 ৭. আপনি কীভাবে CI/CD pipeline এ Angular অ্যাপ deploy করেন?**

**উত্তর:**

* Angular build করি ng build --prod দিয়ে।
* GitHub Actions / GitLab CI / Jenkins দিয়ে:
  + Code linting
  + Unit testing
  + Prod build
  + Artifact publish
  + Deployment to Firebase, AWS S3 বা Azure

**🔹 ৮. আপনি Angular version upgrade কিভাবে handle করেন?**

**উত্তর:**

* ng update কমান্ড ব্যবহার করি।
* Angular Update Guide অনুসরণ করি।
* Dependency গুলো update করি এবং breaking change note করি।
* Test coverage নিশ্চিত করি upgrade এর আগে ও পরে।
* Staging environment এ প্রথমে deploy করি।

**🔹 ৯. আপনি কীভাবে একটি Angular টিম পরিচালনা করেন?**

**উত্তর:**

* Task assign করি ডেভেলপারদের স্কিল অনুযায়ী।
* Daily Stand-up মিট করি।
* Jira বা Trello দিয়ে progress ট্র্যাক করি।
* Quality assurance নিশ্চিত করতে PR policy enforce করি।
* সমস্যা হলে দ্রুত unblock করি।

**🔹 ১০. Angular এর কোন বিষয়টি নতুনদের জন্য বেশি চ্যালেঞ্জিং এবং আপনি কীভাবে সাহায্য করেন?**

**উত্তর:**

* **RxJS (Observables & Operators)** — কারণ এটি async flow handle করা জটিল হতে পারে।
* **Routing with Lazy Loading** — Module structure বোঝা কষ্টকর হয়।

আমি ডায়াগ্রাম, ছোট উদাহরণ, এবং live demo দিয়ে বিষয়গুলো বোঝানোর চেষ্টা করি।

নিচে আমি আরও **বিশদভাবে ব্যাখ্যা** করছি **Lead Software Engineer (Angular)** পদের জন্য গুরুত্বপূর্ণ ইন্টারভিউ প্রশ্ন ও উত্তর **বাংলা ভাষায়**, যেখানে Angular এর advanced ধারণা, architectural সিদ্ধান্ত, এবং leadership দক্ষতার ব্যাখ্যা থাকবে:

**✅ ১. আপনি Angular Component architecture কিভাবে ডিজাইন করেন?**

**উত্তর:**  
Angular Component Architecture ডিজাইনের সময় আমি নিচের বিষয়গুলো অনুসরণ করি:

* **Smart vs Dumb Components**:
  + **Smart (Container) Component**: API call করে, business logic ধরে রাখে।
  + **Dumb (Presentational) Component**: শুধুমাত্র UI দেখায় ও Input/Output এর মাধ্যমে ডেটা পাঠায়।
* **Single Responsibility Principle (SRP)** ফলো করি।
  + প্রতিটি কম্পোনেন্ট একটি নির্দিষ্ট কাজ করে। যেমনঃ ProductListComponent শুধু পণ্যের তালিকা দেখাবে, EditProductComponent শুধু সম্পাদনার কাজ করবে।
* **Component Communication**:
  + @Input() ও @Output() দিয়ে parent-child component এ ডেটা শেয়ার করি।
  + Shared Service বা BehaviorSubject ব্যবহার করি sibling components এর মধ্যে ডেটা শেয়ারের জন্য।

**✅ ২. আপনি Lazy Loading কীভাবে করেন এবং এটি কবে করা উচিত?**

**উত্তর:**  
Lazy Loading হল Angular এর feature যেখানে মডিউলগুলো তখনই লোড হয় যখন প্রয়োজন হয়, এর ফলে initial load অনেক দ্রুত হয়।

**কিভাবে Lazy Load করি?**

* app-routing.module.ts ফাইলে lazy-loaded module define করি:

{

path: 'admin',

loadChildren: () => import('./admin/admin.module').then(m => m.AdminModule)

}

* AdminModule এর routing আলাদাভাবে থাকে (admin-routing.module.ts)

**কখন Lazy Loading ব্যবহার করবো?**

* অ্যাপ যদি বড় হয় এবং অনেক ফিচার থাকে।
* যদি কিছু route শুধুমাত্র নির্দিষ্ট user role এর জন্য access করা যায় (e.g. Admin panel)।

**✅ ৩. আপনি RxJS এর মাধ্যমে asynchronous ডেটা কীভাবে manage করেন?**

**উত্তর:**  
আমি Angular service এর মধ্যে HttpClient এর সঙ্গে RxJS operator গুলো ব্যবহার করে API call করি।

**উদাহরণ:**

this.userService.getUsers()

.pipe(

debounceTime(300),

switchMap(() => this.http.get('/api/users')),

catchError(err => of([]))

)

.subscribe(users => this.userList = users);

**ব্যাখ্যা:**

* **debounceTime**: user input stop করার পর delay করে।
* **switchMap**: আগের request ক্যান্সেল করে নতুন one ব্যবহার করে।
* **catchError**: error হলে fallback ডেটা দেয়।

এছাড়া, আমি Subject, BehaviorSubject, ReplaySubject ইত্যাদি ব্যবহার করে data sharing এবং state management করি।

**✅ ৪. আপনি Angular এ State Management কীভাবে করেন (NgRx ব্যতীত)?**

**উত্তর:**  
NgRx ছাড়াও আমি নিচের উপায়ে state manage করি:

* **Service with BehaviorSubject**:  
  একটি সার্ভিস তৈরি করি যেটা কম্পোনেন্টে subscribe করে। এই সার্ভিস অ্যাপের বিভিন্ন স্থানে ব্যবহৃত হয়।

@Injectable({ providedIn: 'root' })

export class AuthService {

private \_user$ = new BehaviorSubject<User | null>(null);

user$ = this.\_user$.asObservable();

setUser(user: User) {

this.\_user$.next(user);

}

}

* **LocalStorage / SessionStorage**:
  + Long-term persistence এর জন্য ব্যবহার করি।
* **Component level state**:
  + ছোট স্কোপে, কম্পোনেন্টের ভিতরে লোকাল ভ্যারিয়েবল ব্যবহার করে।

**✅ ৫. আপনি কীভাবে Code Maintainability ও Scalability নিশ্চিত করেন?**

**উত্তর:**

* **SOLID principles** অনুসরণ করি।
* DRY (Don’t Repeat Yourself) কোডিং করি।
* Service ও Utility গুলো reusable করি।
* Component গুলো ছোট ও Testable রাখি।
* Reusable form control, modal, button ইত্যাদি shared component এ রাখি।
* forRoot() ও forChild() প্যাটার্ন Module এর জন্য অনুসরণ করি।
* Core module শুধু একবার AppModule এ import করি।

**✅ ৬. আপনি Angular এ Testing কীভাবে করেন?**

**উত্তর:**

**Unit Test:**

* **Karma + Jasmine** ব্যবহার করি।
* Component, Service, Pipe গুলোর behavior test করি।

it('should fetch users from API', fakeAsync(() => {

const mockUsers = [{ name: 'John' }];

userService.getUsers().subscribe(data => expect(data).toEqual(mockUsers));

tick();

}));

**Integration Test:**

* Component এবং DOM behavior একসাথে test করি।
* TestBed দিয়ে configure করি।

**End-to-End Test:**

* Cypress অথবা Playwright ব্যবহার করি।
* ব্যবহারকারী কীভাবে অ্যাপ ব্যবহার করছে তা simulate করে দেখি।

**✅ ৭. আপনি একটি Angular অ্যাপ এর version upgrade করলে কী কী বিষয় খেয়াল রাখেন?**

**উত্তর:**

* Angular Official [Update Guide](https://update.angular.io/) অনুসরণ করি।
* ng update @angular/cli @angular/core দিয়ে শুরু করি।
* সব dependency এর compatibility খতিয়ে দেখি।
* Deprecated API গুলো খুঁজে দেখি ও পরিবর্তন করি।
* সব জায়গায় test চালাই: Unit, Integration, Manual QA।
* GitHub বা CI/CD pipeline এ Auto Build + Test চালিয়ে রেজাল্ট দেখি।

**✅ ৮. আপনি junior developer দের Angular শেখাতে কীভাবে সহায়তা করেন?**

**উত্তর:**

* প্রতিটি টপিককে ছোট ছোট অংশে ভাগ করি (Components, Services, Routing, RxJS ইত্যাদি)।
* Code Review এর সময় ভুল ধরিয়ে দিয়ে তার ব্যাখ্যা দিই।
* Pair Programming এ সহায়তা করি।
* একটি GitHub repo তৈরি করি উদাহরণসহ।
* Live walkthrough দিয়ে UI থেকে API পর্যন্ত কাজ দেখাই।

**✅ ৯. আপনি কীভাবে বড় টিমের Angular কোড কনভেনশন মেইনটেন করেন?**

**উত্তর:**

* Code Style Guide নির্ধারণ করি (Google Angular Style Guide)।
* Prettier, ESLint ব্যবহার করি।
* Husky দিয়ে pre-commit hook তৈরি করি।
* PR (Pull Request) টেমপ্লেট তৈরি করি।
* Code Review checklist তৈরি করি (naming, module split, logic separation ইত্যাদি)।

**✅ ১০. আপনি কীভাবে Angular অ্যাপের Security নিশ্চিত করেন?**

**উত্তর:**

* Input validation client এবং server দু’দিকেই করি।
* DOM sanitize করি যাতে XSS না হয়।
* Route Guards দিয়ে role-based access control করি।
* HttpInterceptor দিয়ে JWT token attach করি।
* Angular built-in Security APIs ব্যবহার করি (e.g., DomSanitizer, innerHTML এর পরিবর্তে [innerHtml] binding)।
* API call এ HTTPS enforced করি এবং CORS configure করি।