SUPPORT DOCUMENT

FOR

**Handling Dropdown Voluminous Options**

If there is requirement of displaying a dropdown with tons of options in a web page. We can't load all the options at a time because it is going to take a lot of time which is very bad experience for user of the site.

In this asset we discuss about the solution for the above problem using the primeNG (framework provided) and Angular material (Manual solution).

**Benefits of this Asset:**

User gets to learn about how to handle dropdown having infinite number of options in it. There are two solutions provided: using primeNG framework and Angular material.

User gets to know about the lazy loading, scroll events and directives which can be used in a dropdown. Lazy loading is a technique for waiting to load certain parts of a webpage until they are needed. Instead of loading everything all at once, known as "eager" loading, the browser does not request certain resources until the user interacts in such a way that the resources are needed.

There are two solutions provided: using primeNG framework and Angular material which can be used in their project by following the steps mentioned in the asset.

**Prerequisites to run the application:**

It’s an angular application which runs on node js, so we need to have node JS installed in your system.

You need to have node modules present in your application folder.

Install by using command: **npm install**

You need to have angular framework installed.

Install it by using command: **npm install -g @angular/cli**

Used primeNG and material UI as styling framework for styling the application so you need to have them installed in modules.

Install it by using command: **npm i primeng**

Install it by using command: **ng add @angular/material**

After installing all required dependencies, you need to serve the application to run it.

Run it by using command: **ng serve**.

By default, application will serve at <http://localhost:4200>

**Application explains about:**

Below are the two different ways which we can achieve the above requirement:

**1.Using PrimeNG dropdown:**

<p-dropdown></p-dropdown>

We have properties [virtualScroll]="true" and [virtualScrollItemSize]="5"

Where 5 stands for number of options to be loaded into dropdown:

<p-dropdown [options]="options"

placeholder="Select Item" [virtualScroll]="true"

[virtualScrollItemSize]="5">

</p-dropdown>

**2.Using Angular material dropdown:**

To implement a Dropdown using Angular material, we need to add the required material tags - "mat-form-field" tag in which we define "mat-label" tag to display required label and "mat-select" tag to display a dropdown and "mat-option" tag to display the required options in a dropdown.  
  
Using "infiniteScroll" event which triggers on scroll action of user over the dropdown options, we can implement lazy loading to add next set of options according to the given limit "msInfiniteScroll" directive is required to trigger the "infiniteScroll" event when it reaches the end of dropdown options.

<mat-form-field>

<mat-label>Select Option</mat-label>

<mat-select placeholder="Select an item"

(infiniteScroll)="getNextBatch()" msInfiniteScroll

[complete]="offset === options.length">

<mat-option \*ngFor="let option of options$ | async" [value]="option">

{{option}}

</mat-option>

</mat-select>

</mat-form-field>

**About application**

This application contains three different components.

Components to refer:

dropdown-scroll component

This component basically contains dropdowns.

dropdown-all component

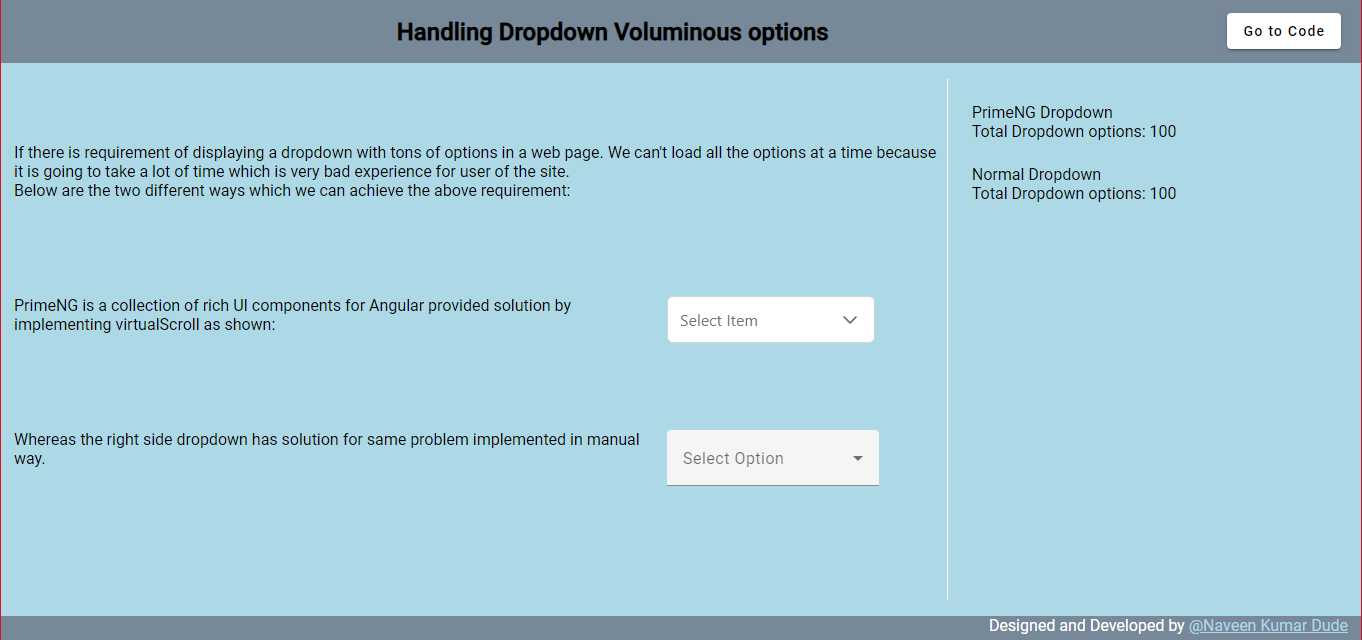
This component contains details of dropdown.

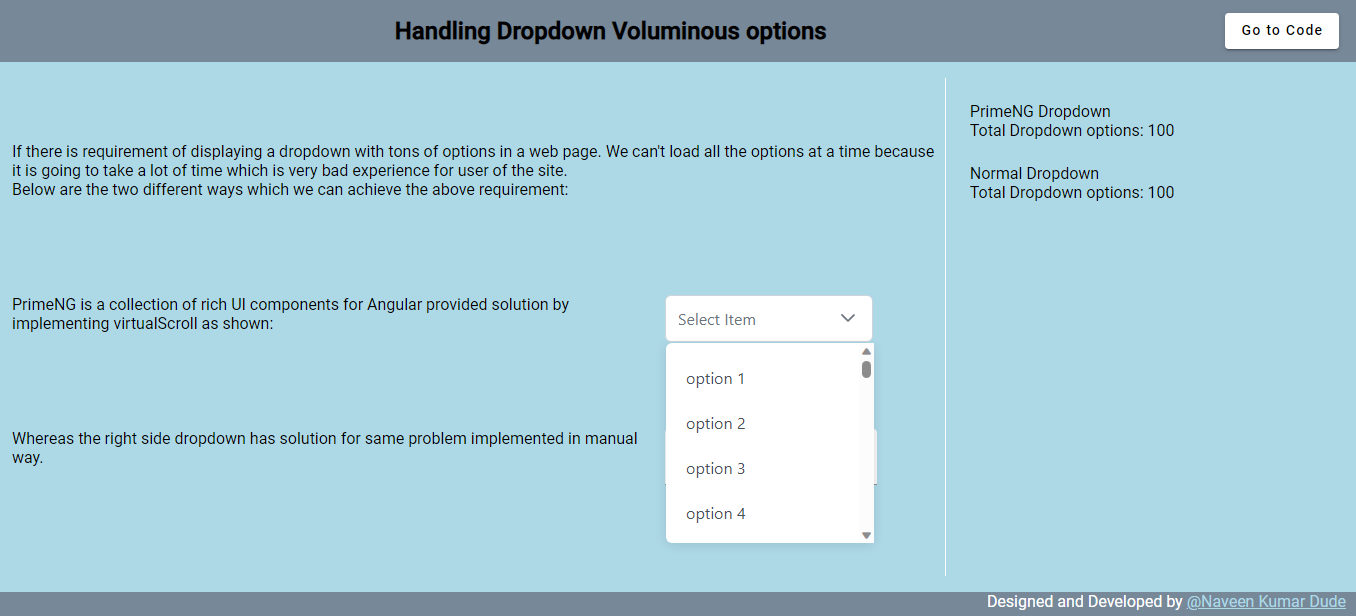
dropdown-code-details component

This component contains code for dropdown and their description.

Snapshots of application are provided in the next page…

Application basically lands on dropdown-scroll component by default:





A screenshot of a computer

Description automatically generated

we can navigate by using ‘Go to code’ button and navigate back by using ‘Home icon’ button in the header of the page.

**Reusability of the dropdowns mentioned in this project:**

Suppose user needs to handle a dropdown of similar kind, they can simply copy

and paste the code I have provided in the asset and import all the necessary modules to run the application.