Section 11- Skill Assesment

Full steps including code pasted below. I mostly just used the eStore project weve been building as a guide, and resued some of the Skill Assesment 8 stuff to build this quickly. Ive done some additional minor steps like input a logo in the index.html so its on all pages, made a favicon, and tweaked the form controls to get different kinds of input.

Start a new Angular Project: in Powershell: ng new henna-solutions
FAIL: Powershell strict cannot run scripts?
Yep: CMD >>> Get-ExecutionPolicy
returns “Restricted”. Change this
Set-ExecutionPolicy RemoteSigned
Create Angular PJ: ng new henna-solutions, use angular routing, and CSS stylesheets
Open in VSC
Make Client Management Component: In VSC Command interface: ng g c client-management
Fail: navigate to PJ folder…
Create 3 components:
ng g c client-management
ng g c meeting-managemer
ng g c home

app.component.html:
Delete all placeholder text in app.component.html
create a basic angular navbar to Home, Meetings, and Client Manager components

app.component.html

```
<nav>
  <ul>
    <li><a routerLink="/home">Home</a></li>
    <li><a routerLink="/meetings">Meetings</a></li>
    <li><a routerLink="/clients">Clients</a></li>
  </ul>
</nav>
<app-home [name]="appName"></app-home>
<router-outlet></router-outlet>
```

app.module.ts
make sure our new components are listed in declarations

```
  declarations: [
```

```
    AppComponent,
    ClientManagementComponent,
    MeetingMangerComponent,
    HomeComponent
  ],
```

Import the ReactiveFormsModule and list it in imports

app.component.ts:
Implement some vars with names:

```
export class AppComponent {
  title = 'Henna_Solutions';
  appName 'Architectural Solutions'
}
```

app.component.css: CURRENTLY: Dropped huge CSS chunk from the Section 11- eStore project. Has a bunch of Bootstrap copied CSS we may use.

app-routing.module.ts:
Add paths to routes

```
import { HomeComponent } from './home/home.component';
import { ClientManagementComponent } from
'./client-management/client-management.component';
import { MeetingMangerComponent } from
'./meeting-manger/meeting-manger.component';

const routes: Routes = [
  { path: 'home', component: HomeComponent },
  { path: 'clients', component: ClientManagementComponent },
  { path: 'meetings', component: MeetingMangerComponent },
];
```

Had to manually import the last two components. Possible naming convention issues. Note the need to specify their path

home.component.ts:
modify class to bind name variable:

```
export class HomeComponent implements OnInit {
  //using input decorator/directive
  // name will get its data from its parent component
  @Input() name = 'NA';
```

```
  //dependency injection for route
  constructor() { }

  ngOnInit(): void { }
}
```

Make sure to update its imports:

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

In index.html add the logo image. Pain in the butt for some reason. Copying relative path didnt work.
Replace favicon

Generate service for meetings: ng generate service meetings

meeting-manager.componnet.ts
IMPLEMENT WHOLE MEETING CLASS
Add reactive form elements from angular:

```
import { Component, OnInit } from '@angular/core';
import { MeetingsService } from '../meetings.service';
import { FormControl } from '@angular/forms';
import { FormGroup } from '@angular/forms';

@Component({
  selector: 'app-meeting-manger',
  templateUrl: './meeting-manger.component.html',
  styleUrls: ['./meeting-manger.component.css']
})
export class MeetingMangerComponent implements OnInit {

  meetings = this.meetingsService.getMeetings();

  showMeetings = true;

  categories = [
    { title: "Meet & Greet" },
    { title: "Design Consult" },
    { title: "Planning" },
    { title: "Questions" },
  ]
```

```
  meetingForm = new FormGroup(
    {
      date: new FormControl('1/1/2023'),
      time: new FormControl("12:30pm"),
      attending: new FormControl('Client Name'),
      category: new FormControl(this.categories[0])
    }
  );

  constructor(private meetingsService: MeetingsService) {

  }


  addMeetingToDB() {

  }

  ngOnInit(): void {
  }
}
```

Make sure to update LOTS of imports

make a modal folder (somehow this was done automatically in the video pj?)
Make a meeting class:

```
export class Meeting {
    date: string;
    attending: string;
    time: string;
    constructor(date: string, attending: string, time: string) {
        this.date = date;
        this.attending = attending;
        this.time = time;
    }

    showMeeting() {
        console.log(this.date + " " + this.time + " " + this.attending);
```

```
    }
}
```

Implement meeting-manager.componenet.html

```
<h2>Manage Meetings</h2>
<div>
    <form [formGroup]="meetingForm" (ngSubmit)="addMeetingToDB">
        <p>Date:</p>
        <input type="date" id="date" placeholder="1-1-2001"
formControlName="date">
        <br>
        <p>Time:</p>
        <input type="time" id="time" placeholder="12:45p"
formControlName="time">
        <br>
        <p>Attending:</p>
        <input type="text" id="attending" placeholder="Client Names"
formControlName="attending">
        <br>
        <p>Meeting Type:</p>
        <select formControlName="category">
            <option *ngFor="let category of categories">
                {{category.title}}
            </option>
        </select>
        <button type="submit">ADD MEETING</button>
        <br>
    </form>
</div>

<h2>FORM DATA</h2>
<p>{{meetingForm.value | json}}</p>

<h2>Current Meetings</h2>
<div *ngIf="showMeetings">
    <p *ngFor='let meeting of meetings'>
        {{meeting.date}} | {{meeting.time}} | {{meeting.attending}} |
{{meeting.category}}
    </p>
</div>
```

meetings.service.ts:
Create a meetings service class with some dummy data and a returnMeetings() func

```
import { Injectable } from '@angular/core';
import { Meeting } from './modal/Meetings';

@Injectable({
  providedIn: 'root'
})
export class MeetingsService {
  meetings: Meeting[] = [
    new Meeting("1-1-2023", "John Smith", "1:00pm", "Design Consult"),
    new Meeting("1-10-2023", "John Smith", "2:00pm", "Planning"),
    new Meeting("1-23-2023", "John Smith", "10:00am", "Plan Approval"),
  ];

  getMeetings() {
    return this.meetings;
  }

  constructor() { }
}
```

Implement meetings.service.ts

```
import { Injectable } from '@angular/core';
import { Meeting } from './modal/Meetings';

@Injectable({
  providedIn: 'root'
})
export class MeetingsService {
```

```
  meetings: Meeting[] = [
    new Meeting(new Date("1-10-2023"), "1:00pm", "John Smith", "Design
Consult"),
    new Meeting(new Date("1-10-2023"), "2:00pm", "John Smith",
"Planning"),
    new Meeting(new Date("1-23-2023"), "10:00am", "John Smith", "Plan
Approval"),
  ];

  getMeetings() {
    return this.meetings;
  }

  constructor() { }
}
```

OH MY GOD IT ACTUALLY WORKS!

Ok, more or less copy this code into a clients component and service with minor tweaking:

Wow!