INTERNSHIP REPORT ON SEARCH ENGINE OPTIMIZATION

Submitted by

Jaya Prakash S P

B. Tech – CSE (3rd Year)

Vellore Institute of Technology

Vellore - 14

Under the guidance of

Mr. Mugilan G

Technology Architect

Mavens-I Software Solution Private Limited.,

Chennai - 96

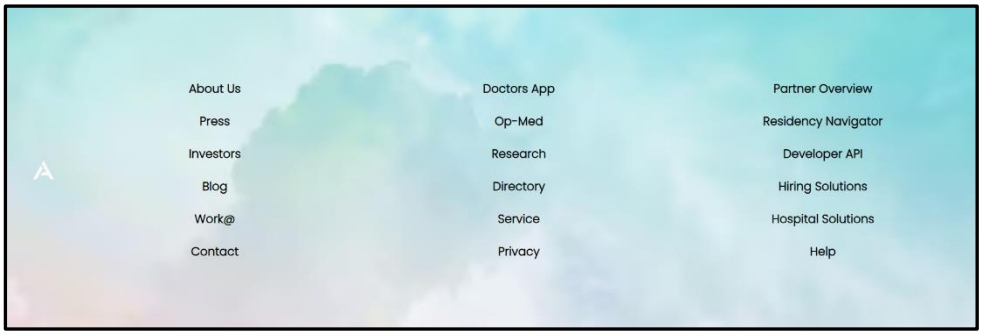
September – November, 2023

CONTENTS

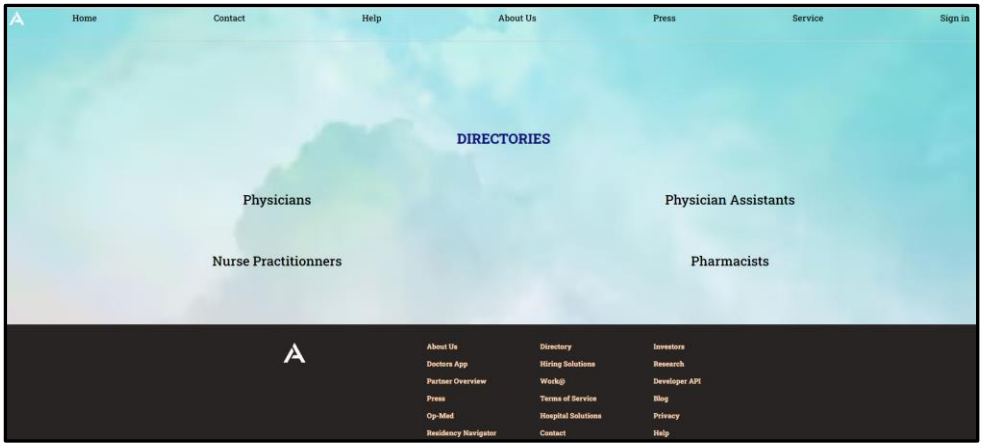
1. Creating a website using Angular:

Initially, I started with the development of 5-page (Home, Directory, Physicians, Anaesthesiology, Physicians-details) website using angular with reference to Doximity existing website (www.doximity.com).

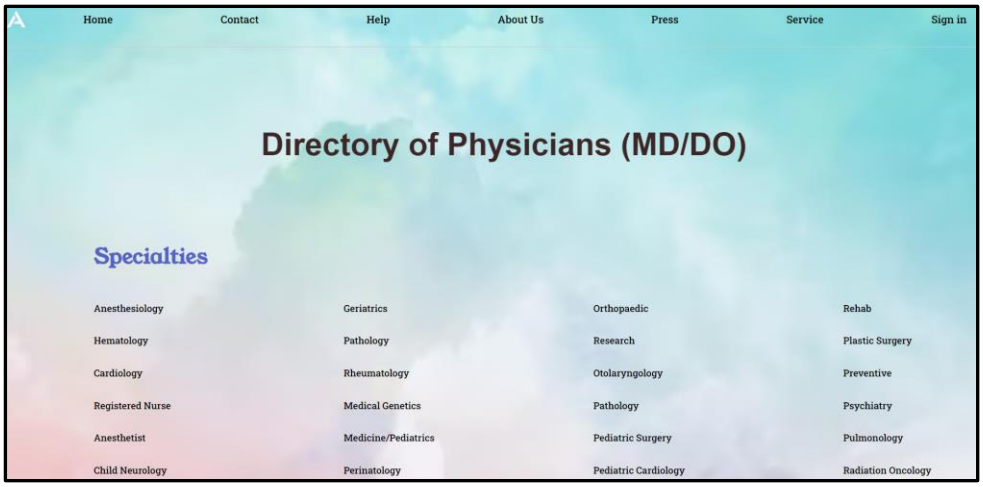
Home page:



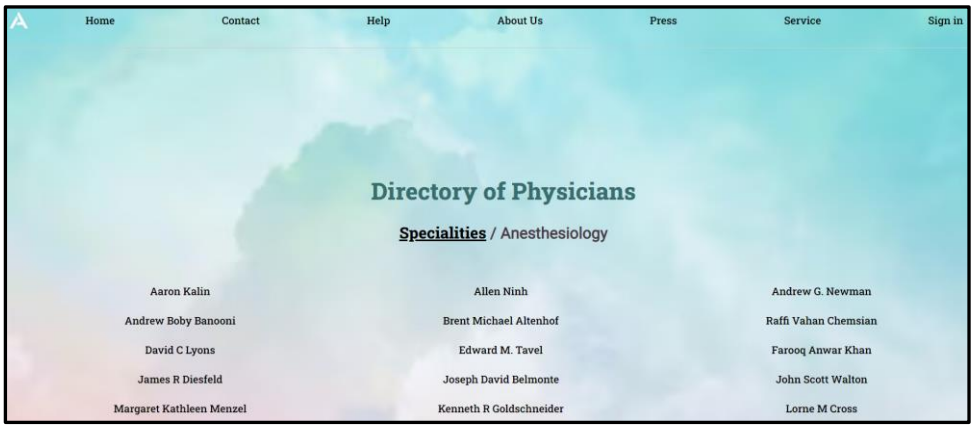
Directory page:



Physicians page:



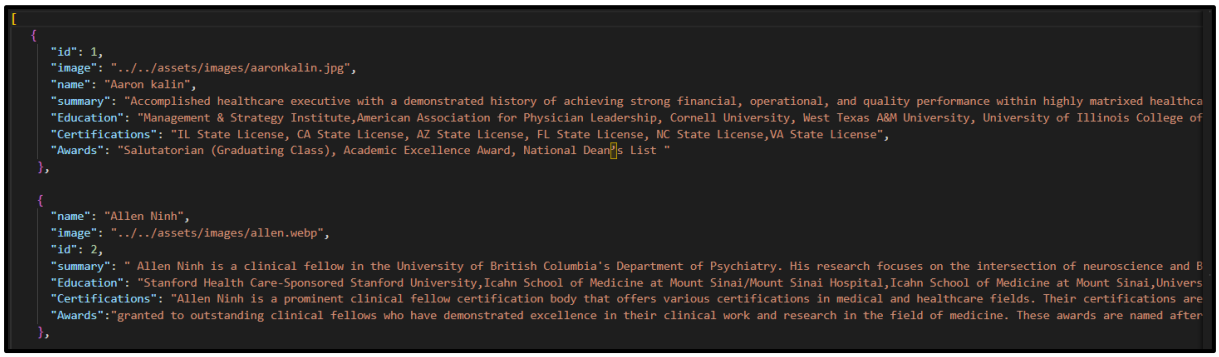
Anaesthesiology page:



Physician-details page:

2. Displaying Doctor's details:

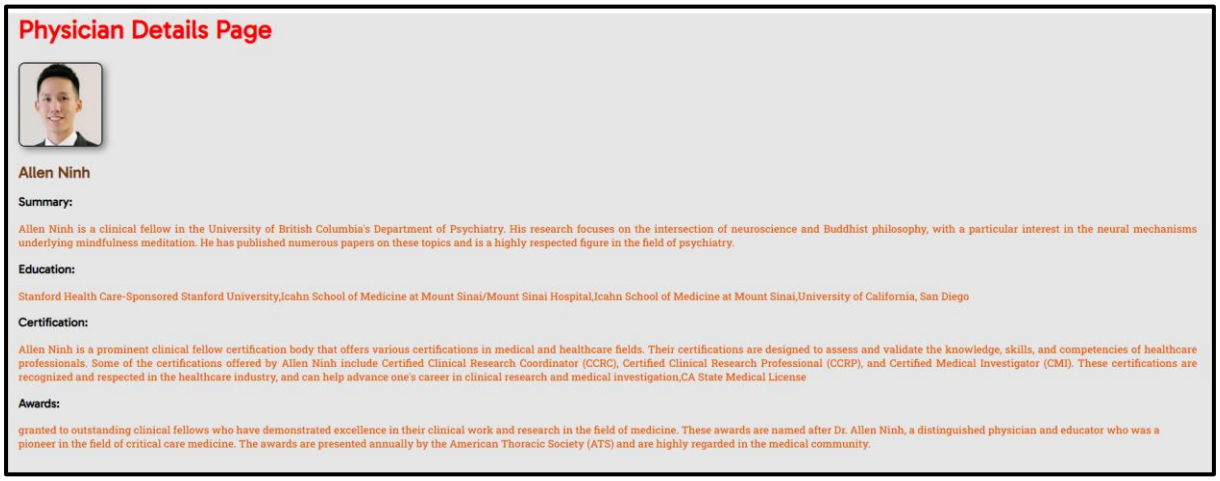
After creating the above 5 pages, I started creating a JSON file and stored the details (Name, Profile pic, Summary, Education, Certification, Award) of the doctors in that JSON file.



After storing the details in the JSON file, I started trying to display the details of the doctors in physician-details page. While selecting the name of the doctors in Anaesthesiology page the details of the specific doctor should be displayed in the physician-details page.

The below mentioned is the initial physician-details page layout created:

While clicking on the doctor's name in Anaesthesiology page the physician-details page looks like given below:



For this, I used to match the id number specified in the JSON file and the page URL. If both JSON and page URL matches the specific details of the doctor will be displayed on the physician-details page.

TypeScript code to display JSON file details in physician-details page:

```
getDetails() {
    this.http.get<any[]>('http://localhost:4200/assets/data/DrList.json').s
ubscribe((doctors) => {
      this.doctors = doctors;
      this.Details = this.doctors.find(doctor => doctor.id ===
this.physicianId);
      console.log(this.Details);
}
```

3. Performing Search Engine Optimization (SEO) in the website:

To make the website SEO-friendly there are many things that we need to take into consideration like Title & Meta tags, Page loading Speed, Optimizing the website responsively and ensure Search Engine Algorithm can crawl and read our page.

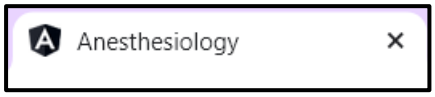
- **Title Tags:**

  Adding Title tag to each page of the website can make the website SEO-friendly. So, I added relevant title tag to the components.

  TypeScript code to add Title tags:

```
export class AnesthesiologyComponent implements OnInit{
  title = 'Anesthesiology';
  constructor( private router: Router,  private titleService:Title) { }

  ngOnInit() {
    this.titleService.setTitle(this.title);

  } }
```

  By adding these comments, we can set the title tag:



  As well as while inspecting the page we can see the title:

- **Meta Tags:**

  Adding Meta Tags using Dynamic method provides information about the web page to Search Engine. This can be improving the visibility and ranking of website in search engine results.

  App-routing.module.ts code to add Meta Tag Dynamically:

```
const routes: Routes = [
  {
    path:'',component: HomeComponent,pathMatch:"full",
    data: {
      title: 'Home',
```

```
      description: 'Welcome to our comprehensive medical platform
offering a quality care.',
      ogTitle: 'Healthcare Innovations: Unveiling Doctors',
      ogDescription: 'Description of Anesthesiology for social media',}
},
  {
    path: 'about-us', component: AboutUsComponent,
    data: {
      title: 'About Us',
      description: 'Title of About Us',
      robots: 'noindex, nofollow',
      ogTitle: 'Explore of About us Component'
    }
}]
```

And also, I need to display the name of the doctor and their details so that Search Engine could address it show to the viewers who are searching for the specific doctor's name and their details.

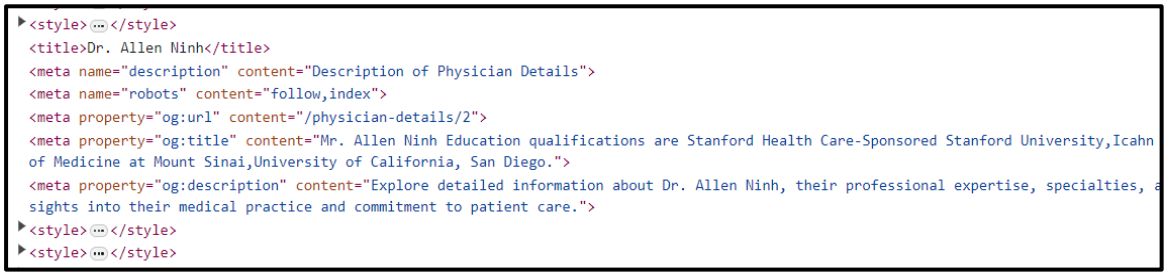
For that, I added the Meta Tag using Meta Service in Angular.

TypeScript code to add doctor's details to Meta Tag using Meta Service:

```
      if (this.Details) {

        this.titleService.setTitle("Dr. " + this.Details.name);

        const meta =
document.querySelector('meta[property="og:title"]') as HTMLMetaElement;
        meta.content = `Mr. ${this.Details.name} Education
qualifications are ${this.Details.Education}.`;;


        const metaDescription =
document.querySelector('meta[property="og:description"]') as
HTMLMetaElement;
       metaDescription.content = `Explore detailed information about Dr.
${this.Details.name}, their professional expertise, specialties, and
experience in providing quality healthcare services. Discover insights
into their medical practice and commitment to patient care. `;

      }
```

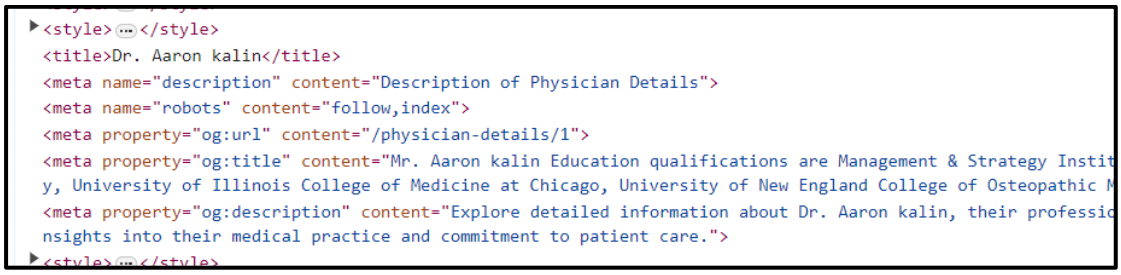
Using this code, we retrieve the name of the doctor's and their details from the JSON file we created and adding it to the meta tag so that it will change according to which doctor we are selecting in anaesthesiology page.

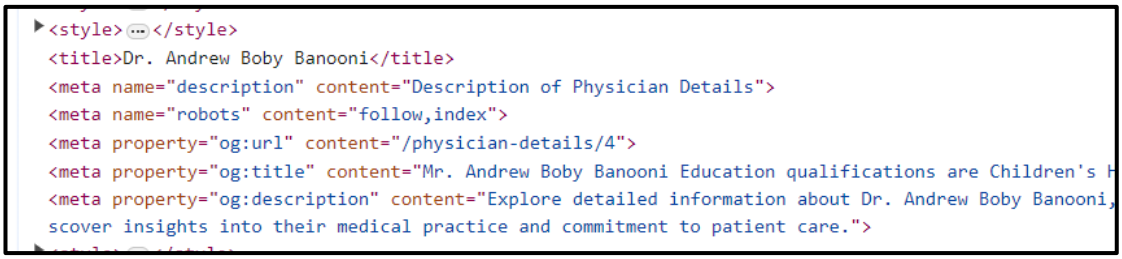
For example, if I select Allen Ninh in Anaesthesiology page it displays:



In this we can see that the name of the doctor is set as the title of the page and other details in the meta tags. This is how we can add the meta tags from the JSON file.

If we select other doctor the meta tag will change accordingly:





4. Server-Side Rendering:

While using Rendering method the user's browser takes the HTML, CSS, JavaScript code and transforms them into visual representation on the user's screen.

- Server-Side Rendering is a technique used to improve the loader speed and Search Engine Optimization (SEO) of the website.
- Search Engine can see the content of our website before sending it to the user.
- SSR display's a fully-rendered HTML page on the first load.

For making our website Server-Side Rendered website at first, we need to install Angular Universal to our application.

Angular Universal is used for Server-Side Rendering and also for Pre-Rendering (Pre-rendering in Angular Universal refers to the process of rendering an Angular application on the Server-Side before it is sent to the client-side).

For installing Angular Universal in our application, we use the following command in terminal:

```
ng add @nguniversal/express-engine@16.2.0
```

In this 16.2.0 refers to the version of angular in your system.

After adding the Angular Universal to the application, we can now execute the bellow command to perform Server-Side Rendering in our website:

```
npm run dev:ssr
```

This command will be specified in your package.json file after installing Angular Universal to your system.

Package.json:

```
{
  "name": "doximity",
  "version": "0.0.0",
  "scripts": {
    "ng": "ng",
    "start": "ng serve --port 1234",
    "build": "ng build",
    "watch": "ng build --watch --configuration development",
    "test": "ng test",
    "dev:ssr": "ng run doximity:serve-ssr",
    "serve:ssr": "node dist/doximity/server/main.js",
    "build:ssr": "ng build && ng run doximity:server",
    "prerender": "ng run doximity:prerender"
  },
```

While executing this command in terminal our website will be now running in Server-Side.

We can Identify this by going to the source page

Before SSR:

```
he wrap
<!doctype html>
<html lang="en">
<head>
  <meta charset="utf-8">
  <!-- <title>home</title> -->
  <base href="/">
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <link rel="icon" type="image/x-icon" href="favicon.ico">
<link rel="stylesheet" href="styles.css"></head>
<body>
  <app-root></app-root>
<script src="runtime.js" type="module"></script><script src="polyfills.js" type="module"></script><scri
</html>
```

After SSR:

```
<body><script id="__bs_script__">//<![CDATA[
  (function() {
    try {
      var script = document.createElement('script');
      if ('async') {
        script.async = true;
      }
      script.src = '/browser-sync/browser-sync-client.js?v=2.29.3'.replace("HOST", location.hostname);
      if (document.body) {
        document.body.appendChild(script);
      } else if (document.head) {
        document.head.appendChild(script);
      }
    } catch (e) {
      console.error("Browsersync: could not append script tag", e);
    }
  })()
//]]></script>

  <app-root _nghost-ng-c632329433 ng-version="16.2.7" ng-server-context="ssr"><router-outlet _ngcontent-ng
  "ng-reflect-ng-for-of": "Home,Contact,Help,About Us,Pre"
}--></div></div><!--bindings={
  "ng-reflect-ng-for-of": "Home,Contact,Help,About Us,Pre"
}--></div></div></header><body _ngcontent-ng-c2195079377><div _ngcontent-ng-c2195079377 class="contents">
<script src="runtime.js" type="module"></script><script src="polyfills.js" type="module"></script><script
```

Now we can see that the HTML code is visible in the source page.

5. Localhost Deployment:

For testing the SEO performance of our website, we need to deploy our website. So that we are going to deploy our website in localhost for temporary purpose. After deploying the website in localhost, we can test the SEO score of the website. If everything works fine, we can proceed to deploy our website.

There are only limited tools to deploy the website in localhost. Even if we deploy in any of the tools there might be a verification page at first. After verification only we will able to see our website.

The drawback of encountering a verification page when entering a website's URL is that when testing the website with SEO tools, only the verification page is considered. Consequently, the SEO score provided is based solely on the verification page. To address this issue, we utilize a tool called Serveo. With Serveo, there is no verification page encountered when entering the URL.

Command to deploy website in localhost (Serveo):

```
ssh -R your-subdomain.serveo.net:80:localhost:4200 serveo.net
```

While running this command in terminal it will provide the URL of the website

```
PS C:\Users\spjp4\OneDrive\Desktop\final doxi> ssh -R your-subdomain.serveo.net:80:localhost:4200 serveo.net
To request a particular subdomain, you first need to generate a key. Use the command
ssh-keygen to generate your key. For more information about generating and using
ssh keys, see https://www.ssh.com/academy/ssh/keygen. Once you've generated a key,
try again, and these instructions will be replaced with instructions on how to
register your key with serveo.

Forwarding HTTP traffic from https://5d159f60fbd28a80f423658ce9daf5dd.serveo.net
```

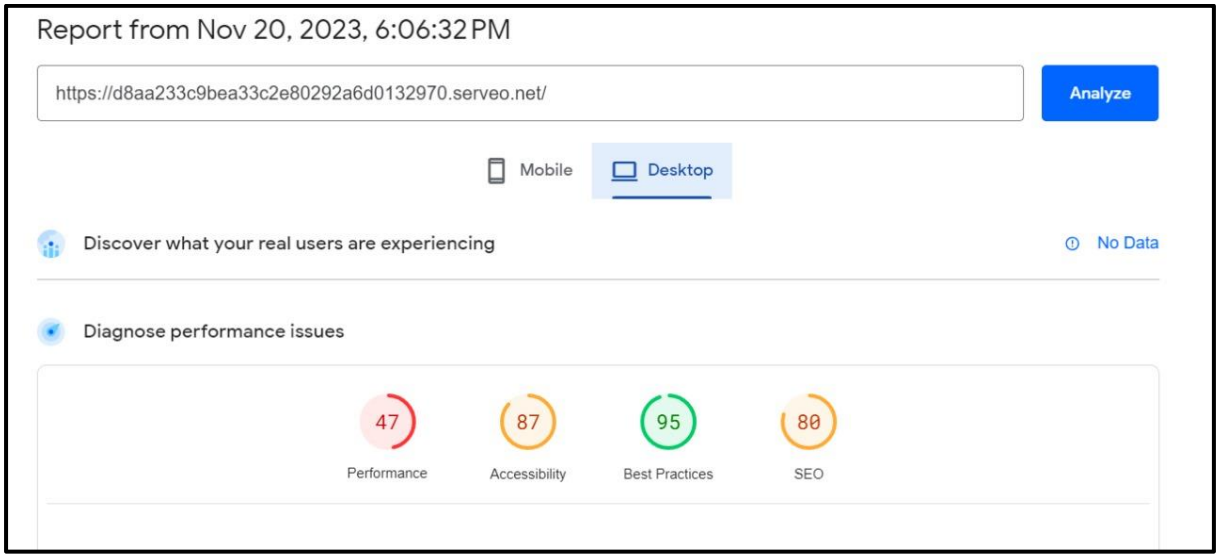
Now we can go to the provided URL and navigate into the website without any verification page.

6. SEO Performance Checking:

After deploying the application in localhost, we can check the performance of the website in Google PageSpeed Insights . It will show the Performance, Accessibility, Best Practices and SEO score of the website.

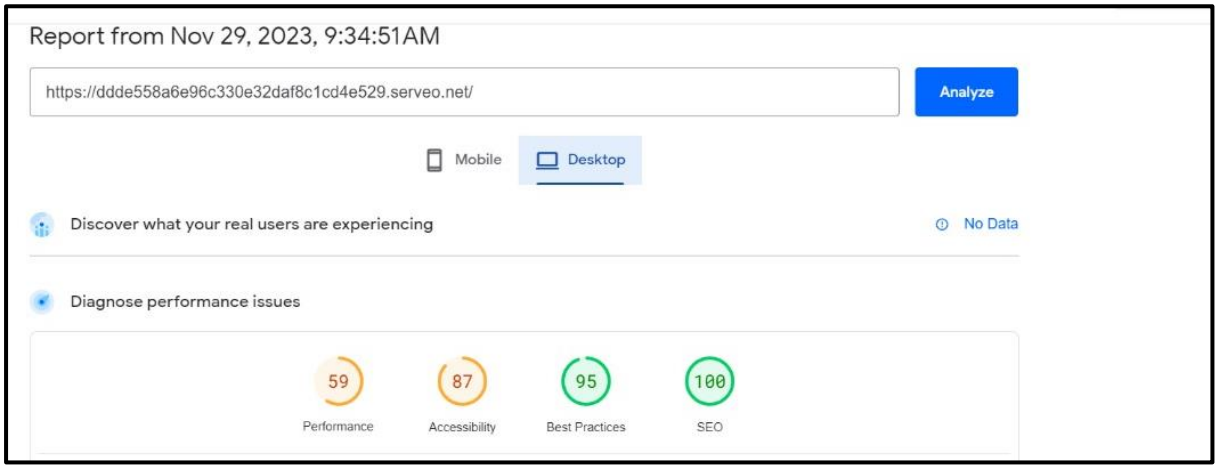
For Testing that we need to paste our page URL in this section:

At first It showed me 80% SEO score:



PageSpeed insights suggested to give the relevant URL in the anaesthesiology page and make the page responsive after doing the changes suggested by the PageSpeed Insights my SEO score increased to 100%.

100% SEO score:



7. Conclusion:

In conclusion, the internship period provided an enriching experience in the field of Search Engine Optimization (SEO). The primary focus was on enhancing the SEO performance of a website developed using Angular. The key aspects covered in this report includes the creation of a comprehensive website, displaying doctor's details dynamically, and implementing SEO strategies for optimal website visibility.

8. References:

- https://angular-university.io
- https://www.tektutorialshub.com/angular-tutorial/
- https://www.w3schools.com/angular/
- https://asperbrothers.com/blog/angular-seo/
- https://aglowiditsolutions.com/blog/angular-seo/
- https://www.youtube.com/watch?v=3EEp4WU7zrk&ab_channel=AngularUniversity
- https://www.searchenginejournal.com/angular-seo-guide/303849/

-----End of the Report----