Student CRM — Rebuild Steps

This guide walks you through recreating the Student CRM project with Angular 20 in non-standalone mode. Follow the steps exactly to avoid template and module mismatches.

> **Prerequisites:** Node 18+, Angular CLI 20+, and PowerShell (Windows). Run all shell commands from a terminal unless noted otherwise.

1. Create the Angular workspace

```
ng new student-crm --standalone=false --routing --style=css
cd student-crm
```

2. Bootstrap AppModule in `src/main.ts`

```
import { platformBrowserDynamic } from '@angular/platform-browser-dynamic';
import { AppModule } from './app/app.module';

platformBrowserDynamic()
  .bootstrapModule(AppModule)
  .catch(err => console.error(err));
```

3. Generate the Home component

```
ng g component home --standalone=false
```

4. Create core services

```
ng g service core/student
ng g service core/logger
```

```
// src/app/core/student.service.ts
import { Injectable } from '@angular/core';

export interface Student {
  id: number;
```

```
  name: string;
  track: 'Front-end' | 'Back-end' | 'Data' | 'DevOps';
  active: boolean;
}

@Injectable({ providedIn: 'root' })
export class StudentService {
  private store: Student[] = [
    { id: 1, name: 'Alice', track: 'Front-end', active: true },
    { id: 2, name: 'Bob', track: 'Data', active: false }
  ];

  list(): Student[] { return [...this.store]; }

  add(s: Omit<Student, 'id'>): Student[] {
    const id = Math.max(0, ...this.store.map(x => x.id)) + 1;
    this.store = [...this.store, { id, ...s }];
    return this.list();
  }

  toggleActive(id: number): Student[] {
    this.store = this.store.map(s =>
      s.id === id ? { ...s, active: !s.active } : s
    );
    return this.list();
  }
}
```

```
// src/app/core/logger.service.ts
import { Injectable } from '@angular/core';

@Injectable({ providedIn: 'root' })
export class LoggerService {
  log(message: string) {
    console.log(`[LOG] ${message}`);
  }
}
```

5. Generate the Students feature module and components

```
ng g module students
ng g component students/add-student --standalone=false
ng g component students/student-card --standalone=false
ng g component students/student-list --standalone=false
```

6. Wire the Students module

```
// src/app/students/students.module.ts
import { NgModule } from '@angular/core';
import { CommonModule } from '@angular/common';
import { FormsModule } from '@angular/forms';

import { AddStudentComponent } from './add-student/add-student.component';
import { StudentCardComponent } from './student-card/student-card.component';
import { StudentListComponent } from './student-list/student-list.component';

@NgModule({
  declarations: [
    AddStudentComponent,
    StudentCardComponent,
    StudentListComponent
  ],
  imports: [CommonModule, FormsModule],
  exports: [AddStudentComponent, StudentListComponent]
})
export class StudentsModule {}
```

7. Implement the Students components

```
// src/app/students/add-student/add-student.component.ts
import { Component, EventEmitter, Output } from '@angular/core';

@Component({
  selector: 'app-add-student',
  templateUrl: './add-student.component.html',
  styleUrls: ['./add-student.component.css']
})
export class AddStudentComponent {
  @Output() create = new EventEmitter<{ name: string; track: string; active: boolean }>();

  name = '';
  track: 'Front-end' | 'Back-end' | 'Data' | 'DevOps' = 'Front-end';
  active = true;

  submit() {
    if (!this.name.trim()) return;
    this.create.emit({ name: this.name, track: this.track, active: this.active });
    this.name = '';
    this.active = true;
  }
}
```

```
// src/app/students/student-card/student-card.component.ts
import { Component, EventEmitter, Input, OnDestroy, OnInit, Output } from '@angular/core';
import { Student } from '../../core/student.service';

@Component({
  selector: 'app-student-card',
  templateUrl: './student-card.component.html',
  styleUrls: ['./student-card.component.css']
})
export class StudentCardComponent implements OnInit, OnDestroy {
  @Input({ required: true }) student!: Student;
  @Output() toggle = new EventEmitter<number>();

  ngOnInit() { console.log(`Card init: ${this.student?.name}`); }
  ngOnDestroy() { console.log(`Card destroy: ${this.student?.name}`); }
}
```

```
// src/app/students/student-list/student-list.component.ts
import { Component, EventEmitter, Input, Output } from '@angular/core';
import { Student } from '../../core/student.service';

@Component({
  selector: 'app-student-list',
  templateUrl: './student-list.component.html',
  styleUrls: ['./student-list.component.css']
})
export class StudentListComponent {
  @Input() students: Student[] = [];
  @Output() toggleActive = new EventEmitter<number>();
}
```

8. Finish the Home feature

```
// src/app/home/home.component.ts
import { Component, OnDestroy, OnInit } from '@angular/core';
import { Student, StudentService } from '../core/student.service';
import { LoggerService } from '../core/logger.service';

@Component({
  selector: 'app-home',
  templateUrl: './home.component.html',
  styleUrls: ['./home.component.css']
})
export class HomeComponent implements OnInit, OnDestroy {
  students: Student[] = [];
```

```
  constructor(private svc: StudentService, private log: LoggerService) {}

  ngOnInit(): void {
    this.students = this.svc.list();
    this.log.log('Home initialized');
  }

  addStudent(data: Omit<Student, 'id'>) {
    this.students = this.svc.add(data);
  }

  toggleStatus(id: number) {
    this.students = this.svc.toggleActive(id);
  }

  ngOnDestroy(): void {
    this.log.log('Home destroyed');
  }
}
```

9. Configure routing

```
// src/app/app-routing.module.ts
import { NgModule } from '@angular/core';
import { RouterModule, Routes } from '@angular/router';
import { HomeComponent } from './home/home.component';

const routes: Routes = [
  { path: '', component: HomeComponent },
  { path: '**', redirectTo: '' }
];

@NgModule({
  imports: [RouterModule.forRoot(routes)],
  exports: [RouterModule]
})
export class AppRoutingModule {}
```

10. Finalize AppComponent and AppModule

```
// src/app/app.component.ts
import { Component } from '@angular/core';

@Component({
  selector: 'app-root',
```

```
  templateUrl: './app.component.html',
  styleUrls: ['./app.component.css']
})
export class AppComponent {
  title = 'Student CRM';
}
```

```
// src/app/app.module.ts
import { NgModule } from '@angular/core';
import { BrowserModule } from '@angular/platform-browser';
import { AppRoutingModule } from './app-routing.module';
import { AppComponent } from './app.component';
import { HomeComponent } from './home/home.component';
import { StudentsModule } from './students/students.module';

@NgModule({
  declarations: [AppComponent, HomeComponent],
  imports: [BrowserModule, StudentsModule, AppRoutingModule],
  providers: [],
  bootstrap: [AppComponent]
})
export class AppModule {}
```

11. Run the application

```
npm install
ng serve -o
```

After these steps, Angular finds the right template and style files, all components are declared in their modules, and the application compiles without errors.