Server-Side Rendering in Angular

Ancient website:

- User navigates to url via bar or href, server call is made for that particular page
- The page is returned (either static html or html rendered on server by ASP.NET MVC, etc
- EVERY page reloads everything, slow, reason to go to SPA

Angular SPA:

- User navigates to url via bar or router
- A server call is made for the component's html/javascript
- ONLY the stuff within the router outlet is loaded, not the navbar, etc (main advantage of SPAs)
- HOWEVER, html is not actually received from server as is, Angular 2 code/markup is - then this markup is processed on the CLIENT before it can be displayed as plain HTML, which the browser can understand - SLOW? Enter Angular Universal?

Angular Universal:

First time users of your application will instantly see a server rendered view which greatly improves perceived performance and the overall user experience.

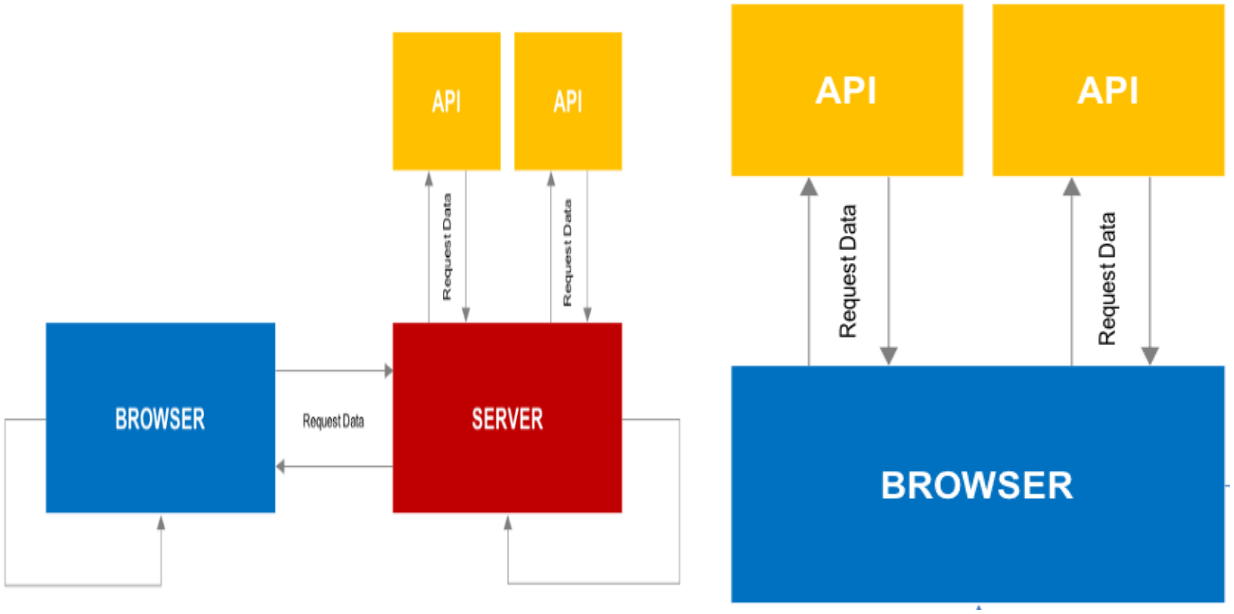
So, in short:

- User navigates to url via search bar or router
- Instead of returning Angular components, Angular Universal actually turns those components into html AND then sends them to the client. This is the ONLY advantage.

What is Angular Universal?

Angular Universal is a technology that renders Angular application on Server-Side Rendering (SSR for short) with Angular. Technically the package is now found under.

SSR vs CSR



SSR **CSR**

Why use server-side rendering?

There are three main reasons to create a Universal version of your app.

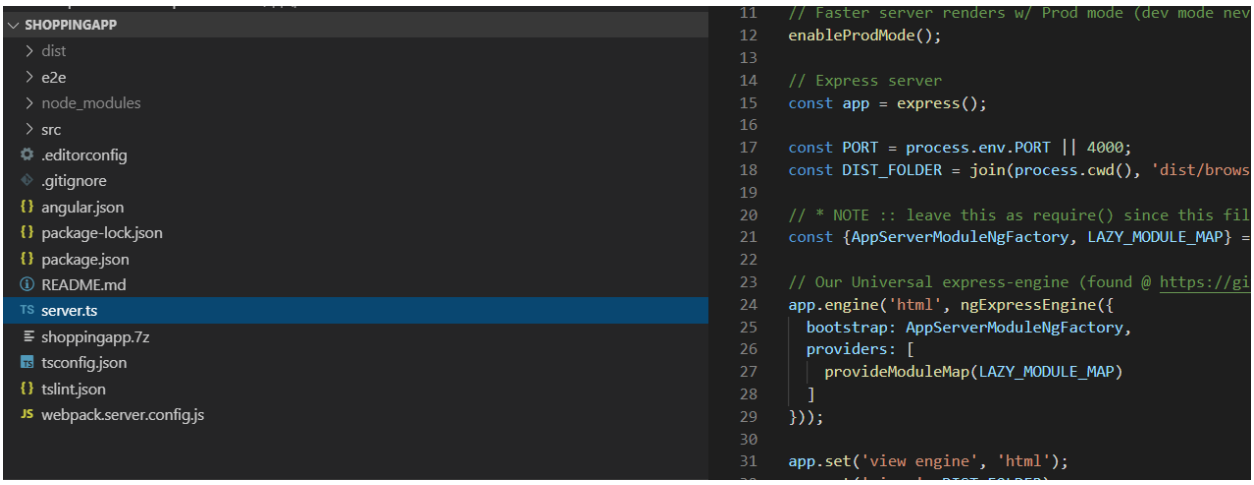
- Facilitate web crawlers through search engine optimization (SEO)
- Improve performance on mobile and low-powered devices
- Show the first page quickly with a first-contentful paint (FCP)

Angular Universal schematic installation

Go to your Angular root project directory where package.json file exist and run below command.

```
ng add @nguniversal/express-engine --clientProject [name]
```

After Running this command it will add server.ts file in our codebase.



How do we run Angular Universal for our App?

Generally, we run angular app using **ng serve** command. To run our code on client side (CSR).

But when we want to run our code on SSR version, we're going to have to use few scripts that were added to our package. json. Before going to that scripts let's discuss to topics of SSR world.

Dynamic SSR & Static Pre-rendering

Dynamic SSR is the concept that there will be a live Node server spun up that whenever a Route is hit, it will dynamically generate and serialize the application — returning that String to the browser.

Static Pre-rendering is when we want to pre-render a list of routes, and create static files, (i.e.: index.html, about-us.html, etc.) and then use a server of our choosing to serve up those files later.

How to start up Angular Universal

Going back to our package. json, we can see we have a few new scripts that were added. There were many new scripts added, but if you take a look below — these will give you both methods of SSR right away!

// Dynamic SSR
npm run build:ssr && npm run serve:ssr

This will compile your application and *spin up a Node Express server* to serve your Universal application on http://localhost:4000

// Static Pre-Rendering
npm run build:prerender && npm run serve:prerender

This script compiles your application and *pre-renders your applications files*, spinning up a demo http-server so you can view it on **http://localhost:8080**

So how do we know it worked?

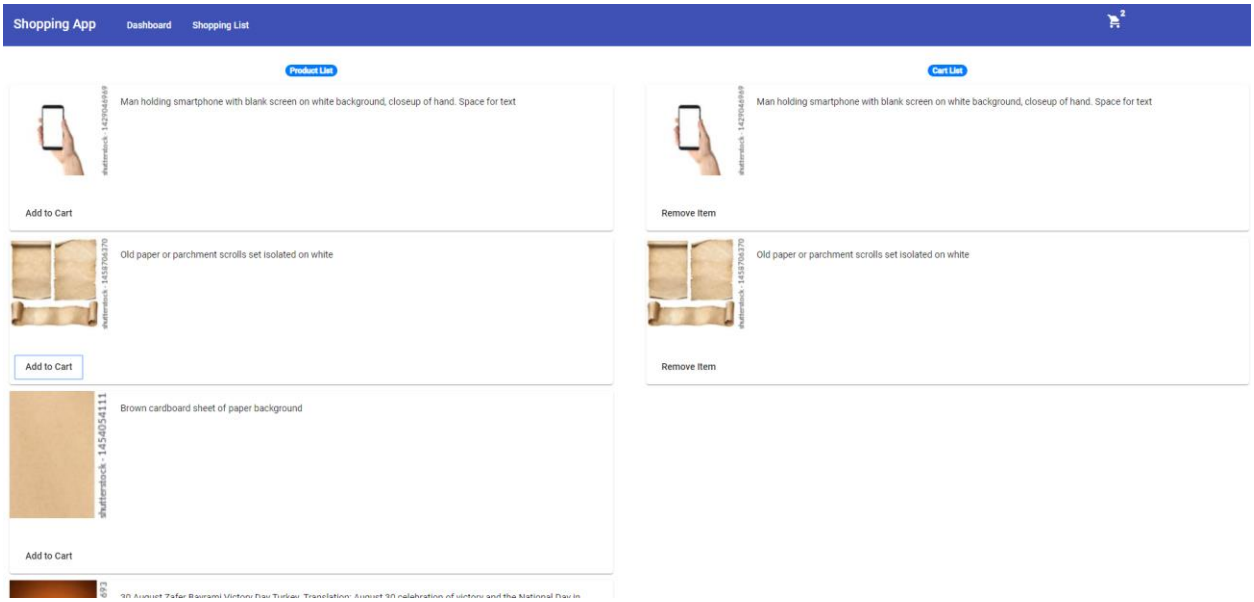
Depending on which version (dynamic/static) you fired up, if you access that localhost URL, you should now see that the typical empty <app-root></app-root> will now have content inside it!

You can set your <title>ShoppingApp- ProductList </title> .
You can set your <meta></meta> tags.

Shopping App on SSR:-

I have created a small application based on cart functionality where we have list of products coming from api and we can add products into cart with routes, material design, Server-Side rendering.

Below I have attached screenshot of Shopping app.



How does our application works Without ssr:-

Here in this screenshot of angular application we can see **<app-root></app-root>** is not having any content or html.

```
<!doctype html>
<html lang="en">
<head>
  <meta charset="utf-8">
  <title>Shoppingapp</title>
  <base href="/">

  <meta name="viewport" content="width=device-width, initial-scale=1">
  <link rel="icon" type="image/x-icon" href="favicon.ico">
  <link href="https://fonts.googleapis.com/css?family=Roboto:300,400,500" rel="stylesheet">
  <link href="https://fonts.googleapis.com/icon?family=Material+Icons" rel="stylesheet">
</head>
<body>
  <app-root></app-root>
<script type="text/javascript" src="runtime.js"></script><script type="text/javascript" src="es2015-polyfills.js" no
type="text/javascript" src="vendor.js"></script><script type="text/javascript" src="main.js"></script></body>
</html>
```

How does our application works With SSR:-

Here in this screenshot of angular application we can see **<app-root></app-root>** is now having content or html.