Admissions & Support:
support@mockattempt.com
Phone / WhatsApp: +91 98765 43210
Course Code: ANGL • **Updated:** 01 Nov 2026
https://mockattempt.com/courses/angular-training

ANGL • ENTERPRISE FRONTEND & TYPESCRIPT SPA **BEGINNER TO SENIOR ANGULAR ARCHITECT**

100% LIVE INDUSTRY EXPERT COHORT + INTERNSHIP LETTER

Angular Training

Master Enterprise Angular from Ivy AOT Compilation, CLI Workspaces, Standalone Bootstrapping (`bootstrapApplication`) & Dependency Injection (`inject()`) to Structural Directives (`*ngIf`/`@if`, `*ngFor`/`@for` track), Signals, Data Binding, Typed Reactive Forms & Full-Stack Spring Boot JWT CRUD — with TypeScript programs, 10×50 MCQ Mock Tests per topic, and Spoken Voice + Gemini AI Interviews at ₹3,999.

Duration: 15–17 Weeks • Live + Hands-On Projects (208 Hrs) • **Modules:** 16 • **Live Topics:** 42 • **Labs & Tasks:** 84 • **Mocks / AI Vivas:** 244 / 244 • **Projects:** 16 Module + 1 Live Client Prod

Fixed Non-Negotiable Tuition Fee: Base Fee **₹3,999** + 18% GST (₹720) = **Total Incl. GST: ₹4,719**

Live Batch Tracks: Weekday Live Online Batch (Mon – Fri) (Monday to Friday (5 Live Interactive Classes / Week) • 1.5 Hours / Day (60 Mins Live Architecture & Coding + 30 Mins Hands-On Lab) • 17–19 Weeks Fast-Track) | Weekend Intensive Live Online Batch (Sat – Sun) (Saturday & Sunday (Designed for Working Professionals & College Students) • 3.0 Hours / Day (2 Hrs Deep-Dive Live Coding + 1 Hr Guided Project Build) • 19–21 Weeks Weekend Track)

Core Technologies & Engineering Stack: Angular (Ivy AOT & Standalone APIs) • Fine-Grained Angular Signals • TypeScript 5 Strict Mode • Built-in Control Flow & Directives • RxJS Observables & Operators • Strictly Typed Reactive Forms • HttpClient, Interceptors & Route Guards

1. PROGRAM MASTERY OUTCOMES & ELIGIBILITY

End-to-End Engineering Outcomes:

1. Architect modular Angular SPAs using Standalone Components, hierarchical Dependency Injection (`inject()`), Ivy AOT tree-shaking, and lazy-loaded routes (`loadComponent`)
2. Eliminate unnecessary DOM re-renders using `ChangeDetectionStrategy.OnPush`, Angular Signals (`signal`, `computed`, `model`), and `trackBy` / `@for (...; track item.id)`
3. Build strongly typed `FormGroup` Reactive Forms, custom validators, and full-stack CRUD + JWT `HttpInterceptorFn` authentication pipelines with Spring Boot REST APIs
4. Clear 200 Timed Mock Exams (10,000 Angular MCQs) and 200 Spoken Voice + Gemini AI Frontend & Full-Stack Angular Interviews
5. Standalone component architecture (NgModule-free), fine-grained Angular Signals, and `OnPush` change detection.
6. Reactive Forms with typed form controls, cross-field async validators, and reusable dynamic form builders.
7. Clean RxJS reactive data flows (`switchMap`, `catchError`, `takeUntilDestroyed`), functional HTTP interceptors, and guards.
8. Route-level lazy loading, `@defer` viewport blocks, and strict domain feature folder boundaries.

Eligibility & Prerequisites: Basic HTML5, CSS3, and modern JavaScript (ES6+ arrow functions, classes, modules) — teaches TypeScript & Angular architecture from scratch • Any laptop/PC with Node.js LTS, npm, `@angular/cli` (`ng`), and VS Code with Angular Language Service • No prior Angular industry experience required — Module 1 starts from environment setup, core mental models, and plain-English fundamentals. • Basic computer & internet familiarity on a Windows, macOS, or Linux laptop (8 GB+ RAM recommended for smooth live coding labs). • Ideal for B.Tech / BCA / MCA / B.Sc IT students, non-CS beginners, career switchers, and working developers upgrading to 2026 production standards. • Commitment to attend live Weekday or Weekend classes, complete daily hands-on lab checklists, and practice topic-wise Mock Tests & AI Interviews.

2. DETAILED MODULE & TOPIC-WISE SYLLABUS (16 MODULES • 42 LIVE TOPICS)

MODULE 1 OF 16 • 11 LIVE + LAB HOURS • 2 TOPICS • 1 CAPSTONE PROJECT (100 MARKS)

Module 1: Angular Platform Foundations, SPA Evolution & Opinionated Framework Design

Understand Angular’s evolution from AngularJS (1.x) to modern TypeScript-based Angular, Single-Page Application routing, and batteries-included enterprise conventions.

Objectives: ✓ Contrast digest-cycle AngularJS 1.x against modern component-tree TypeScript Angular (v2 through modern Ivy/Signals) • ✓ Explain how Single-Page Applications update DOM views dynamically without full-page browser reloads • ✓ Apply reactive state derivation using Angular Signals (`signal`, `computed`) and strict TypeScript interfaces

1.1 Angular Platform Overview — TypeScript SPA Evolution, Semantic Versioning & Batteries-Included Ecosystem [50 mins • Assignment (100M) • Hands-On Lab (50M) • 10 Mock Sets • 10 AI Vivas]

Purpose: To understand and apply Angular Platform Overview — TypeScript SPA Evolution, Semantic Versioning & Batteries-Included Ecosystem in real Angular systems — specifically: H...

1.2 Angular Introduction — Component Trees, Declarative Templates & Zone vs Zoneless Reactivity [50 mins • Assignment (100M) • Hands-On Lab (50M) • 10 Mock Sets • 10 AI Vivas]

Purpose: To understand and apply Angular Introduction — Component Trees, Declarative Templates & Zone vs Zoneless Reactivity in real Angular systems — specifically: Structuring en...

Module 1 Hands-On Capstone Project (100 Marks): Reactive Course Catalog Summary Engine with Angular Signals — Build an Angular state model using `signal()` and `computed()` that filters published courses by track and derives live enrollment totals without manual imperative recalculation.

MODULE 2 OF 16 • 11 LIVE + LAB HOURS • 2 TOPICS • 1 CAPSTONE PROJECT (100 MARKS)

Module 2: Ivy Compiler Internals, Lazy Loading & Angular CLI Workspace Tooling

Explore Ivy tree-shakable locality compilation, dynamic ES `import()` route lazy loading, and Angular CLI (`angular.json`, budgets, environments, esbuild/Vite).

Objectives: ✓ Understand how the Ivy rendering engine achieves locality compilation, smaller bundle sizes, and strict template type-checking • ✓ Split initial main bundles using dynamic ES `import()` with `loadComponent` and `loadChildren` • ✓ Configure Angular CLI workspaces (`ng new`, `ng generate`, `angular.json` bundle budgets, and `strictTemplates`)

2.1 Core Angular Compiler & Runtime Features — Ivy Tree-Shaking, Dynamic Lazy Imports & Web Workers [50 mins • Assignment (100M) • Hands-On Lab (50M) • 10 Mock Sets • 10 AI Vivas]

Purpose: To understand and apply Core Angular Compiler & Runtime Features — Ivy Tree-Shaking, Dynamic Lazy Imports & Web Workers in real Angular systems — specifically: How Angula...

2.2 Angular CLI & Workspace Architecture — Node LTS, `@angular/cli`, `angular.json` & `tsconfig` [50 mins • Assignment (100M) • Hands-On Lab (50M) • 10 Mock Sets • 10 AI Vivas]

Purpose: To understand and apply Angular CLI & Workspace Architecture — Node LTS, `@angular/cli`, `angular.json` & `tsconfig` in real Angular systems — specifically: Setting up re...

Module 2 Hands-On Capstone Project (100 Marks): Lazy-Loaded Enterprise Route Map & CLI Bundle Budget Guard — Configure a strongly typed Angular `Routes` table using dynamic `loadComponent` lazy chunks and enforce `angular.json` initial bundle size budgets + `strictTemplates`.

MODULE 3 OF 16 • 11 LIVE + LAB HOURS • 2 TOPICS • 1 CAPSTONE PROJECT (100 MARKS)

Module 3: First Angular App Anatomy & Application Bootstrapping Pipeline

Trace every step from `index.html` (`<app-root>`) and `main.ts` (`bootstrapApplication` / `AppConfig`) to root component mounting and lifecycle initialization.

Objectives: ✓ Build clean Standalone root components with selector, imports, templateUrl/template, and scoped styles • ✓ Trace the exact startup sequence from angular.json -> main.ts -> bootstrapApplication(AppComponent, appConfig) -> <app-root> • ✓ Order Angular component lifecycle hooks accurately (constructor -> ngOnChanges -> ngOnInit -> ngOnDestroy / DestroyRef)

3.1 First Angular Application — `@Component` Metadata, Selectors, Standalone `imports` & Lifecycle Hooks [50 mins • Assignment (100M) • Hands-On Lab (50M) • 10 Mock Sets • 10 AI Vivas]

Purpose: To understand and apply First Angular Application — `@Component` Metadata, Selectors, Standalone `imports` & Lifecycle Hooks in real Angular systems — specifically: Anato...

3.2 How an Angular App Loads & Starts — `index.html`, `main.ts`, `bootstrapApplication` & `appConfig` [50 mins • Assignment (100M) • Hands-On Lab (50M) • 10 Mock Sets • 10 AI Vivas]

Purpose: To understand and apply How an Angular App Loads & Starts — `index.html`, `main.ts`, `bootstrapApplication` & `appConfig` in real Angular systems — specifically: Step-by-...

Module 3 Hands-On Capstone Project (100 Marks): Standalone Shell Bootstrapper with `appConfig` Providers & `DestroyRef` Cleanup — Create an Angular startup pipeline wiring `provideRouter(routes)` and `provideHttpClient(withFetch())` in `appConfig` and verifying deterministic `DestroyRef` subscription cleanup.

MODULE 4 OF 16 • 12 LIVE + LAB HOURS • 2 TOPICS • 1 CAPSTONE PROJECT (100 MARKS)

Module 4: Hierarchical Dependency Injection Architecture & Custom Directives

Master Angular's building blocks (`@Injectable`, `providedIn: "root"`, `inject()`, Services) and distinguish Component, Attribute, and Structural Directives.

Objectives: ✓ Design tree-shakable singleton services with `@Injectable({ providedIn: "root" })` and functional `inject()` • ✓ Understand injector resolution scopes (`root` EnvironmentInjector vs component-level `providers: [...]` instances) • ✓ Build safe custom Attribute Directives using `@Directive`, `ElementRef`, `@HostListener`, and `@HostBinding` (never direct unsafe DOM hacks)

4.1 Angular Architecture — Components, Templates, `@Injectable` Services & Hierarchical DI (`inject()`) [50 mins • Assignment (100M) • Hands-On Lab (50M) • 10 Mock Sets • 10 AI Vivas]

Purpose: To understand and apply Angular Architecture — Components, Templates, `@Injectable` Services & Hierarchical DI (`inject()`) in real Angular systems — specifically: How An...

4.2 Angular Directives — Component vs Structural vs Attribute Directives, `@HostBinding` & `@HostListener` [50 mins • Assignment (100M) • Hands-On Lab (50M) • 10 Mock Sets • 10 AI Vivas]

Purpose: To understand and apply Angular Directives — Component vs Structural vs Attribute Directives, `@HostBinding` & `@HostListener` in real Angular systems — specifically: Com...

Module 4 Hands-On Capstone Project (100 Marks): Hierarchical Theme Service & Role-Aware Attribute Directive Suite — Build a `providedIn: "root"` state service consumed via `inject()` alongside a reusable Attribute Directive (`[appHighlightRole]`) using `@HostBinding` and `@HostListener`.

MODULE 5 OF 16 • 12 LIVE + LAB HOURS • 2 TOPICS • 1 CAPSTONE PROJECT (100 MARKS)

Module 5: Conditional & List Rendering — `*ngIf` / `@if` and `*ngFor` / `@for` (`track`)

Control DOM creation/destruction with `*ngIf` (and modern `@if` / `@else`) and render 10,000-row lists cleanly with `*ngFor` `trackBy` and `@for (...; track item.id)`.

Objectives: ✓ Distinguish true DOM creation/destruction (`*ngIf` / `@if`) from CSS `[hidden]` toggling and handle async observable unwrapping cleanly • ✓ Master `<ng-template>` microsyntax desugaring and zero-wrapper `<ng-container>` grouping • ✓ Eliminate full-list DOM thrashing on array updates using `trackBy` functions and mandatory `@for (...; track item.id)`

5.1 Angular `*ngIf` & `@if` Control Flow — DOM Lifecycle Destruction, `else` Templates & `async` Aliasing [50 mins • Assignment (100M) • Hands-On Lab (50M) • 10 Mock Sets • 10 AI Vivas]

Purpose: To understand and apply Angular `*ngIf` & `@if` Control Flow — DOM Lifecycle Destruction, `else` Templates & `async` Aliasing in real Angular systems — specifically: Phys...

5.2 Angular `*ngFor` & `@for` — `trackBy` / Mandatory `track` Diffing, `@empty` & Context Variables [50 mins • Assignment (100M) • Hands-On Lab (50M) • 10 Mock Sets • 10 AI Vivas]

Purpose: To understand and apply Angular `*ngFor` & `@for` — `trackBy` / Mandatory `track` Diffing, `@empty` & Context Variables in real Angular systems — specifically: Rendering...

Module 5 Hands-On Capstone Project (100 Marks): High-Throughput Exam Leaderboard with `@if` Skeleton States & `track` Identity Diffing — Build an Angular leaderboard template & view-model handling loading/empty/error states via `@if` / `@else` and diffing live score refreshes via stable primary-key `track` identities.

MODULE 6 OF 16 • 12 LIVE + LAB HOURS • 2 TOPICS • 1 CAPSTONE PROJECT (100 MARKS)

Module 6: Multi-Branch Switching (`ngSwitch` / `@switch`) & Angular Data-Binding Architecture

Render clean multi-branch state machines via `[ngSwitch]` / `@switch` (`===`) and master all 4 directions of Angular's data-binding mental model.

Objectives: ✓ Replace deeply nested `*ngIf` ladders with exhaustive `@switch` / `[ngSwitch]` views using strict TypeScript union types • ✓ Map all 4 Angular binding syntaxes (`{{ }}`, `[prop]`, `(event)`, `[(ngModel)]` / `[(value)]`) by data-flow direction • ✓ Understand unidirectional top-down change detection and prevent `NG0100: ExpressionChangedAfterItHasBeenCheckedError`

6.1 Angular `[ngSwitch]` & `@switch` — Multi-Branch Strict Equality (`===`), `@case` & `@default` [50 mins • Assignment (100M) • Hands-On Lab (50M) • 10 Mock Sets • 10 AI Vivas]

Purpose: To understand and apply Angular `[ngSwitch]` & `@switch` — Multi-Branch Strict Equality (`===`), `@case` & `@default` in real Angular systems — specifically: Declarative...

6.2 Data Binding in Angular — The 4 Binding Modes, Unidirectional Flow & `NG0100` Prevention [50 mins • Assignment (100M) • Hands-On Lab (50M) • 10 Mock Sets • 10 AI Vivas]

Purpose: To understand and apply Data Binding in Angular — The 4 Binding Modes, Unidirectional Flow & `NG0100` Prevention in real Angular systems — specifically: Unifying Interpol...

Module 6 Hands-On Capstone Project (100 Marks): Multi-Stage AI Interview Status View with Exhaustive `@switch` & Unidirectional Flow — Build an Angular status view-model switching cleanly across `"idle" | "listening" | "verifying" | "scored"` union states while keeping parent-to-child data flow strictly unidirectional.

MODULE 7 OF 16 • 12 LIVE + LAB HOURS • 2 TOPICS • 1 CAPSTONE PROJECT (100 MARKS)

Module 7: DOM Property Binding (`[ ]`), Accessibility Attributes & XSS-Safe Interpolation (`{{ }}`)

Master boolean/object DOM Property Binding (`[disabled]`, `[attr.aria-*]`, `[class.active]`) vs string Interpolation (`{{ }}`) and Angular's built-in XSS sanitization engine.

Objectives: ✓ Understand why boolean DOM properties MUST use `[disabled]="isInvalid"` rather than string interpolation `disabled="{{isInvalid}}"` • ✓ Bind ARIA/SVG/table attributes cleanly using `[attr.aria-label]` and `[attr.colspan]` when no matching DOM property exists • ✓ Leverage Angular's automatic HTML context sanitization in `{{ }}` and `[innerHTML]` without misusing `DomSanitizer.bypassSecurityTrustHtml`

7.1 Property Binding `[ ]` — DOM Properties vs HTML Attributes, Boolean `[disabled]` & `[attr.*]` [50 mins • Assignment (100M) • Hands-On Lab (50M) • 10 Mock Sets • 10 AI Vivas]

Purpose: To understand and apply Property Binding `[ ]` — DOM Properties vs HTML Attributes, Boolean `[disabled]` & `[attr.*]` in real Angular systems — specifically: Binding live...

7.2 String Interpolation `{{ }}` — Pure Template Expressions, Null-Safety (`?.`, `??`) & XSS Auto-Escaping [50 mins • Assignment (100M) • Hands-On Lab (50M) • 10 Mock Sets • 10 AI Vivas]

Purpose: To understand and apply String Interpolation `{{ }}` — Pure Template Expressions, Null-Safety (`?.`, `??`) & XSS Auto-Escaping in real Angular systems — specifically: Ren...

Module 7 Hands-On Capstone Project (100 Marks): Accessible Exam Question Card with ARIA Attribute Binding & XSS Sanitizer Guard — Implement an accessible question card binding boolean `[disabled]`, `[attr.aria-expanded]`, `[class.selected]`, and safe text interpolation while blocking `<script>`/`onerror` XSS payloads.

MODULE 8 OF 16 • 11 LIVE + LAB HOURS • 2 TOPICS • 1 CAPSTONE PROJECT (100 MARKS)

Module 8: DOM Event Binding (`( )`), Key Filters & Two-Way Binding (`[(ngModel)]` / `model()`)

Capture DOM & custom child events with typed `$event`, key pseudo-events (`(keyup.enter)`), `output()`, and build custom two-way bindings using `[( )]` / Signal `model()`.

Objectives: ✓ Handle typed DOM events (`MouseEvent`, `KeyboardEvent`), key-filtered pseudo-events (`(keydown.escape)`), and child `output<T>()` emitters • ✓ Understand how banana-in-a-box `[(x)]="val"` desugars into `[x]="val"` + `(xChange)="val = $event"` • ✓ Build modern custom two-way bindings in 1 line using Angular's `model<T>()` writable signal input/output primitive

8.1 Event Binding `( )` — Typed `$event`, Key Pseudo-Events `(keyup.enter)` & Child `output()` [50 mins • Assignment (100M) • Hands-On Lab (50M) • 10 Mock Sets • 10 AI Vivas]

Purpose: To understand and apply Event Binding `( )` — Typed `$event`, Key Pseudo-Events `(keyup.enter)` & Child `output()` in real Angular systems — specifically: Listening to na...

8.2 Two-Way Data Binding `[()]` — Banana-in-a-Box Desugaring, `FormsModule` `ngModel` & Signal `model()` [50 mins • Assignment (100M) • Hands-On Lab (50M) • 10 Mock Sets • 10 AI Vivas]

Purpose: To understand and apply Two-Way Data Binding `[()]` — Banana-in-a-Box Desugaring, `FormsModule` `ngModel` & Signal `model()` in real Angular systems — specifically: Under...

Module 8 Hands-On Capstone Project (100 Marks): Keyboard-Accessible Rating & Search Bar with `output()` and `model()` Two-Way Sync — Create a reusable `<app-score-slider [(score)]="currentScore">` component using Angular's `x` / `xChange` (`model()`) contract plus `(keydown.arrowUp)` keyboard accessibility.

MODULE 9 OF 16 • 12 LIVE + LAB HOURS • 2 TOPICS • 1 CAPSTONE PROJECT (100 MARKS)

Module 9: Strictly Typed Reactive Forms & Enterprise Architecture Trade-Offs (Angular vs React)

Build immutable, unit-testable `NonNullableFormBuilder` Reactive Forms with cross-field/async validators and evaluate Angular vs React architectural trade-offs objectively.

Objectives: ✓ Prefer strictly typed Reactive Forms (`FormBuilder.nonNullable`, `FormGroup`, `FormArray`, `Validators`) for complex enterprise workflows • ✓ Implement synchronous cross-field validators (password confirm) and debounced `AsyncValidatorFn` uniqueness checks • ✓ Compare Angular (opinionated DI + Ivy AOT + Signals + RxJS) and React (UI library + VDOM + JSX ecosystem) across team governance and SSR

9.1 Angular Forms — Template-Driven vs Strictly Typed Reactive Forms (`FormGroup`, `FormArray`, `Validators`) [50 mins • Assignment (100M) • Hands-On Lab (50M) • 10 Mock Sets • 10 AI Vivas]

Purpose: To understand and apply Angular Forms — Template-Driven vs Strictly Typed Reactive Forms (`FormGroup`, `FormArray`, `Validators`) in real Angular systems — specifically:...

9.2 Angular vs React — Opinionated Platform vs UI Library, Signals/Ivy vs Virtual DOM & DI [50 mins • Assignment (100M) • Hands-On Lab (50M) • 10 Mock Sets • 10 AI Vivas]

Purpose: To understand and apply Angular vs React — Opinionated Platform vs UI Library, Signals/Ivy vs Virtual DOM & DI in real Angular systems — specifically: Architectural compa...

Module 9 Hands-On Capstone Project (100 Marks): Strictly Typed Multi-Skill Enrollment `FormGroup` & Cross-Field Validator — Build a `NonNullableFormBuilder` checkout form model with email/phone validators, a dynamic `FormArray` of skill tags, and a cross-field discount-cap validator testable without DOM rendering...

MODULE 10 OF 16 • 12 LIVE + LAB HOURS • 2 TOPICS • 1 CAPSTONE PROJECT (100 MARKS)

Module 10: Full-Stack Angular + Spring Boot REST CRUD, JWT Interceptors & Route Guards

Connect Angular `HttpClient` services to Spring Boot REST APIs with RxJS `catchError`, cold Observable lifecycle discipline, functional `HttpInterceptorFn` JWT headers, and `CanActivateFn` guards.

Objectives: ✓ Architect typed `HttpClient` CRUD services (`GET`, `POST`, `PUT`, `DELETE`) backed by RxJS `Observable<T>` and `catchError` recovery • ✓ Attach `Authorization: Bearer <jwt>` headers centrally via immutable `req.clone()` inside a functional `HttpInterceptorFn` • ✓ Protect authenticated routes with `CanActivateFn` (`UrlTree` redirect) and implement clean logout state & token teardown

10.1 Angular + Spring Boot CRUD Integration — Typed `HttpClient`, Cold Observables, CORS & `catchError` [50 mins • Assignment (100M) • Hands-On Lab (50M) • 10 Mock Sets • 10 AI Vivas]

Purpose: To understand and apply Angular + Spring Boot CRUD Integration — Typed `HttpClient`, Cold Observables, CORS & `catchError` in real Angular systems — specifically: Wiring...

10.2 Angular + Spring Security Login & Logout — Functional `HttpInterceptorFn`, `req.clone()` & `CanActivateFn` [50 mins • Assignment (100M) • Hands-On Lab (50M) • 10 Mock Sets • 10 AI Vivas]

Purpose: To understand and apply Angular + Spring Security Login & Logout — Functional `HttpInterceptorFn`, `req.clone()` & `CanActivateFn` in real Angular systems — specifically:...

Module 10 Hands-On Capstone Project (100 Marks): Full-Stack Angular + Spring Boot Student Portal Client with JWT `HttpInterceptorFn` & `CanActivateFn` — Implement a typed Angular `StudentApiService` CRUD layer, a functional JWT `authInterceptor` that clones requests immutably for `/api/**` calls, and an `authGuard` returning a `/login` `UrlT...

MODULE 11 OF 16 • 16 LIVE + LAB HOURS • 4 TOPICS • 1 CAPSTONE PROJECT (100 MARKS)

Modern Angular Signals, Zoneless Change Detection & Reactive Stores

Production-grade Angular Training engineering module covering Modern Angular Signals, Zoneless Change Detection & Reactive Stores with hands-on labs and capstone build.

Objectives: ✓ signal(), computed(), linkedSignal() & Explicit effect() Boundaries — Purpose: Update exact DOM nodes surgically without scanning the entire component tree. • ✓ OnPush & Zoneless Change Detection Architecture — Purpose: Eliminate Zone.js overhead and achieve ultra-snappy Core Web Vitals INP scores. • ✓ NgRx SignalStore / Entity Management for Complex Domain Workflows — Purpose: Coordinate loading, optimistic updates, and normalized collections cleanly. • ✓ Bridging RxJS Streams & Signals (toSignal / toObservable) — Purpose: Combine powerful RxJS debounce/switchMap HTTP operators with ergonomic Signal templates.

11.1 signal(), computed(), linkedSignal() & Explicit effect() Boundaries [65 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Purpose: To understand and apply signal(), computed(), linkedSignal() & Explicit effect() Boundaries in real Angular systems — specifically: Update exact DOM nodes surgically with...

11.2 OnPush & Zoneless Change Detection Architecture [65 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Purpose: To understand and apply OnPush & Zoneless Change Detection Architecture in real Angular systems — specifically: Eliminate Zone.js overhead and achieve ultra-snappy Core W...

11.3 NgRx SignalStore / Entity Management for Complex Domain Workflows [65 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Purpose: To understand and apply NgRx SignalStore / Entity Management for Complex Domain Workflows in real Angular systems — specifically: Coordinate loading, optimistic updates,...

11.4 Bridging RxJS Streams & Signals (toSignal / toObservable) [65 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Purpose: To understand and apply Bridging RxJS Streams & Signals (toSignal / toObservable) in real Angular systems — specifically: Combine powerful RxJS debounce/switchMap HTTP op...

Module 11 Hands-On Capstone Project (100 Marks): Fine-Grained Signal-Driven Enterprise Operations Console — Design, implement, test, and document 'Fine-Grained Signal-Driven Enterprise Operations Console' applying every production concept from Module 11 (Modern Angular Signals, Zoneless Change Det...

MODULE 12 OF 16 • 16 LIVE + LAB HOURS • 4 TOPICS • 1 CAPSTONE PROJECT (100 MARKS)

Enterprise Typed Reactive Forms, Functional Guards & HTTP Interceptors

Production-grade Angular Training engineering module covering Enterprise Typed Reactive Forms, Functional Guards & HTTP Interceptors with hands-on labs and capstone build.

Objectives: ✓ Strictly Typed FormGroup/FormArray & Reusable ControlValueAccessor — Purpose: Build custom date-pickers, currency inputs, and dynamic rule forms with full type safety. • ✓ Async Cross-Field Validators & Accessible Validation Summaries — Purpose: Validate PAN/GST/email uniqueness live without spamming backend APIs. • ✓ Functional CanActivateFn / CanMatchFn Route Guards & Preloading — Purpose: Enforce RBAC permissions before lazy route bundles are even downloaded. • ✓ Functional HttpInterceptorFn for JWT Refresh Queuing & Retry Backoff — Purpose: Handle 401 token renewal transparently across concurrent inflight HTTP calls.

12.1 Strictly Typed FormGroup/FormArray & Reusable ControlValueAccessor [65 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Purpose: To understand and apply Strictly Typed FormGroup/FormArray & Reusable ControlValueAccessor in real Angular systems — specifically: Build custom date-pickers, currency inp...

12.2 Async Cross-Field Validators & Accessible Validation Summaries [65 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Purpose: To understand and apply Async Cross-Field Validators & Accessible Validation Summaries in real Angular systems — specifically: Validate PAN/GST/email uniqueness live with...

12.3 Functional CanActivateFn / CanMatchFn Route Guards & Preloading [65 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Purpose: To understand and apply Functional CanActivateFn / CanMatchFn Route Guards & Preloading in real Angular systems — specifically: Enforce RBAC permissions before lazy route...

12.4 Functional HttpInterceptorFn for JWT Refresh Queuing & Retry Backoff [65 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Purpose: To understand and apply Functional HttpInterceptorFn for JWT Refresh Queuing & Retry Backoff in real Angular systems — specifically: Handle 401 token renewal transparentl...

Module 12 Hands-On Capstone Project (100 Marks): Dynamic Multi-Step Banking KYC Portal in Angular — Design, implement, test, and document 'Dynamic Multi-Step Banking KYC Portal in Angular' applying every production concept from Module 12 (Enterprise Typed Reactive Forms, Functional Guards...

MODULE 13 OF 16 • 16 LIVE + LAB HOURS • 4 TOPICS • 1 CAPSTONE PROJECT (100 MARKS)

Deferrable Views (@defer), SSR Hydration, Virtual Scroll & Accessibility

Production-grade Angular Training engineering module covering Deferrable Views (@defer), SSR Hydration, Virtual Scroll & Accessibility with hands-on labs and capstone build.

Objectives: ✓ Template @defer Blocks (on viewport, on interaction, on idle) — Purpose: Lazy-load heavy charts and below-the-fold widgets declaratively with zero boilerplate. • ✓ Angular CDK Virtual Scroll, Overlays, FocusTrap & Drag-and-Drop — Purpose: Render 25,000-row enterprise tables and accessible Kanban/modals effortlessly. • ✓ Server-Side Rendering (Angular SSR), TransferState & Non-Destructive Hydration — Purpose: Deliver instant SEO-ready first paint without double-fetching initial API data. • ✓ NgOptimizedImage, Bundle Budget Enforcement & Source-Map Audits — Purpose: Fail CI automatically if an accidental library import bloats the main JS chunk.

13.1 Template @defer Blocks (on viewport, on interaction, on idle) [65 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Purpose: To understand and apply Template @defer Blocks (on viewport, on interaction, on idle) in real Angular systems — specifically: Lazy-load heavy charts and below-the-fold wi...

13.2 Angular CDK Virtual Scroll, Overlays, FocusTrap & Drag-and-Drop [65 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Purpose: To understand and apply Angular CDK Virtual Scroll, Overlays, FocusTrap & Drag-and-Drop in real Angular systems — specifically: Render 25,000-row enterprise tables and ac...

13.3 Server-Side Rendering (Angular SSR), TransferState & Non-Destructive Hydration [65 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Purpose: To understand and apply Server-Side Rendering (Angular SSR), TransferState & Non-Destructive Hydration in real Angular systems — specifically: Deliver instant SEO-ready f...

13.4 NgOptimizedImage, Bundle Budget Enforcement & Source-Map Audits [65 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Purpose: To understand and apply NgOptimizedImage, Bundle Budget Enforcement & Source-Map Audits in real Angular systems — specifically: Fail CI automatically if an accidental lib...

Module 13 Hands-On Capstone Project (100 Marks): High-Speed Accessible Angular Storefront & Data Grid — Design, implement, test, and document 'High-Speed Accessible Angular Storefront & Data Grid' applying every production concept from Module 13 (Deferrable Views (@defer), SSR Hydration, Virtu...

MODULE 14 OF 16 • 16 LIVE + LAB HOURS • 4 TOPICS • 1 CAPSTONE PROJECT (100 MARKS)

Nx Workspace Architecture, Component Harnesses, Playwright & CI/CD

Production-grade Angular Training engineering module covering Nx Workspace Architecture, Component Harnesses, Playwright & CI/CD with hands-on labs and capstone build.

Objectives: ✓ Feature-Sliced Standalone Folder Structure & Library Boundaries — Purpose: Keep large 40-screen Angular apps maintainable across multiple agile squads. • ✓ Unit Testing Signals, Services & HttpTestingController Specs — Purpose: Verify interceptors, guards, and stores fast with deterministic HTTP mocks. • ✓ CDK Test Harnesses & Playwright End-to-End User Journey Suites — Purpose: Write resilient UI tests decoupled from internal CSS class names. • ✓ Multi-Environment Docker Nginx Builds, CSP Headers & Live Deployment — Purpose: Ship cache-optimized Angular builds with strict Content-Security-Policy protection.

14.1 Feature-Sliced Standalone Folder Structure & Library Boundaries [65 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Purpose: To understand and apply Feature-Sliced Standalone Folder Structure & Library Boundaries in real Angular systems — specifically: Keep large 40-screen Angular apps maintain...

14.2 Unit Testing Signals, Services & HttpTestingController Specs [65 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Purpose: To understand and apply Unit Testing Signals, Services & HttpTestingController Specs in real Angular systems — specifically: Verify interceptors, guards, and stores fast...

14.3 CDK Test Harnesses & Playwright End-to-End User Journey Suites [65 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Purpose: To understand and apply CDK Test Harnesses & Playwright End-to-End User Journey Suites in real Angular systems — specifically: Write resilient UI tests decoupled from int...

14.4 Multi-Environment Docker Nginx Builds, CSP Headers & Live Deployment [65 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Purpose: To understand and apply Multi-Environment Docker Nginx Builds, CSP Headers & Live Deployment in real Angular systems — specifically: Ship cache-optimized Angular builds w...

Module 14 Hands-On Capstone Project (100 Marks): Modular Angular Monorepo Pipeline with Automated E2E Gate — Design, implement, test, and document 'Modular Angular Monorepo Pipeline with Automated E2E Gate' applying every production concept from Module 14 (Nx Workspace Architecture, Component Harne...

MODULE 15 OF 16 • 14 LIVE + LAB HOURS • 3 TOPICS • 1 CAPSTONE PROJECT (100 MARKS)

Angular Signals, Fine-Grained Reactivity & Enterprise State Architecture

Self-explanatory list-and-bullet production mastery module covering Angular Signals, Fine-Grained Reactivity & Enterprise State Architecture in Angular Training, paired with hands-on labs, capstone repository, Topic Mock Tests, and Spoken Gemini AI Viva.

Objectives: ✓ Angular Signals (signal, computed, effect), input()/output() & Zoneless Readiness — Purpose: Build predictable, ultra-fast reactive Angular components without manual ChangeDetectorRef juggling. • ✓ RxJS Interop (toSignal, toObservable), Cancellation & Functional HTTP Interceptors — Purpose: Combine debounced search streams, JWT refresh queues, and centralized retry/error handling. • ✓ Typed Reactive Forms, Cross-Field Validators & Reusable ControlValueAccessor Inputs — Purpose: Handle complex enterprise multi-step workflows with strict compile-time form type safety. • ✓ Build, verify, and document 'Signals + RxJS Interop Operations Console with Optimistic HTTP Interceptors' with clean Git commits and edge-case protection. • ✓ Clear 100% of Topic Timed Mock Tests and Spoken Voice + Google Gemini AI Technical Interviews.

15.1 Angular Signals (signal, computed, effect), input()/output() & Zoneless Readiness [65 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Purpose: To understand and apply Angular Signals (signal, computed, effect), input()/output() & Zoneless Readiness in real Angular systems — specifically: Build predictable, ultra...

15.2 RxJS Interop (toSignal, toObservable), Cancellation & Functional HTTP Interceptors [65 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Purpose: To understand and apply RxJS Interop (toSignal, toObservable), Cancellation & Functional HTTP Interceptors in real Angular systems — specifically: Combine debounced searc...

15.3 Typed Reactive Forms, Cross-Field Validators & Reusable ControlValueAccessor Inputs [65 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Purpose: To understand and apply Typed Reactive Forms, Cross-Field Validators & Reusable ControlValueAccessor Inputs in real Angular systems — specifically: Handle complex enterpr...

Module 15 Hands-On Capstone Project (100 Marks): Signals + RxJS Interop Operations Console with Optimistic HTTP Interceptors — Architect, implement, test, and document 'Signals + RxJS Interop Operations Console with Optimistic HTTP Interceptors' applying every bullet-wise engineering concept from Module 15 (Angular...

MODULE 16 OF 16 • 14 LIVE + LAB HOURS • 3 TOPICS • 1 CAPSTONE PROJECT (100 MARKS)

Deferrable Views (@defer), SSR Hydration, Testing & Angular Interviews

Self-explanatory list-and-bullet production mastery module covering Deferrable Views (@defer), SSR Hydration, Testing & Angular Interviews in Angular Training, paired with hands-on labs, capstone repository, Topic Mock Tests, and Spoken Gemini AI Viva.

Objectives: ✓ Template @defer Blocks, Route Lazy Loading, NgOptimizedImage & Bundle Budgets — Purpose: Slash initial bundle size and achieve top Core Web Vitals on large enterprise Angular portals. • ✓ Testing Standalone Components, Signals, Guards & HTTP Flows with TestBed / Harness — Purpose: Automate regression tests for user interactions, route guards, and API error states. • ✓ Angular DI Hierarchy, Change Detection Internals & Senior Frontend Interviews — Purpose: Explain OnPush vs Signals, memory-leak-free takeUntilDestroyed(), and monorepo architecture. • ✓ Build, verify, and document 'Bundle-Optimized Standalone Angular App with Deferrable Charts & Component Harness Tests' with clean Git commits and edge-case protection. • ✓ Clear 100% of Topic Timed Mock Tests and Spoken Voice + Google Gemini AI Technical Interviews.

16.1 Template @defer Blocks, Route Lazy Loading, NgOptimizedImage & Bundle Budgets [65 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Purpose: To understand and apply Template @defer Blocks, Route Lazy Loading, NgOptimizedImage & Bundle Budgets in real Angular systems — specifically: Slash initial bundle size an...

16.2 Testing Standalone Components, Signals, Guards & HTTP Flows with TestBed / Harness [65 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Purpose: To understand and apply Testing Standalone Components, Signals, Guards & HTTP Flows with TestBed / Harness in real Angular systems — specifically: Automate regression tes...

16.3 Angular DI Hierarchy, Change Detection Internals & Senior Frontend Interviews [65 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Purpose: To understand and apply Angular DI Hierarchy, Change Detection Internals & Senior Frontend Interviews in real Angular systems — specifically: Explain OnPush vs Signals, m...

Module 16 Hands-On Capstone Project (100 Marks): Bundle-Optimized Standalone Angular App with Deferrable Charts & Component Harness Tests — Architect, implement, test, and document 'Bundle-Optimized Standalone Angular App with Deferrable Charts & Component Harness Tests' applying every bullet-wise engineering concept from Module...

3. LIVE CLIENT PRODUCTION PROJECT, GLOBAL CERTIFICATIONS & CAREER SCOPE

End-of-Course Live Client Production Project + Official Internship Confirmation Letter

Project: FinDesk Enterprise Wealth Management & Role-Gated Operations SPA • **Confirmed Role:** Angular 18 & TypeScript Enterprise Frontend Intern

Domain & Production Infra: WealthTech & Corporate Treasury SaaS Client (Angular 18 Standalone Components + Signals + RxJS + Reactive Forms + Cloud CDN HTTPS)

- Built modular Angular 18 feature domains using Standalone Components, Signals, and OnPush change detection.
- Engineered complex multi-step Reactive Forms with async custom validators, HTTP interceptors, and route guards.
- Managed real-time data streams cleanly using RxJS operators (switchMap, combineLatest, shareReplay) without memory leaks.
- Shipped lazy-loaded, AOT-compiled production bundles passing strict TypeScript and unit test gates.

Aligned Global Vendor Certifications: **ANG-L2** (Certified Mid-to-Senior Angular Enterprise Developer) • **META-FE** (Meta Certified Front-End & React Developer Credential)

Target Job Roles: Angular Frontend Engineer, Enterprise UI Software Engineer, TypeScript & Angular SPA Developer, Frontend Application Specialist

Realistic Annual CTC Scope: **Fresher / Entry-Level (0–1 Yr):** ₹5.0 LPA – ₹9.2 LPA **Mid-Level Angular Product Engineer (1–3 Yrs):** ₹9.5 LPA – ₹19.5 LPA **Senior Enterprise Frontend / Full-Stack Architect (3–6+ Yrs):** ₹20.0 LPA – ₹38.0+ LPA

MockAttempt Academy Pvt. Ltd. • Email: support@mockattempt.com • Phone/WhatsApp: +91 98765 43210 • Web: https://mockattempt.com/courses/angular-training

Try 2 Free Topic-Wise 50-MCQ Mock Tests & Spoken Gemini AI Interviews daily on every topic above before enrolling.