**Angular**

Exercise -- 1

1. See the following picture and create the required components, templates



1. Create movie-list component, which should be having the movies list as shown in the above picture.
   1. You can create a movies array to hold the movies as shown below

movies = ['The Manchurian Candidate', 'Oceans 8',’The Incredibles’, ‘Hidden Figures’];

* 1. Change the movie list text by adjusting the code in movie-list.component.css to accomplish the following:
     1. The text for the heading and list items can be any color EXCEPT black.
     2. The movie list should have a centered heading.
     3. The font size should be large enough to easily read.

Ans:

movies/movie-list.component.ts

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

@Component({

selector: 'movie-list',

templateUrl: './movie-list.component.html',

styleUrls: ['./movie-list.component.css']

})

export class MovieListComponent implements OnInit {

movies = ['The Manchurian Candidate', 'Oceans 8',’The Incredibles’, ‘Hidden Figures’];

constructor() { }

ngOnInit() {

}

}

movies/movie-list.component.css

.movies {

color: purple;

font-size: 30px;

}

h3 {

text-align: center;

}

movies/movie-list.component.html

<div class="movies">

<h3>Movies to Watch</h3>

<ol>

<li>{{movies[0]}}</li>

<li>{{movies[1]}}</li>

<li>{{movies[2]}}</li>

<li>{{movies[3]}}</li>

</ol>

</div>

1. Create chore-list component, which should be having the chores list as shown in the above picture.
   1. Change the chore list text by adjusting the code in chores-list.component.css to accomplish the following:
   2. Use a different font, with a size large enough to easily read.
   3. The text color should be different from the movie list, but not black.
   4. The chores list should have an underlined heading.
   5. The chores in the list should be italicized

chores/chores-list.component.ts

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

@Component({

selector: 'chores-list',

templateUrl: './chores-list.component.html',

styleUrls: ['./chores-list.component.css']

})

export class ChoresListComponent implements OnInit {

chores = ['Empty dishwasher', 'Complete LaunchCode prep work', 'Buy groceries'];

constructor() { }

ngOnInit() {

}

}

chores/chores-list.component.html

<div class="chores">

<h3>Today's Chores</h3>

<ul>

<li>{{chores[0]}}</li>

<li>{{chores[1]}}</li>

<li>{{chores[2]}}</li>

</ul>

</div>

chores/chores-list.component.css

.

h3{

underline the text

}

.chores {

color:red

.

.

}

1. Create fav-photos component, which should be having a single photo as shown in the above picture. ('https://www.launchcode.org/assets/icons/target-00c7a0bc7776901e3c5cdc4c36c465f19e41a249bc7267a595f0bfa14ab944b7.png')

fav-photos.component.ts

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

@Component({

selector: 'fav-photos',

templateUrl: './fav-photos.component.html',

styleUrls: ['./fav-photos.component.css']

})

export class FavPhotosComponent implements OnInit {

photosTitle = 'Photo/Image Section Title Here';

image1 = 'https://www.launchcode.org/assets/icons/target-00c7a0bc7776901e3c5cdc4c36c465f19e41a249bc7267a595f0bfa14ab944b7.png';

image2 = '';

image3 = '';

constructor() { }

ngOnInit() {

}

}

fav-photos.component.html

<div class="photos">

<h3>{{photosTitle}}</h3>

<img src="{{image1}}" alt="Oops! Missing photo!">

<img src="" alt="Oops! Missing photo!"> <br>

<img src="" alt="Oops! Missing photo!">

</div>

fav-photos.component.css

h3 {

color: brown;

}

img {

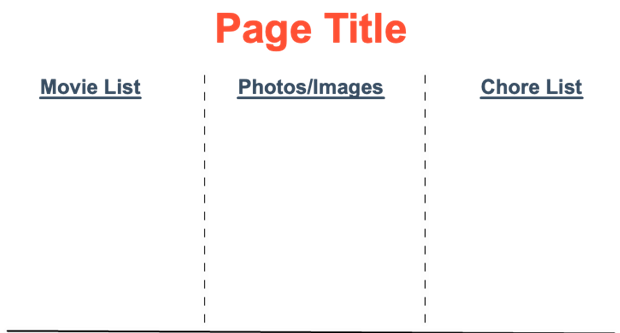
}

.photos {

text-align: center;

}

1. You need to get a plan to display total output as shown below



1. The page needs a title.

Put inside app.component.ts

title=”Angular Exercises”;

1. Finally you need to include all these components inside app.component.html to get picture shown above

app.component.html

<h1>{{title}}</h1>

<div class="col-3">

<movie-list></movie-list>

</div>

<div class="col-3">

<fav-photos></fav-photos>

</div>

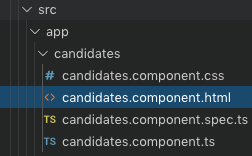
<div class="col-3">

<chores-list></chores-list>

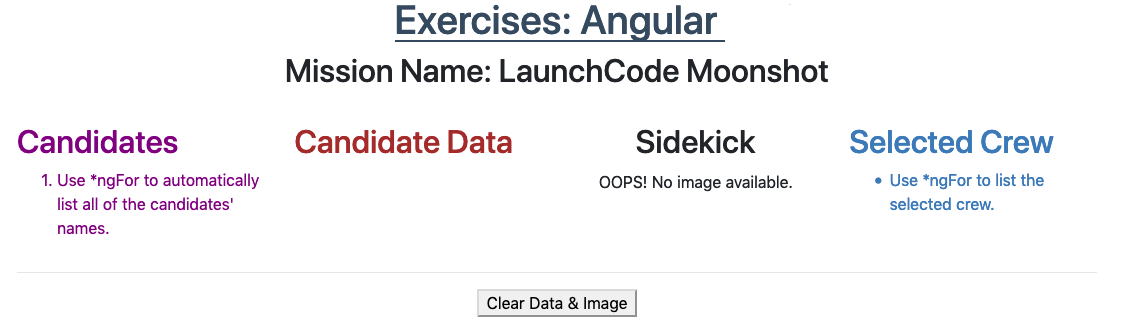
</div>

Exercise – 2

1. Let's build an interactive web page that allows us to review data for our astronaut candidates and select crew members for a space mission
2. Create a folder “candidates” under app folder and do the requirements. Get the help from the picture shown below



1. You have to design your application such that you need to get an html output as shown below



**candidates.component.ts**

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

@Component({

selector: 'candidates',

templateUrl: './candidates.component.html',

styleUrls: ['./candidates.component.css']

})

export class CandidatesComponent implements OnInit {

missionName = "LaunchCode Moonshot"

candidates = [

{name: 'Rusty Rutabaga', data: {age: 5, mass: '0.75 kg', sidekick: 'Blake'}, image: 'assets/images/Blake.png'},

{name: 'Tessa Tortoise', data: {age: 126, mass: '113 kg', sidekick: 'Sally'}, image: 'assets/images/Sally.png'},

{name: 'Bernie Beagle', data: {age: 4, mass: '15 kg', sidekick: 'Paul'}, image: 'assets/images/Paul.png'},

{name: 'Jackalope', data: {age: 45, mass: '33 kg', sidekick: 'Chris'}, image: 'assets/images/Chris.png'},

{name: 'Champ', data: {age: 9, mass: '36 kg', sidekick: 'Carly'}, image: 'assets/images/Carly.png'}

];

crew = [];

constructor() { }

ngOnInit() {

}

// Code the addToCrew function here:

// BONUS: Code the changeMissionName function here:

}

## ****candidates.component.css****

.candidates {

color: purple;

}

.data {

color: brown;

}

img {

width: 40%;

height: auto;

}

.centered {

text-align: center;

}

.crew {

color: #3c79b8;

}

## ****candidates.component.html****

<div class="container">

<h2 class="centered">Mission Name: {{missionName}}</h2>

<br>

<div class="row justify-content-center">

<div class="candidates col-3">

<h2>Candidates</h2>

<ol>

<li>Use \*ngFor to automatically list all of the candidates' names.</li>

</ol>

</div>

<div class="data col-3">

<h2>Candidate Data</h2>

<p> <!-- Add an \*ngIf to this tag to display the selected candidate's data-->

{{selected.name}}

</p>

</div>

<div class="centered col-3">

<h2>Sidekick</h2>

<!-- Use \*ngIf and the correct placeholder to display a sidekick's image -->

<img src="{{placeholder}}" alt="OOPS! No image available."/>

</div>

<div class="crew col-3">

<h2>Selected Crew</h2>

<ul>

<li>Use \*ngFor to list the selected crew.</li>

</ul>

<!-- Add a "Clear Crew List" button here. -->

</div>

</div>

<hr>

<div class = 'centered'>

<button>Clear Data & Image</button>

<!-- Add a "Send on Mission" button here. -->

</div>

</div>

<!-- The code below is for one of the Bonus Missions -->

<ng-template #editMission>

<h2 class="centered">

Mission Name:

<!-- Add a keyup.enter event that calls the changeMissionName function. -->

<input #newName type="text"/>

</h2>

</ng-template>

1. Candidates Column

Examine the candidates array in candidates.component.ts. It contains one object for each animal astronaut. We want to start by listing the names of the animals in the "Candidates" column of the web page.

Use \*ngFor in the <li> tag to loop over the candidates array and display each name in an ordered list.

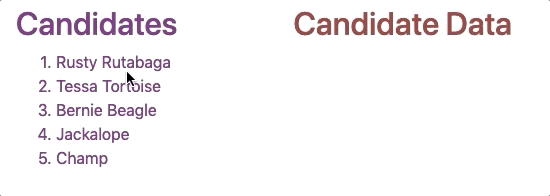
Use **\*ngFor** in the **<li>** tag to loop over the candidates array and display each name in an ordered list.

**src/app/candidates/candidates.component.html**:

|  |  |
| --- | --- |
| 1  2  3  4  5  6 | <**div** **class**="candidates col-3">  <**h2**>Candidates</**h2**>  <**ol**>  <**li** \***ngFor**="let candidate of candidates" (**click**)="selected = candidate">***{{***candidate.name***}}***</**li**>  </**ol**>  </**div**> |

We want each name to be interactive. Add a click event to the <li> tag. When a user clicks on a name, set the variable selected to be equal to the chosen candidate.

Properly done, your output should behave something like this



After mouseovering on an item from the Candidates list you should show same name details in Candidate Data column as shown below

Name: Rusty Rutabaga

Age: 5

Mass: 0.75 kg

Sidekick: Blake

1. Sidekick Image Column

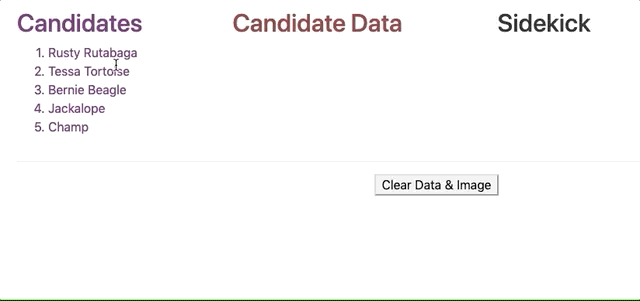
Every good hero needs a loyal sidekick, and our candidates are no exception!

When we click on a candidate's name, we want an image of their sidekick to appear under the "Sidekick" column. If no candidate is selected, we want this area to remain blank.

In the <img> tag, use \*ngIf to check if a candidate has been selected.

Replace the generic {{placeholder}} with the image property of the candidate.

Properly done, your output should behave something like this:



After clicking an item from the candidates, you need show that candidates information as shown above, and also the sidekick picture belonging to the candidates. (I have send that picture “lesson2-exercises-sidekicks.gif” to you, Please refer it.

1. Selected Crew Column

Once we select a candidate, we want an option to add them to the crew of the next space mission.

In candidates.component.ts, code an addToCrew function that takes an object as a parameter.

If the candidate is NOT part of the crew, the function should push them into the crew array. Candidates who are already part of the crew should be ignored.

In candidates.component.html, add a "Send on Mission" button next to the "Clear Data & Image" button.

Add a click event to the button to call the addToCrew function. When clicked, pass the selected candidate as the argument.

Under the "Selected Crew" section, use \*ngFor to loop over the crew array and display each name.

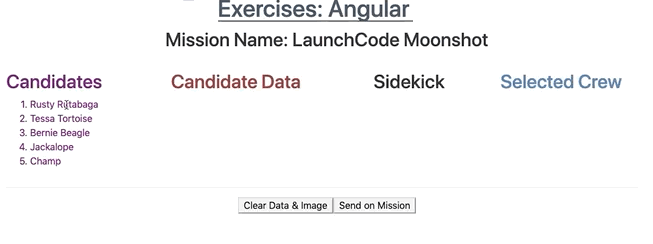
1. Clear Crew List

Add a "Clear Crew List" button under the "Selected Crew" list.

This button should only appear when the crew array contains data. Use \*ngIf to make this happen.

Add a click event that clears the crew array.

Properly done, your output should behave something like this:



(See the final output in the picture, “lesson2-exercises-crew.gif”)

1. Fffg
2. Fgfgfg
3. \fgfgfg
4. Fgfg
5. Fgfg