**Here is a step by step explanation to practice the modules we learned in the class.**

**Step 1- Components:**

1. Go to the angular-seed-pure folder
2. Add new folder according to component name
3. Add Typescript file and create your first component
4. Remember the component should have @import, metadata and @export keywords
5. Choose a selector name in the component metadata
6. Add html file to view the data about an artist: Name, album name
7. Add CSS file if needed
8. Add the component in app.component.html file

**Step 2- Data Binding:**

1. Go to the step 2.1 folder
2. In your Typescript file add an object named album and use this JSON to display real data.

{  
 **"name"**: **"Revolver"**,  
 **"url"**:**"https://www.last.fm/music/The+Beatles/Revolver"**,  
 **"artist"**: {  
 **"name"**: **"The Beatles"**,},  
 **"image"**:   
 {  
 **"src"**: **"https://lastfm-img2.akamaized.net/i/u/174s/fafc74a8f45241acc10158be6e2d8270.png"**,  
 }  
}

1. Add binding in your component html according to the album object.

**Step 3- Directives:**

1. Go to the step 3.1 folder
2. Use ng-if directive to show the album artist name , only if the name is ‘The Beatles’

**Step 4- Services:**

1. Go to the step 4.1 folder
2. Add resources folder into app folder
3. Add beatles-albums.json file into resources folder
4. Copy the JSON from the step 4.2/src/app/resources/beatles-albums.json file
5. Add album.ts file into playlist folder and create 3 classes ImageObject, Album, Artist

according to the hard coded album object you created in a previous step.

1. Add playlist.service.ts file in playlist folder, import Injectable, Http, Response, Observable modules and album class. Use albumsUrl = **'app/resources/beatles-albums.json'** and create a service to get the albums from JSON file.
2. Change the component to get albums from the service.

**Step 5- Pipes:**

1. Go to the step 5.1 folder
2. Add ellipsis.pipe Typescript file to the playlist folder
3. **import** { Pipe, PipeTransform } **from '@angular/core'**;
4. Create a pipe that takes a number (length of string) and if the text is longer than a number, cuts the text and adds … to the end of the text.
5. Add the pipe for the relevant property in the component html
6. Don’t forget to import the pipe in app.module file and add it as declaration
7. You can write another pipe if you want

**Step 6- Forms:**

1. Go to the step 6.1 folder
2. Add contact folder
3. Add contact model class with name, email, content properties
4. Add contact component with html and CSS files
5. Use [(ngModel)] binding to bind the contact model and input elements in html.
6. Use [hidden] attribute to show/ hide validation errors, according to the mode.valid, model.pristine or another input element status.
7. Add validation rule as you want.

**Step 7- Routing:**

1. Go to the step 7.1 folder
2. Add navbar folder
3. Add navbar component with html and CSS files
4. In navbar component html add routerLink directive and set the name of component.
5. Add routerConfig Typescript file , insert routing like this

**export const** RouterConfig:**any**[] =  
[  
 { **path**:**''**, **redirectTo**:**'/home'**, **pathMatch**:**'full'**},   
 { **path**:**'home'**, **component**: PlaylistComponent},  
 { **path**:**'contact'**, **component**: ContactComponent}   
}]

1. Inport RouterConfig in app.module.ts and add RouterModule.*forRoot*(RouterConfig) in imports section.