Student CRM — Module 3 Coverage

Topic: Adding Interactivity to Your Applications

Overview

This document maps the Student CRM application's code to common learning objectives for "Module 3: Adding Interactivity to Your Applications." Based on the repository code, the app demonstrates core interactivity patterns: event handling, data binding, template-driven forms, component communication, routing, services/DI, and lifecycle hooks.

Note: If your course PDF defines Module 3 differently, please share it (or confirm its location) so we can align this mapping precisely to its sections and terminology.

Typical Module 3 Topics

- Event binding and handlers
- Property, attribute, and class/style binding
- Two-way binding with `[(ngModel)]`
- Structural directives: `*ngIf`, `*ngFor`
- Template-driven forms and basic validation
- Component communication with `@Input()`/`@Output()`
- Services and dependency injection for stateful logic
- Routing and navigation
- Lifecycle hooks: `OnInit`, `OnDestroy`
- (Optional) Signals and hydration/event replay in modern Angular

Where the App Covers These

Routing and Navigation

- Navigation link and outlet: `student-crm/src/app/app.html:5`, `student-crm/src/app/app.html:12`
- Route configuration: `student-crm/src/app/app-routing-module.ts:5`, `student-crm/src/app/app-routing-module.ts:11`

Template-Driven Forms and Validation

- Form submit handler and two-way binding: `student-crm/src/app/students/add-student/add-student.html:1`, `:2`, `:3`, `:10`, `:12`
- Component logic and event emission: `student-crm/src/app/students/add-student/add-student.ts:11`, `:17`, `:20`
- Forms module import: `student-crm/src/app/students/students-module.ts:4` (and included in `imports`)

Two-Way, Property, and Class Binding

- Two-way binding and disabling submit until valid: `student-crm/src/app/students/add-student/add-student.html:2`, `:12`
- Class binding that reflects state: `student-crm/src/app/students/student-card/student-card.html:4`
- Property bindings to pass data: `student-crm/src/app/students/student-list/student-list.html:6`

Structural Directives

- Conditional and list rendering: `student-crm/src/app/students/student-list/student-list.html:1`, `:3`, `:4`

Component Communication and Events

- Child emits toggle; parent handles: `student-crm/src/app/students/student-card/student-card.ts:12`, `student-crm/src/app/students/student-card/student-card.html:10`
- Event bubbling via list: `student-crm/src/app/students/student-list/student-list.html:7`
- Parent handlers wire up app behavior: `student-crm/src/app/home/home.html:6`, `student-crm/src/app/home/home.html:10`, `student-crm/src/app/home/home.ts:21`, `student-crm/src/app/home/home.ts:26`

Services, State, and DI

- Service with state and methods: `student-crm/src/app/core/student.ts:12` (service), `:18` (`list()`), `:20` (`add()`), `:27` (`toggleActive()`)
- Injecting and using the service in a component: `student-crm/src/app/home/home.ts:14`, `:18`, `:22`, `:27`

Lifecycle Hooks

- Component initialization and teardown: `student-crm/src/app/students/student-card/student-card.ts:14`, `:18`
- Parent component lifecycle: `student-crm/src/app/home/home.ts:11`, `:16`, `:31`

Signals and Hydration (Modern Angular)

- Signal example (title): `student-crm/src/app/app.ts:9`
- Client hydration with event replay: `student-crm/src/app/app-module.ts:18`

User Flow Walkthrough

1. Home loads and calls `ngOnInit()`, pulling students from the service and showing a count.
2. User fills the add-student form and submits (`ngSubmit`), which emits a `create` event to the parent; the parent calls `add()` on the service, then re-reads and displays the list.
3. User clicks “Toggle Active” on a card; the child emits an ID, the parent calls `toggleActive()` on the service; UI updates class bindings and labels accordingly.

Gaps and Enhancement Ideas

- **Validation UX:** Add error messages and stricter validation on the form.
- **Async data flow:** If Module 3 covers HTTP/RxJS, adapt the service to use `HttpClient` and observables.
- **Signals in UI:** Bind the app title signal in the template to demonstrate reactive reads.
- **Routing depth:** Add routes for details/edit to cover route params and navigation extras.

File Map (Quick Reference)

- `student-crm/src/app/home/home.ts`, `student-crm/src/app/home/home.html`
- `student-crm/src/app/students/add-student/add-student.ts`, …/`add-student.html`

- `student-crm/src/app/students/student-list/student-list.ts`, `…/student-list.html`
- `student-crm/src/app/students/student-card/student-card.ts`, `…/student-card.html`
- `student-crm/src/app/core/student.ts`, `student-crm/src/app/core/logger.ts`
- `student-crm/src/app/app.html`, `student-crm/src/app/app-routing-module.ts`, `student-crm/src/app/app-module.ts`