Lecture 08

Installing angular Material (as of today 17.3.6)

1. In the terminal, run the following command

```
ng add @angular/material@17.3.6
```

2. Choose a theme of your choice.
3. Setup global Angular Material typography styles: **y**.
4. Include and enable animations: **y**.
5. Verify that the library has been installed by browsing to the *angular.json* file and look for the *styles* property.

Getting familiar with Material

1. Browse to https://material.angular.io and browse through the various components to see how to use them in the code. This link always refers to the latest version, BUT because we are using v17, so we will select that.

Implement a Toolbar for your application to serve as a main menu.

1. Edit *app.component.html* and add the following at the top most.

```
<mat-toolbar>
   <button mat-icon-button class="example-icon" aria-label="Example icon-button with menu
icon">
     <mat-icon>menu</mat-icon>
   </button>
   <span>Course Admin System</span>
</mat-toolbar>
...
```

2. Edit *app.component.ts* and add the following to the **imports** list.

```
import { MatIconModule } from '@angular/material/icon';
import { MatButtonModule } from '@angular/material/button';
import { MatToolbarModule } from '@angular/material/toolbar';


...
@Component({
...
   imports: [MatToolbarModule, MatButtonModule, MatIconModule ... ]
...
})
```

3. If we need to provide som custom colour, then we can do that by editing the *app.component.css* file as follows.

```
.mat-toolbar {
   background-color: red;
}
```

Moving the menu items inside the Menu

1. On Material website, look for the *Menu* component to see how to add it to your application.
2. Edit *app.component.html* and update the *mat-toolbar* as follows.

```
<mat-toolbar>
   <button mat-icon-button class="example-icon" aria-label="Example icon-button with menu
icon" [matMenuTriggerFor]="menu">
     <mat-icon>menu</mat-icon>
   </button>
   <mat-menu #menu="matMenu">
      <button mat-menu-item>Students</button>
    </mat-menu>
   <span>Welcome Applied Programming Course</span>
</mat-toolbar>
```

3. Edit *app.component.ts* and update the **imports** as follows.

```
import { MatMenuModule } from '@angular/material/menu';
...
...
@Component({
...
   imports: [MatMenuModule, ... ]
...
})
```

4. Notice that the links do not work. It's because the buttons do not have *routerLink* specified. Change that by updating the buttons as follows.

```
<button routerLink="/student" mat-menu-item>Students</button>
```

5. Verify that the links work.

Add a new component to add new student

1. Add a new component using the following command.

```
ng g c AddStudent
```

2. On Material website, look for the *Input component* to see how to add it to your application.
3. Any *input* that needs to be added has the general form as follows.

```
<mat-form-field>
   <mat-label>Some Label For your Input</mat-label>
   <input matInput>
</mat-form-field>
```

4. Edit *add-student.component.html* and update all inputs to *mat-form-field*s as follows.

```
<mat-form-field>
  <mat-label>First name</mat-label>
  <input matInput type="text">
</mat-form-field><br />
<mat-form-field>
  <mat-label>Last name</mat-label>
  <input matInput type="text">
</mat-form-field><br />
<mat-form-field>
  <mat-label>Study Program</mat-label>
  <input matInput type="text">
</mat-form-field><br />
<mat-form-field>
  <mat-label>DOB</mat-label>
  <input matInput type="text">
</mat-form-field><br />
<mat-form-field>
  <mat-label>Email</mat-label>
  <input matInput type="text">
</mat-form-field><br />
<mat-form-field>
  <mat-label>Phone</mat-label>
  <input matInput type="text">
</mat-form-field><br />
<button mat-raised-button (click)="addStudent()">Add</button>
```

5. Edit *add-student.component.ts* and add the following to the **imports** list.

```
...
import { MatFormFieldModule } from '@angular/material/form-field';
import { MatInputModule } from '@angular/material/input';
import { MatButtonModule } from '@angular/material/button';

@Component({
...
   imports: [MatButtonModule, MatFormFieldModule, MatInputModule, ...]
...
})
```

6. Edit *add-student.ts* and add a new *addStudent()* function as follows.

```
...
```

```
addStudent() {
}
...
```

7. Edit the *app.routes.ts* file and add a new route to the routes array as follows.

```
...
{ path: 'add-student', component: AddStudentComponent },
...
```

8. Edit *app.component.html* and add a new menu item for this new route as follows.

```
...
<mat-menu #menu="matMenu">
   <button mat-menu-item routerLink="/">Dashboard</button>
   <button mat-menu-item routerLink="/student">Students</button>
   <button mat-menu-item routerLink="/add-student">Add student</button>
</mat-menu>
...
```

Add Basic validation using Reactive Forms approach

1. Edit *add-student.component.html* and wrap the complete code inside a *form* tag as follows.

```
<form [formGroup]="studentFormGroup">
....
</form>
```

2. Edit all the elements by adding a *formControl* attribute toe them as shown below.

```
...
<mat-form-field>
   <mat-label>First name</mat-label>
   <input matInput type="text" [formControl]="firstName">
 </mat-form-field><br />
 <mat-form-field>
   <mat-label>Last name</mat-label>
   <input matInput type="text" [formControl]="lastName">
 </mat-form-field><br />
 <mat-form-field>
   <mat-label>Study Program</mat-label>
   <input matInput type="text" [formControl]="studyProgram">
 </mat-form-field><br />
 <mat-form-field>
```

```
   <mat-label>DOB</mat-label>
   <input matInput type="text" [formControl]="dob">
 </mat-form-field><br />
 <mat-form-field>
   <mat-label>Email</mat-label>
   <input matInput type="text" [formControl]="email">
 </mat-form-field><br />
 <mat-form-field>
   <mat-label>Phone</mat-label>
   <input matInput type="text" [formControl]="phone">
 </mat-form-field><br />


...
```

3. Edit the *add-student.component.ts* and add the following to **imports** list

```
...
import { FormControl, FormGroup, FormsModule, ReactiveFormsModule, Validators } from
'@angular/forms';
...
@Component({
...
   imports: [, ReactiveFormsModule, ...]
...
})
```

4. Edit the *add-student component.ts* and add the following *FormGroup* and *FormControl*.

```
// Form Controls
firstName: FormControl = new FormControl('', [Validators.required]);
lastName: FormControl = new FormControl('', [Validators.required]);
studyProgram: FormControl = new FormControl('1', [Validators.required,
Validators.pattern('\\d+')]); // Only digits
dob: FormControl = new FormControl('', [Validators.required]);
email: FormControl = new FormControl('', [Validators.required, Validators.email]);
phone: FormControl = new FormControl('', [Validators.required, Validators.pattern('\
\d{8}')]); // Only 8 digits


studentFormGroup: FormGroup = new FormGroup({
  firstName: this.firstName,
  lastName: this.lastName,
  studyProgram: this.studyProgram,
  dob: this.dob,
  email: this.email,
```

```
  phone: this.phone
});
```

5. Edit the *add-student.component.ts* and update the *addStudent* method as follows, so that it only updates the data if it is valid.

```
  addStudent() {
    if (!this.studentFormGroup.valid) {
      console.log('Data not valid');
      return;
    }

    this.studentService.addStudent({
      firstName: this.firstName.value,
      lastName: this.lastName.value,
      studyProgramId: this.studyProgram.value,
      dob: new Date(this.dob.value),
      id: 0,
      email: this.email.value,
      phone: this.phone.value
    }).subscribe({
      next: () => console.log('Done'),
      error: (err) => console.error('Somethig went wrong: ' + err)
    })

  }
```

6. Update the *student.service.ts* with a new method to add student as follows.

```
...
addStudent(student: Student) : Observable<any> {
  return this.httpClient.post(`${this.baseUrl}/student`, student);
}
...
```

7. Optionally, we can choose to enable the *Update* button ONLY if the data is valid (as shown below), otherwise it will be disabled.

```
<button [disabled]="!this.studentFormGroup.valid" (click)="addStudent()">Add</button>
```

Convert DOB to a Date Picker

1. Install a new package MomentDateAdapter using the following command.

```
ng add @angular/material-moment-adapter
```

2. Edit *student.component.html* and update the dob as follows. Remember to add **CommonModule** to list of Imports in the *student.component*.ts file.

```
...
<b>Dob:</b><span>{{student.dob | date:'dd-MM-yyyy'}}</span>
...
```

3. Edit *add-student.component.html* and replace the dob field as follows.

```
...
<mat-form-field>
   <mat-label>DOB</mat-label>
   <input matInput [matDatepicker]="picker" [formControl]="dob">
   <mat-datepicker-toggle matIconSuffix [for]="picker"></mat-datepicker-toggle>
   <mat-datepicker #picker></mat-datepicker>
</mat-form-field><br/>
...
```

4. Edit *add-student.component.ts* and update the following to the **imports** and **providers** list.

```
...
import { MatDatepickerModule } from '@angular/material/datepicker';
import { MatMomentDateModule, provideMomentDateAdapter } from '@angular/material-moment-
adapter';


@Component({
...
   providers: [provideMomentDateAdapter(
      {
        parse: {
          dateInput: ['DD-MM-YYYY'],
        },
        display: {
          dateInput: 'DD-MM-YYYY',
          monthYearLabel: 'MMM YYYY',
          dateA11yLabel: 'LL',
          monthYearA11yLabel: 'MMMM YYYY',
        },
      },
      { useUtc: true })
    ],
    imports: [, MatDatepickerModule, MatMomentDateModule, ...],
...
})
```

```
export class AddStudent implements OnInit {
   ...
}
```

5. Try selecting a date to make sure that our new control works.

Exercise

1. Implement validation for other components.
2. Implement DatePicker control in EditStudent component.
3. Explore other angular material e.g. Card to improve the user interface.