**Login.component.html**

<div class="container-fluid">

<form [formGroup]="loginForm">

<div class="login\_container">

<h4 class="text-center mb-2">ChatBot Login</h4>

<div class="input-group mb-3">

<div class="input-group-text">

<i class="fa fa-user" aria-hidden="true"></i>

</div>

<input type="text" class="form-control" placeholder="Username" formControlName="username" aria-label="Text input with checkbox">

</div>

<div class="input-group mb-3">

<div class="input-group-text">

<i class="fa fa-key" aria-hidden="true"></i>

</div>

<input type="password" class="form-control" formControlName="password" placeholder="Password"

aria-label="Text input with radio button">

</div>

<div class="text-warning mb-2" \*ngIf="isLoginClicked && !service.loggedIn">

Incorrect username or password

</div>

<div class="form-group text-center">

<button type="button" class="btn btn-primary mx-2" (click)="loginUser($event)">Login</button>

</div>

</div>

</form>

</div>

**Login.component.ts**

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

import { FormControl, FormGroup, Validators } from '@angular/forms';

import { Router } from '@angular/router';

import { CommondataService } from '../services/commondata.service';

@Component({

selector: 'app-login',

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

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

})

export class LoginComponent {

username:any;

password:any;

isLoginClicked:boolean = false;

loginForm = new FormGroup({

username: new FormControl('', Validators.required),

password: new FormControl('', Validators.required),

});

constructor(public service:CommondataService, private router:Router) { }

loginUser(event:any){

this.isLoginClicked = true;

this.username = this.loginForm.value.username;

this.password = this.loginForm.value.password;

if(this.username === 'test' && this.password === 'password'){

this.service.loggedIn = true;

}else{

this.service.loggedIn = false;

}

if(this.service.loggedIn)

{

this.router.navigate(['chat'])

}else{

alert("Please contact administrator.")

}

}

signUpUser(event:any){

console.log("Sign Up screen to be done as well.")

}

ngOnInit(): void {

}

}

**Login.component.css**

.login\_container{

position: absolute;

top: 50%; right: 50%;

transform: translate(50%,-50%);

background: blue;

padding: 2em;

}

.container-fluid{

position: absolute;

height:100%;

width:100%;

background:linear-gradient( to left top, blue, red);

}

**Chat.component.html**

<div class="container-fluid">

<div class="card\_position">

<div class="card style-card">

<div class="card-header">

<div class="row">

<div class="col-2">

<img src="../../../assets/images/bot.jpg.jpg" class="circle" />

</div>

<div class="col-10 text\_position">

<h5 class="bold style-agent">Agent X</h5>

</div>

</div>

</div>

<div class="card-body" #scrollMe [scrollTop]="scrollMe.scrollHeight">

<div \*ngIf="isData">

<div \*ngFor="let item of chatHistory.result">

<div><p \*ngIf="item.isBot" class="bot\_style">{{item.message}}</p></div>

<div><p \*ngIf="!item.isBot" class="user\_style">{{item.message}}</p></div>

</div>

</div>

</div>

<div class="card-footer">

<form [formGroup]="chatForm">

<div class="row">

<div class="col-11">

<input type="text" class="form-control" id="text" placeholder="Type your message..."

formControlName="message">

</div>

<div class="col-1">

<span class="icon-style" (click)="onSend($event)"><mat-icon

class="icon\_pos">send</mat-icon></span>

</div>

</div>

</form>

</div>

</div>

</div>

</div>

<div class="row position\_button">

<div class="col-3">

<button mat-raised-button color="primary" class="mx-2" (click)="selectUser($event,1)">Bot</button>

<button mat-raised-button color="primary" (click)="selectUser($event,2)">User</button>

</div>

</div>

**Chat.component.ts**

import { Component, ElementRef, OnInit, ViewChild } from '@angular/core';

import { FormControl, FormGroup, Validators } from '@angular/forms';

import { ScrollToBottomDirective } from 'src/app/directive/scroll-to-bottom.directive';

import { ApiService } from 'src/app/services/api.service';

@Component({

selector: 'app-chat',

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

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

})

export class ChatComponent implements OnInit {

constructor(private service:ApiService){}

@ViewChild(ScrollToBottomDirective)

scroll!: ScrollToBottomDirective;

isBot:boolean = true;

chatForm = new FormGroup({

message: new FormControl('', Validators.required)

})

chatHistory:any=[];

isData:boolean=false;

getChatData:any=[];

selectUser(event:any, user:any){

if(user === 1){

this.isBot = true;

}else{

this.isBot = false;

}

}

onSend(event:any){

console.log(this.chatHistory);

console.log(this.getChatData);

let length;

if(this.getChatData.result.length===0){

length=0;

}else{

length=this.getChatData.result.length;

}

let latestId= length;

if(this.chatForm.value.message){

if(latestId===0){

this.isData = true;

}

let data = {id:latestId+1, message:this.chatForm.value.message,isBot:this.isBot};

this.service.saveChat(data).subscribe((res:any)=>{

this.chatForm.reset();

if(this.chatHistory.length === 0){

this.chatHistory = this.getChatData;

}

this.chatHistory.result.push(data);

}, (err:any)=>{

console.log(err);

});

}else{

alert("Enter your questions.")

}

}

getData(){

this.service.getChatHistory().subscribe((res:any)=>{

this.getChatData = res;

if(this.getChatData.result.length===0){

this.isData = false;

}else{

this.chatHistory = this.getChatData;

this.isData = true;

}

}, (err:any)=>{

console.log(err);

})

}

ngOnInit(): void {

this.getData();

}

}

interface Data{

id:number;

message:string;

isBot:boolean;

}

**Chat.component.css**

.style-card{

width: 50em;

height: 40em;

}

.container-fluid{

position: absolute;

}

.card\_position{

position: relative;

top: 50%;

left: 50%;

transform: translate(-20%,20%);

}

.position\_button{

position: relative;

top: 90%;

left: 50%;

}

.style-agent{

color: white;

font-size: 2em;

padding: 0.25em;

}

.card-header{

background-color: rgba(10, 10, 173, 0.959);

}

.text\_position{

position: relative;

left: 22.5%;

margin-top: 0.5em;

}

.circle{

height: 4em;

width: 4em;

background-color: #bbb;

border-radius: 50%;

display: inline-block;

}

.icon-style{

height: 3em;

width: 3em;

background-color: rgb(12, 12, 148);

border-radius: 50%;

display: inline-block;

}

.icon\_pos{

margin-top: 0.35em;

margin-left: 0.4em;

scale: 1.25;

color: white;

}

.user\_style{

color: Orange;

background-color: lightblue;

float:right;

padding: 0.5em;

word-wrap: break-word;

width: 80%;

}

.card-body{

overflow-y:scroll;

max-height: 100em;

}

.bot\_style{

color:white;

background-color: darkblue;

padding: 0.5em;

float: left;

width: 80%;

}

**App.component.html**

<router-outlet></router-outlet>

**App.component.ts**

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

@Component({

selector: 'app-root',

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

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

})

export class AppComponent {

title = 'chatbot';

}

**App-routing.module.ts**

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

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

import { ChatComponent } from './components/chat/chat.component';

import { LoginComponent } from './login/login.component';

const routes: Routes = [{path:'', component:LoginComponent}, {path:'chat', component:ChatComponent}];

@NgModule({

imports: [RouterModule.forRoot(routes)],

exports: [RouterModule]

})

export class AppRoutingModule { }

**App.module.ts**

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

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

import { FormsModule, ReactiveFormsModule } from '@angular/forms';

import {HttpClientModule} from '@angular/common/http';

import {MatIconModule} from '@angular/material/icon';

import {MatButtonModule} from '@angular/material/button';

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

import { AppComponent } from './app.component';

import { LoginComponent } from './login/login.component';

import { ChatComponent } from './components/chat/chat.component';

import { BrowserAnimationsModule } from '@angular/platform-browser/animations';

import { ScrollToBottomDirective } from './directive/scroll-to-bottom.directive';

@NgModule({

declarations: [

AppComponent,

LoginComponent,

ChatComponent,

ScrollToBottomDirective

],

imports: [

BrowserModule,

AppRoutingModule,

FormsModule,

ReactiveFormsModule,

HttpClientModule,

BrowserAnimationsModule,

MatIconModule,

MatButtonModule,

HttpClientModule

],

providers: [],

bootstrap: [AppComponent]

})

export class AppModule { }

**Api.service.ts**

import { HttpClient } from '@angular/common/http';

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

@Injectable({

providedIn: 'root'

})

export class ApiService {

url:string='http://localhost:8000/api';

constructor(private http:HttpClient) { }

getChatHistory(){

return this.http.get(this.url)

}

saveChat(data:any){

return this.http.post(this.url+'/save',data,{responseType: 'text'})

}

}

**Commondata.service.ts**

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

@Injectable({

providedIn: 'root'

})

export class CommondataService {

loggedIn:boolean = false;

constructor() { }

}

**NODEJS CODE**

**Chat.js**

const mongoose = require('mongoose');

const Schema = mongoose.Schema;

let chat = new Schema({

id: {

type: Number

},

message: {

type: String

},

isBot: {

type: Boolean

}

}, {

collection: 'Chat'

})

module.exports = mongoose.model('Chat', chat)

**Chat.routes.js**

const express = require('express');

const app = express();

const chatRoute = express.Router();

let chatModel = require('../models/chat');

// Get all Book

chatRoute.route('/').get((req, res) => {

chatModel.find({}, function (err, result) {

if (err) {

console.log(err);

} else {

res.send({ result })

}

})

})

// Save Book

chatRoute.route('/save').post((req, res) => {

var myData = new chatModel(req.body);

console.log(req.body);

console.log(req);

myData.save().then(item => {

res.send("Data saved to database");

})

.catch(err => {

res.status(400).send("unable to save to database");

});

});

module.exports = chatRoute;

**Index.js**

const express = require('express')

const path = require('path')

const mongoose = require('mongoose')

const cors = require('cors')

const bodyParser = require('body-parser')

mongoose

.connect('mongodb://localhost:27017',

{useNewUrlParser: true})

.then((x) => {

console.log(`Connected to Mongo! Database name: "${x.connections[0].name}"`)

})

.catch((err) => {

console.error('Error connecting to mongo', err.reason)

})

const chatRoute = require('./routes/chat.routes')

const app = express()

// parse application/x-www-form-urlencoded

app.use(bodyParser.urlencoded({ extended: false }))

// parse application/json

app.use(bodyParser.json())

app.use(cors())

// API root

app.use('/api', chatRoute)

// PORT

const port = process.env.PORT || 8000

app.listen(port, () => {

console.log('Listening on port ' + port)

})

// error handler

app.use(function (err, req, res, next) {

console.error(err.message)

if (!err.statusCode) err.statusCode = 500

res.status(err.statusCode).send(err.message)

})