ANGULAR2

Set up the Development Environment

First we need to install **Nodejs and npm to**  set up our development environment  
Then **install the**[Angular CLI](https://github.com/angular/angular-cli) globally.

1. **npm install -g @angular/cli**
2. Create a new project using below command

* ng new web-app  
  It takes some time to install npm packages

1. Go to the project directory and launch the server using below command  
   -> cd web-app  
   -> ng serve

The ng serve command launches the server, rebuilds the app as you make changes to those files.

1. Open browser and enter  <http://localhost:4200/>

The CLI created the first Angular component and automatically we get Angular2 package structure in which one root component is already there

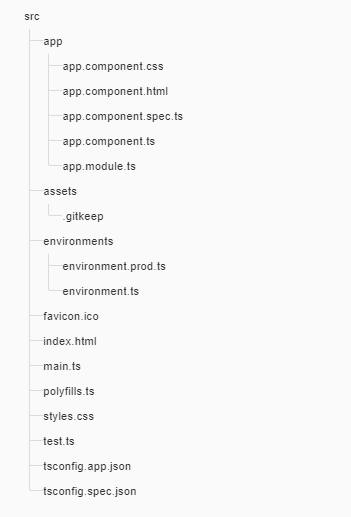
AppComponent-> Root Component

This is the root component and it is named app-root. You can find it in ./src/app/app.component.ts.

Our app lives in the src folder. All Angular components, templates, styles, images, and anything else your app needs go here. Any files outside of this folder are meant to support building our app.

Please find the below npm commands

* run npm install to install dependencies
* npm start to fire up dev server



app/app.component. ->

|  |
| --- |
| Defines the AppComponent along with an HTML template, CSS stylesheet, and a unit test. It is the root component of what will become a tree of nested components as the application evolves. |
|  |

app/app.module.ts -> Defines AppModule, the [root module](https://angular.io/guide/bootstrapping) that tells Angular how to assemble the application. Right now it declares only the AppComponent. Soon there will be more components to declare.

assets/\* -> A folder where you can put images and anything else to be copied wholesale when you build your application.

index.html-> The main HTML page that is served when someone visits your site. Most of the time you'll never need to edit it. The CLI automatically adds all js and css files when building your app so you never need to add any <script> or <link> tags here manually.

main.ts ->   
The main entry point for your app. Compiles the application with the [JIT compiler](https://angular.io/guide/glossary#jit)and bootstraps the application's root module (AppModule) to run in the browser.

node\_modules/ -> Node.js creates this folder and puts all third party modules listed in package.json inside of it.

### App Module

### The app module configures our app its a root module (used to bootstrap the application).App module will declare comment module as an import so that comment module exports can be available to the app module members:

LOCOMOTIVE DATA VISION

So in Locomotive data vision project we created five components

* header Component
* footer Component
* data Component
* data-details Component
* Map-child Component

We have added the component using below command so that we can get all the related files also  
ng g component my-new-component

**Data Component**

This is the component which we see first when we run the application with header and footer component, we make header and footer separate component so that we can manage it throughout the application

In these component we will show the different types of locomotive data through different elements like using Video, Graphs, Table and Google Map

For each element we have integrated external packages with Angular2 application

**Graph**

For Graph we are using the angular2-highcharts which we have integrated to our application, so for using it first we have to install it using npm installation commands

npm install angular2-highcharts –save

then we import it in our component  
import { ChartModule } from 'angular2-highcharts';  
After installation its dependency will automatically inserted in package.json file

**Table**

For Graph we are using the angular2-datatable which we have integrated it and installed it using below npm installation command

npm install angular2-datatable

Then we are importing it in our component

import {DataTableModule} from "angular2-datatable";

**MAP**

For map we are using Angular Google Maps (AGM) to show the latitude and longitude on the map so initially we are showing some position to the user and after that we are fetching it from table .Like when user clicks on a particular row of table then we are fetching the latitude and longitude and displaying it on the google map

Installation->npm install agm/core

**import** { AgmCoreModule } **from** '@agm/core'; //INSIDE THE COMPONENT

We have to add some apiKey also for running this Google MAP

**Data Details Component**

This is the component which user can see by clicking on the File which is there in the navigation header

So in this component user is uploading the file using Select file button then this file we are sending it to the backend for further processing(for converting the .xml file java then JSON object) so that we can populate the file data in the web page

User can select one among three files to upload

* FRA Data
* PTC Data
* Locomotive Video Data

So the upload button will be disabled till user don’t select any radio button ,after selecting the radio button upload button will be enabled and that time we are validating the file to .xml only  
User wont be able to select or upload any other file except .xml after that then this file we are sending it to the backend for further processing

We are calling the REST API using HTTP so in previous version of Angular we handle request using Promises but in this version we can also handle it by Observables.  
So each method takes in a url and a payload as the case may be and returns a generic observable response type

We are importing the required libraries for making HTTP calls. With the map operator, we call the .json method on the response because the actual response is not a collection of data but a JSON string, so we are converting the response data into the JSON so that we can populate it on the client side.



Please see the sample above in which we are calling the method fileUpload then first we are checking the fileList(if it has atleast 1 file then only it will proceed)

So after adding the the file with selected radio button to formData we are calling the REST API “api/locoData” which is taking the .xml file and gives the data , that data we are converting by .map operator into JSON .

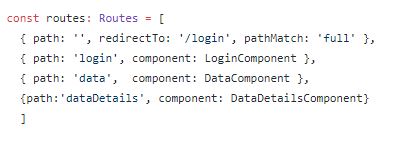
We have to handle exception or errors also so that our application won’t crash if we get any type of exception

ROUTING AND NAVIGATION

### For Angular2 routing first we have to Configure routes. *Routes* tell the router which views to display when a user clicks a link or pastes a URL into the browser address bar. We need to import some built-in packages to work on routing

import { RouterModule } from '@angular/router';  
import { HttpModule } from '@angular/http';  
import { RouterModule,Routes } from '@angular/router';

All the above packages we can import it in app.module.ts so that it will eflect throughout the   
application. Please look over the routing structure of our application



Currently, the browser launches with / in the address bar. When the app starts, it should show the login component after redirecting to login path and display a /login URL in the browser address bar.

Let’s say we need to navigate to data component so for that we have to add some link or button in html file and add routerLink.

<a routerLink="/data">View</a>

So to navigate between two component we can approach the same procedure

SERVICES

Services are generally used to make the code reusable and maintain the readability of the application and Instead of copying and pasting the same code over and over, you'll create a single reusable data service and inject it into the components that need it.

So for using the service we have to inject the particular service by Injectable decorator  
import { Injectable } from '@angular/core'; //import it in service

@Injectable()

export class DataService { }

In our application we are maintaining service class (data.service.ts) also which will call the REST API and give back the response to the components which uses that service

Suppose we need to call the REST API from our component ,for that we can directly call the services by passing the URL and some parameters(if required) then we get the backend response directly from services only . By doing this we can make the code reusable



In the below services we have two methods which will handle API calls and gives response to the component which will use these two methods