**PROGRAM:-01** **REG NO:-24251116**

**DATE: 18-01-2025**

**AIM:**

1. Create a simple Angular application that displays "Hello, Angular!" on the screen. Follow the steps below:
   • Create a new component named hello-angular.
   • Modify the hello-angular component to display the message "Hello, Angular!" using data binding.
   • Include the hello-angular component in the AppComponent to display the message in the browser.

---

**Step – 1: Create a new angular application**

**ng new myfirst**

**Step – 2: Create a new Component**

**ng generate component hello-angular**

**Step – 3: Serve the application**

**ng serve**

**open your browser and go to http://localhost:4200/ to see the angular application.**

**index.html**

```
<!doctype html>
<html lang="en">
<head>
  <meta charset="utf-8">
  <title>Myfirst</title>
  <base href="/">
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <link rel="icon" type="image/x-icon" href="favicon.ico">
</head>
<body>
  <app-root></app-root>
</body>
</html>
```

**src/app/hello-angular/hello-angular.component.ts**

```
import { Component } from '@angular/core';

@Component({
  selector: 'app-hello-angular',
  imports: [],
  template: '<h1>{{message}}</h1>',
  styleUrl: './hello-angular.component.css'
})
```

```
export class HelloAngularComponent {
  message: string = 'Hello Angular!';
}
```

**src/app/hello-angular/hello-angular.component.html**

```
<h1>{{message}}</h1>
```

**src/app/hello-angular/hello-angular.component.css**

```
h1{
    color: blue;
    text-align:   center;
    font-weight:   bold;
    font-size: xx-large;

}
```

**Output:**

**PROGRAM:-02** **REG NO:-24251116**

**DATE: 18-01-2025**

**AIM:**

**2.** Create an Angular application to demonstrate two-way data binding. The application should allow a user to input text in a textbox, and the text should dynamically update and display below the textbox.

- Create a component named dynamic-binding.
- In the dynamic-binding component, use [(ngModel)] to bind a variable to the textbox input field and display the variable's value dynamically below the textbox.

---

**src\app\dynamicbinding \dynamicbinding.component.html**

```
<div>
   <h3>Two way data binding </h3>
   <input type="text" [(ngModel)]="userInput" placeholder="Type Something Here....."/>
   <p>Youu typed: {{userInput}}</p>
</div>
```

**src\app\dynamicbinding\dynamicbinding.component.ts**

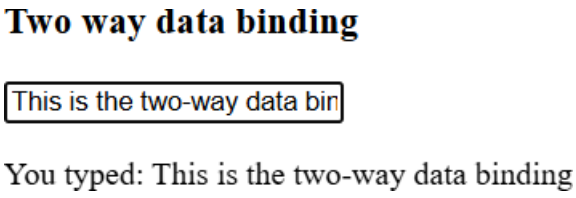
```
import { Component } from '@angular/core';


@Component({
selector: 'app-dynamicbinding',
standalone: false,


templateUrl: './dynamicbinding.component.html',
styleUrl: './dynamicbinding.component.css'
})
export class DynamicbindingComponent {
userInput: string = '';
}
```

**src\app\app.component**

```
<app-dynamicbinding></app-dynamicbinding>
```

**Output:**


Two way data binding
Type Something Here....
You typed:


Two way data binding
This is the two-way data bin
You typed: This is the two-way data binding

**PROGRAM:-03** **REG NO:-24251116**

**DATE: 18-01-2025**

**AIM:**

**3.** Conditional Rendering with *ngIf

- Create an Angular application that displays a button labeled "Toggle Message."
- When the button is clicked, toggle the visibility of a message "Hello, this is a conditional message!" using the *ngIf directive.

---

**src\app\app.component.html**

```
<div style="text-align: center; margin-top: 50px;">
  <button (click)="toogleMessage1()">Toogle Message</button>
  <p *ngIf="isMessageVisible">Hello, this is a conditional message</p>
</div>
```

**src\app\app.component.css**

```
button {
    margin: 20px;
    padding: 10px 20px;
    font-size: 16px;
  }

  p {
    font-size: 18px;
    color: red;
    font-weight: bold;
    font-family: 'Times New Roman', Times, serif;
  }
```

**src\app\app.component.ts**

```
import { Component } from '@angular/core';

@Component({
  selector: 'app-root',
  templateUrl: './app.component.html',
  standalone: false,
  styleUrl: './app.component.css'
})
export class AppComponent {
  title = 'toogleMessage1';
  isMessageVisible: boolean = false;

  toogleMessage1(){
    this.isMessageVisible = !this.isMessageVisible;
  }}
```

**Output:**



**Hello, this is a conditional message**

**PROGRAM:-04** **REG NO:-24251116**

**DATE: 18-01-2025**

**AIM:**

**4.** Using *ngFor for Displaying a List

- Create an Angular application to display a list of student names dynamically.
- Use the *ngFor directive to loop through an array of student names and display them as a list in the browser.

Include a button to add a new name to the list.

---

**src/app/student-list/student-list.component.html**

```
<h2>Student list</h2>

<ul>
   <li *ngFor="let student of students Names">{{students}}</li>
</ul>

<input
type="text"[(ngModel)]="newStudent" placeholder="Enter a new student name"/>

<button (click)="addStudent()">Add Student</button>
```

**src/app/student-list/student-list.component.css**

```
h2 {
   color: #2c3e50;
 }

 ul {
   list-style-type: none;
   padding: 0;
 }

 li {
   background: #ecf0f1;
   margin: 5px 0;
   padding: 10px;
   border-radius: 5px;
 }

 input {
   margin-right: 10px;
   border-color: black;
 }

 button{
   border-color: black;
   font-weight: bold;
    }
```

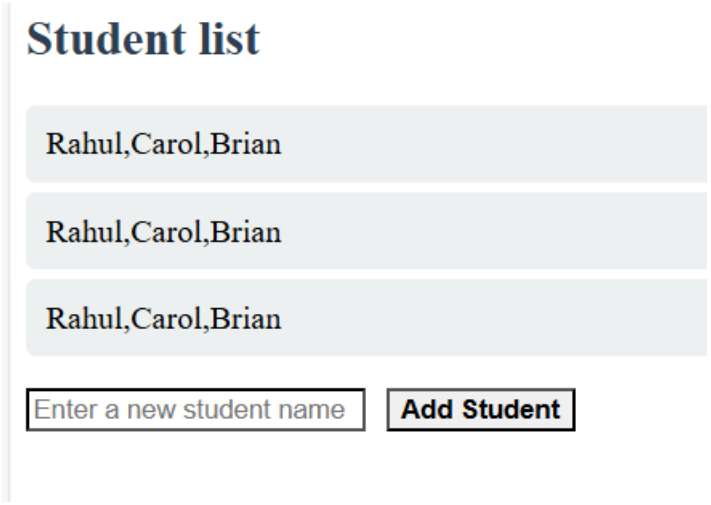
**src/app/student-list/student-list.component.ts**

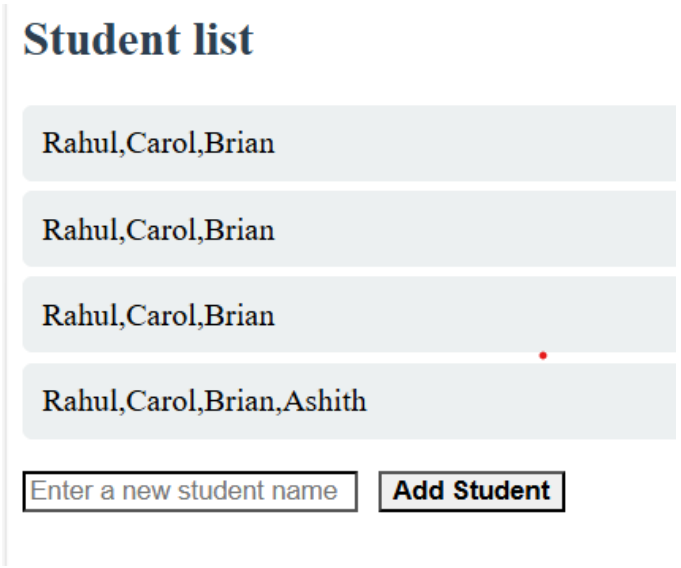
```
import { Component } from '@angular/core';

@Component({
  selector: 'app-student-list',
  standalone: false,

  templateUrl: './student-list.component.html',
  styleUrl: './student-list.component.css'
})
export class StudentListComponent {
  students: string[] = ['Rahul','Carol','Brian'];
  newStudent: string = '';

  addStudent(){
    if(this.newStudent.trim()){
      this.students.push(this.newStudent.trim());
      this.newStudent = '';
    }
  }
}
```

**src/app/app.component.html**

```
<app-student-list></app-student-list>
```

**Output:**


Student list
Rahul,Carol,Brian
Rahul,Carol,Brian
Rahul,Carol,Brian
Enter a new student name
Add Student


Student list
Rahul,Carol,Brian
Rahul,Carol,Brian
Rahul,Carol,Brian
Rahul,Carol,Brian,Ashith
Enter a new student name
Add Student

**PROGRAM:-05** **REG NO:-24251116**

**DATE: 18-01-2025**

**AIM:**

**5.** Program Title: Displaying Day of the Week Using *ngSwitch Create an Angular application that displays the name of the day based on a number input (1 to 7).

---

**src/app/day-display/day-display.component.html**

```
<div>
   <h2>Enter a number (1-7) to get the day of the week</h2>
   <input type="number" [(ngModel)]="dayNumber" min="1" max="7"/>

   <div [ngSwitch]="dayNumber">
      <div *ngSwitchCase="1">Monday</div>
      <div *ngSwitchCase="2">Tuesday</div>
      <div *ngSwitchCase="3">Wednesday</div>
      <div *ngSwitchCase="4">Thursday</div>
      <div *ngSwitchCase="5">Friday</div>
      <div *ngSwitchCase="6">Saturday</div>
      <div *ngSwitchCase="7">Sunday</div>
      <div *ngSwitchDefault>Please enter a number between 1 to 7</div>
   </div>
</div>
```

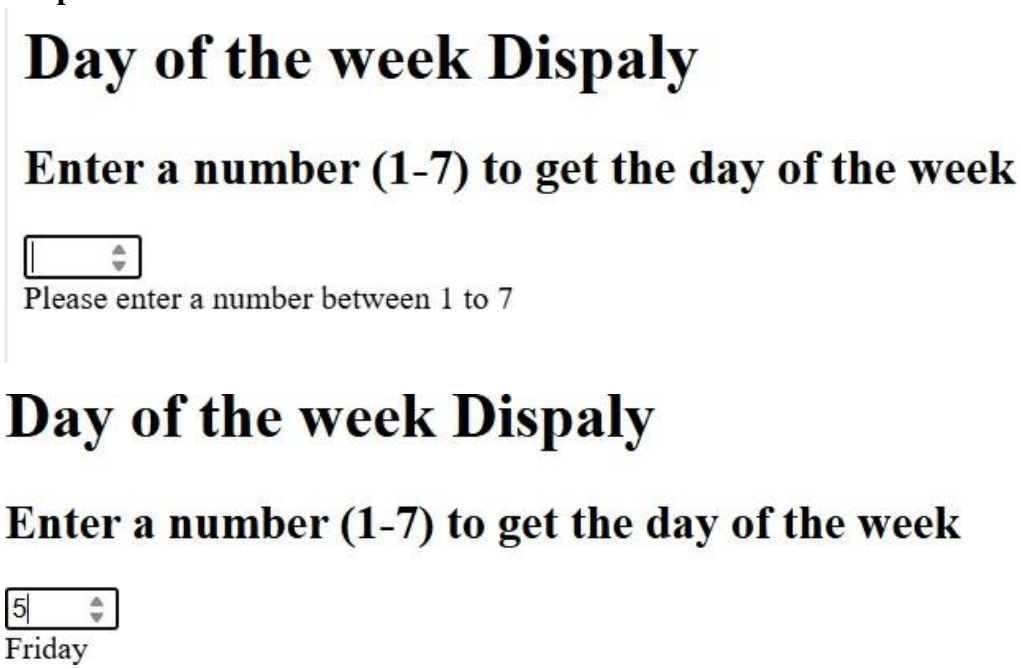
**src/app/day-display/day-display.component.ts**

```
import { Component } from '@angular/core';

@Component({
  selector: 'app-day-display',
  standalone: false,

  templateUrl: './day-display.component.html',
  styleUrl: './day-display.component.css'
})
export class DayDisplayComponent {
  dayNumber: number = 1;

}
```

**src/app/app.component.html**

```
<h1>Day of the week Dispaly</h1>

<app-day-display></app-day-display>
```

**Output:**


Day of the week Dispaly
Enter a number (1-7) to get the day of the week
Please enter a number between 1 to 7
Day of the week Dispaly
Enter a number (1-7) to get the day of the week
5
Friday

**PROGRAM:-06** **REG NO:-24251116**

**DATE: 18-01-2025**

**AIM:**

**6.** Develop an Angular program to create a simple to-do list application. Allow users to add, edit, and delete tasks. Note: The default values for tasks may be included in the program.

---

**src/app/todo/todo.component.html**

```
<div class="todo-container">
   <h1>To-Do List</h1>
   <input [(ngModel)]="newTaskTitle" placeholder="Add a new task" />   
   <button (click)="editTaskId ? updateTask() : addTask()">
    {{ editTaskId ? 'Update Task' : 'Add Task' }}
   </button>

   <ul>
    <li *ngFor="let task of tasks">
      <input type="checkbox" [(ngModel)]="task.completed" (change)="toogleCompletion(task)" />
      <span [class.completed]="task.completed">{{ task.title }}</span>
      <button (click)="editTask(task)">Edit</button>
      <button (click)="deleteTask(task.id)">Delete</button>
    </li>
   </ul>
 </div>
```

**src/app/todo/todo.component.ts**

```
import { Component } from '@angular/core';
import { Task } from './task.model';
import { title } from 'process';

@Component({
  selector: 'app-todo',
  standalone: false,
```

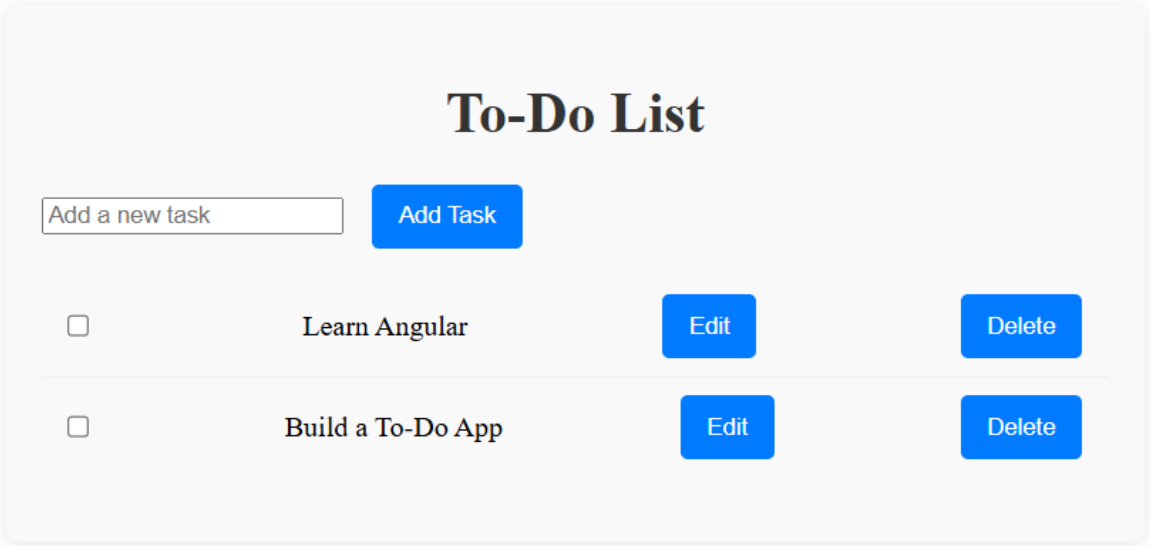
```
  templateUrl: './todo.component.html',
  styleUrl: './todo.component.css'
})
export class TodoComponent {
  tasks: Task[]=[
    {id: 1, title: 'Learn Angular', completed:false},
    {id: 2, title: 'Build a To-Do App', completed: false},
  ];
  newTaskTitle: string ='';
  editTaskId: number | null = null;

  addTask(){
    if(this.newTaskTitle.trim()){
    const newTask: Task ={
        id:  this.tasks.length+1,
        title:this.newTaskTitle,
        completed:false,
      };
      this.tasks.push(newTask);
      this.newTaskTitle  = '';
    }
  }
  editTask(task: Task){
    this.editTaskId = task.id;
    this.newTaskTitle=task.title;
  }
  updateTask(){
    if(this.editTaskId!=null){
      const task = this.tasks.find(t=>t.id == this.editTaskId);
      if(task){
        task.title = this.newTaskTitle;
        this.newTaskTitle='';
```

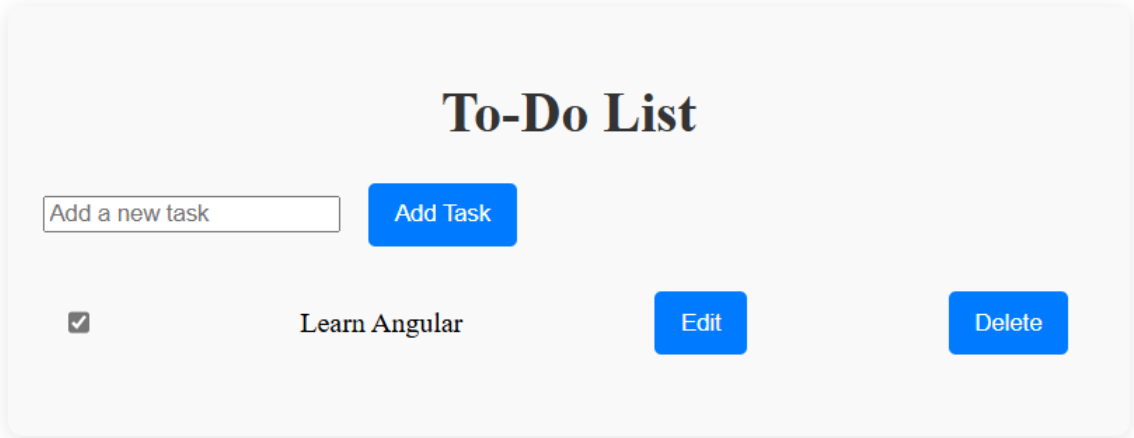
```
        this.editTaskId=null;
      }
    }
  }
  deleteTask(taskId: number){
    this.tasks = this.tasks.filter(task => task.id!=taskId);
  }
toogleCompletion(task: Task){
  task.completed = !task.completed;
}
}
```

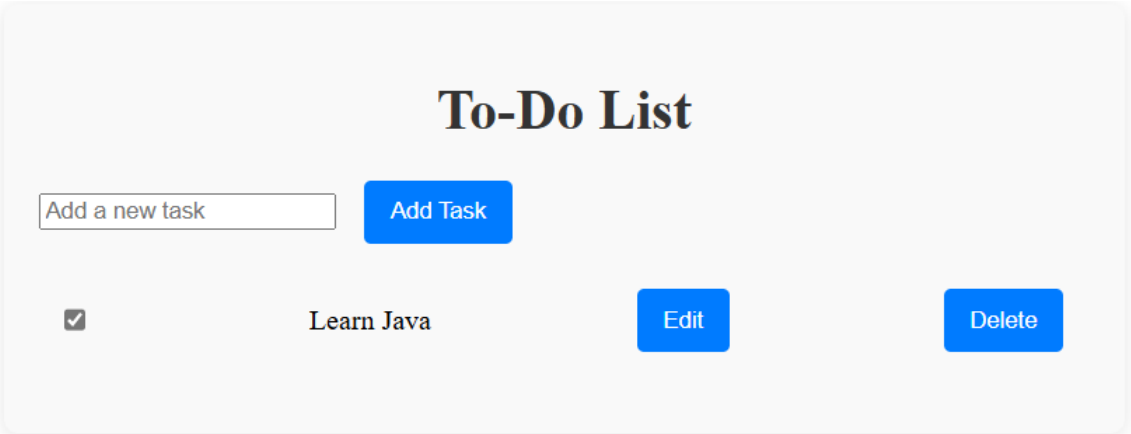
**src/app/app.component.html**

```
<app-todo></app-todo>
```

**Output:**


To-Do List
Add a new task
Add Task
Learn Angular
Edit
Delete
Build a To-Do App
Edit
Delete


To-Do List
Add a new task
Add Task
Learn Angular
Edit
Delete


To-Do List
Add a new task
Add Task
Learn Java
Edit
Delete

**PROGRAM:-07** **REG NO:-24251116**

**DATE: 18-01-2025**

**AIM:**

**7.** Angular Pipes

- Create Angular application to convert student details to Uppercase using angular filters. Note: The default details of students may be included in the program.
- Create an Angular application that displays the date by using date filter parameters.

---

**src/app/date-display/date-display.component.html**

```
<h2>Current Date</h2>
<p>Default Format: {{ currentDate | date }}</p>
<p>Short Date: {{ currentDate | date:'shortDate' }}</p>
<p>Medium Date: {{ currentDate | date:'mediumDate' }}</p>
<p>Long Date: {{ currentDate | date:'longDate' }}</p>
<p>Custom Format: {{ currentDate | date:'yyyy-MM-dd HH:mm:ss' }}</p>
```

**src/app/date-display/date-display.component.css**

```
h2 {
   color: #007bff;
   margin-top: 20px;
 }

 p {
   background: #fff;
   margin: 10px 0;
   padding: 10px;
   border-radius: 5px;
   box-shadow: 0 2px 5px rgba(0, 0, 0, 0.1);
 }
```

**src/app/date-display/date-display.component.ts**

```
import { Component } from '@angular/core';

@Component({
  selector: 'app-date-display',
  standalone: false,

  templateUrl: './date-display.component.html',
  styleUrl: './date-display.component.css'
})
export class DateDisplayComponent {
  currentDate: Date = new Date();
}
```

**src/app/student-details/student-details.component.html**

```
<h2>Student Details</h2>
<ul>
  <li *ngFor="let student of students">
```

```
   Name: {{ student.name | uppercase }} | Age: {{ student.age }} | Course: {{ student.course | uppercase }}
  </li>
</ul>
```

**src/app/student-details/student-details.component.css**

```
h2 {
   color: #007bff;
   margin-top: 20px;
  }

  ul {
   list-style-type: none;
   padding: 0;
  }

  li {
   background: #fff;
   margin: 10px 0;
   padding: 10px;
   border-radius: 5px;
   box-shadow: 0 2px 5px rgba(0, 0, 0, 0.1);
  }
```

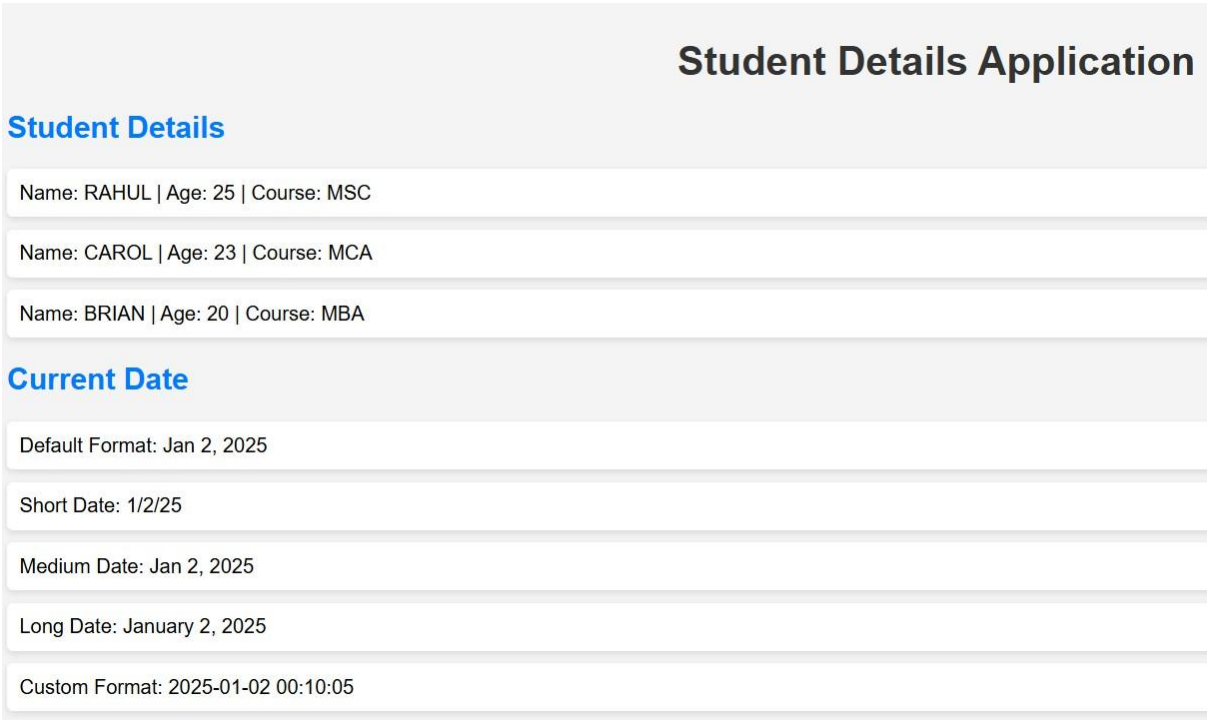
**src/app/student-details/student-details.component.ts**

```
import { Component } from '@angular/core';

@Component({
  selector: 'app-student-details',
  standalone: false,

  templateUrl: './student-details.component.html',
  styleUrl: './student-details.component.css'
})
export class StudentDetailsComponent {
  students = [
    { name: 'Rahul', age:'25', course:'Msc'},
    { name: 'Carol', age:'23', course:'MCA'},
    { name: 'Brian', age:'20', course:'MBA'}

  ];}
```

**src/app/app.component.html**

```
<h1>Student Details Application</h1>
<app-student-details></app-student-details>
<app-date-display></app-date-display>
```

**Output:**


Student Details Application
Student Details
Name: RAHUL | Age: 25 | Course: MSC
Name: CAROL | Age: 23 | Course: MCA
Name: BRIAN | Age: 20 | Course: MBA
Current Date
Default Format: Jan 2, 2025
Short Date: 1/2/25
Medium Date: Jan 2, 2025
Long Date: January 2, 2025
Custom Format: 2025-01-02 00:10:05