**Web-App for showing movie reviews**

1. **Requirement:** Building a web-app using angular 2/4 which helps user to see details of ongoing and other movies with ratings in order to identify the quality of the movies. There should a homepage to display the brief info of movie and one other page which will show the detail info of selected movie.
2. **Architecture of the App:** This app is divided into two tiers. Below are these tiers.
3. **Tier 1:-** Containing client side part which include modules, component and services.
4. **Tier 2:**- Containing server side part which include a fake backend for processing user request.
5. **Tier 1 in detail:** This tier is further divided into below parts.
6. **Modules:** There are two modules in this app.
   1. **App Module:** This module is the starting point of this app. It loads other modules and component for uses interface. This module also provides dependencies for different services to be injected. “app.module.ts” is the file used for this module.
   2. **App Routing Module:** This module provides routing to our app. This module deals with the components which are rendered on the browser tab according to the URL passed in the browser. “app-routing.module.ts” is the file used for this module.
7. **Components:** There are three components in this app.
   1. **App Component:** This component is used displaying the navigation bar and rendering other components based on the routing. “app.component.ts, app.component.html, app.component.css,” are the files used by this component.
   2. **Dashboard Component:** This component is used to display all the movies. This component display “on going” movies above the other movies. Here we can click on any movie for the detail description and view rating as well. “dashboard.component.ts, dashboard.component.html, dashboard.component.css,” are the files used by this component.
   3. **Movie Detail Component:** This component is used to display the selected movie in detail with a banner image and rating. “movie-detail.component.ts, movie-detail.component.html, movie-detail.component.css,” are the files used by this component.
8. **Services:** There are two services in this app.
   1. **Dashboard Service:** This service is used to make http request for all the movies. This service is used by “Dashboard Component”. When “Dashboard Component” is loaded, this service’s getMovies() method is called. “dashboard.service.ts” is the file used by this service.
   2. **Movie Detail Service:** This service is used to make http request for a particular movie. While making request an ID is passed to the server to identify the movie, which is requested. This service is used by “Movie Detail Component”. When “Movie Detail Component” is loaded, this service’s getMovieDetail() method is called. “movie-detail.service.ts” is the file used by this service.
9. **Models:**  There is only one model is used in this app.
   1. **Movie:** This model is used for defining the structure for holding data for movie object. “movie.ts” is the file used by this model.
10. **Tier 2 in detail:** This tier contain a fake backend for testing our app. Whenever an http request is made by our services, these request are redirected to this fake backend. It receives those request and matches the URL of the request. On the bases of match performed, this backed serves our request and sends our data in the form of JSON. This backend serves two type requests
11. Request for all movies
12. Request for one movie
13. **Routing in the App:** The job of routing is depends on the App Routing Module. This module used in built “Router Module” for handling routing. We call forRoot() method of the “Router Module” by passing an array of objects, of type Routes.

The individual object in Routes array has two properties set, path and component. Path holds a string which match to the URL and component holds the Component which will be rendered on matching the URL.

In our app, for we have empty path for our Dashboard Component which means that Dashboard Component will be loaded at the root URL. For Movie Detail Component we have ‘movie/:id’ string which means that when URL contains movie, after the root URL, and followed by a slash and any arbitrary string for indicating an id for the request movie, this arbitrary string will be passed as the parameter and can be accessed by the Movie Detail Component.