ANGULAR ASSIGNMENT

1) Write a program to configure routing with lazy loading technique?

App.module.ts:

```
import { NgModule } from '@angular/core';
import { BrowserModule } from '@angular/platform-browser';

import { AppRoutingModule } from './app-routing.module';
import { AppComponent } from './app.component';
```

```
@NgModule({
  declarations: [
    AppComponent
  ],
  imports: [
    BrowserModule,
    AppRoutingModule
  ],
  providers: [],
  bootstrap: [AppComponent]
})
export class AppModule { }
```

App.component.html:

```
<nav>
 <button class="home-button"><a routerLink="/home" class="nav-link">Home</a></button>
 <button class="about-button"> <a routerLink="/about" class="nav-link">About</a></button>
 <button class="contact-button"> <a routerLink="/contact" class="nav-
link">Contact</a></button>
</nav>
<router-outlet></router-outlet>
```

App.component.css :

```css
.nav-container {
    display: flex;
    gap: 20px; /* Space between tabs */
  }

  .nav-link {
    padding: 10px 20px; /* Padding inside the tab */
    text-decoration: none; /* Remove underline from links */
    color: black; /* Text color */
    background-color: rgba(253, 253, 253, 0.726); /* Background color */
    border: 2px solid rgba(238, 238, 241, 0); /* Border properties */
    border-radius: 10px; /* Rounded corners */
    transition: background-color 0.3s ease, box-shadow 0.3s ease; /* Smooth transition for hover effects */
  }

  .nav-link:hover {
    background-color: rgb(78, 77, 77); /* Background color on hover */
    box-shadow: 0px 4px 8px rgba(0, 0, 0, 0.2); /* Shadow effect on hover */
  }

  .nav-link.active {
    background-color: rgb(128, 128, 128); /* Background color for active tab */
    color: white; /* Text color for active tab */
  }



  .about-button {
    background-color: green;
    color: white;
    border: none;
    padding: 10px 20px;
    cursor: pointer;

    border-radius: 15px;

  }

  .contact-button {

    background-color: rgb(238, 7, 7);

    color: white;

    border: none;

    padding: 10px 20px;

    cursor: pointer;

    border-radius: 16px;

  }

  .home-button {

    background-color: blue;

    color: white;

    border: none;

    padding: 10px 20px;

    cursor: pointer;

    border-radius: 16px;

  }
```

App-routing.module.ts :

```
import { NgModule } from '@angular/core';

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


const routes: Routes = [

  {

    path: 'home',

    loadChildren: () => import('./home/home.module').then(m => m.HomeModule)

  },

  {

    path: 'about',
    loadChildren: () => import('./about/about.module').then(m => m.AboutModule)
  },
  {
    path: 'contact',
    loadChildren: () => import('./contact/contact.module').then(m => m.ContactModule)
  },
  { path: '', redirectTo: '/home', pathMatch: 'full' },
  { path: '**', redirectTo: '/home' }
];
```

```
@NgModule({
  imports: [RouterModule.forRoot(routes)],
  exports: [RouterModule]
})
export class AppRoutingModule { }
```

Home-routing.Module.ts :

```
import { NgModule } from '@angular/core';
import { RouterModule, Routes } from '@angular/router';
import { HomeComponent } from './home.component';

const routes: Routes = [
  { path: '', component: HomeComponent }
];
@NgModule({
  imports: [RouterModule.forChild(routes)],
  exports: [RouterModule]
})
export class HomeRoutingModule { }
```

Home .component.html :

```
<h1>Home Page</h1>
<p>Welcome to the home page!</p>
<button class="home-button">Home Button</button>
```

Home.module.ts :

```
import { NgModule } from '@angular/core';
import { CommonModule } from '@angular/common';

import { HomeRoutingModule } from './home-routing.module';
import { HomeComponent } from './home.component';
@NgModule({
  declarations: [
    HomeComponent
  ],
  imports: [
    CommonModule,
    HomeRoutingModule
  ]
})
export class HomeModule { }
```

About-routing.Module.ts :

```
import { NgModule } from '@angular/core';
import { RouterModule, Routes } from '@angular/router';
import { AboutComponent } from './about.component';

const routes: Routes = [{ path: '', component: AboutComponent }];
@NgModule({
  imports: [RouterModule.forChild(routes)],
  exports: [RouterModule]
})
export class AboutRoutingModule { }
```

About .component.html :

```
<h1>About Page</h1>
<p>This is the about page.</p>
<button >About Button</button>
```

About.module.ts :

```
import { NgModule } from '@angular/core';
import { CommonModule } from '@angular/common';

import { AboutRoutingModule } from './about-routing.module';
import { AboutComponent } from './about.component';
@NgModule({
  declarations: [
    AboutComponent
  ],
  imports: [
    CommonModule,
    AboutRoutingModule
  ]
})
export class AboutModule { }
```

Contact-routing.Module.ts :

```
import { NgModule } from '@angular/core';
import { RouterModule, Routes } from '@angular/router';
import { ContactComponent } from './contact.component';

const routes: Routes = [{ path: '', component: ContactComponent }];
@NgModule({
  imports: [RouterModule.forChild(routes)],
  exports: [RouterModule]
})
export class ContactRoutingModule { }
```

Contact.component.html :

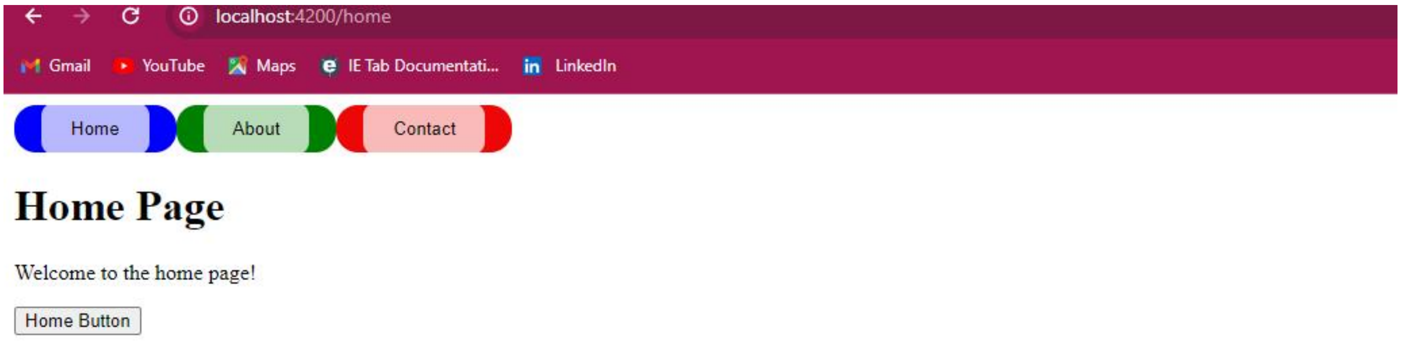
```
<h1>Contact Page</h1>
<p>Get in touch with us.</p>
<button class="contact-button">Contact Button</button>
```

Contact.module.ts :

```
import { NgModule } from '@angular/core';
import { CommonModule } from '@angular/common';

import { ContactRoutingModule } from './contact-routing.module';
import { ContactComponent } from './contact.component';
@NgModule({
  declarations: [
    ContactComponent
  ],
  imports: [
    CommonModule,
    ContactRoutingModule
  ]
})
export class ContactModule { }
```

Output:

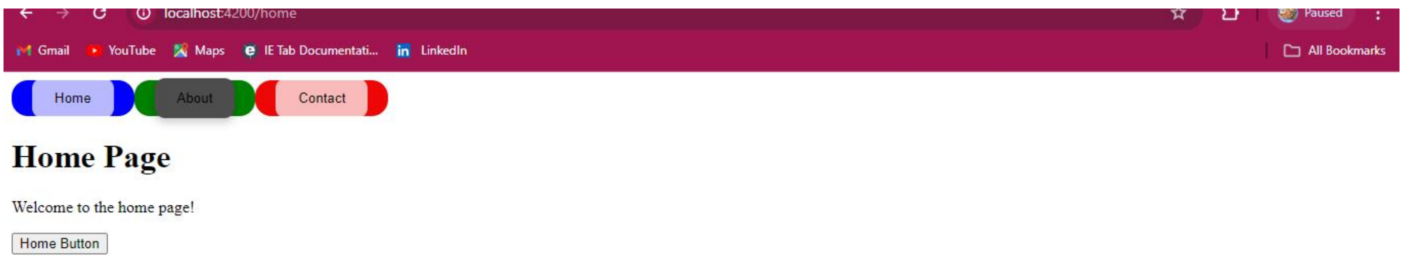
Home tab ;



Home Page

Welcome to the home page!

About.Tab :

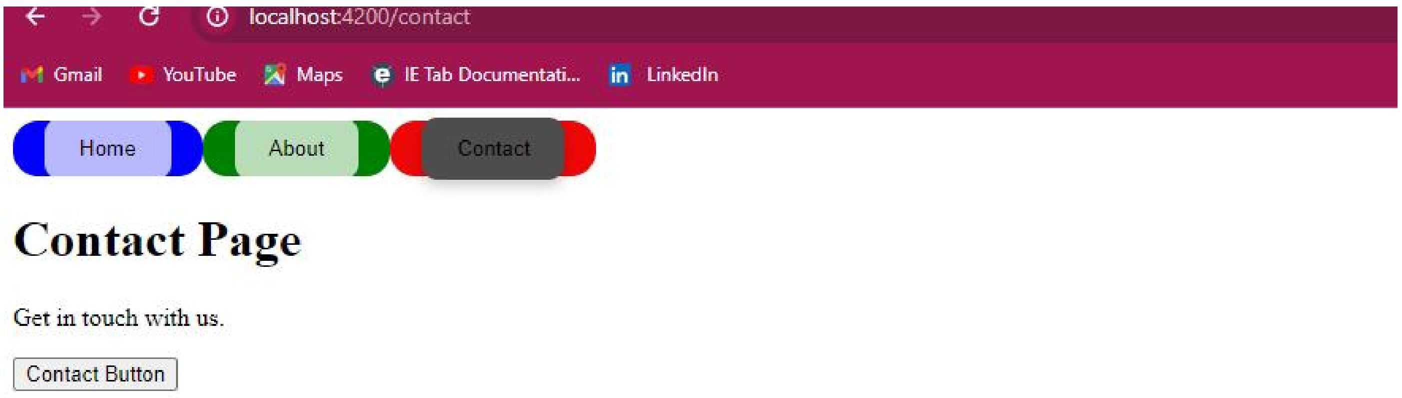


Home Page

Welcome to the home page!

Home Button

Contact.tab ;



Contact Page

Get in touch with us.

Contact Button

2) Write a program to create user with address fields with validation using reactive forms?

Program;

app.module.ts;

```
import { NgModule } from '@angular/core';
import { ReactiveFormsModule } from '@angular/forms';
import { BrowserModule } from '@angular/platform-browser';
import { BrowserAnimationsModule } from '@angular/platform-browser/animations';
import { MatInputModule } from '@angular/material/input';
import { MatFormFieldModule } from '@angular/material/form-field';
import { MatButtonModule } from '@angular/material/button';

import { AppComponent } from './app.component';
import { UserFormComponent } from './user-form/user-form.component';
```

```
@NgModule({
  declarations: [
    AppComponent,
    UserFormComponent
  ],
  imports: [
    BrowserModule,
    ReactiveFormsModule,
    BrowserAnimationsModule,
    MatInputModule,
    MatFormFieldModule,
    MatButtonModule
  ],
  providers: [],
  bootstrap: [AppComponent]
})
export class AppModule { }
```

App.component.html;

```

<app-user-form></app-user-form>
```

User-form.component.html;

```
<h1 >User Address Form </h1>
<form [formGroup]="userForm" (ngSubmit)="onSubmit()">
  <mat-form-field appearance="fill">
    <mat-label>Name</mat-label>
    <input matInput formControlName="name"  placeholder="manjula"/>
    <mat-error *ngIf="name?.invalid && (name?.dirty || name?.touched)">
      <div *ngIf="name?.errors?.['required']">Name is required.</div>
      <div *ngIf="name?.errors?.['minlength']">Name must be at least 3 characters
long.</div>
    </mat-error>
  </mat-form-field>

  <mat-form-field appearance="fill">
```

```

    <mat-label>Email</mat-label>
    <input matInput formControlName="email"  placeholder="abc@gmail.com"/>
    <mat-error *ngIf="email?.invalid && (email?.dirty || email?.touched)">
      <div *ngIf="email?.errors?.['required']">Email is required.</div>
      <div *ngIf="email?.errors?.['email']">Email must be a valid email address.</div>
    </mat-error>
  </mat-form-field>
```

```
  <div formGroupName="address">
    <mat-form-field appearance="fill">
      <mat-label>Street</mat-label>
      <input matInput formControlName="street" placeholder="North" />
      <mat-error *ngIf="street?.invalid && (street?.dirty || street?.touched)">
        <div *ngIf="street?.errors?.['required']">Street is required.</div>
      </mat-error>
    </mat-form-field>
```

```
    <mat-form-field appearance="fill">
      <mat-label>City</mat-label>
      <input matInput formControlName="city"  placeholder="Attur"/>
      <mat-error *ngIf="city?.invalid && (city?.dirty || city?.touched)">
        <div *ngIf="city?.errors?.['required']">City is required.</div>
      </mat-error>
    </mat-form-field>
```

```
    <mat-form-field appearance="fill">
      <mat-label>State</mat-label>
      <input matInput formControlName="state" placeholder="Bangalore"/>
      <mat-error *ngIf="state?.invalid && (state?.dirty || state?.touched)">
        <div *ngIf="state?.errors?.['required']">State is required.</div>
      </mat-error>
    </mat-form-field>
```

```
    <mat-form-field appearance="fill">
      <mat-label>Zip</mat-label>
      <input matInput formControlName="zip" placeholder="000000"/>
      <mat-error *ngIf="zip?.invalid && (zip?.dirty || zip?.touched)">
        <div *ngIf="zip?.errors?.['required']">Zip is required.</div>
        <div *ngIf="zip?.errors?.['pattern']">Zip must be a valid 6-digit code.</div>
      </mat-error>
    </mat-form-field>
  </div>
```

```
  <button mat-raised-button type="submit" [disabled]="userForm.invalid">Submit</button>
</form>
```

User-form.component.html;

```
<h1 >User Address Form </h1>
<form [formGroup]="userForm" (ngSubmit)="onSubmit()">
  <mat-form-field appearance="fill">
    <mat-label>Name</mat-label>
    <input matInput formControlName="name"  placeholder="manjula"/>
    <mat-error *ngIf="name?.invalid && (name?.dirty || name?.touched)">
      <div *ngIf="name?.errors?.['required']">Name is required.</div>
      <div *ngIf="name?.errors?.['minlength']">Name must be at least 3 characters
long.</div>
    </mat-error>
  </mat-form-field>

  <mat-form-field appearance="fill">
```

```


    <mat-label>Email</mat-label>
    <input matInput formControlName="email"  placeholder="abc@gmail.com"/>
    <mat-error *ngIf="email?.invalid && (email?.dirty || email?.touched)">
      <div *ngIf="email?.errors?.['required']">Email is required.</div>
      <div *ngIf="email?.errors?.['email']">Email must be a valid email address.</div>
    </mat-error>
  </mat-form-field>
```

```
  <div formGroupName="address">
    <mat-form-field appearance="fill">
      <mat-label>Street</mat-label>
      <input matInput formControlName="street" placeholder="North" />
      <mat-error *ngIf="street?.invalid && (street?.dirty || street?.touched)">
        <div *ngIf="street?.errors?.['required']">Street is required.</div>
      </mat-error>
    </mat-form-field>
```

```
    <mat-form-field appearance="fill">
      <mat-label>City</mat-label>
      <input matInput formControlName="city"  placeholder="Attur"/>
      <mat-error *ngIf="city?.invalid && (city?.dirty || city?.touched)">
        <div *ngIf="city?.errors?.['required']">City is required.</div>
      </mat-error>
    </mat-form-field>
```

```
    <mat-form-field appearance="fill">
      <mat-label>State</mat-label>
      <input matInput formControlName="state" placeholder="Bangalore"/>
      <mat-error *ngIf="state?.invalid && (state?.dirty || state?.touched)">
        <div *ngIf="state?.errors?.['required']">State is required.</div>
      </mat-error>
    </mat-form-field>
```

```
    <mat-form-field appearance="fill">
      <mat-label>Zip</mat-label>
      <input matInput formControlName="zip" placeholder="000000"/>
      <mat-error *ngIf="zip?.invalid && (zip?.dirty || zip?.touched)">
        <div *ngIf="zip?.errors?.['required']">Zip is required.</div>
        <div *ngIf="zip?.errors?.['pattern']">Zip must be a valid 6-digit code.</div>
      </mat-error>
    </mat-form-field>
  </div>
```

```
  <button mat-raised-button type="submit" [disabled]="userForm.invalid">Submit</button>
</form>
```

User-form.component.ts ;

```
import { Component, OnInit } from '@angular/core';
import { FormBuilder, FormGroup, Validators } from '@angular/forms';

@Component({
  selector: 'app-user-form',
  templateUrl: './user-form.component.html',
  styleUrls: ['./user-form.component.css']
})
export class UserFormComponent implements OnInit {
  userForm: any[string]=FormGroup;
```

```
  constructor(private fb: FormBuilder) { }
```

```
  ngOnInit(): void {
    this.userForm = this.fb.group({
      name: ['', [Validators.required, Validators.minLength(3)]],
      email: ['', [Validators.required, Validators.email]],
      address: this.fb.group({
        street: ['', Validators.required],
        city: ['', Validators.required],
        state: ['', Validators.required],
        zip: ['', [Validators.required, Validators.pattern('^[0-9]{6}$')]]
      })
    });
  }
```

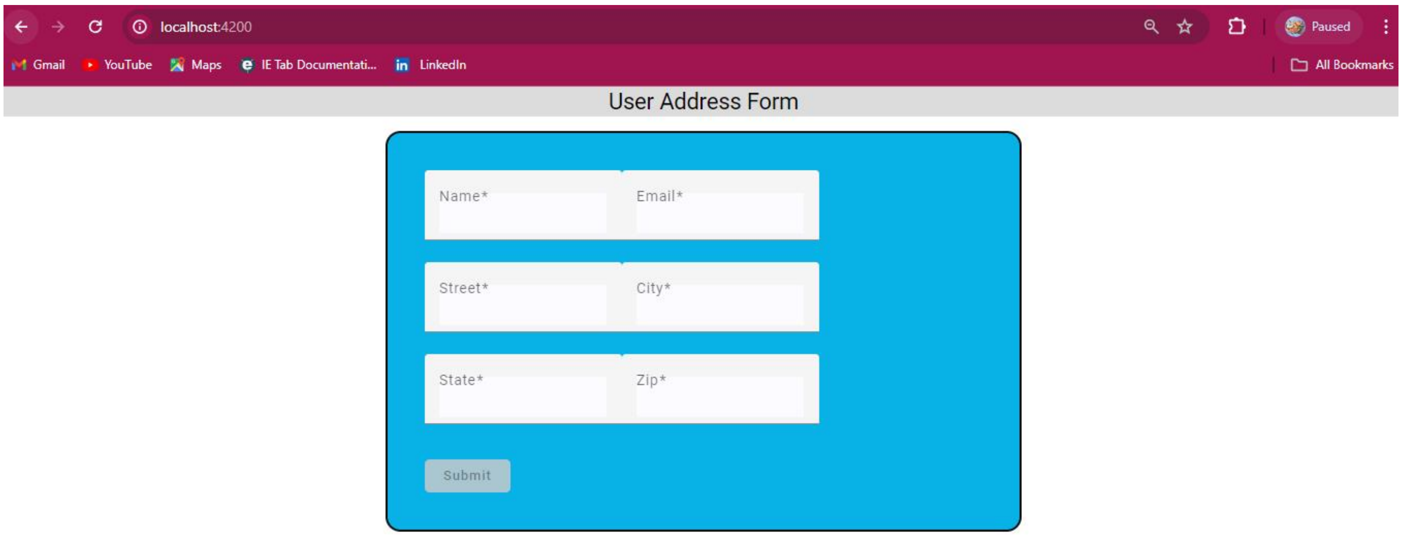
```
  onSubmit(): void {
    if (this.userForm.valid) {
      console.log('Form Submitted', this.userForm.value);
    }
  }
```

```
  get name() { return this.userForm.get('name'); }
  get email() { return this.userForm.get('email'); }
  get street() { return this.userForm.get('address.street'); }
  get city() { return this.userForm.get('address.city'); }
  get state() { return this.userForm.get('address.state'); }
  get zip() { return this.userForm.get('address.zip'); }
}
```

User-form.component.css;

```css
form {
  max-width: 600px;
  margin: 0 auto;
  padding: 40px;
  border: 3px solid #080a0c;
  border-radius: 16px;
  background: #09b2e6;
 }

 h1{
   color: #080a0c bold;
   text-align: center;
   background-color: gainsboro;
   font-size: x-large;
 }
 div {
  margin-bottom: 15px;
 }

 label {
  display:flex;
  margin-bottom: 15px;
  border-radius:5px ;
  border-color: #9c0995;
  border-width: 40%;
 }

 input {
  width: 100%;
  padding: 10px;
  box-sizing: border-box;
  background-color: #fcfbff;
  align-self: auto;
 }

 .error {
  color: red;
  font-size: 0.875em;
 }

 button{
  padding: 10px 20px;
  color: rgb(14, 1, 1);
  border: none;
  border-radius: 5px;
  cursor: pointer;
 }

 button[disabled] {
  background-color: #ccccccd2;
  cursor: not-allowed;
 }
```

Output;



User Address Form

Name*

Email*

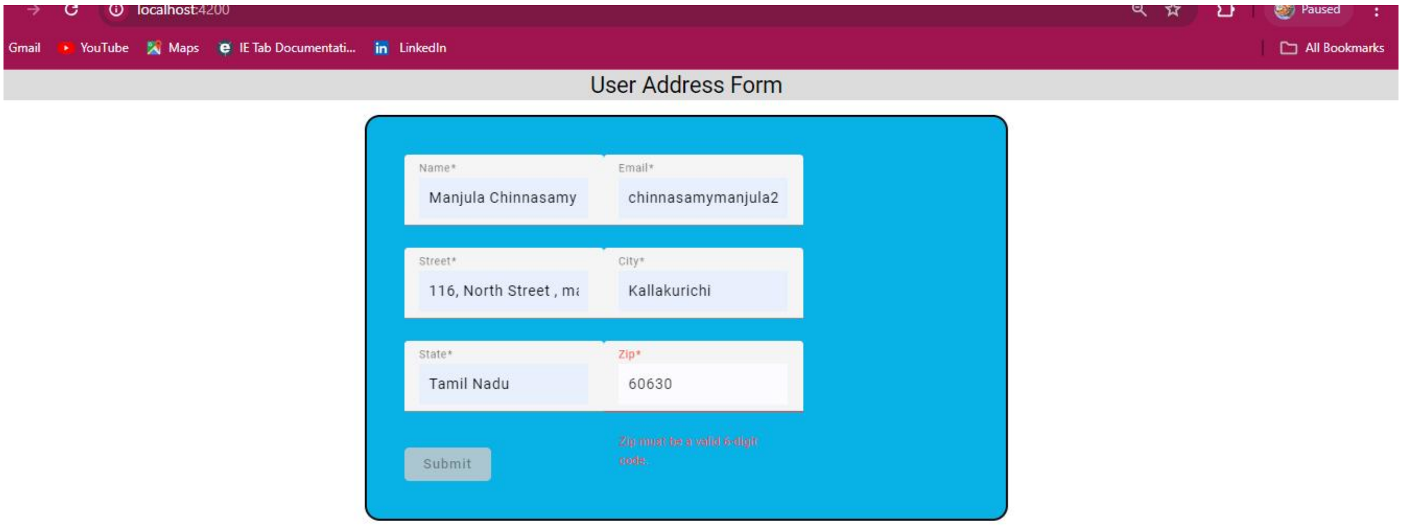
Street*

City*

State*

Zip*

Submit



User Address Form

Name* Manjula Chinnasamy

Email* chinnasamymanjula2

Street* 116, North Street , ma

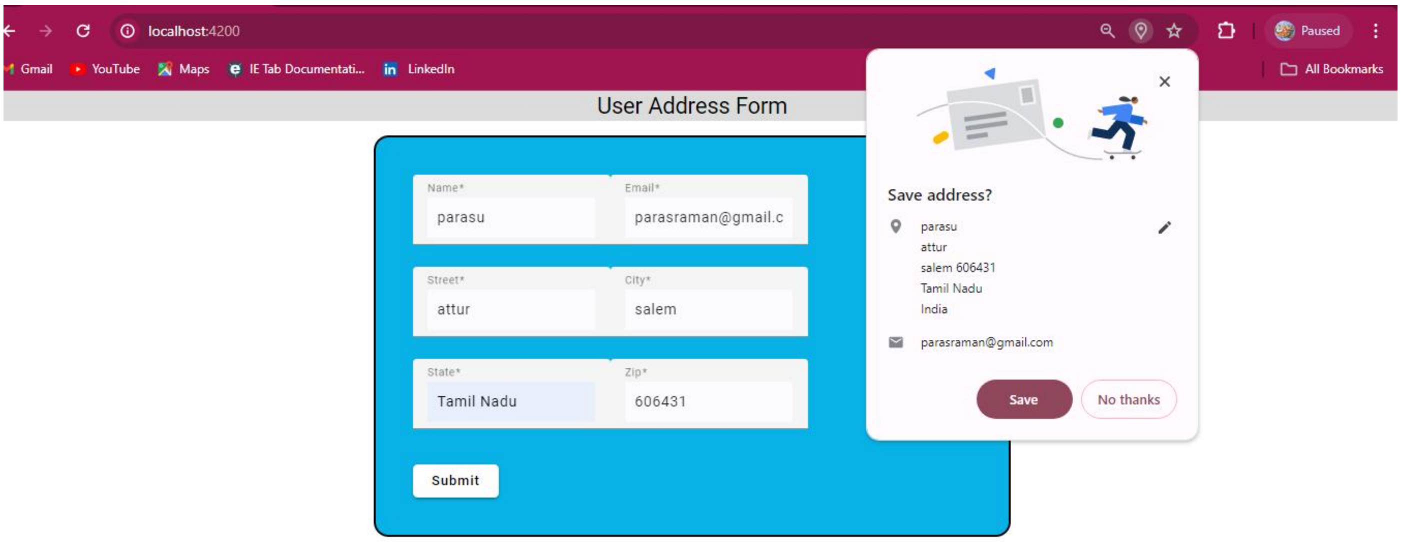
City* Kallakurichi

State* Tamil Nadu

Zip* 60630

Zip must be a valid 6-digit code.

Submit



User Address Form

Name* parasu

Email* parasraman@gmail.c

Street* attur

City* salem

State* Tamil Nadu

Zip* 606431

Submit

Save address?

parasu
attur
salem 606431
Tamil Nadu
India

parasraman@gmail.com

Save No thanks

3) Write a program to add and retrieve super heroes using services?

Program:

App.component.ts ;

```
// app.component.ts
import { Component } from '@angular/core';

@Component({
  selector: 'app-root',
  template: `
    <h1>Welcome to Superhero Tracker!</h1>
    <app-add-hero></app-add-hero>
    <app-display-heroes></app-display-heroes>
  `
})
export class AppComponent {}
```

App.component.html ;

```
<!-- src/app/app.component.html -->
<div>
<h1>Superhero App</h1>

    <app-add-hero></app-add-hero>
    <app-display-heroes> </app-display-heroes>
</div>
```

App.component.css ;

```
/* styles.css */

/* Global Styles */
div {
    font-family: Arial, sans-serif;
    margin: 0;
    padding: 0;
    background-color:rgb(80, 176, 240);
```

```
  }


  /* Header Styles */
  h1 {
    background-color: #333;
    color: #fff;
    padding: 20px;
    text-align: center;
  }
```

App.module.ts ;

```

import { NgModule } from '@angular/core';
import { BrowserModule } from '@angular/platform-browser';
import { CommonModule } from '@angular/common'; // Import CommonModule
import { AppComponent } from './app.component';
import { AddHeroComponent } from './add-hero/add-hero.component';
import { DisplayHeroesComponent } from './display-heroes/display-heroes.component';
import { FormsModule } from '@angular/forms';
```

```
@NgModule({
  declarations: [
    AppComponent,
    AddHeroComponent,
    DisplayHeroesComponent
  ],
  imports: [
    BrowserModule,
    CommonModule, // Include CommonModule in imports
    FormsModule
  ],
  providers: [],
  bootstrap: [AppComponent]
})
export class AppModule { }
```

Hero.model.ts ;

```
export interface Hero {

  id: number;

  name: string;

  powers: string[];

}
```

Hero.service.ts ;

```
import { Injectable } from '@angular/core';

import { Hero } from './hero.model';


@Injectable({

  providedIn: 'root'

})

export class HeroService {

  private heroes: Hero[] = [];

  constructor() { }

  addHero(hero: Hero) {

    this.heroes.push(hero);

  }

  getHeroes(): Hero[] {

    return this.heroes;

  }

}
```

Add-hero.component.ts;

```
// add-hero.component.ts

import { Component } from '@angular/core';

import { HeroService } from '../hero.service';

import { Hero } from '../hero.model';


@Component({

  selector: 'app-add-hero',
  templateUrl: './add-hero.component.html',
  styleUrls: ['./add-hero.component.css']
})
export class AddHeroComponent {
  constructor(private heroService: HeroService) {}
```

```
  addHero(name: string, powers: string[]) {
    const id = this.heroService.getHeroes().length + 1; // Generate unique ID
    const hero: Hero = { id, name, powers };
    this.heroService.addHero(hero);
  }
}
```

Add-hero.component.html;

```
<!-- add-hero.component.html -->
<div class="form-container">
    <h2>Add Hero</h2>
    <div class="form-group">
      <label>Name:
        <input type="text" #name>
      </label>
    </div>
    <div class="form-group">
      <label>Powers:
        <input type="text" #powers> </label>
    </div>
    <button (click)="addHero(name.value, powers.value.split(','))">Add Hero</button>
  </div>
```

Add-hero.component.css;

```
/* Form Styles */
.form-container {
```

```css
  background-color: #64b6ec;
  padding: 20px;
  border-radius: 10px;
  box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);
}

.form-group {
  margin-bottom: 20px;
}

.form-group label {
  display: block;
  font-weight: bold;
}

.form-group input[type="text"],
.form-group input[type="password"],
.form-group textarea {
  width: 70%;
  padding: 10px;
  border: 1px solid #9b6e6ea9;
  border-radius: 10px;
  box-sizing: border-box;
  background-color:rgb(202, 194, 194);
}

.form-group button {
  padding: 10px 20px;
  background-color: #246ac5;
  color: #fff;
  border: none;
  border-radius: 5px;
  cursor: pointer;
}
```

```
  .form-group button:hover {
    background-color: #0056b3;
  }
  button{
    background-color:rgba(247, 242, 242, 0.781);
    color:black;
    border-radius: 5px;
  }
```

Display-heroes.component.ts;

```
// display-heroes.component.ts
import { Component } from '@angular/core';
import { HeroService } from '../hero.service';
import { Hero } from '../hero.model';

@Component({
  selector: 'app-display-heroes',
  templateUrl: './display-heroes.component.html',
  styleUrls: ['./display-heroes.component.css']
})
export class DisplayHeroesComponent {
  heroes: Hero[] = [];
```

```
  constructor(private heroService: HeroService) {
    this.heroes = this.heroService.getHeroes();
  }
}
```

Display-heroes.component.html;

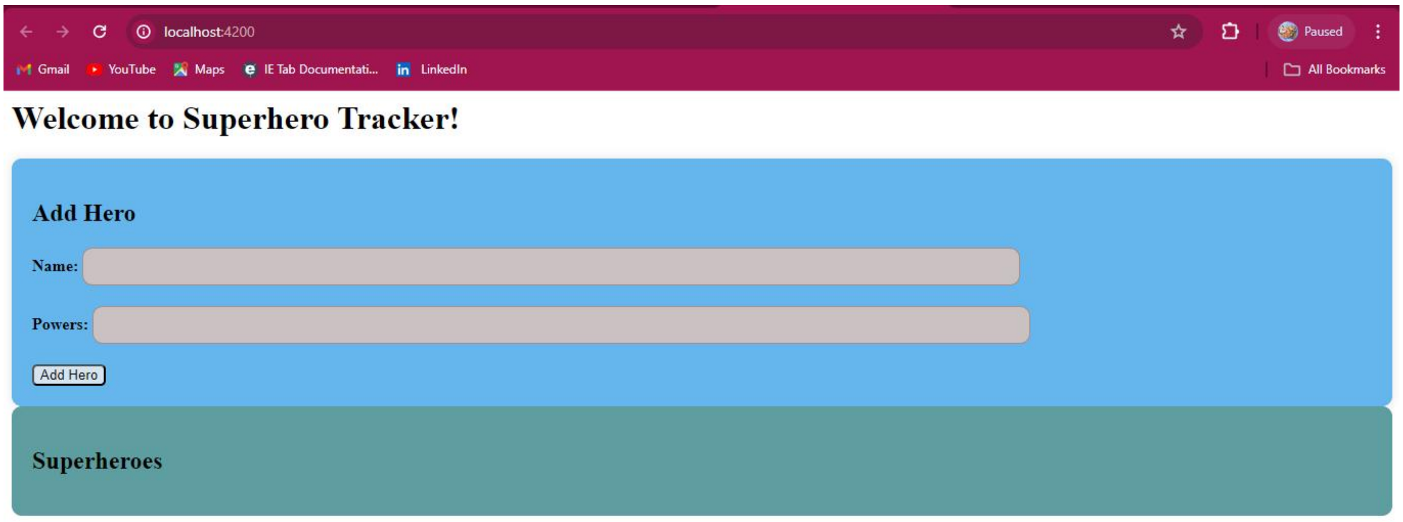
```
<!-- display-heroes.component.html -->
<div class="container">
```

```
    <h2>Superheroes</h2>

    <ul class="hero-list">

      <li *ngFor="let hero of heroes">

        <strong>{{hero.name}}</strong> - Powers: {{hero.powers.join(', ')}}

      </li>

    </ul>

  </div>
```

Display-heroes.component.css;

```
/* Hero List Styles */

.hero-list {

    list-style-type: none;

    padding: 0;

  }


  .hero-list li {

    margin-bottom: 10px;

    padding: 10px;

    background-color: #e4a78f;

    border-radius: 5px;

    box-shadow: 0 2px 4px rgba(0, 0, 0, 0.1);

  }

  .hero-list li:hover {

    background-color: #eaeaea;

  }

  /* Container Styles */

  .container {

    max: width 100%;;

    margin: auto;

    padding: 20px;

    background-color: cadetblue;

    border-radius: 10px;

  }
```

Output:



Welcome to Superhero Tracker!

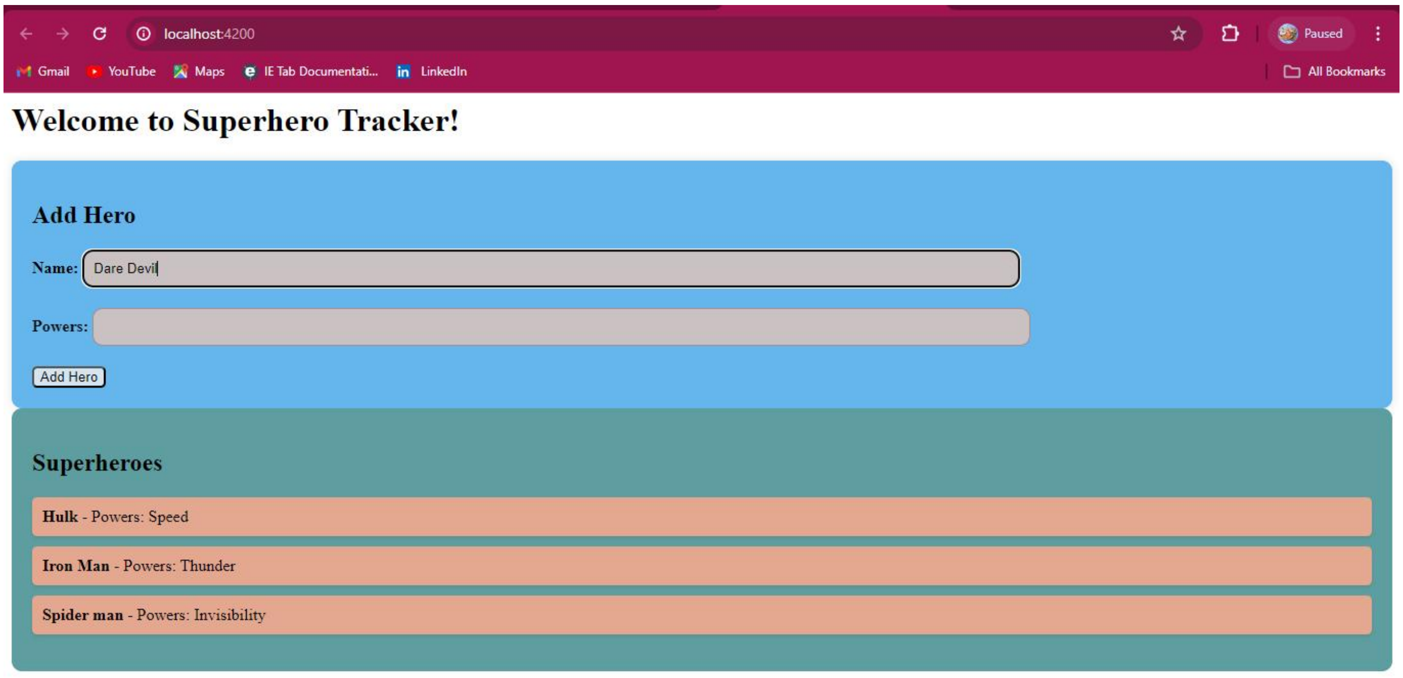
Add Hero

Name:

Powers:

Add Hero

Superheroes



Welcome to Superhero Tracker!

Add Hero

Name: Dare Devil

Powers:

Add Hero

Superheroes

Hulk - Powers: Speed

Iron Man - Powers: Thunder

Spider man - Powers: Invisibility

4) Write a program to demonstrate angular life cycle hooks with component communication?

Program;

Parent.component.ts;

```
import { Component } from '@angular/core';


@Component({
  selector: 'app-parent',
  templateUrl: './parent.component.html',
  styleUrls: ['./parent.component.css']
})
export class ParentComponent {
  parentData: string = 'Initial data from parent';
  showChild: boolean = true;
  toggleChild(): void {
    this.showChild = !this.showChild;
  }
  changeData(): void {
    this.parentData = 'Updated data from parent';
  }
}
```

Parent.component.html;

```
<button (click)="toggleChild()">Toggle Child Component</button>
<button (click)="changeData()">Change Data</button>


<app-child *ngIf="showChild" [data]="parentData"></app-child>
```

Child.component.ts;

```
import { Component, OnInit, OnChanges, DoCheck, AfterContentInit, AfterContentChecked,
AfterViewInit, AfterViewChecked, OnDestroy, Input, SimpleChanges } from '@angular/core';


@Component({
  selector: 'app-child',
  templateUrl: './child.component.html',
  styleUrls: ['./child.component.css']
})

export class ChildComponent implements OnInit, OnChanges, DoCheck, AfterContentInit,
AfterContentChecked, AfterViewInit, AfterViewChecked, OnDestroy {

  @Input() data: any[string] = [];
```

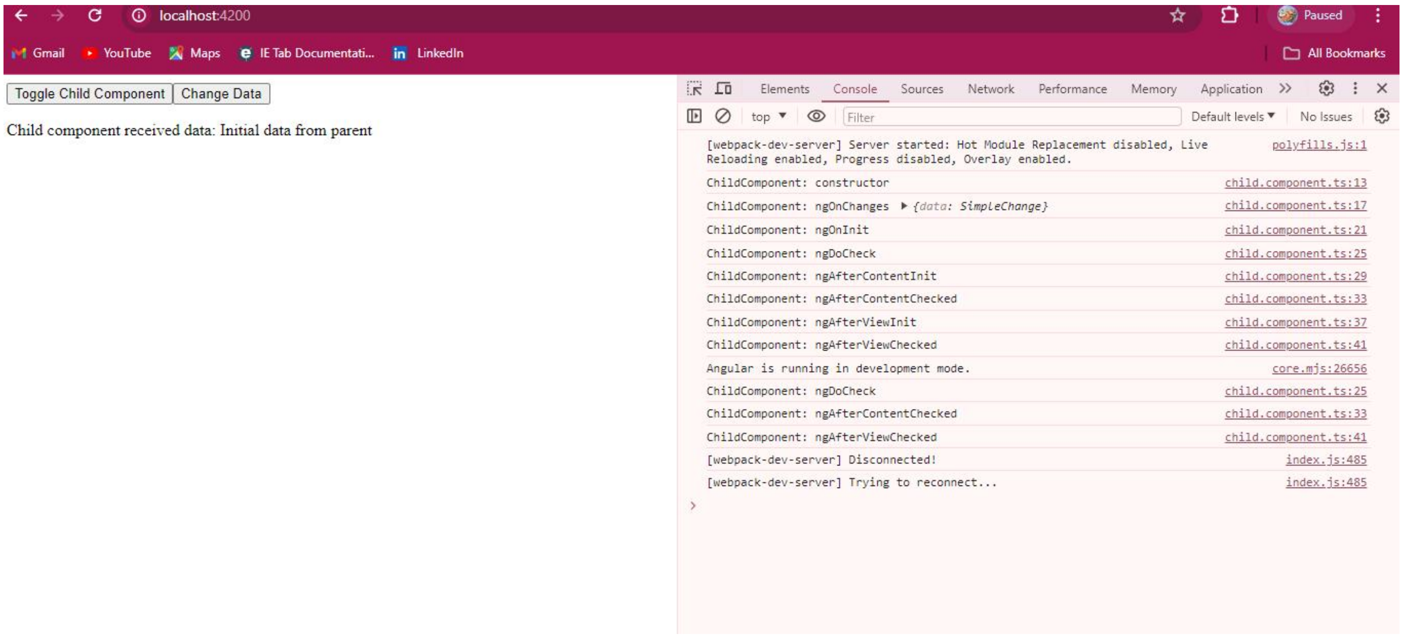
```
  constructor() {

    console.log('ChildComponent: constructor');

  }

  ngOnChanges(changes: SimpleChanges) {

    console.log('ChildComponent: ngOnChanges', changes);

  }

  ngOnInit(): void {

    console.log('ChildComponent: ngOnInit');

  }

  ngDoCheck(): void {

    console.log('ChildComponent: ngDoCheck');

  }

  ngAfterContentInit(): void {

    console.log('ChildComponent: ngAfterContentInit');

  }

  ngAfterContentChecked(): void {

    console.log('ChildComponent: ngAfterContentChecked');

  }

  ngAfterViewInit(): void {

    console.log('ChildComponent: ngAfterViewInit');

  }

  ngAfterViewChecked(): void {

    console.log('ChildComponent: ngAfterViewChecked');

  }

  ngOnDestroy(): void {

    console.log('ChildComponent: ngOnDestroy');

  }

}
```

Child.component.html;

```html
<p>Child component received data: {{ data }}</p>
```

Output ;

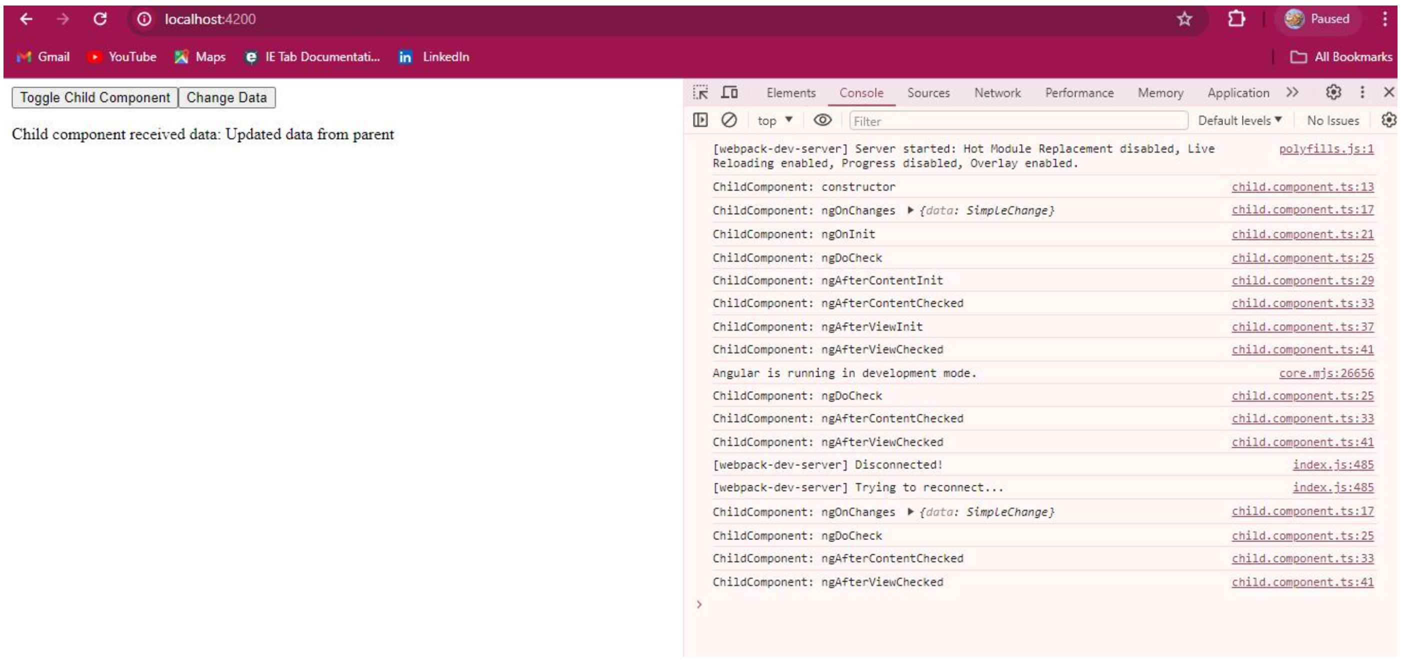
Initial data from parent ;



Child component received data: Initial data from parent

```
[webpack-dev-server] Server started: Hot Module Replacement disabled, Live     polyfills.js:1
Reloading enabled, Progress disabled, Overlay enabled.
ChildComponent: constructor                                              child.component.ts:13
ChildComponent: ngOnChanges ▶ {data: SimpleChange}                       child.component.ts:17
ChildComponent: ngOnInit                                                 child.component.ts:21
ChildComponent: ngDoCheck                                                child.component.ts:25
ChildComponent: ngAfterContentInit                                       child.component.ts:29
ChildComponent: ngAfterContentChecked                                    child.component.ts:33
ChildComponent: ngAfterViewInit                                          child.component.ts:37
ChildComponent: ngAfterViewChecked                                       child.component.ts:41
Angular is running in development mode.                                  core.mjs:26656
ChildComponent: ngDoCheck                                                child.component.ts:25
ChildComponent: ngAfterContentChecked                                    child.component.ts:33
ChildComponent: ngAfterViewChecked                                       child.component.ts:41
[webpack-dev-server] Disconnected!                                       index.js:485
[webpack-dev-server] Trying to reconnect...                              index.js:485
```

Update data from parent ;



Child component received data: Updated data from parent

```
[webpack-dev-server] Server started: Hot Module Replacement disabled, Live     polyfills.js:1
Reloading enabled, Progress disabled, Overlay enabled.
ChildComponent: constructor                                              child.component.ts:13
ChildComponent: ngOnChanges ▶ {data: SimpleChange}                       child.component.ts:17
ChildComponent: ngOnInit                                                 child.component.ts:21
ChildComponent: ngDoCheck                                                child.component.ts:25
ChildComponent: ngAfterContentInit                                       child.component.ts:29
ChildComponent: ngAfterContentChecked                                    child.component.ts:33
ChildComponent: ngAfterViewInit                                          child.component.ts:37
ChildComponent: ngAfterViewChecked                                       child.component.ts:41
Angular is running in development mode.                                  core.mjs:26656
ChildComponent: ngDoCheck                                                child.component.ts:25
ChildComponent: ngAfterContentChecked                                    child.component.ts:33
ChildComponent: ngAfterViewChecked                                       child.component.ts:41
[webpack-dev-server] Disconnected!                                       index.js:485
[webpack-dev-server] Trying to reconnect...                              index.js:485
ChildComponent: ngOnChanges ▶ {data: SimpleChange}                       child.component.ts:17
ChildComponent: ngDoCheck                                                child.component.ts:25
ChildComponent: ngAfterContentChecked                                    child.component.ts:33
ChildComponent: ngAfterViewChecked                                       child.component.ts:41
```

5) Write a program to send message using angular subject

Program:

App.Module.ts;

```
import { NgModule } from '@angular/core';
import { BrowserModule } from '@angular/platform-browser';

import { AppComponent } from './app.component';
import { SenderComponent } from './sender/sender.component';
import { ReceiverComponent } from './receiver/receiver.component';
```

```
@NgModule({
  declarations: [
    AppComponent,
    SenderComponent,
    ReceiverComponent
  ],
  imports: [
    BrowserModule
  ],
  providers: [],
  bootstrap: [AppComponent]
})
export class AppModule { }
```

App.component.html ;

```
<h1>Message Service Program</h1>
<app-sender></app-sender>
<app-receiver></app-receiver>
```

App.component.css ;

```
  .message-container {
    display: flex;
    flex-direction: column;
    align-items: flex-start;
    padding: 20px;
  }

  .message-container .sender:last-child {
    margin-bottom: 20px; /* Add extra space after the last sender message */
  }
```

Sender.component.ts ;

```
import { Component } from '@angular/core';
import { MessageService } from '../message.service';

@Component({
  selector: 'app-sender',
  templateUrl: './sender.component.html',
  styleUrls: ['./sender.component.css']
})
export class SenderComponent {
```

```
  constructor(private messageService: MessageService) {}
  sendMessage(message: string): void {
    this.messageService.sendMessage(message);
  }
}
```

Sender.component.html ;

```
<div class="sender">

    <input #messageInput type="text" placeholder="Enter message">

    <button (click)="sendMessage(messageInput.value)">Send Message</button>


</div>
```

Sender.component.css ;

```

.sender {

    background-color: #c6f8ec; /* Light green */

    padding: 10px;

    border-radius: 10px;

    margin-bottom: 10px;

    max-width: 70%;

    align-self: flex-end; /* Align to the right */

  }

```

Receiver.component.ts ;

```
import { Component, OnInit, OnDestroy } from '@angular/core';

import { Subscription } from 'rxjs';

import { MessageService } from '../message.service';


@Component({

  selector: 'app-receiver',

  templateUrl: './receiver.component.html',

  styleUrls: ['./receiver.component.css']

})

export class ReceiverComponent implements OnInit, OnDestroy {

  message: any[string] = [];

  subscription: any[string] = Subscription;

  constructor(private messageService: MessageService) {}

  ngOnInit(): void {
    this.subscription = this.messageService.message$.subscribe(message => {
      this.message = message;
    });
  }
```

```
  ngOnDestroy(): void {
    this.subscription.unsubscribe();
  }
}
```

Receiver.component.html ;

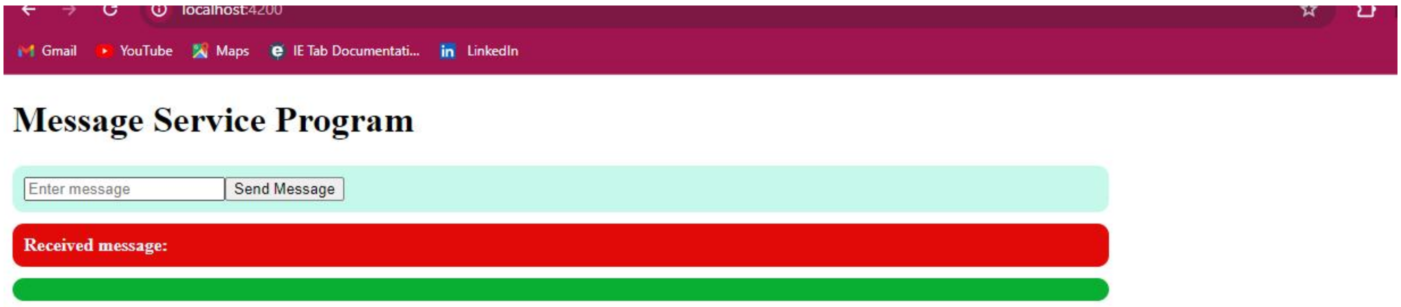
```
<div class="receiver" >
    <b >Received message: </b>
    <br>
</div>
<div class="msg" >
    <i >{{ message }}</i>
</div>
```

Receiver.component.css ;

```
/* sender-receiver.component.css */

.receiver {
    background-color: #e20909; /* Light gray */
    padding: 10px;
    border-radius: 10px;
    margin-bottom: 10px;
    max-width: 70%;
    color:aliceblue;
  }
  .msg{
    background-color: #08af32; /* Light gray */
    padding: 10px;
    border-radius: 10px;

    margin-bottom: 30px;

    max-width: 70%;

    color:rgb(240, 245, 255);

  }
```

Output ;

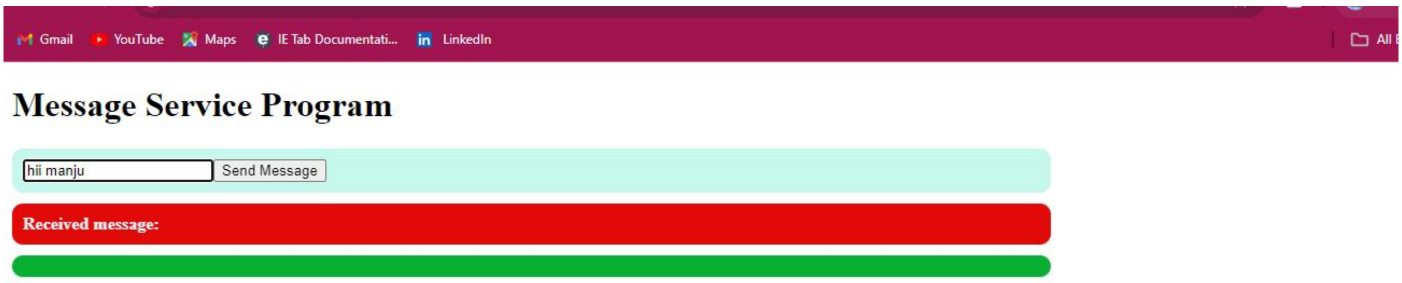
User Interface;



Message Service Program

Received message:

Sender ; **Enter message**

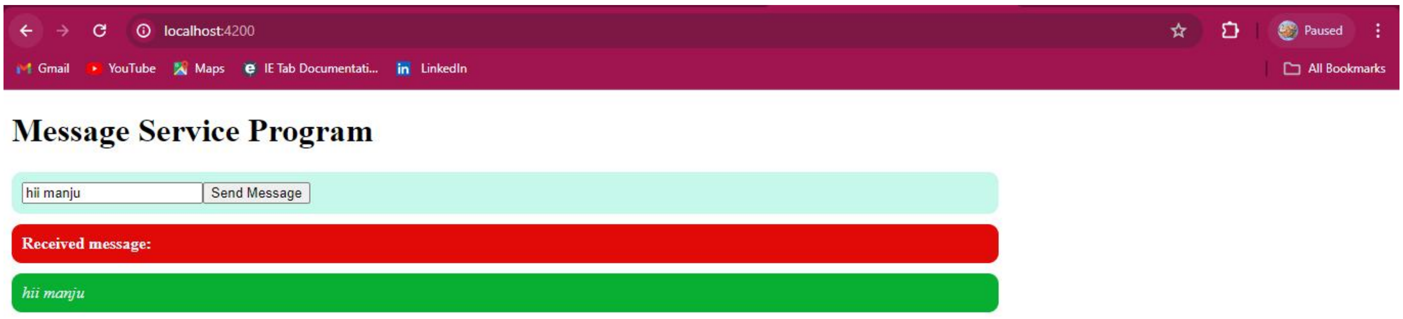


Message Service Program

hii manju

Received message:

Receiver ; The messages comes from sender is “ hii manju “


localhost:4200

Gmail YouTube Maps IE Tab Documentati... LinkedIn All Bookmarks

Message Service Program

hii manju Send Message

Received message:

hii manju