# Angular Fundamentals You Need to Learn

## Angular Architecture Basics

* What is Angular?
* Components, Templates, Modules, Services — How they work together
* Angular CLI (Command Line Interface) basics: creating, building, serving apps

## Components

* Creating components
* Component decorator and metadata (@Component)
* Component class and template (HTML) binding
* Component styles (CSS/SCSS)
* Lifecycle hooks (e.g., ngOnInit, ngOnDestroy)

## Templates & Data Binding

* Interpolation ({{ }})
* Property binding ([property]="value")
* Event binding ((event)="handler()")
* Two-way binding ([(ngModel)]="property") — requires FormsModule
* Template expressions and directives

## Directives

* Structural directives:
* \* ngIf (conditional rendering)
* \* ngFor (list rendering)
* \* ngSwitch
* Attribute directives:
* \* ngClass
* \* ngStyle
* Creating custom directives (optional, advanced)

## Modules

* Understanding NgModules (@NgModule)
* Root module (AppModule) and feature modules
* Importing/exporting modules
* Lazy loading modules (advanced)

## Services and Dependency Injection

* What are services?
* Creating and providing services (@Injectable)
* Injecting services into components
* Singleton services and scopes

## Routing

* Configuring routes (RouterModule)
* Route parameters
* Navigation via code and router links
* Child routes and nested routing
* Route guards (CanActivate, CanDeactivate) (intermediate)

## Forms

* Template-driven forms (simple)
* Reactive forms (model-driven) (more powerful)
* Form validation (built-in and custom)
* Form controls and groups

## HTTP Client

* Making HTTP requests with HttpClientModule
* GET, POST, PUT, DELETE requests
* Handling responses and errors
* Observables basics with RxJS
* Using async pipe for subscription management

## Pipes

* Using built-in pipes (e.g., date, currency, uppercase)
* Creating custom pipes (optional, advanced)

## Angular CLI and Project Structure

* Generating components, services, modules with CLI
* Understanding Angular workspace and configuration
* Build and deploy basics

## Basic State Management (Optional for Beginners)

* Using component properties to manage state
* Sharing data between components via services
* Introduction to RxJS subjects (optional)

## Debugging and Testing (Intro)

* Debugging Angular apps with browser dev tools
* Basic unit testing with Jasmine/Karma (optional)

## Summary: Start Learning These in This Order

* 1. Angular architecture & CLI
* 2. Components & templates
* 3. Data binding & directives
* 4. Modules & services
* 5. Routing basics
* 6. Forms (template-driven then reactive)
* 7. HTTP client & observables
* 8. Pipes
* 9. Bonus: Lazy loading, guards, custom directives/pipes, testing