src\index.html

```
<!doctype html>
<html lang="en">
<head>
  <meta charset="utf-8">
  <title>ClientManagementApp</title>
  <base href="/">
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <link rel="icon" type="image/x-icon" href="favicon.ico">
</head>
<body>
  <app-root></app-root>
</body>
</html>
```

src\main.ts

```
import { enableProdMode } from '@angular/core';
import { platformBrowserDynamic } from '@angular/platform-browser-dynamic';

import { AppModule } from './app/app.module';
import { environment } from './environments/environment';

if (environment.production) {
  enableProdMode();
}

platformBrowserDynamic().bootstrapModule(AppModule)
  .catch(err => console.error(err));
```

src\polyfills.ts

```
/**
 * This file includes polyfills needed by Angular and is loaded before the app.
 * You can add your own extra polyfills to this file.
 *
 * This file is divided into 2 sections:
 *   1. Browser polyfills. These are applied before loading ZoneJS and are sorted by browsers.
 *   2. Application imports. Files imported after ZoneJS that should be loaded before your main
 *      file.
 *
 * The current setup is for so-called "evergreen" browsers; the last versions of browsers that
 * automatically update themselves. This includes recent versions of Safari, Chrome (including
 * Opera), Edge on the desktop, and iOS and Chrome on mobile.
 *
 * Learn more in https://angular.io/guide/browser-support
 */

/***************************************************************************************************
```

```
 * BROWSER POLYFILLS
 */

/**
 * By default, zone.js will patch all possible macroTask and DomEvents
 * user can disable parts of macroTask/DomEvents patch by setting following flags
 * because those flags need to be set before `zone.js` being loaded, and webpack
 * will put import in the top of bundle, so user need to create a separate file
 * in this directory (for example: zone-flags.ts), and put the following flags
 * into that file, and then add the following code before importing zone.js.
 * import './zone-flags';
 *
 * The flags allowed in zone-flags.ts are listed here.
 *
 * The following flags will work for all browsers.
 *
 * (window as any).__Zone_disable_requestAnimationFrame = true; // disable patch requestAnimationFr
 * (window as any).__Zone_disable_on_property = true; // disable patch onProperty such as onclick
 * (window as any).__zone_symbol__UNPATCHED_EVENTS = ['scroll', 'mousemove']; // disable patch spec
 *
 *  in IE/Edge developer tools, the addEventListener will also be wrapped by zone.js
 *  with the following flag, it will bypass `zone.js` patch for IE/Edge
 *
 *  (window as any).__Zone_enable_cross_context_check = true;
 *
 */

/***************************************************************************************************
 * Zone JS is required by default for Angular itself.
 */
import 'zone.js';  // Included with Angular CLI.


/***************************************************************************************************
 * APPLICATION IMPORTS
 */
```

src\styles.css

```
/* You can add global styles to this file, and also import other style files */
```

src\test.ts

```
// This file is required by karma.conf.js and loads recursively all the .spec and framework files

import 'zone.js/testing';
import { getTestBed } from '@angular/core/testing';
import {
  BrowserDynamicTestingModule,
  platformBrowserDynamicTesting
} from '@angular/platform-browser-dynamic/testing';
```

```
declare const require: {
  context(path: string, deep?: boolean, filter?: RegExp): {
    <T>(id: string): T;
    keys(): string[];
  };
};

// First, initialize the Angular testing environment.
getTestBed().initTestEnvironment(
  BrowserDynamicTestingModule,
  platformBrowserDynamicTesting(),
);

// Then we find all the tests.
const context = require.context('./', true, /\.spec\.ts$/);
// And load the modules.
context.keys().forEach(context);
```

src\app\app-routing.module.ts

```
import { NgModule } from '@angular/core';
import { RouterModule, Routes } from '@angular/router';
import { LoginComponent } from './auth/login/login.component';

const routes: Routes = [
  {
    path: 'auth',
    loadChildren: () => import('./auth/auth.module').then(m => m.AuthModule)
  },
  {
    path: 'dash',
    loadChildren: () => import('./dash/dash.module').then(m => m.DashModule)
  },
  {
    path: 'client',
    loadChildren: () => import('./client/client.module').then(m => m.ClientModule)
  },
  {
    path: 'project',
    loadChildren: () => import('./project/project.module').then(m => m.ProjectModule)
  },
  {
    path: 'meetings',
    loadChildren: () => import('./meetings/meetings.module').then(m => m.MeetingsModule)
  },
  {
    path:'shared',
    loadChildren:() => import('./shared/shared.module').then(m=>m.SharedModule)
  },
  { path: 'login', component: LoginComponent },
  { path: '', redirectTo: 'login', pathMatch: 'full' }, // Redirect to login as default
  { path: '**', redirectTo: 'login' } // Redirect for all undefined routes
];
@NgModule({
```

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

src\app\app.component.css

src\app\app.component.html

```
<router-outlet></router-outlet>
```

src\app\app.component.spec.ts

```
import { TestBed } from '@angular/core/testing';
import { RouterTestingModule } from '@angular/router/testing';
import { AppComponent } from './app.component';

describe('AppComponent', () => {
  beforeEach(async () => {
    await TestBed.configureTestingModule({
      imports: [
        RouterTestingModule
      ],
      declarations: [
        AppComponent
      ],
    }).compileComponents();
  });

  it('should create the app', () => {
    const fixture = TestBed.createComponent(AppComponent);
    const app = fixture.componentInstance;
    expect(app).toBeTruthy();
  });

  it(`should have as title 'client-management-app'`, () => {
    const fixture = TestBed.createComponent(AppComponent);
    const app = fixture.componentInstance;
    expect(app.title).toEqual('client-management-app');
  });

  it('should render title', () => {
    const fixture = TestBed.createComponent(AppComponent);
    fixture.detectChanges();
    const compiled = fixture.nativeElement as HTMLElement;
    expect(compiled.querySelector('.content span')?.textContent).toContain('client-management-app a
  });
```

```
});
```

src\app\app.component.ts

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

@Component({
  selector: 'app-root',
  templateUrl: './app.component.html',
  styleUrls: ['./app.component.css']
})
export class AppComponent {
  title = 'client-management-app';
}
```

src\app\app.module.ts

```
import { NgModule } from '@angular/core';
import { BrowserModule } from '@angular/platform-browser';
import { AppComponent } from './app.component';
import { DashModule } from './dash/dash.module';
import { ClientModule } from './client/client.module';
import { ProjectModule } from './project/project.module';
import { MeetingsModule } from './meetings/meetings.module';
import { AuthModule } from './auth/auth.module';
import { AppRoutingModule } from './app-routing.module';
import { SharedModule } from './shared/shared.module';
import { HttpClientModule } from '@angular/common/http';

@NgModule({
  declarations: [
    AppComponent
  ],
  imports: [
    BrowserModule,
    DashModule,
    ClientModule,
    ProjectModule,
    MeetingsModule,
    AuthModule,
    AppRoutingModule,
    SharedModule,
    HttpClientModule
  ],
  providers: [],
  bootstrap: [AppComponent],

  exports:[
    SharedModule
  ]
})
```

```
export class AppModule { }
```

src\app\auth\auth-routing.module.ts

```
import { NgModule } from '@angular/core';
import { RouterModule, Routes } from '@angular/router';
import { LoginComponent } from './login/login.component';
import { RegisterComponent } from './register/register.component';
import { ForgotPasswordComponent } from './forgot-password/forgot-password.component';

const routes: Routes = [
  {path:'login',component:LoginComponent},
  {path:'register',component:RegisterComponent},
  {path:'forgot-password',component:ForgotPasswordComponent}
];

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

src\app\auth\auth.module.ts

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

import { AuthRoutingModule } from './auth-routing.module';

import { ReactiveFormsModule } from '@angular/forms';
import { LoginComponent } from './login/login.component';
import { RegisterComponent } from './register/register.component';
import { ForgotPasswordComponent } from './forgot-password/forgot-password.component';
import { FormsModule } from '@angular/forms';
import { EmailService } from './email.service';


@NgModule({
  declarations: [
    LoginComponent,
    RegisterComponent,
    ForgotPasswordComponent

  ],
  imports: [
    CommonModule,
    AuthRoutingModule,
    ReactiveFormsModule,
    FormsModule
  ],
  providers:[EmailService]
```

```
})
export class AuthModule { }
```

src\app\auth\email.service.ts

```
import { Injectable } from '@angular/core';
import { HttpClient } from '@angular/common/http';
import { Observable } from 'rxjs';

@Injectable({
  providedIn: 'root'
})
export class EmailService {
  constructor(private http: HttpClient) {}

  sendVerificationEmail(email: string, token: string): Observable<any> {
    const verificationLink = `http://localhost:3000/api/send-email?token=${token}`; // Use your loc
    const emailPayload = {
      to: email, // Use the provided email
      subject: 'Email Verification',
      body: `Click this link to verify your email: ${verificationLink}`
    };

    return this.http.post('http://localhost:3000/api/send-email', emailPayload); // Use your backen
  }
}
```

src\app\auth\forgot-password\forgot-password.component.css

```
/* Forgot Password Container */
.forgot-password-container {
  display: flex;
  justify-content: center;
  align-items: center;
  height: 100vh;
  background-color: #121212; /* Dark background */
}

/* Forgot Password Form */
.forgot-password-form {
  background-color: #1a1a1a; /* Dark form background */
  padding: 30px;
  border-radius: 10px;
  box-shadow: 0 4px 8px rgba(0, 0, 0, 0.3);
  width: 100%;
  max-width: 400px; /* Maximum width for the form */
  text-align: center;
  color: #fff; /* White text */
}

/* Heading */
```

```
.forgot-password-form h2 {
  color: #00b4d8; /* Reference accent color for heading */
  margin-bottom: 20px;
}

/* Form Group */
.forgot-password-form .form-group {
  margin-bottom: 20px;
}

/* Label */
.forgot-password-form .form-group label {
  font-size: 16px;
  color: #ddd; /* Light gray text */
  display: block;
  margin-bottom: 8px;
}

/* Input */
.forgot-password-form .form-group input {
  width: 100%;
  padding: 10px;
  font-size: 16px;
  border: 1px solid #444; /* Dark border */
  border-radius: 5px;
  background-color: #2a2a2a; /* Dark input background */
  color: #fff; /* White text */
  box-sizing: border-box;
}

/* Input Focus */
.forgot-password-form .form-group input:focus {
  outline: none;
  border-color: #00b4d8; /* Reference accent color on focus */
}

/* Button */
.forgot-password-form button {
  background-color: #00b4d8; /* Reference accent color for button */
  color: black;
  padding: 12px 20px;
  font-size: 16px;
  border: none;
  border-radius: 5px;
  width: 100%;
  cursor: pointer;
  transition: background-color 0.3s ease;
}

/* Button Hover */
.forgot-password-form button:hover {
  background-color: #008ba3; /* Darker accent color on hover */
}

/* Links */
.forgot-password-form .links {
  margin-top: 20px;
```

```
}

/* Link Text */
.forgot-password-form .links a {
  color: #00b4d8; /* Reference accent color for links */
  font-size: 14px;
  text-decoration: none;
}

/* Link Hover */
.forgot-password-form .links a:hover {
  text-decoration: underline;
}
```

src\app\auth\forgot-password\forgot-password.component.html

```
<div class="forgot-password-container" style="background-color: black; height: 900px;">
    <form class="forgot-password-form" (ngSubmit)="onSubmit()">
      <h2>Forgot Password</h2>

      <div class="form-group">
        <label for="email">Email</label>
        <input
          id="email"
          type="email"
          class="form-control"
          [(ngModel)]="email"
          name="email"
          required
        />
      </div>

      <button type="submit" class="btn btn-primary">Send Reset Link</button>

      <div class="links">
        <a routerLink="/auth/login">Remembered your password? Login</a>
      </div>
    </form>
  </div>
```

src\app\auth\forgot-password\forgot-password.component.spec.ts

```
import { ComponentFixture, TestBed } from '@angular/core/testing';

import { ForgotPasswordComponent } from './forgot-password.component';

describe('ForgotPasswordComponent', () => {
  let component: ForgotPasswordComponent;
  let fixture: ComponentFixture<ForgotPasswordComponent>;

  beforeEach(async () => {
```

```
    await TestBed.configureTestingModule({
      declarations: [ ForgotPasswordComponent ]
    })
    .compileComponents();

    fixture = TestBed.createComponent(ForgotPasswordComponent);
    component = fixture.componentInstance;
    fixture.detectChanges();
  });

  it('should create', () => {
    expect(component).toBeTruthy();
  });
});
```

src\app\auth\forgot-password\forgot-password.component.ts

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

@Component({
  selector: 'app-forgot-password',
  templateUrl: './forgot-password.component.html',
  styleUrls: ['./forgot-password.component.css']
})
export class ForgotPasswordComponent {
  email: string = '';

  constructor(private router: Router) {}


  onSubmit() {
    if (this.email) {
      console.log('Password reset link sent to:', this.email);


      this.router.navigate(['/auth/login']);
    } else {
      console.error('Please enter your email');
    }
  }
}
```

src\app\auth\login\login.component.css

```
body {
  margin: 0;
  font-family: Arial, sans-serif;
  background-color: #000000; /* Black background */
  color: #fff;               /* White text */
  display: flex;
```

```
  justify-content: center;   /* Center content horizontally */
  align-items: center;       /* Center content vertically */
  height: 100vh;             /* Full viewport height */
}

/* Login Container */
.login-container {
  display: flex;
  justify-content: center;   /* Center the form inside the container */
  align-items: center;       /* Center the form inside the container */
  width: 100%;
  height: 100%;             /* Full height of the screen */
}

/* Login Form */
.login-form {
  background-color: #1a1a1a; /* Dark form background */
  padding: 30px;
  border-radius: 8px;
  box-shadow: 0 4px 8px rgba(0, 0, 0, 0.3);
  width: 100%;
  max-width: 400px;          /* Maximum width of the form */
  font-size: 16px;
  border: 1px solid #444;   /* Dark border around form */
}

/* Heading */
h2 {
  text-align: center;
  font-size: 24px;
  margin-bottom: 20px;
  color: #00b4d8;            /* Reference accent color */
}

/* Form Group */
.form-group {
  margin-bottom: 20px;
}

/* Label */
.form-group label {
  font-weight: bold;
  font-size: 16px;
  color: #ddd;               /* Light gray label text */
}

/* Input Fields */
.form-group input {
  width: 100%;
  padding: 12px;
  font-size: 16px;
  border-radius: 6px;
  border: 1px solid #444;   /* Dark border */
  margin-top: 5px;
  background-color: #2a2a2a; /* Dark input background */
  color: #fff;               /* White text */
  box-sizing: border-box;
```

```
}

/* Input Focus */
.form-group input:focus {
  outline: none;
  border-color: #00b4d8;    /* Reference accent color on focus */
}

/* Submit Button */
button {
  width: 100%;
  padding: 12px;
  background-color: #00b4d8; /* Reference accent color */
  color: black;
  border: none;
  border-radius: 6px;
  font-size: 18px;
  cursor: pointer;
  transition: background-color 0.3s ease;
}

/* Button Hover */
button:hover {
  background-color: #008ba3; /* Darker accent color on hover */
}

/* Links */
.links {
  text-align: center;
  margin-top: 15px;
}

/* Link Style */
.links a {
  color: #00b4d8;            /* Accent color for links */
  font-size: 14px;
  text-decoration: none;
}

/* Link Hover */
.links a:hover {
  text-decoration: underline;
}

/* Forgot Password and Register Links */
.links .forgot-password,
.links .register {
  display: block;
  margin-top: 10px;
}
```

src\app\auth\login\login.component.html

```
<div class="container" style="background-color: black; height: 900px;">
```

```
  <div class="login-container">
    <div class="login-form">
      <h2>Login</h2>
      <form (ngSubmit)="onSubmit()">
        <div class="form-group">
          <label for="username">Username</label>
          <input
            id="username"
            type="text"
            class="form-control"
            [(ngModel)]="username"
            name="username"
            required
          />
        </div>

        <div class="form-group">
          <label for="password">Password</label>
          <input
            id="password"
            type="password"
            class="form-control"
            [(ngModel)]="password"
            name="password"
            required
          />
        </div>

        <button type="submit" class="btn btn-primary">Login</button>
      </form>

      <div class="links">
        <a routerLink="/auth/forgot-password">Forgot Password?</a> |
        <a routerLink="/auth/register">Register</a>
      </div>
    </div>
  </div>

</div>
```

src\app\auth\login\login.component.spec.ts

```
import { ComponentFixture, TestBed } from '@angular/core/testing';

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

describe('LoginComponent', () => {
  let component: LoginComponent;
  let fixture: ComponentFixture<LoginComponent>;

  beforeEach(async () => {
    await TestBed.configureTestingModule({
      declarations: [ LoginComponent ]
    })
```

```
    .compileComponents();

    fixture = TestBed.createComponent(LoginComponent);
    component = fixture.componentInstance;
    fixture.detectChanges();
  });

  it('should create', () => {
    expect(component).toBeTruthy();
  });
});
```

src\app\auth\login\login.component.ts

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

@Component({
  selector: 'app-login',
  templateUrl: './login.component.html',
  styleUrls: ['./login.component.css']
})
export class LoginComponent {
  username: string = '';
  password: string = '';

  constructor(private router: Router) {}


  onSubmit() {

    if (this.username && this.password) {

      this.router.navigate(['/dash/dashboard']);
    } else {

      console.error('Invalid username or password');
    }
  }
}
```

src\app\auth\register\register.component.css

```
body {
  height: 100vh;
  margin: 0;
  font-family: Arial, sans-serif;
  background-color: #000000; /* Black background */
  color: #fff; /* White text */
  display: flex;
  justify-content: center; /* Center horizontally */
```

```
  align-items: center; /* Center vertically */
}

.register-container {
  display: flex;
  justify-content: center; /* Center the form inside the container */
  align-items: center; /* Center the form inside the container */
  height: 100%; /* Ensure the container takes full height */
  width: 100%; /* Full width */
}

.register-form {
  background-color: #1a1a1a; /* Dark background for form */
  padding: 30px;
  border-radius: 8px;
  box-shadow: 0 4px 8px rgba(0, 0, 0, 0.3);
  width: 100%;
  max-width: 400px; /* Maximum width of the form */
  font-size: 16px;
}

h2 {
  text-align: center;
  font-size: 24px;
  margin-bottom: 20px;
  color: #00b4d8; /* Cyan-blue color */
}

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

.form-group label {
  font-weight: bold;
  font-size: 16px;
  color: #ddd; /* Light gray label text */
}

.form-group input {
  width: 100%;
  padding: 12px;
  font-size: 16px;
  border-radius: 6px;
  border: 1px solid #444; /* Dark border */
  margin-top: 5px;
  background-color: #2a2a2a; /* Dark input background */
  color: #fff; /* White text */
  box-sizing: border-box;
}

.form-group input:focus {
  outline: none;
  border-color: #00b4d8; /* Cyan-blue border on focus */
}

button {
  width: 100%;
```

```
  padding: 12px;
  background-color: #00b4d8; /* Cyan-blue button */
  color: black;
  border: none;
  border-radius: 6px;
  font-size: 18px;
  cursor: pointer;
  transition: background-color 0.3s;
}

button:hover {
  background-color: #0074a2; /* Darker blue on hover */
}

.links {
  text-align: center;
  margin-top: 15px;
}

.links a {
  color: #00b4d8; /* Cyan-blue link color */
  font-size: 14px;
  text-decoration: none;
}

.links a:hover {
  text-decoration: underline;
}

.links .forgot-password,
.links .register {
  display: block;
  margin-top: 10px;
}
```

src\app\auth\register\register.component.html

```
<div class="register-container" style="background-color: black; height: 900px;">
  <form class="register-form" (ngSubmit)="onSubmit()">
    <h2>Registerrrrr</h2>

    <div class="form-group">
      <label for="username">Username</label>
      <input
        id="username"
        type="text"
        class="form-control"
        [(ngModel)]="username"
        name="username"
        required
      />
    </div>

    <div class="form-group">
```

```
      <label for="email">Email</label>
      <input
        id="email"
        type="email"
        class="form-control"
        [(ngModel)]="email"
        name="email"
        required
      />
    </div>

    <div class="form-group">
      <label for="password">Password</label>
      <input
        id="password"
        type="password"
        class="form-control"
        [(ngModel)]="password"
        name="password"
        required
      />
    </div>

    <button type="submit" class="btn btn-primary">Register</button>

    <div class="links">
      <a routerLink="/auth/login">Already have an account? Login</a>
    </div>
  </form>
</div>
```

src\app\auth\register\register.component.spec.ts

```
import { ComponentFixture, TestBed } from '@angular/core/testing';

import { RegisterComponent } from './register.component';

describe('RegisterComponent', () => {
  let component: RegisterComponent;
  let fixture: ComponentFixture<RegisterComponent>;

  beforeEach(async () => {
    await TestBed.configureTestingModule({
      declarations: [ RegisterComponent ]
    })
    .compileComponents();

    fixture = TestBed.createComponent(RegisterComponent);
    component = fixture.componentInstance;
    fixture.detectChanges();
  });

  it('should create', () => {
    expect(component).toBeTruthy();
```

```
  });
});
```

src\app\auth\register\register.component.ts

```
import { Component } from '@angular/core';
import { Router } from '@angular/router';
import { EmailService } from '../email.service'; // Import the email service

@Component({
  selector: 'app-register',
  templateUrl: './register.component.html',
  styleUrls: ['./register.component.css']
})
export class RegisterComponent {
  username: string = '';
  email: string = '';
  password: string = '';

  constructor(private router: Router, private emailService: EmailService) {} // Inject EmailService

  onSubmit() {
    if (this.username && this.email && this.password) {
      const verificationToken = this.generateVerificationToken(); // Generate token
      this.emailService.sendVerificationEmail(this.email, verificationToken)
        .subscribe(
          () => {
            console.log('Verification email sent successfully');
            alert('A verification email has been sent to your email address.');
            this.router.navigate(['/auth/login']); // Redirect to login page
          },
          (error: any) => {
            console.error('Failed to send verification email:', error);
            alert('Failed to send verification email.');
          }
        );
    } else {
      console.error('Please fill in all fields');
    }
  }

  private generateVerificationToken(): string {
    return Math.random().toString(36).substr(2); // Generate a random token
  }
}
```

src\app\client\client-routing.module.ts

```
import { NgModule } from '@angular/core';
import { RouterModule, Routes } from '@angular/router';
import { ClientListComponent } from './client-list/client-list.component';
```

```
import { ClientDetailsComponent } from './client-details/client-details.component';
import { ClientEditComponent } from './client-edit/client-edit.component';

const routes: Routes = [
  { path: 'client-list', component: ClientListComponent },
  { path: 'client-details/:id', component: ClientDetailsComponent },
  { path: 'client-edit/:id', component: ClientEditComponent }
];

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

src\app\client\client.module.ts

```
// client.module.ts
import { NgModule } from '@angular/core';
import { CommonModule } from '@angular/common';
import { ClientListComponent } from './client-list/client-list.component';
import { ClientDetailsComponent } from './client-details/client-details.component';
import { ClientEditComponent } from './client-edit/client-edit.component';
import { ClientRoutingModule } from './client-routing.module';
import { FormsModule } from '@angular/forms';
import { SharedModule } from '../shared/shared.module';


@NgModule({
  declarations: [
    ClientListComponent,
    ClientDetailsComponent,
    ClientEditComponent,

  ],
  imports: [
    CommonModule,
    ClientRoutingModule,
    FormsModule,
    SharedModule

  ]
})
export class ClientModule {}
```

src\app\client\client-details\client-details.component.css

```
/* Client Details Section */
.client-details {
  background: linear-gradient(145deg, rgba(42, 42, 59, 0.9), rgba(42, 42, 59, 0.7));
  padding: 25px;
```

```
  border-radius: 15px;
  box-shadow: 0 4px 20px rgba(0, 0, 0, 0.3);
  border: 1px solid rgba(0, 247, 255, 0.1);
  backdrop-filter: blur(5px);
  margin-top: 20px;
  position: relative;
  overflow: hidden;
}

.client-details::before {
  content: '';
  position: absolute;
  top: 0;
  left: 0;
  width: 100%;
  height: 2px;
  background: linear-gradient(90deg, transparent, #00f7ff, transparent);
  transform: translateX(-100%);
  transition: transform 0.6s ease;
}

.client-details:hover::before {
  transform: translateX(100%);
}

.client-details h2 {
  font-size: 2rem;
  color: #ffffff;
  text-shadow: 0 0 10px rgba(0, 247, 255, 0.5);
  margin-bottom: 20px;
  position: relative;
  padding-bottom: 10px;
}

.client-details h2::after {
  content: '';
  position: absolute;
  bottom: 0;
  left: 0;
  width: 50px;
  height: 2px;
  background: #00f7ff;
}

.client-details p {
  margin-bottom: 15px;
  font-size: 1.1rem;
  color: #b3b3c1;
}

.client-details button {
  background: linear-gradient(135deg, #00f7ff, #2e59d9);
  color: #ffffff;
  border: none;
  padding: 12px 25px;
  border-radius: 8px;
  cursor: pointer;
```

```
  transition: all 0.3s ease;
  text-shadow: 0 0 10px rgba(0, 247, 255, 0.3);
}

.client-details button:hover {
  transform: translateY(-2px);
  box-shadow: 0 4px 15px rgba(0, 247, 255, 0.2);
}
```

src\app\client\client-details\client-details.component.html

```
<app-navbar></app-navbar>

    <div class="client-details" style="height: 500px; ">
        <h2>Client Details</h2>
        <div *ngIf="client">
          <p><strong>Name:</strong> {{ client.name }}</p>
          <p><strong>Contact:</strong> {{ client.contact }}</p>
          <p><strong>Summary:</strong> {{ client.summary }}</p>
          <p><strong>Additional Details:</strong> {{ client.additionalDetails }}</p><br><br>

          <button (click)="goToEdit()">Edit Client</button>
        </div>
        <div *ngIf="!client">
          <p>Loading client details...</p>
        </div>
      </div>

<app-footer></app-footer>
```

src\app\client\client-details\client-details.component.spec.ts

```
import { ComponentFixture, TestBed } from '@angular/core/testing';

import { ClientDetailsComponent } from './client-details.component';

describe('ClientDetailsComponent', () => {
  let component: ClientDetailsComponent;
  let fixture: ComponentFixture<ClientDetailsComponent>;

  beforeEach(async () => {
    await TestBed.configureTestingModule({
      declarations: [ ClientDetailsComponent ]
    })
    .compileComponents();

    fixture = TestBed.createComponent(ClientDetailsComponent);
    component = fixture.componentInstance;
    fixture.detectChanges();
  });
  it('should create', () => {
```

```
    expect(component).toBeTruthy();
  });
});
```

src\app\client\client-details\client-details.component.ts

```
import { Component, OnInit } from '@angular/core';
import { ActivatedRoute, Router } from '@angular/router';

@Component({
  selector: 'app-client-details',
  templateUrl: './client-details.component.html',
  styleUrls: ['./client-details.component.css']
})
export class ClientDetailsComponent implements OnInit {
  clientId!: number;
  client: any;

  constructor(private route: ActivatedRoute, private router: Router) {}

  ngOnInit(): void {
    this.clientId = Number(this.route.snapshot.paramMap.get('id')); // Get the client ID from the r
    this.fetchClientDetails();
  }

  fetchClientDetails(): void {
    // Simulate fetching client data (replace with actual API call if needed)
    const clients = [
      { id: 1, name: 'Alice Johnson', contact: 'alice@example.com', summary: 'Regular client from N
      { id: 2, name: 'Bob Smith', contact: 'bob@example.com', summary: 'Long-term client.', additio
      { id: 3, name: 'Charlie Brown', contact: 'charlie@example.com', summary: 'New client in Calif
      { id: 4, name: 'David Green', contact: 'david@example.com', summary: 'Frequent buyer.', addit
      { id: 5, name: 'Eva White', contact: 'eva@example.com', summary: 'VIP client from LA.', addit
    ];

    this.client = clients.find(client => client.id === this.clientId); // Fetch client data based o
  }

  goToEdit(): void {
    this.router.navigate(['/client-edit', this.clientId]); // Navigate to the client edit page
  }
}
```

src\app\client\client-edit\client-edit.component.css

```
/* Client Edit Container */
.client-edit {
  background: linear-gradient(145deg, rgba(42, 42, 59, 0.9), rgba(42, 42, 59, 0.7));
  padding: 30px;
  border-radius: 15px;
  box-shadow: 0 4px 20px rgba(0, 0, 0, 0.3);
```

```
  border: 1px solid rgba(0, 247, 255, 0.1);
  backdrop-filter: blur(5px);
  position: relative;
  overflow: hidden;
  margin: 20px auto;
  max-width: 800px;
}

.client-edit::before {
  content: '';
  position: absolute;
  top: 0;
  left: 0;
  width: 100%;
  height: 2px;
  background: linear-gradient(90deg, transparent, #00f7ff, transparent);
  transform: translateX(-100%);
  transition: transform 0.6s ease;
}

.client-edit:hover::before {
  transform: translateX(100%);
}

/* Header Styling */
.client-edit h2 {
  font-size: 2.5rem;
  color: #ffffff;
  text-shadow: 0 0 10px rgba(0, 247, 255, 0.5);
  margin-bottom: 30px;
  text-align: center;
  position: relative;
  padding-bottom: 15px;
}

.client-edit h2::after {
  content: '';
  position: absolute;
  bottom: 0;
  left: 50%;
  transform: translateX(-50%);
  width: 150px;
  height: 2px;
  background: linear-gradient(90deg, transparent, #00f7ff, transparent);
}

/* Form Layout */
.client-edit form {
  display: flex;
  flex-direction: column;
  gap: 25px;
  width: 100%;
  max-width: 700px;
  margin: 0 auto;
}

/* Form Group */
```

```
.form-group {
  position: relative;
}

/* Label Styling */
.client-edit label {
  display: block;
  font-size: 1.1rem;
  color: #e6e6f0;
  margin-bottom: 8px;
  text-shadow: 0 0 5px rgba(0, 247, 255, 0.3);
}

/* Input and Textarea Styles */
.client-edit input,
.client-edit textarea {
  width: 100%;
  background: rgba(30, 30, 47, 0.9);
  border: 1px solid rgba(0, 247, 255, 0.1);
  padding: 12px 15px;
  border-radius: 8px;
  color: #ffffff;
  font-size: 1rem;
  transition: all 0.3s ease;
}

.client-edit input:focus,
.client-edit textarea:focus {
  border-color: #00f7ff;
  box-shadow: 0 0 15px rgba(0, 247, 255, 0.1);
  outline: none;
  background: rgba(30, 30, 47, 0.95);
}

.client-edit textarea {
  min-height: 120px;
  resize: vertical;
}

/* Button Container */
.button-group {
  display: flex;
  gap: 15px;
  justify-content: center;
  margin-top: 20px;
}

/* Button Styles */
.client-edit button {
  background: linear-gradient(135deg, #00f7ff, #2e59d9);
  color: #ffffff;
  border: none;
  padding: 12px 30px;
  border-radius: 8px;
  cursor: pointer;
  font-size: 1.1rem;
  font-weight: 500;
```

```
  transition: all 0.3s ease;
  text-shadow: 0 0 10px rgba(0, 247, 255, 0.3);
  position: relative;
  overflow: hidden;
}

.client-edit button:hover {
  transform: translateY(-2px);
  box-shadow: 0 4px 15px rgba(0, 247, 255, 0.2);
}

.client-edit button:active {
  transform: translateY(0);
}

/* Cancel Button */
.client-edit button.cancel {
  background: linear-gradient(135deg, #ff4d4d, #cc0000);
}

/* Loading State */
.client-edit .loading {
  text-align: center;
  font-size: 1.2rem;
  color: #00f7ff;
  text-shadow: 0 0 10px rgba(0, 247, 255, 0.5);
  margin: 20px 0;
  font-style: italic;
}

/* Error Messages */
.error-message {
  color: #ff4d4d;
  font-size: 0.9rem;
  margin-top: 5px;
  text-shadow: 0 0 5px rgba(255, 77, 77, 0.3);
}

/* Success Message */
.success-message {
  color: #00ff9d;
  font-size: 0.9rem;
  margin-top: 5px;
  text-shadow: 0 0 5px rgba(0, 255, 157, 0.3);
}

/* Optional Fields */
.optional-field::after {
  content: '(optional)';
  font-size: 0.8rem;
  color: #b3b3c1;
  margin-left: 5px;
}

/* Required Fields */
.required-field::after {
  content: '*';
```

```
  color: #ff4d4d;
  margin-left: 5px;
}

/* Responsive Design */
@media (max-width: 768px) {
  .client-edit {
      padding: 20px;
      margin: 15px;
  }

  .client-edit h2 {
      font-size: 2rem;
  }

  .client-edit form {
      gap: 20px;
  }

  .button-group {
      flex-direction: column;
  }

  .client-edit button {
      width: 100%;
  }
}

/* Animation for Form Submission */
@keyframes submitPulse {
  0% {
      box-shadow: 0 0 0 0 rgba(0, 247, 255, 0.4);
  }
  70% {
      box-shadow: 0 0 0 10px rgba(0, 247, 255, 0);
  }
  100% {
      box-shadow: 0 0 0 0 rgba(0, 247, 255, 0);
  }
}

.submitting button {
  animation: submitPulse 1.5s infinite;
}
```

src\app\client\client-edit\client-edit.component.html

```
<body style="background-color: black;">
<app-navbar></app-navbar>
<div class="client-edit">
    <h2>Edit Client</h2>
    <div *ngIf="client">
      <form (ngSubmit)="onSubmit()">
        <label for="name">Name:</label>
```

```
        <input type="text" id="name" [(ngModel)]="client.name" name="name" required />

        <label for="contact">Contact:</label>
        <input type="text" id="contact" [(ngModel)]="client.contact" name="contact" required />

        <label for="summary">Summary:</label>
        <textarea id="summary" [(ngModel)]="client.summary" name="summary"></textarea>

        <button type="submit">Save</button>
      </form>
    </div>
    <div *ngIf="!client">
      <p>Loading client details...</p>
    </div>
  </div>
<app-footer></app-footer>
</body>
```

src\app\client\client-edit\client-edit.component.spec.ts

```
import { ComponentFixture, TestBed } from '@angular/core/testing';

import { ClientEditComponent } from './client-edit.component';

describe('ClientEditComponent', () => {
  let component: ClientEditComponent;
  let fixture: ComponentFixture<ClientEditComponent>;

  beforeEach(async () => {
    await TestBed.configureTestingModule({
      declarations: [ ClientEditComponent ]
    })
    .compileComponents();

    fixture = TestBed.createComponent(ClientEditComponent);
    component = fixture.componentInstance;
    fixture.detectChanges();
  });

  it('should create', () => {
    expect(component).toBeTruthy();
  });
});
```

src\app\client\client-edit\client-edit.component.ts

```
import { Component, OnInit } from '@angular/core';
import { ActivatedRoute, Router } from '@angular/router';

@Component({
  selector: 'app-client-edit',
```

```
  templateUrl: './client-edit.component.html',
  styleUrls: ['./client-edit.component.css']
})
export class ClientEditComponent implements OnInit {
  clientId!: number;
  client: any;

  constructor(private route: ActivatedRoute, private router: Router) {}


  ngOnInit(): void {
    this.clientId = Number(this.route.snapshot.paramMap.get('id')); // Get the client ID from the r
    this.fetchClientDetails();
  }

  fetchClientDetails(): void {
    // Simulate fetching client data (replace with actual API call if needed)
    const clients = [
      { id: 1, name: 'Alice Johnson', contact: 'alice@example.com', summary: 'Regular client from N
      { id: 2, name: 'Bob Smith', contact: 'bob@example.com', summary: 'Long-term client.' },
      { id: 3, name: 'Charlie Brown', contact: 'charlie@example.com', summary: 'New client in Calif
      { id: 4, name: 'David Green', contact: 'david@example.com', summary: 'Frequent buyer.' },
      { id: 5, name: 'Eva White', contact: 'eva@example.com', summary: 'VIP client from LA.' }
    ];

    this.client = clients.find(client => client.id === this.clientId); // Fetch client data based o
  }

  onSubmit(): void {
    // Handle form submission logic here (e.g., save client data)
    console.log('Updated Client:', this.client);
    alert('Client details updated successfully!');
    this.router.navigate(['/client-list']); // Navigate back to client list after saving
  }
}
```

src\app\client\client-list\client-list.component.css

```
/* Client List Container */
.client-list {
  padding: 20px;
  min-height: 100vh;
  background: linear-gradient(135deg, rgba(30, 30, 47, 0.95), rgba(30, 30, 47, 0.85));
  backdrop-filter: blur(10px);
}

.client-list h2 {
  text-align: center;
  font-size: 2.5rem;
  color: #ffffff;
  text-shadow: 0 0 10px rgba(0, 247, 255, 0.5);
  margin-bottom: 30px;
  position: relative;
  padding-bottom: 15px;
```

```
}

.client-list h2::after {
  content: '';
  position: absolute;
  bottom: 0;
  left: 50%;
  transform: translateX(-50%);
  width: 150px;
  height: 2px;
  background: linear-gradient(90deg, transparent, #00f7ff, transparent);
}

/* Search Bar */
.search-bar {
  margin-bottom: 30px;
  text-align: center;
}

.search-bar input {
  width: 100%;
  max-width: 500px;
  padding: 12px 20px;
  border: 1px solid rgba(0, 247, 255, 0.2);
  border-radius: 8px;
  background: rgba(42, 42, 59, 0.7);
  color: #ffffff;
  font-size: 1rem;
  transition: border-color 0.3s ease, box-shadow 0.3s ease;
}

.search-bar input:focus {
  outline: none;
  border-color: #00f7ff;
  box-shadow: 0 0 10px rgba(0, 247, 255, 0.2);
}

/* Add Client Section */
.add-client {
  background: linear-gradient(145deg, rgba(42, 42, 59, 0.9), rgba(42, 42, 59, 0.7));
  padding: 25px;
  border-radius: 15px;
  margin-bottom: 30px;
  border: 1px solid rgba(0, 247, 255, 0.1);
  box-shadow: 0 4px 20px rgba(0, 0, 0, 0.3);
}

.add-client h3 {
  font-size: 1.5rem;
  margin-bottom: 20px;
  color: #ffffff;
}

.add-client input {
  width: 100%;
  padding: 10px;
  margin-bottom: 15px;
```

```
  border: 1px solid rgba(0, 247, 255, 0.2);
  border-radius: 8px;
  background: rgba(42, 42, 59, 0.7);
  color: #ffffff;
}

.add-client input:focus {
  outline: none;
  border-color: #00f7ff;
}

/* Client Cards */
.client-card-container {
  display: grid;
  grid-template-columns: repeat(auto-fit, minmax(300px, 1fr));
  gap: 20px;
}

.client-card .card {
  background: linear-gradient(145deg, rgba(42, 42, 59, 0.9), rgba(42, 42, 59, 0.7));
  padding: 25px;
  border-radius: 15px;
  border: 1px solid rgba(0, 247, 255, 0.1);
  box-shadow: 0 4px 20px rgba(0, 0, 0, 0.3);
  transition: transform 0.3s ease, box-shadow 0.3s ease;
  position: relative;
}

.client-card .card:hover {
  transform: translateY(-5px);
  box-shadow: 0 8px 30px rgba(0, 247, 255, 0.15);
}

.client-card h3 {
  font-size: 1.4rem;
  color: #00f7ff;
  margin-bottom: 15px;
  text-shadow: 0 0 10px rgba(0, 247, 255, 0.3);
}

.client-card p {
  color: #b3b3c1;
  margin-bottom: 10px;
}

.client-card strong {
  color: #ffffff;
}

/* Buttons */
button {
  padding: 8px 20px;
  margin-right: 10px;
  border: none;
  border-radius: 5px;
  background: rgba(0, 247, 255, 0.1);
  color: #00f7ff;
```

```
  cursor: pointer;
  transition: transform 0.3s ease, box-shadow 0.3s ease;
  border: 1px solid rgba(0, 247, 255, 0.2);
}

button:hover {
  background: rgba(0, 247, 255, 0.2);
  transform: translateY(-2px);
  box-shadow: 0 0 10px rgba(0, 247, 255, 0.3);
}

button:active {
  transform: translateY(0);
}

/* Navbar and Footer */
app-navbar, app-footer {
  background-color: rgba(18, 18, 33, 0.9);
  color: #b3b3c1;
  padding: 15px 0;
  backdrop-filter: blur(10px);
  border-bottom: 1px solid rgba(0, 247, 255, 0.1);
}

app-footer {
  border-top: 1px solid rgba(0, 247, 255, 0.1);
  text-align: center;
}

/* Responsive Design */
@media (max-width: 768px) {
  .client-list h2 {
    font-size: 2rem;
  }

  .client-card-container {
    grid-template-columns: 1fr;
  }

  .add-client {
    padding: 20px;
  }

  button {
    width: 100%;
    margin-bottom: 10px;
    margin-right: 0;
  }
}
```

src\app\client\client-list\client-list.component.html

```
<app-navbar></app-navbar>
<div class="client-list" style="background-color: black;">
```

```
  <h2 style="color:white;">Client List</h2>

  <div class="search-bar">
    <input type="text" [(ngModel)]="searchQuery" placeholder="Search clients by name..." (input)="f
  </div>

  <div class="add-client">
    <h3  style="color: aliceblue;">Add New Client</h3>
    <input type="text" [(ngModel)]="newClientName" placeholder="Enter client name" />
    <input type="text" [(ngModel)]="newClientContact" placeholder="Enter contact info" />
    <input type="text" [(ngModel)]="newClientSummary" placeholder="Enter client summary" />
    <br /><br />
    <button  (click)="addClient()">Add Client</button>
  </div>



  <div class="client-card-container">
    <div class="client-card" *ngFor="let client of filteredClients">
      <div class="card">
        <h3>{{ client.name }}</h3>
        <p><strong>Contact:</strong> {{ client.contact }}</p>
        <p><strong>Summary:</strong> {{ client.summary }}</p><br><br><br>

        <button (click)="viewClient(client.id)">View</button>
        <button (click)="goToEdit(client.id)">Edit</button>
        <button (click)="deleteClient(client.id)">Delete</button>
      </div>
    </div>
  </div>
</div>
<app-footer></app-footer>
```

src\app\client\client-list\client-list.component.spec.ts

```
import { ComponentFixture, TestBed } from '@angular/core/testing';

import { ClientListComponent } from './client-list.component';

describe('ClientListComponent', () => {
  let component: ClientListComponent;
  let fixture: ComponentFixture<ClientListComponent>;

  beforeEach(async () => {
    await TestBed.configureTestingModule({
      declarations: [ ClientListComponent ]
    })
    .compileComponents();

    fixture = TestBed.createComponent(ClientListComponent);
    component = fixture.componentInstance;
    fixture.detectChanges();
  });
  it('should create', () => {
```

```
    expect(component).toBeTruthy();
  });
});
```

src\app\client\client-list\client-list.component.ts

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

@Component({
  selector: 'app-client-list',
  templateUrl: './client-list.component.html',
  styleUrls: ['./client-list.component.css']
})
export class ClientListComponent implements OnInit {
  clients = [
    { id: 1, name: 'Alice Johnson', contact: 'alice@example.com', summary: 'Regular client from NY.
    { id: 2, name: 'Bob Smith', contact: 'bob@example.com', summary: 'Long-term client.' },
    { id: 3, name: 'Charlie Brown', contact: 'charlie@example.com', summary: 'New client in Califor
    { id: 4, name: 'David Green', contact: 'david@example.com', summary: 'Frequent buyer.' },
    { id: 5, name: 'Eva White', contact: 'eva@example.com', summary: 'VIP client from LA.' }
  ];

  filteredClients = [...this.clients]; // Holds the filtered list of clients based on search query
  searchQuery = '';

  newClientName = '';
  newClientContact = '';
  newClientSummary = '';

  constructor(private router: Router) {}

  ngOnInit(): void {}

  addClient(): void {
    if (this.newClientName && this.newClientContact && this.newClientSummary) {
      const newClient = {
        id: this.clients.length + 1,
        name: this.newClientName,
        contact: this.newClientContact,
        summary: this.newClientSummary
      };
      this.clients.push(newClient);
      this.filteredClients.push(newClient);
      this.clearNewClientFields();
      alert('Client added successfully!');
    } else {
      alert('Please fill in all fields to add a new client.');
    }
  }

  clearNewClientFields(): void {
    this.newClientName = '';
    this.newClientContact = '';
```

```
    this.newClientSummary = '';
  }

  filterClients(): void {
    if (this.searchQuery.trim()) {
      this.filteredClients = this.clients.filter(client =>
        client.name.toLowerCase().includes(this.searchQuery.toLowerCase())
      );
    } else {
      this.filteredClients = [...this.clients]; // Reset to all clients if search is empty
    }
  }

  viewClient(id: number): void {
    this.router.navigate(['/client-details', id]);
  }

  goToEdit(id: number): void {
    this.router.navigate(['/client-edit', id]);
  }

  deleteClient(id: number): void {
    this.clients = this.clients.filter(client => client.id !== id);
    this.filteredClients = this.filteredClients.filter(client => client.id !== id);
    alert('Client deleted successfully!');
  }
}
```

src\app\dash\dash-routing.module.ts

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

const routes: Routes = [
  { path: 'dashboard', component: DashboardComponent }  // The route for dashboard inside DashModul
];


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

src\app\dash\dash.module.ts

```
import { NgModule } from '@angular/core';
import { CommonModule } from '@angular/common';
import { SharedModule } from '../shared/shared.module';
import { DashboardComponent } from './dashboard/dashboard.component';
```

```
import { DashRoutingModule } from './dash-routing.module';

@NgModule({
  declarations: [
    DashboardComponent
  ],
  imports: [
    CommonModule,
    SharedModule,
    DashRoutingModule
  ]
})
export class DashModule { }
```

src\app\dash\dashboard\dashboard.component.css

```
/* General Styles */
body {
    font-family: 'Arial', sans-serif;
    margin: 0;
    padding: 0;
    background-color: #1e1e2f;
    color: #fff;
    position: relative;
}

body::before {
    content: '';
    position: fixed;
    top: 0;
    left: 0;
    width: 100%;
    height: 100%;
    background: url('image1.jpg') no-repeat center center;
    background-size: cover;
    opacity: 0.2;
    z-index: -1;
}

#app-root {
    display: flex;
    flex-direction: column;
    min-height: 100vh;
    background: linear-gradient(135deg, rgba(30, 30, 47, 0.95), rgba(30, 30, 47, 0.85));
    backdrop-filter: blur(10px);
}

main {
    flex: 1;
    padding: 20px;
}

/* Dashboard Header */
.dashboard-header {
```

```
    text-align: center;
    margin-bottom: 20px;
    position: relative;
    padding: 30px 0;
}

.dashboard-header::after {
    content: '';
    position: absolute;
    bottom: 0;
    left: 50%;
    transform: translateX(-50%);
    width: 150px;
    height: 2px;
    background: linear-gradient(90deg, transparent, #00f7ff, transparent);
}

.dashboard-header h1 {
    font-size: 2.5rem;
    color: #ffffff;
    text-shadow: 0 0 10px rgba(0, 247, 255, 0.5);
    margin-bottom: 15px;
}

.dashboard-header p {
    font-size: 1.1rem;
    color: #b3b3c1;
    max-width: 600px;
    margin: 0 auto;
}

/* Summary Cards */
.summary-cards {
    display: grid;
    grid-template-columns: repeat(auto-fit, minmax(250px, 1fr));
    gap: 20px;
    margin-bottom: 30px;
}

.card {
    background: linear-gradient(145deg, rgba(42, 42, 59, 0.9), rgba(42, 42, 59, 0.7));
    padding: 25px;
    border-radius: 15px;
    text-align: center;
    box-shadow: 0 4px 20px rgba(0, 0, 0, 0.3);
    border: 1px solid rgba(0, 247, 255, 0.1);
    backdrop-filter: blur(5px);
    transition: transform 0.3s ease, box-shadow 0.3s ease;
    position: relative;
    overflow: hidden;
}

.card::before {
    content: '';
    position: absolute;
    top: 0;
    left: 0;
```

```
    width: 100%;
    height: 2px;
    background: linear-gradient(90deg, transparent, #00f7ff, transparent);
    transform: translateX(-100%);
    transition: transform 0.6s ease;
}

.card:hover {
    transform: translateY(-5px);
    box-shadow: 0 8px 30px rgba(0, 247, 255, 0.15);
}

.card:hover::before {
    transform: translateX(100%);
}

.card h3 {
    font-size: 1.2rem;
    margin-bottom: 15px;
    color: #e6e6f0;
}

.card p {
    font-size: 2rem;
    font-weight: bold;
    color: #00f7ff;
    text-shadow: 0 0 10px rgba(0, 247, 255, 0.3);
}

/* Recent Activity and Quick Links */
.container-row {
    display: flex;
    flex-wrap: wrap;
    gap: 20px;
}

.recent-activity, .quick-links {
    flex: 1;
    background: linear-gradient(145deg, rgba(42, 42, 59, 0.9), rgba(42, 42, 59, 0.7));
    padding: 25px;
    border-radius: 15px;
    box-shadow: 0 4px 20px rgba(0, 0, 0, 0.3);
    border: 1px solid rgba(0, 247, 255, 0.1);
    backdrop-filter: blur(5px);
    min-width: 300px;
}

.recent-activity h2, .quick-links h2 {
    font-size: 1.5rem;
    margin-bottom: 20px;
    color: #ffffff;
    position: relative;
    padding-bottom: 10px;
}

.recent-activity h2::after, .quick-links h2::after {
    content: '';
```

```
    position: absolute;
    bottom: 0;
    left: 0;
    width: 50px;
    height: 2px;
    background: #00f7ff;
}

.recent-activity ul, .quick-links ul {
    list-style: none;
    padding: 0;
}

.recent-activity li {
    margin-bottom: 15px;
    font-size: 1rem;
    color: #b3b3c1;
    padding: 10px;
    border-radius: 8px;
    transition: background-color 0.3s ease;
}

.recent-activity li:hover {
    background-color: rgba(0, 247, 255, 0.05);
}

.quick-links li {
    margin-bottom: 15px;
}

.quick-links a {
    color: #00f7ff;
    text-decoration: none;
    font-size: 1rem;
    display: block;
    padding: 10px;
    border-radius: 8px;
    transition: all 0.3s ease;
}

.quick-links a:hover {
    background-color: rgba(0, 247, 255, 0.05);
    transform: translateX(10px);
    text-shadow: 0 0 10px rgba(0, 247, 255, 0.5);
}

/* Footer */
app-footer {
    background-color: rgba(18, 18, 33, 0.9);
    color: #b3b3c1;
    text-align: center;
    padding: 15px 0;
    font-size: 0.9rem;
    backdrop-filter: blur(10px);
    border-top: 1px solid rgba(0, 247, 255, 0.1);
}
/* Responsive Design */
```

```
@media (max-width: 768px) {
    .container-row {
        flex-direction: column;
    }

    .card, .recent-activity, .quick-links {
        padding: 20px;
    }

    .dashboard-header h1 {
        font-size: 2rem;
    }
}
```

src\app\dash\dashboard\dashboard.component.html

```
<div id="app-root" style="background: linear-gradient(rgba(0, 0, 0, 0.92), rgba(0, 0, 0, 0.92)), ur
    <main>
        <div class="dashboard-container">
            <div class="dashboard-header">
                <h1>Anna's Dashboard</h1>
                <p>Your one-stop overview for client and project management</p>
            </div>

            <app-navbar></app-navbar>

            <div class="summary-cards" style="color: white;">
                <div class="card">
                    <h3>Total Clients</h3>
                    <p>{{ totalClients }}</p>
                </div>
                <div class="card">
                    <h3>Active Projects</h3>
                    <p>{{ activeProjects }}</p>
                </div>
                <div class="card">
                    <h3>Upcoming Meetings</h3>
                    <p>{{ upcomingMeetings }}</p>
                </div>
            </div>

            <!-- Recent Activity and Quick Links -->
            <div class="container-row">
                <!-- Recent Activity Section -->
                <div class="recent-activity">
                    <h2>Recent Activity</h2>
                    <ul>
                        <li>Client "ABC Corp" added new project</li>
                        <li>Meeting with "XYZ Ltd" scheduled for tomorrow</li>
                        <li>Client "DEF Inc" updated billing info</li>
                    </ul>
                </div>

                <!-- Quick Links Section -->
```

```
                <div class="quick-links">
                    <h2>Quick Links</h2>
                    <ul>
                        <li><a routerLink="/client/client-list">Clients</a></li>
                        <li><a routerLink="/project/project-list">Projects</a></li>
                        <li><a routerLink="/meetings/meeting-list">Meetings</a></li>
                    </ul>
                </div>
            </div>

            <!-- Router Outlet -->
        </div>
    </main>
 </div>
 <app-footer></app-footer>
```

src\app\dash\dashboard\dashboard.component.spec.ts

```
import { ComponentFixture, TestBed } from '@angular/core/testing';

import { DashboardComponent } from './dashboard.component';

describe('DashboardComponent', () => {
  let component: DashboardComponent;
  let fixture: ComponentFixture<DashboardComponent>;

  beforeEach(async () => {
    await TestBed.configureTestingModule({
      declarations: [ DashboardComponent ]
    })
    .compileComponents();

    fixture = TestBed.createComponent(DashboardComponent);
    component = fixture.componentInstance;
    fixture.detectChanges();
  });

  it('should create', () => {
    expect(component).toBeTruthy();
  });
});
```

src\app\dash\dashboard\dashboard.component.ts

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

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

```
export class DashboardComponent implements OnInit {
  // Declare the properties
  totalClients: number = 0;  // Example: You can replace 0 with the actual value
  activeProjects: number = 0; // Example: Replace with the actual count of active projects
  upcomingMeetings: number = 0; // Example: Replace with the actual count of upcoming meetings

  constructor() { }

  ngOnInit(): void {
    // Here you can fetch or set the actual values for these properties
    // For now, we set them statically as examples.
    this.totalClients = 100; // Set this dynamically if needed
    this.activeProjects = 50; // Set this dynamically if needed
    this.upcomingMeetings = 10; // Set this dynamically if needed
  }
}
```

src\app\meetings\meetings-routing.module.ts

```
import { NgModule } from '@angular/core';
import { RouterModule, Routes } from '@angular/router';
import { MeetingCalendarComponent } from './meeting-calender/meeting-calendar.component';
import { MeetingDetailsComponent } from './meeting-details/meeting-details.component';
import { MeetingListComponent } from './meeting-list/meeting-list.component';

const routes: Routes = [
  {path:'meeting-calender',component:MeetingCalendarComponent},
  {path:'meeting-details',component:MeetingDetailsComponent},
  {path:'meeting-list',component:MeetingListComponent},

];

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

src\app\meetings\meetings.model.ts

```
// meeting.model.ts
export interface Meeting {
    id: string;
    title: string;
    date: Date;
    duration: number;
    description?: string;
    status: 'Scheduled' | 'Cancelled' | 'Completed';
    participants: string[];
}
```

src\app\meetings\meetings.module.ts

```
import { NgModule } from '@angular/core';
import { CommonModule,DatePipe } from '@angular/common';
import { FormsModule, ReactiveFormsModule } from '@angular/forms';
import { MeetingCalendarComponent } from './meeting-calender/meeting-calendar.component';
import { MeetingDetailsComponent } from './meeting-details/meeting-details.component';
import { MeetingListComponent } from './meeting-list/meeting-list.component';
import { MeetingsRoutingModule } from './meetings-routing.module';
import { SharedModule } from '../shared/shared.module';

@NgModule({
  declarations: [
    MeetingCalendarComponent,
    MeetingDetailsComponent,
    MeetingListComponent,
  ],
  imports: [
    CommonModule,
    FormsModule,
    MeetingsRoutingModule,
    SharedModule,
    ReactiveFormsModule
  ],
  exports: [
    MeetingCalendarComponent,
    MeetingDetailsComponent,
    MeetingListComponent,
  ],
  providers:[DatePipe]
})
export class MeetingsModule { }
```

src\app\meetings\shared.css

```
:root {
    --primary-color: #2196f3;
    --primary-hover: #1976d2;
    --danger-color: #f44336;
    --success-color: #4caf50;
    --warning-color: #ff9800;
    --background-dark: #121212;
    --background-card: #1e1e1e;
    --background-hover: #2c2c2c;
    --text-primary: #ffffff;
    --text-secondary: #b3b3b3;
    --border-color: #333333;
  }

  body {
```

```
  background-color: var(--background-dark);
  color: var(--text-primary);
  font-family: 'Segoe UI', Roboto, Arial, sans-serif;
}

.container {
  max-width: 1200px;
  margin: 0 auto;
  padding: 2rem;
}

.header {
  display: flex;
  justify-content: space-between;
  align-items: center;
  margin-bottom: 2rem;
}

.header h2 {
  color: var(--text-primary);
  margin: 0;
}

button {
  background-color: var(--primary-color);
  color: white;
  border: none;
  padding: 0.75rem 1.5rem;
  border-radius: 4px;
  cursor: pointer;
  transition: background-color 0.3s ease;
}

button:hover {
  background-color: var(--primary-hover);
}

button:disabled {
  opacity: 0.6;
  cursor: not-allowed;
}

.status-badge {
  padding: 0.25rem 0.75rem;
  border-radius: 12px;
  font-size: 0.875rem;
  font-weight: 500;
}

.status-badge.scheduled {
  background-color: var(--primary-color);
  color: white;
}

.status-badge.completed {
  background-color: var(--success-color);
  color: white;
```

```
  }

  .status-badge.cancelled {
    background-color: var(--danger-color);
    color: white;
  }

  .status-badge.pending {
    background-color: var(--warning-color);
    color: black;
  }
```

src\app\meetings\meeting-calender\meeting-calendar.component.css

```
.calendar-container {
  padding: 20px;
  background: #1a1a1a;
  border-radius: 8px;
  box-shadow: 0 2px 10px rgba(0, 0, 0, 0.3);
  max-width: 1200px;
  margin: 0 auto;
  color: #e0e0e0;
}

.calendar-header {
  display: flex;
  justify-content: space-between;
  align-items: center;
  margin-bottom: 20px;
}

.calendar-header h2 {
  margin: 0;
  font-size: 1.5rem;
  color: #ffffff;
}

.nav-button {
  background: none;
  border: none;
  font-size: 1.2rem;
  padding: 8px 16px;
  cursor: pointer;
  border-radius: 4px;
  color: #e0e0e0;
}

.nav-button:hover {
  background-color: #333333;
}

.calendar-grid {
  display: grid;
```

```
  grid-template-columns: repeat(7, 1fr);
  gap: 4px;
  margin-bottom: 20px;
}

.weekday-header {
  text-align: center;
  font-weight: 500;
  padding: 10px;
  color: #b0b0b0;
  background-color: #2a2a2a;
}

.calendar-day {
  aspect-ratio: 1;
  padding: 8px;
  border: 1px solid #333333;
  cursor: pointer;
  display: flex;
  flex-direction: column;
  align-items: center;
  background-color: #242424;
}

.date-number {
  font-size: 0.9rem;
  margin-bottom: 4px;
  color: #e0e0e0;
}

.today {
  background-color: #1a3045;
  border-color: #0077ee;
}

.selected {
  background-color: #0077ee;
  color: white;
}

.other-month {
  opacity: 0.3;
}

.meeting-indicators {
  display: flex;
  gap: 2px;
  justify-content: center;
}

.meeting-dot {
  width: 6px;
  height: 6px;
  border-radius: 50%;
  background-color: #0077ee;
}
.meeting-dot.completed {
```

```
  background-color: #2ea043;
}

.meeting-dot.cancelled {
  background-color: #e5534b;
}

.selected-date-info {
  border-top: 1px solid #333333;
  padding-top: 20px;
}

.meetings-list {
  margin-top: 15px;
  display: flex;
  flex-direction: column;
  gap: 10px;
}

.meeting-card {
  display: flex;
  padding: 15px;
  background-color: #2a2a2a;
  border-radius: 6px;
  border-left: 4px solid #0077ee;
}

.meeting-time {
  min-width: 120px;
  font-weight: 500;
  color: #e0e0e0;
}

.meeting-content {
  flex-grow: 1;
}

.meeting-content h4 {
  margin: 0 0 5px 0;
  color: #ffffff;
}

.meeting-content p {
  margin: 0;
  color: #b0b0b0;
  font-size: 0.9rem;
}

.status-badge {
  display: inline-block;
  padding: 2px 8px;
  border-radius: 12px;
  font-size: 0.8rem;
  margin-top: 5px;
}

.status-badge.scheduled {
```

```
  background-color: #1a3045;
  color: #0077ee;
}

.status-badge.completed {
  background-color: #1b2a1e;
  color: #2ea043;
}

.status-badge.cancelled {
  background-color: #2d1f1f;
  color: #e5534b;
}

.add-meeting-btn {
  width: 100%;
  padding: 10px;
  background-color: #0077ee;
  color: white;
  border: none;
  border-radius: 4px;
  cursor: pointer;
  font-size: 1rem;
}

.add-meeting-btn:hover {
  background-color: #0066cc;
}

.add-meeting-form {
  position: fixed;
  top: 50%;
  left: 50%;
  transform: translate(-50%, -50%);
  background: #1a1a1a;
  padding: 20px;
  border-radius: 8px;
  box-shadow: 0 2px 20px rgba(0, 0, 0, 0.4);
  width: 90%;
  max-width: 500px;
  z-index: 1000;
}

.form-group {
  margin-bottom: 15px;
}

.form-group label {
  display: block;
  margin-bottom: 5px;
  color: #e0e0e0;
}

.form-group input,
.form-group textarea {
  width: 100%;
  padding: 8px;
```

```
  border: 1px solid #333333;
  border-radius: 4px;
  background-color: #242424;
  color: #e0e0e0;
}

.form-group input:focus,
.form-group textarea:focus {
  border-color: #0077ee;
  outline: none;
}

.form-buttons {
  display: flex;
  gap: 10px;
  justify-content: flex-end;
  margin-top: 20px;
}

.form-buttons button {
  padding: 8px 16px;
  border: none;
  border-radius: 4px;
  cursor: pointer;
}

.form-buttons button[type="submit"] {
  background-color: #0077ee;
  color: white;
}

.form-buttons button[type="button"] {
  background-color: #333333;
  color: #e0e0e0;
}

.form-buttons button:hover {
  opacity: 0.9;
}

.form-buttons button:disabled {
  background-color: #333333;
  color: #666666;
  cursor: not-allowed;
  opacity: 0.7;
}
```

src\app\meetings\meeting-calender\meeting-calendar.component.html

```
<body style="background-color: black;">

  <app-navbar></app-navbar>
<div class="calendar-container" style="background-color: black;">
```

```
<!-- Calendar Header -->
<div class="calendar-header">
  <button class="nav-button" (click)="previousMonth()">&lt;</button>
  <h2>{{ formatDate(currentMonth) }}</h2>
  <button class="nav-button" (click)="nextMonth()">&gt;</button>
</div>

<!-- Calendar Grid -->
<div class="calendar-grid">
  <!-- Weekday Headers -->
  <div class="weekday-header" *ngFor="let day of weekDays">
    {{ day }}
  </div>

  <!-- Calendar Days -->
  <div class="calendar-day"
       *ngFor="let date of calendarDays"
       [class.today]="isToday(date)"
       [class.selected]="selectedDate === date"
       [class.other-month]="!isCurrentMonth(date)"
       (click)="selectDate(date)">
    <span class="date-number">{{ date.getDate() }}</span>
    <!-- Meeting Indicators -->
    <div class="meeting-indicators">
      <span class="meeting-dot"
            *ngFor="let meeting of getMeetingsForDay(date)"
            [class]="meeting.status">
      </span>
    </div>
  </div>
</div>

<!-- Selected Date Meetings -->
<div class="selected-date-info" *ngIf="selectedDate">
  <h3>Meetings for {{ selectedDate | date:'mediumDate' }}</h3>

  <!-- Meeting List -->
  <div class="meetings-list">
    <div class="meeting-card" *ngFor="let meeting of getMeetingsForDay(selectedDate)">
      <div class="meeting-time">
        {{ meeting.startTime }} - {{ meeting.endTime }}
      </div>
      <div class="meeting-content">
        <h4>{{ meeting.title }}</h4>
        <p>Client: {{ meeting.clientName }}</p>
        <p>Project: {{ meeting.projectName }}</p>
        <span class="status-badge" [class]="meeting.status">
          {{ meeting.status }}
        </span>
      </div>
    </div>

    <!-- Add Meeting Button -->
    <button class="add-meeting-btn" (click)="toggleAddMeetingForm()">
      + Add Meeting
    </button>
  </div>
```

```
    </div>

    <!-- Simple Add Meeting Form -->
    <div class="add-meeting-form" *ngIf="showAddMeetingForm">
      <form #meetingForm="ngForm" (ngSubmit)="addMeeting(meetingForm.value)">
        <div class="form-group">
          <label for="title">Title</label>
          <input type="text" id="title" name="title" ngModel required>
        </div>

        <div class="form-group">
          <label for="startTime">Start Time</label>
          <input type="time" id="startTime" name="startTime" ngModel required>
        </div>

        <div class="form-group">
          <label for="endTime">End Time</label>
          <input type="time" id="endTime" name="endTime" ngModel required>
        </div>

        <div class="form-group">
          <label for="clientName">Client</label>
          <input type="text" id="clientName" name="clientName" ngModel required>
        </div>

        <div class="form-group">
          <label for="projectName">Project</label>
          <input type="text" id="projectName" name="projectName" ngModel required>
        </div>

        <div class="form-group">
          <label for="description">Description</label>
          <textarea id="description" name="description" ngModel></textarea>
        </div>

        <div class="form-buttons">
          <button type="submit" [disabled]="!meetingForm.valid">Save</button>
          <button type="button" (click)="toggleAddMeetingForm()">Cancel</button>
        </div>
      </form>
    </div>
  </div>

<app-footer></app-footer>

</body>
```

src\app\meetings\meeting-calender\meeting-calendar.component.ts

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

interface Meeting {
  id: number;
  title: string;
```

```
  date: Date;
  startTime: string;
  endTime: string;
  clientName: string;
  projectName: string;
  attendees: string[];
  status: 'scheduled' | 'completed' | 'cancelled';
  description?: string;
}

@Component({
  selector: 'app-meeting-calendar',
  templateUrl: './meeting-calendar.component.html',
  styleUrls: ['./meeting-calendar.component.css']
})
export class MeetingCalendarComponent implements OnInit {
  meetings: Meeting[] = [];
  selectedDate: Date = new Date();
  calendarDays: Date[] = [];
  currentMonth: Date = new Date();
  showAddMeetingForm = false;
  weekDays = ['Sun', 'Mon', 'Tue', 'Wed', 'Thu', 'Fri', 'Sat'];

  // Sample data - replace with actual API call
  sampleMeetings: Meeting[] = [
    {
      id: 1,
      title: 'Project Kickoff',
      date: new Date(),
      startTime: '10:00',
      endTime: '11:00',
      clientName: 'Acme Corp',
      projectName: 'Website Redesign',
      attendees: ['John Doe', 'Jane Smith'],
      status: 'scheduled',
      description: 'Initial project meeting'
    },
    {
      id: 2,
      title: 'Sprint Planning',
      date: new Date(new Date().setDate(new Date().getDate() + 2)),
      startTime: '14:00',
      endTime: '15:30',
      clientName: 'Tech Corp',
      projectName: 'Mobile App',
      attendees: ['Alice Johnson', 'Bob Wilson'],
      status: 'scheduled',
      description: 'Sprint planning session'
    }
  ];

  constructor() {}

  ngOnInit() {
    this.meetings = this.sampleMeetings;
    this.generateCalendarDays();
  }
```

```
generateCalendarDays() {
  this.calendarDays = [];
  const firstDay = new Date(this.currentMonth.getFullYear(), this.currentMonth.getMonth(), 1);
  const lastDay = new Date(this.currentMonth.getFullYear(), this.currentMonth.getMonth() + 1, 0);

  // Add previous month's days
  const firstDayOfWeek = firstDay.getDay();
  for (let i = firstDayOfWeek - 1; i >= 0; i--) {
    const day = new Date(firstDay);
    day.setDate(day.getDate() - i);
    this.calendarDays.push(day);
  }

  // Add current month's days
  for (let i = 1; i <= lastDay.getDate(); i++) {
    const day = new Date(this.currentMonth.getFullYear(), this.currentMonth.getMonth(), i);
    this.calendarDays.push(day);
  }
}

getMeetingsForDay(date: Date): Meeting[] {
  return this.meetings.filter(meeting =>
    this.isSameDay(new Date(meeting.date), date)
  );
}

private isSameDay(date1: Date, date2: Date): boolean {
  return date1.getDate() === date2.getDate() &&
         date1.getMonth() === date2.getMonth() &&
         date1.getFullYear() === date2.getFullYear();
}

nextMonth() {
  this.currentMonth = new Date(
    this.currentMonth.getFullYear(),
    this.currentMonth.getMonth() + 1,
    1
  );
  this.generateCalendarDays();
}

previousMonth() {
  this.currentMonth = new Date(
    this.currentMonth.getFullYear(),
    this.currentMonth.getMonth() - 1,
    1
  );
  this.generateCalendarDays();
}

isToday(date: Date): boolean {
  return this.isSameDay(date, new Date());
}

selectDate(date: Date) {
  this.selectedDate = date;
}
```

```
  toggleAddMeetingForm() {
    this.showAddMeetingForm = !this.showAddMeetingForm;
  }

  formatDate(date: Date): string {
    return date.toLocaleDateString('en-US', {
      month: 'long',
      year: 'numeric'
    });
  }

  isCurrentMonth(date: Date): boolean {
    return date.getMonth() === this.currentMonth.getMonth();
  }

  addMeeting(meetingData: Partial<Meeting>) {
    const newMeeting: Meeting = {
      id: this.meetings.length + 1,
      ...meetingData,
      date: this.selectedDate,
      status: 'scheduled',
      attendees: meetingData.attendees || []
    } as Meeting;

    this.meetings.push(newMeeting);
    this.showAddMeetingForm = false;
  }
}
```

src\app\meetings\meeting-calender\meeting-calender.component.spec.ts

```
import { ComponentFixture, TestBed } from '@angular/core/testing';

import { MeetingCalenderComponent } from './meeting-calendar.component';

describe('MeetingCalenderComponent', () => {
  let component: MeetingCalenderComponent;
  let fixture: ComponentFixture<MeetingCalenderComponent>;

  beforeEach(async () => {
    await TestBed.configureTestingModule({
      declarations: [ MeetingCalenderComponent ]
    })
    .compileComponents();

    fixture = TestBed.createComponent(MeetingCalenderComponent);
    component = fixture.componentInstance;
    fixture.detectChanges();
  });

  it('should create', () => {
    expect(component).toBeTruthy();
  });
```

```
});
```

src\app\meetings\meeting-details\meeting-details.component.css

```
/* Meeting Details Component */
.meeting-details {
  background-color: #0a0a0a; /* Dark background */
  border-radius: 8px;
  padding: 2rem;
  box-shadow: 0 4px 12px rgba(0, 0, 0, 0.5);
  color: #e0e0e0; /* Text color from the black and blue theme */
}

.detail-card {
  background-color: #0f0f0f; /* Darker background for detail cards */
  border-radius: 8px;
  padding: 1.5rem;
  border: 1px solid #1a1a1a; /* Subtle border */
}

.detail-section {
  margin-bottom: 2rem;
}

.detail-section h3 {
  color: #808080; /* Muted gray for section headers */
  margin-bottom: 1rem;
  font-size: 1.25rem;
  border-bottom: 1px solid #1a1a1a; /* Divider below headers */
  padding-bottom: 0.5rem;
}

.detail-row {
  display: flex;
  margin-bottom: 0.75rem;
}

.label {
  width: 120px;
  color: #808080; /* Label color for contrast */
}

.value {
  color: #e0e0e0; /* Text color for values */
}

.description {
  color: #cccccc; /* Light gray for descriptions */
  line-height: 1.6;
  background-color: #0a0a0a; /* Dark background */
  padding: 1rem;
  border-radius: 4px;
}
.participants-list {
```

```
  display: grid;
  grid-template-columns: repeat(auto-fill, minmax(200px, 1fr));
  gap: 1rem;
}

.participant {
  display: flex;
  align-items: center;
  gap: 0.5rem;
  padding: 0.75rem;
  background-color: #1a1a1a; /* Darker background for participants */
  border-radius: 4px;
  color: #e0e0e0; /* Text color */
  border: 1px solid #262626; /* Subtle border for participant cards */
}

.participant i {
  color: #00c3ff; /* Accent color for icons */
}

.loading {
  text-align: center;
  padding: 2rem;
  color: #808080; /* Muted gray for loading text */
}

.loading i {
  font-size: 2rem;
  margin-bottom: 1rem;
  color: #00c3ff; /* Accent color for icons */
}

/* Actions Container */
.actions {
  display: flex;
  justify-content: space-between;
  align-items: center;
  padding: 10px 20px;
  background-color: #222; /* Dark background for actions */
  border-radius: 8px;
  box-shadow: 0 4px 8px rgba(0, 0, 0, 0.5);
  margin-bottom: 15px;
}

.back-btn {
  display: flex;
  align-items: center;
  background-color: #333; /* Darker background for buttons */
  color: #fff; /* White text */
  border: none;
  border-radius: 6px;
  padding: 8px 16px;
  font-size: 14px;
  cursor: pointer;
  transition: background-color 0.3s ease, transform 0.2s ease;
  box-shadow: 0 2px 4px rgba(0, 0, 0, 0.3);
}
```

```
.back-btn:hover {
  background-color: #444; /* Darker hover effect */
  transform: scale(1.05);
}

.back-btn:active {
  background-color: #555; /* Active state background */
  transform: scale(0.95);
}

.back-btn i {
  margin-right: 8px;
  font-size: 16px;
}

.right-actions {
  display: flex;
  align-items: center;
}

/* Cancel Button */
.cancel-btn {
  display: flex;
  align-items: center;
  background-color: #660000; /* Dark red background */
  color: #fff; /* White text */
  border: none;
  border-radius: 6px;
  padding: 8px 16px;
  font-size: 14px;
  cursor: pointer;
  transition: background-color 0.3s ease, transform 0.2s ease;
  box-shadow: 0 2px 4px rgba(0, 0, 0, 0.3);
  margin-left: 10px;
}

.cancel-btn:hover {
  background-color: #990000; /* Darker red on hover */
  transform: scale(1.05);
}

.cancel-btn:active {
  background-color: #330000; /* Darkest red on active state */
  transform: scale(0.95);
}

.cancel-btn i {
  margin-right: 8px;
  font-size: 16px;
}
```

src\app\meetings\meeting-details\meeting-details.component.html

```
<app-navbar></app-navbar>
```

```
<div class="container">
    <div class="meeting-details">
        <div class="header">
            <h2>Meeting Details</h2>
        </div>

        <div class="content" *ngIf="meeting; else loading">
            <div class="detail-card">
                <div class="detail-section">
                    <h3>Basic Information</h3>
                    <div class="detail-row">
                        <span class="label">Title:</span>
                        <span class="value">{{ meeting.title }}</span>
                    </div>
                    <div class="detail-row">
                        <span class="label">Date & Time:</span>
                        <span class="value">{{ meeting.date | date: 'MMM d, y, h:mm a' }}</span>
                    </div>
                    <div class="detail-row">
                        <span class="label">Duration:</span>
                        <span class="value">{{ meeting.duration }} minutes</span>
                    </div>
                    <div class="detail-row">
                        <span class="label">Status:</span>
                        <span class="status-badge" [ngClass]="meeting.status.toLowerCase()">
                            {{ meeting.status }}
                        </span>
                    </div>
                </div>

                <div class="detail-section">
                    <h3>Description</h3>
                    <p class="description">{{ meeting.description || 'No description provided' }}</
                </div>

                <div class="detail-section">
                    <h3>Participants</h3>
                    <div class="participants-list">
                        <div class="participant" *ngFor="let participant of meeting.participants">
                            <i class="fas fa-user"></i>
                            <span>{{ participant }}</span>
                        </div>
                        <p *ngIf="!meeting.participants?.length">No participants added</p>
                    </div>
                </div>
            </div>

            <div class="actions">
                <button class="back-btn" (click)="goBack()">
                    <i class="fas fa-arrow-left"></i> Back to List
                </button>
                <div class="right-actions">

                    <button class="cancel-btn" (click)="cancelMeeting()" *ngIf="meeting.status ===
                        <i class="fas fa-times"></i> Cancel Meeting
                    </button>
                </div>
```

```
            </div>
        </div>

        <ng-template #loading>
            <div class="loading">
                <i class="fas fa-spinner fa-spin"></i>
                <p>Loading meeting details...</p>
            </div>
        </ng-template>
    </div>
</div>
<app-footer></app-footer>
```

src\app\meetings\meeting-details\meeting-details.component.spec.ts

```
import { ComponentFixture, TestBed } from '@angular/core/testing';

import { MeetingDetailsComponent } from './meeting-details.component';

describe('MeetingDetailsComponent', () => {
  let component: MeetingDetailsComponent;
  let fixture: ComponentFixture<MeetingDetailsComponent>;

  beforeEach(async () => {
    await TestBed.configureTestingModule({
      declarations: [ MeetingDetailsComponent ]
    })
    .compileComponents();

    fixture = TestBed.createComponent(MeetingDetailsComponent);
    component = fixture.componentInstance;
    fixture.detectChanges();
  });

  it('should create', () => {
    expect(component).toBeTruthy();
  });
});
```

src\app\meetings\meeting-details\meeting-details.component.ts

```
import { Component, OnInit } from '@angular/core';
import { ActivatedRoute, Router } from '@angular/router';

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

```
export class MeetingDetailsComponent implements OnInit {
    meeting: any = {
        id: '1',
        title: 'Team Standup',
        date: new Date('2024-12-23T10:00:00'),
        duration: 30,
        description: 'Daily team sync-up meeting',
        status: 'Scheduled',
        participants: ['john@example.com', 'jane@example.com']
    };

    constructor(
        private route: ActivatedRoute,
        private router: Router
    ) {}

    ngOnInit(): void {
        const id = this.route.snapshot.paramMap.get('id');
        console.log('Meeting ID:', id);
    }



    cancelMeeting(): void {
        if (this.meeting && confirm('Are you sure you want to cancel this meeting?')) {
            this.meeting.status = 'Cancelled';
            setTimeout(() => {
                this.router.navigate(['/meetings/meeting-list']);
            }, 1000);
        }
    }

    goBack(): void {
        this.router.navigate(['/meetings/meeting-list']);
    }
}
```

src\app\meetings\meeting-list\meeting-list.component.css

```
/* Body background and global styles */
body {
  background-color: #1e1e2f; /* Dark grey background */
  color: white;
  font-family: Arial, sans-serif;
}

/* Container styling for the meeting list */
.container {
  max-width: 1200px;
  margin: 0 auto;
  padding: 20px;
}

/* Meeting list header styles */
```

```
.list-header {
  display: flex;
  justify-content: space-between;
  align-items: center;
  margin-bottom: 20px;
}

.list-header h2 {
  color: white;
}

/* Table container styling */
.table-container {
  background-color: rgba(30, 30, 47, 0.9); /* Darker semi-transparent background */
  border-radius: 10px;
  overflow-x: auto;
}

/* Table styling */
table {
  width: 100%;
  border-collapse: collapse;
  color: white;
}

th, td {
  padding: 10px;
  text-align: left;
}

th {
  background-color: rgba(255, 255, 255, 0.1); /* Slightly transparent background for header */
}

/* Actions column buttons */
.actions-cell .action-btn {
  background-color: transparent;
  border: none;
  color: white;
  cursor: pointer;
  padding: 5px;
  margin-right: 5px;
}

.action-btn:hover {
  color: #00f7ff; /* Blue accent on hover */
}

/* Status badge */
.status-badge {
  padding: 5px 10px;
  border-radius: 5px;
  font-size: 0.8em;
}

.status-badge.scheduled {
  background-color: #4caf50; /* Green */
```

```
}

.status-badge.completed {
  background-color: #ff9800; /* Orange */
}

/* Empty state styling */
.empty-state {
  display: flex;
  flex-direction: column;
  justify-content: center;
  align-items: center;
  height: 100px;
  color: white;
  text-align: center;
}

.empty-icon {
  font-size: 50px;
  margin-bottom: 10px;
}

/* New meeting form */
.new-meeting-form {
  background-color: rgba(30, 30, 47, 0.95); /* Darker semi-transparent background for form */
  padding: 20px;
  border-radius: 8px;
  margin-top: 20px;
}

.new-meeting-form h3 {
  color: white;
  margin-bottom: 20px;
}

.new-meeting-form label {
  display: block;
  margin-bottom: 10px;
  color: white;
}

.new-meeting-form input {
  width: 100%;
  padding: 10px;
  margin-bottom: 15px;
  background-color: rgba(255, 255, 255, 0.1); /* Slightly transparent background for inputs */
  border: 1px solid rgba(255, 255, 255, 0.2);
  color: white;
  border-radius: 4px;
}

.new-meeting-form button {
  background-color: #007bff; /* Blue button color */
  border: none;
  padding: 10px 15px;
  color: white;
  border-radius: 4px;
```

```
  cursor: pointer;
  margin-right: 10px;
}

.new-meeting-form button:hover {
  background-color: #0056b3;
}

/* Footer */
app-footer {
  background-color: rgba(30, 30, 47, 0.9);
  color: white;
  text-align: center;
  padding: 10px;
}

/* Add Button */
.add-button {
  display: flex;
  align-items: center;
  justify-content: center;
  background-color: #333; /* Dark grey background */
  color: white;
  border: none;
  border-radius: 8px; /* Rounded corners */
  padding: 10px 20px; /* Spacing inside the button */
  font-size: 16px; /* Adjust the font size */
  cursor: pointer;
  transition: background-color 0.3s ease, transform 0.2s ease;
  box-shadow: 0 4px 6px rgba(0, 0, 0, 0.3); /* Subtle shadow for depth */
}

.add-button:hover {
  background-color: #444; /* Slightly lighter grey on hover */
  transform: scale(1.05); /* Slightly enlarge the button on hover */
}

.add-button:active {
  background-color: #555; /* Darker grey for active state */
  transform: scale(0.95); /* Slightly reduce size on click */
}

.add-button i {
  margin-right: 8px; /* Space between the icon and text */
  font-size: 18px; /* Adjust icon size */
}

.add-button p {
  margin: 0;
  font-weight: bold; /* Make text bold */
  font-family: 'Arial', sans-serif; /* Clean, modern font */
}
```

src\app\meetings\meeting-list\meeting-list.component.html

```
<body style="background-color: black;">
    <app-navbar></app-navbar>
<div class="container" style="height: 700px;">
    <div class="meeting-list">
        <div class="list-header">
            <h2>Meeting List</h2>
            <button class="add-button" (click)="addMeeting()">
                <i class="fas fa-plus"></i>
                <p >New Meeting</p>
            </button>
        </div>

        <div class="table-container">
            <table *ngIf="meetings.length > 0">
                <thead>
                    <tr>
                        <th>Title</th>
                        <th>Date & Time</th>
                        <th>Duration</th>
                        <th>Status</th>
                        <th>Participants</th>
                        <th class="actions-header">Actions</th>
                    </tr>
                </thead>
                <tbody>
                    <tr *ngFor="let meeting of meetings">
                        <td class="meeting-title">{{ meeting.title }}</td>
                        <td>{{ meeting.date | date: 'MMM d, y, h:mm a' }}</td>
                        <td>{{ meeting.duration }} minutes</td>
                        <td>
                            <span class="status-badge" [ngClass]="meeting.status.toLowerCase()">
                                {{ meeting.status }}
                            </span>
                        </td>
                        <td class="participants-cell">
                            <span class="participant-count">
                                <i class="fas fa-users"></i>
                                {{ meeting.participants.length || 0 }}
                            </span>
                        </td>
                        <td class="actions-cell">
                            <button class="action-btn view" (click)="viewDetails(meeting.id)" title
                                <i class="fas fa-eye"></i>
                            </button>
                            <button class="action-btn edit" (click)="editMeeting(meeting.id)" title
                                <i class="fas fa-edit"></i>
                            </button>
                            <button class="action-btn delete" (click)="cancelMeeting(meeting.id)" t
                                <i class="fas fa-times"></i>
                            </button>
                        </td>
                    </tr>
                </tbody>
            </table>

            <div *ngIf="meetings.length === 0" class="empty-state">
```

```
                <i class="fas fa-calendar-plus empty-icon"></i>
                <p>No meetings scheduled</p>
                <button class="add-button" (click)="addMeeting()">Schedule Your First Meeting</butt
            </div>

            <div *ngIf="showForm" class="new-meeting-form">
                <h3>Add New Meeting</h3>
                <label>
                    Title:
                    <input [(ngModel)]="newMeeting.title" placeholder="Enter title" />
                </label>
                <label>
                    Date & Time:
                    <input type="datetime-local" [(ngModel)]="newMeeting.date" />
                </label>
                <label>
                    Duration (minutes):
                    <input type="number" [(ngModel)]="newMeeting.duration" />
                </label>
                <label>
                    Participants:
                    <input [(ngModel)]="newMeeting.participants" placeholder="Enter participants" /
                </label>
                <button (click)="saveMeeting()">Save Meeting</button>
                <button (click)="cancelMeeting()">Cancel</button>
            </div>
        </div>
    </div>
</div>
<app-footer></app-footer>
</body>
```

src\app\meetings\meeting-list\meeting-list.component.spec.ts

```
import { ComponentFixture, TestBed } from '@angular/core/testing';

import { MeetingListComponent } from './meeting-list.component';

describe('MeetingListComponent', () => {
  let component: MeetingListComponent;
  let fixture: ComponentFixture<MeetingListComponent>;

  beforeEach(async () => {
    await TestBed.configureTestingModule({
      declarations: [ MeetingListComponent ]
    })
    .compileComponents();

    fixture = TestBed.createComponent(MeetingListComponent);
    component = fixture.componentInstance;
    fixture.detectChanges();
  });

  it('should create', () => {
```

```
    expect(component).toBeTruthy();
  });
});
```

src\app\meetings\meeting-list\meeting-list.component.ts

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

@Component({
  selector: 'app-meeting-list',
  templateUrl: './meeting-list.component.html',
  styleUrls: ['./meeting-list.component.css']
})
export class MeetingListComponent implements OnInit {

  meetings = [
    // Example meetings data
    {
      id: 1,
      title: 'Team Sync',
      date: new Date('2024-12-25T10:00:00'),
      duration: 60,
      status: 'Scheduled',
      participants: ['Alice', 'Bob']
    },
    {
      id: 2,
      title: 'Project Planning',
      date: new Date('2024-12-26T14:00:00'),
      duration: 120,
      status: 'Completed',
      participants: ['Charlie', 'Dave', 'Eve']
    }
    // Add more meetings as needed
  ];

  newMeeting = {
    title: '',
    date: new Date(),
    duration: 60,
    status: 'Scheduled',
    participants: []
  };

  showForm = false;

  constructor() { }

  ngOnInit(): void {
    // Initialization logic if needed
  }

  addMeeting(): void {
    this.showForm = true;
```

```
  }

  saveMeeting(): void {
    // Validate form inputs
    if (this.newMeeting.title && this.newMeeting.date) {
      const newMeeting = {
        id: this.meetings.length + 1, // Increment ID
        ...this.newMeeting
      };

      this.meetings.push(newMeeting);
      this.newMeeting = { title: '', date: new Date(), duration: 60, status: 'Scheduled', participa
      this.showForm = false; // Hide form after adding
    }
  }

  cancelMeeting(meetingId?: number): void {
    if (meetingId !== undefined) {
      this.meetings = this.meetings.filter(meeting => meeting.id !== meetingId);
      console.log(`Cancelled meeting with ID: ${meetingId}`);
    } else {
      this.showForm = false; // Hide form without saving
    }
  }

  viewDetails(meetingId: number): void {
    console.log(`View details for meeting ID: ${meetingId}`);
    // Implement navigation or modal for meeting details
  }

  editMeeting(meetingId: number): void {
    console.log(`Edit meeting with ID: ${meetingId}`);
    // Implement navigation or modal for editing a meeting
  }
}
```

src\app\project\project-routing.module.ts

```
import { NgModule } from '@angular/core';
import { RouterModule, Routes } from '@angular/router';
import { ProjectListComponent } from './project-list/project-list.component';
import { ProjectDetailsComponent } from './project-details/project-details.component';
import { ProjectEditComponent } from './project-edit/project-edit.component';

const routes: Routes = [
  { path: 'project-list', component: ProjectListComponent },
  { path: 'project-details/:id', component: ProjectDetailsComponent },
  { path: 'project-edit/:id', component: ProjectEditComponent }
];

@NgModule({
  imports: [RouterModule.forChild(routes)],
  exports: [RouterModule]
})
```

```
export class ProjectRoutingModule {}
```

src\app\project\project.module.ts

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

import { ProjectRoutingModule } from './project-routing.module';
import { ProjectListComponent } from './project-list/project-list.component';
import { ProjectEditComponent } from './project-edit/project-edit.component';
import { ProjectDetailsComponent } from './project-details/project-details.component';
import { FormsModule } from '@angular/forms';
import { SharedModule } from '../shared/shared.module';



@NgModule({
  declarations: [
    ProjectListComponent,
    ProjectEditComponent,
    ProjectDetailsComponent,

  ],
  imports: [
    CommonModule,
    ProjectRoutingModule,
    FormsModule,
    SharedModule

  ]
})
export class ProjectModule { }
```

src\app\project\project-details\project-details.component.css

```
/* Project Details Section */
.project-details {
    background: linear-gradient(145deg, rgba(42, 42, 59, 0.9), rgba(42, 42, 59, 0.7));
    padding: 25px;
    border-radius: 15px;
    box-shadow: 0 4px 20px rgba(0, 0, 0, 0.3);
    border: 1px solid rgba(0, 247, 255, 0.1);
    backdrop-filter: blur(5px);
    margin-top: 20px;
    position: relative;
    overflow: hidden;
  }

  .project-details::before {
    content: '';
    position: absolute;
```

```
  top: 0;
  left: 0;
  width: 100%;
  height: 2px;
  background: linear-gradient(90deg, transparent, #00f7ff, transparent);
  transform: translateX(-100%);
  transition: transform 0.6s ease;
}

.project-details:hover::before {
  transform: translateX(100%);
}

.project-details h2 {
  font-size: 2rem;
  color: #ffffff;
  text-shadow: 0 0 10px rgba(0, 247, 255, 0.5);
  margin-bottom: 20px;
  position: relative;
  padding-bottom: 10px;
}

.project-details h2::after {
  content: '';
  position: absolute;
  bottom: 0;
  left: 0;
  width: 50px;
  height: 2px;
  background: #00f7ff;
}

.project-details p {
  margin-bottom: 15px;
  font-size: 1.1rem;
  color: #b3b3c1;
}

.project-details button {
  background: linear-gradient(135deg, rgba(42, 42, 59, 0.7), rgba(0, 247, 255, 0.5));
  color: #ffffff;
  border: none;
  padding: 12px 25px;
  border-radius: 8px;
  cursor: pointer;
  transition: all 0.3s ease;
  text-shadow: 0 0 10px rgba(0, 247, 255, 0.3);
}

.project-details button:hover {
  transform: translateY(-2px);
  box-shadow: 0 4px 15px rgba(0, 247, 255, 0.2);
}

.project-details .loading {
  font-style: italic;
  color: #00f7ff; /* Maintains the neon blue for loading text */
```

```
    text-align: center;
    margin-top: 20px;
  }
```

src\app\project\project-details\project-details.component.html

```
<app-navbar></app-navbar>
<div class="project-details">
    <h2>Project Details</h2>

    <div>
        <p><strong>Name:</strong> {{ project?.name }}</p>
        <p><strong>Deadline:</strong> {{ project?.deadline }}</p>
        <p><strong>Summary:</strong> {{ project?.summary }}</p>
        <p><strong>Additional Details:</strong> {{ project?.additional_details }}</p>

        <button (click)="goToEdit()">Edit Client</button>
    </div>
</div>
<app-footer></app-footer>
```

src\app\project\project-details\project-details.component.spec.ts

```
import { ComponentFixture, TestBed } from '@angular/core/testing';

import { ProjectDetailsComponent } from './project-details.component';

describe('ProjectDetailsComponent', () => {
  let component: ProjectDetailsComponent;
  let fixture: ComponentFixture<ProjectDetailsComponent>;

  beforeEach(async () => {
    await TestBed.configureTestingModule({
      declarations: [ ProjectDetailsComponent ]
    })
    .compileComponents();

    fixture = TestBed.createComponent(ProjectDetailsComponent);
    component = fixture.componentInstance;
    fixture.detectChanges();
  });

  it('should create', () => {
    expect(component).toBeTruthy();
  });
});
```

src\app\project\project-details\project-details.component.ts

```
import { Component, OnInit } from '@angular/core';
import { ActivatedRoute, Router } from '@angular/router';

@Component({
  selector: 'app-project-details',
  templateUrl: './project-details.component.html',
  styleUrls: ['./project-details.component.css']
})
export class ProjectDetailsComponent implements OnInit {

  projectId!: number;
  project: any;
  constructor(private route: ActivatedRoute, private router: Router) {}

  ngOnInit(): void {
    this.projectId = Number(this.route.snapshot.paramMap.get('id'));
    this.fetchProjectDetails();
  }

  fetchProjectDetails(): void {
    const projects = [
      { id: 1, name: "Project 1", deadline: "20/12/24", summary: "Summary of Project 1", additional
      { id: 2, name: "Project 2", deadline: "22/12/24", summary: "Summary of Project 2", additional
      { id: 3, name: "Project 3", deadline: "25/12/24", summary: "Summary of Project 3", additional
    ];

    this.project = projects.find(project => project.id === this.projectId);
    if (!this.project) {
      console.error('Project not found!');
    }
  }

  goToEdit(): void {
    this.router.navigate(['/project-edit', this.projectId]);
  }
}
```

src\app\project\project-edit\project-edit.component.css

```
.project-edit {
  background: linear-gradient(145deg, rgba(42, 42, 59, 0.9), rgba(42, 42, 59, 0.7));
  padding: 30px;
  border-radius: 15px;
  box-shadow: 0 4px 20px rgba(0, 0, 0, 0.3);
  border: 1px solid rgba(0, 247, 255, 0.1);
  backdrop-filter: blur(5px);
  position: relative;
  overflow: hidden;
  margin: 20px auto;
  max-width: 800px;
}

.project-edit::before {
```

```
  content: '';
  position: absolute;
  top: 0;
  left: 0;
  width: 100%;
  height: 2px;
  background: linear-gradient(90deg, transparent, #00f7ff, transparent);
  transform: translateX(-100%);
  transition: transform 0.6s ease;
}

.project-edit:hover::before {
  transform: translateX(100%);
}

/* Header Styling */
.project-edit h2 {
  font-size: 2.5rem;
  color: #ffffff;
  text-shadow: 0 0 10px rgba(0, 247, 255, 0.5);
  margin-bottom: 30px;
  text-align: center;
  position: relative;
  padding-bottom: 15px;
}

.project-edit h2::after {
  content: '';
  position: absolute;
  bottom: 0;
  left: 50%;
  transform: translateX(-50%);
  width: 150px;
  height: 2px;
  background: linear-gradient(90deg, transparent, #00f7ff, transparent);
}

/* Form Layout */
.project-edit form {
  display: flex;
  flex-direction: column;
  gap: 25px;
  width: 100%;
  max-width: 700px;
  margin: 0 auto;
}

/* Label Styling */
.project-edit label {
  display: block;
  font-size: 1.1rem;
  color: #e6e6f0;
  margin-bottom: 8px;
  text-shadow: 0 0 5px rgba(0, 247, 255, 0.3);
}

/* Input and Textarea Styles */
```

```
.project-edit input,
.project-edit textarea {
  width: 100%;
  background: rgba(30, 30, 47, 0.9);
  border: 1px solid rgba(0, 247, 255, 0.1);
  padding: 12px 15px;
  border-radius: 8px;
  color: #ffffff;
  font-size: 1rem;
  transition: all 0.3s ease;
}

.project-edit input:focus,
.project-edit textarea:focus {
  border-color: #00f7ff;
  box-shadow: 0 0 15px rgba(0, 247, 255, 0.1);
  outline: none;
  background: rgba(30, 30, 47, 0.95);
}

.project-edit textarea {
  min-height: 120px;
  resize: vertical;
}

/* Button Container */
.button-group {
  display: flex;
  gap: 15px;
  justify-content: center;
  margin-top: 20px;
}

/* Button Styles */
.project-edit button {
  background: linear-gradient(135deg, #00f7ff, #2e59d9);
  color: #ffffff;
  border: none;
  padding: 12px 30px;
  border-radius: 8px;
  cursor: pointer;
  font-size: 1.1rem;
  font-weight: 500;
  transition: all 0.3s ease;
  text-shadow: 0 0 10px rgba(0, 247, 255, 0.3);
  position: relative;
  overflow: hidden;
}

.project-edit button:hover {
  transform: translateY(-2px);
  box-shadow: 0 4px 15px rgba(0, 247, 255, 0.2);
}

.project-edit button:active {
  transform: translateY(0);
}
```

```
/* Cancel Button */
.project-edit button.cancel {
  background: linear-gradient(135deg, #ff4d4d, #cc0000);
}

/* Loading State */
.project-edit .loading {
  text-align: center;
  font-size: 1.2rem;
  color: #00f7ff;
  text-shadow: 0 0 10px rgba(0, 247, 255, 0.5);
  margin: 20px 0;
  font-style: italic;
}

/* Error Messages */
.error-message {
  color: #ff4d4d;
  font-size: 0.9rem;
  margin-top: 5px;
  text-shadow: 0 0 5px rgba(255, 77, 77, 0.3);
}

/* Success Message */
.success-message {
  color: #00ff9d;
  font-size: 0.9rem;
  margin-top: 5px;
  text-shadow: 0 0 5px rgba(0, 255, 157, 0.3);
}

/* Optional Fields */
.optional-field::after {
  content: '(optional)';
  font-size: 0.8rem;
  color: #b3b3c1;
  margin-left: 5px;
}

/* Required Fields */
.required-field::after {
  content: '*';
  color: #ff4d4d;
  margin-left: 5px;
}

/* Responsive Design */
@media (max-width: 768px) {
  .project-edit {
      padding: 20px;
      margin: 15px;
  }

  .project-edit h2 {
      font-size: 2rem;
  }
  .project-edit form {
```

```
        gap: 20px;
    }

    .button-group {
        flex-direction: column;
    }

    .project-edit button {
        width: 100%;
    }
}

/* Animation for Form Submission */
@keyframes submitPulse {
    0% {
        box-shadow: 0 0 0 0 rgba(0, 247, 255, 0.4);
    }
    70% {
        box-shadow: 0 0 0 10px rgba(0, 247, 255, 0);
    }
    100% {
        box-shadow: 0 0 0 0 rgba(0, 247, 255, 0);
    }
}

.submitting button {
    animation: submitPulse 1.5s infinite;
}
```

src\app\project\project-edit\project-edit.component.html

```
<body style="background-color: black;">
<app-navbar></app-navbar>
<div class="project-edit" >
  <h2>Edit Project</h2>
  <div *ngIf="project">
    <form (ngSubmit)="onSubmit()">
      <label for="name">Name:</label>
      <input type="text" id="name" [(ngModel)]="project.name" name="name" required>
      <label for="deadline">Deadline:</label>
      <input type="text" id="deadline" [(ngModel)]="project.deadline" name="deadline" required>
      <label for="summary">Summary:</label>
      <input type="text" id="summary" [(ngModel)]="project.summary" name="summary" required>
      <button type="submit">Save</button>
    </form>
  </div>
</div>
<app-footer></app-footer>
</body>
```

src\app\project\project-edit\project-edit.component.spec.ts

```
import { ComponentFixture, TestBed } from '@angular/core/testing';

import { ProjectEditComponent } from './project-edit.component';

describe('ProjectEditComponent', () => {
  let component: ProjectEditComponent;
  let fixture: ComponentFixture<ProjectEditComponent>;

  beforeEach(async () => {
    await TestBed.configureTestingModule({
      declarations: [ ProjectEditComponent ]
    })
    .compileComponents();

    fixture = TestBed.createComponent(ProjectEditComponent);
    component = fixture.componentInstance;
    fixture.detectChanges();
  });

  it('should create', () => {
    expect(component).toBeTruthy();
  });
});
```

src\app\project\project-edit\project-edit.component.ts

```
import { Component, OnInit } from '@angular/core';
import { ActivatedRoute, Router } from '@angular/router';

@Component({
  selector: 'app-project-edit',
  templateUrl: './project-edit.component.html',
  styleUrls: ['./project-edit.component.css']
})
export class ProjectEditComponent implements OnInit {
  projectId!: number;
  project: any; // Make sure this is initialized

  constructor(private route: ActivatedRoute, private router: Router) { }

  ngOnInit(): void {
    this.projectId = Number(this.route.snapshot.paramMap.get('id'));
    this.fetchProjectDetails();
  }

  fetchProjectDetails(): void {
    // Mock project data
    const projects = [
      { id: 1, name: 'project 1', deadline: '15th December', summary: 'Summary of project 1' },
      { id: 2, name: 'project 2', deadline: '16th December', summary: 'Summary of project 2' },
      { id: 3, name: 'project 3', deadline: '17th December', summary: 'Summary of project 3' },
    ];
    this.project = projects.find(project => project.id === this.projectId);
```

```
  }

  onSubmit(): void {
    console.log('Updated Project:', this.project);
    alert('Project Details updated successfully');
    this.router.navigate(['./project-list']);
  }
}
```

src\app\project\project-list\project-list.component.css

```
/* Project List Container */
.project-list {
  padding: 20px;
  min-height: 100vh;
  background: linear-gradient(135deg, rgba(30, 30, 47, 0.95), rgba(30, 30, 47, 0.85));
  backdrop-filter: blur(10px);
}

.project-list h2 {
  text-align: center;
  font-size: 2.5rem;
  color: #ffffff;
  text-shadow: 0 0 10px rgba(0, 247, 255, 0.5);
  margin-bottom: 30px;
  position: relative;
  padding-bottom: 15px;
}

.project-list h2::after {
  content: '';
  position: absolute;
  bottom: 0;
  left: 50%;
  transform: translateX(-50%);
  width: 150px;
  height: 2px;
  background: linear-gradient(90deg, transparent, #00f7ff, transparent);
}

/* Search Bar */
.search-bar {
  margin-bottom: 30px;
  text-align: center;
}

.search-bar input {
  width: 100%;
  max-width: 500px;
  padding: 12px 20px;
  border: 1px solid rgba(0, 247, 255, 0.2);
  border-radius: 8px;
  background: rgba(42, 42, 59, 0.7);
  color: #ffffff;
```

```
  font-size: 1rem;
  transition: border-color 0.3s ease, box-shadow 0.3s ease;
}

.search-bar input:focus {
  outline: none;
  border-color: #00f7ff;
  box-shadow: 0 0 10px rgba(0, 247, 255, 0.2);
}

/* Add Project Section */
.add-project {
  background: linear-gradient(145deg, rgba(42, 42, 59, 0.9), rgba(42, 42, 59, 0.7));
  padding: 25px;
  border-radius: 15px;
  margin-bottom: 30px;
  border: 1px solid rgba(0, 247, 255, 0.1);
  box-shadow: 0 4px 20px rgba(0, 0, 0, 0.3);
}

.add-project h3 {
  font-size: 1.5rem;
  margin-bottom: 20px;
  color: #ffffff;
}

.add-project input {
  width: 100%;
  padding: 10px;
  margin-bottom: 15px;
  border: 1px solid rgba(0, 247, 255, 0.2);
  border-radius: 8px;
  background: rgba(42, 42, 59, 0.7);
  color: #ffffff;
}

.add-project input:focus {
  outline: none;
  border-color: #00f7ff;
}

.add-project button {
  padding: 10px 20px;
  border: none;
  border-radius: 8px;
  background: rgba(0, 247, 255, 0.1);
  color: #00f7ff;
  cursor: pointer;
  transition: transform 0.3s ease, box-shadow 0.3s ease;
  border: 1px solid rgba(0, 247, 255, 0.2);
}

.add-project button:hover {
  background: rgba(0, 247, 255, 0.2);
  transform: translateY(-2px);
  box-shadow: 0 0 10px rgba(0, 247, 255, 0.3);
}
```

```
.add-project button:active {
  transform: translateY(0);
}

/* Project Cards */
.project-card-container {
  display: grid;
  grid-template-columns: repeat(auto-fit, minmax(300px, 1fr));
  gap: 20px;
}

.project-card {
  background: linear-gradient(145deg, rgba(42, 42, 59, 0.9), rgba(42, 42, 59, 0.7));
  padding: 25px;
  border-radius: 15px;
  border: 1px solid rgba(0, 247, 255, 0.1);
  box-shadow: 0 4px 20px rgba(0, 0, 0, 0.3);
  transition: transform 0.3s ease, box-shadow 0.3s ease;
}

.project-card:hover {
  transform: translateY(-5px);
  box-shadow: 0 8px 30px rgba(0, 247, 255, 0.15);
}

.project-card h3 {
  font-size: 1.4rem;
  color: #00f7ff;
  margin-bottom: 15px;
  text-shadow: 0 0 10px rgba(0, 247, 255, 0.3);
}

.project-card p {
  color: #b3b3c1;
  margin-bottom: 10px;
}

.project-card button {
  padding: 8px 20px;
  border: none;
  border-radius: 5px;
  background: rgba(0, 247, 255, 0.1);
  color: #00f7ff;
  cursor: pointer;
  transition: transform 0.3s ease, box-shadow 0.3s ease;
  border: 1px solid rgba(0, 247, 255, 0.2);
}

.project-card button:hover {
  background: rgba(0, 247, 255, 0.2);
  transform: translateY(-2px);
  box-shadow: 0 0 10px rgba(0, 247, 255, 0.3);
}

.project-card button:active {
  transform: translateY(0);
}
```

```
/* Responsive Design */
@media (max-width: 768px) {
  .project-list h2 {
    font-size: 2rem;
  }

  .project-card-container {
    grid-template-columns: 1fr;
  }

  .add-project {
    padding: 20px;
  }

  button {
    width: 100%;
    margin-bottom: 10px;
    margin-right: 0;
  }
}
```

src\app\project\project-list\project-list.component.html

```
<app-navbar></app-navbar>
<div class="project-list" style="background-color: black;">
  <h2>Project List</h2>

  <div class="search-bar">
    <input type="text" [(ngModel)]="searchQuery" placeholder="Search projects by name..." (input)="
  </div>

  <div class="add-project">
    <h3>Add New Project</h3>
    <input type="text" [(ngModel)]="newProjectName" placeholder="Enter project name" />
    <input type="text" [(ngModel)]="newProjectDeadline" placeholder="Enter project deadline" />
    <input type="text" [(ngModel)]="newProjectSummary" placeholder="Enter project summary" />
    <br /><br />
    <button (click)="addProject()">Add Project</button>
  </div>

  <div class="project-card-container">
    <div class="project-card" *ngFor="let project of filteredProjects">
      <div class="card">
        <h3>{{ project.name }}</h3>
        <p><strong>Deadline:</strong> {{ project.deadline }}</p>
        <p><strong>Summary:</strong> {{ project.summary }}</p><br><br><br>

        <button (click)="viewProject(project.id)">View</button>
        <button (click)="editProject(project.id)">Edit</button>
        <button (click)="deleteProject(project.id)">Delete</button>
      </div>
    </div>
  </div>
</div>
```

```
<app-footer>
</app-footer>
```

src\app\project\project-list\project-list.component.spec.ts

```
import { ComponentFixture, TestBed } from '@angular/core/testing';

import { ProjectListComponent } from './project-list.component';

describe('ProjectListComponent', () => {
  let component: ProjectListComponent;
  let fixture: ComponentFixture<ProjectListComponent>;

  beforeEach(async () => {
    await TestBed.configureTestingModule({
      declarations: [ ProjectListComponent ]
    })
    .compileComponents();

    fixture = TestBed.createComponent(ProjectListComponent);
    component = fixture.componentInstance;
    fixture.detectChanges();
  });

  it('should create', () => {
    expect(component).toBeTruthy();
  });
});
```

src\app\project\project-list\project-list.component.ts

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

@Component({
  selector: 'app-project-list',
  templateUrl: './project-list.component.html',
  styleUrls: ['./project-list.component.css']
})
export class ProjectListComponent implements OnInit {
  projects = [
    { id: 1, name: 'Project 1', deadline: '10th December', summary: 'Project 1 Summary' },
    { id: 2, name: 'Project 2', deadline: '12th December', summary: 'Project 2 Summary' },
    { id: 3, name: 'Project 3', deadline: '15th December', summary: 'Project 3 Summary' },
  ];

  searchQuery = '';
  filteredProjects = [...this.projects]; // Initially show all projects
  newProjectName = '';
  newProjectDeadline = '';
  newProjectSummary = '';
```

```
  constructor(private router: Router) {}

  ngOnInit(): void {}

  // Method to filter projects based on search query
  filterProjects(): void {
    this.filteredProjects = this.projects.filter(project =>
      project.name.toLowerCase().includes(this.searchQuery.toLowerCase())
    );
  }

  // Call this method whenever the search query changes
  onSearchChange(): void {
    this.filterProjects();
  }

  addProject(): void {
    if (this.newProjectName && this.newProjectDeadline && this.newProjectSummary) {
      const newProject = {
        id: this.projects.length + 1,
        name: this.newProjectName,
        deadline: this.newProjectDeadline,
        summary: this.newProjectSummary,
      };
      this.projects.push(newProject);
      this.filteredProjects.push(newProject);
      this.clearNewProjectFields();
      alert('Project added successfully!');
    } else {
      alert('Please fill all the fields to add a new project.');
    }
  }

  clearNewProjectFields(): void {
    this.newProjectName = '';
    this.newProjectDeadline = '';
    this.newProjectSummary = '';
  }

  viewProject(id: number): void {
    this.router.navigate(['/project-details', id]);
  }

  editProject(id: number): void {
    this.router.navigate(['/project-edit', id]);
  }

  deleteProject(id: number): void {
    this.projects = this.projects.filter(project => project.id !== id);
    this.filteredProjects = this.filteredProjects.filter(project => project.id !== id);
    alert('Project deleted successfully.');
  }
}
```

src\app\shared\shared-routing.module.ts

```
import { NgModule } from '@angular/core';
import { RouterModule, Routes } from '@angular/router';
import { NavbarComponent } from './navbar/navbar.component';
import { FooterComponent } from './footer/footer.component';

const routes: Routes = [
  {path:'navbar',component:NavbarComponent},
  {path:'footer',component:FooterComponent}
];

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

src\app\shared\shared.module.ts

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

import { SharedRoutingModule } from './shared-routing.module';
import { NavbarComponent } from './navbar/navbar.component';
import { FooterComponent } from './footer/footer.component';


@NgModule({
  declarations: [
    NavbarComponent,
    FooterComponent
  ],
  imports: [
    CommonModule,
    SharedRoutingModule
  ],
  exports:[
    NavbarComponent,
    FooterComponent
  ]
})
export class SharedModule { }
```

src\app\shared\footer\footer.component.css

```
/* Footer container */
.footer {
    background-color: #1a1a1a;
    color: #ffffff;
    padding: 3rem 2rem;
    font-family: Arial, sans-serif;
```

```
}

/* Footer content wrapper */
.footer-content {
  max-width: 1200px;
  margin: 0 auto;
  display: grid;
  grid-template-columns: repeat(3, 1fr);
  gap: 2rem;
}

/* Company info section */
.company-info h3 {
  color: #00b4d8;
  font-size: 1.5rem;
  margin-bottom: 1rem;
}

.company-info p {
  color: #999;
  font-size: 0.9rem;
}

/* Footer links section */
.footer-links ul {
  list-style: none;
  padding: 0;
}

.footer-links li {
  margin-bottom: 1rem;
  cursor: pointer;
  transition: color 0.3s ease;
}

.footer-links li:hover {
  color: #00b4d8;
}

/* Social media section */
.social-media a {
  display: inline-flex;
  align-items: center;
  color: #ffffff;
  text-decoration: none;
  margin-bottom: 1rem;
  transition: color 0.3s ease;
}

.social-media a:hover {
  color: #00b4d8;
}

.social-media i {
  margin-right: 0.5rem;
  font-size: 1.2rem;
}
```

```
  /* Hide br tags on larger screens */
  .social-media br {
    display: none;
  }

  /* Responsive design */
  @media (max-width: 768px) {
    .footer-content {
      grid-template-columns: 1fr;
      text-align: center;
      gap: 2rem;
    }

    .social-media {
      display: flex;
      flex-direction: column;
      align-items: center;
    }

    .social-media br {
      display: block;
    }

    .footer-links ul {
      display: flex;
      flex-direction: column;
      align-items: center;
    }
  }

  /* Optional: Add a subtle top border */
  .footer::before {
    content: '';
    display: block;
    height: 1px;
    background: linear-gradient(to right, #1a1a1a, #333, #1a1a1a);
    margin-bottom: 2rem;
  }
```

src\app\shared\footer\footer.component.html

```
<footer class="footer">
  <div class="footer-content">
    <div class="company-info">
      <h3>Client Management App</h3>
      <p>&copy; 2024 All Rights Reserved. Developed by Henna Solutions.</p>
    </div>

    <div class="footer-links">
      <ul>
        <li><a href="#">Privacy Policy</a></li>
        <li><a href="#">Terms of Service</a></li>
        <li><a href="#">Contact Us</a></li>
      </ul>
```

```
    </div>

    <div class="social-media">
      <a href="#" aria-label="Facebook"><i class="fab fa-facebook"></i>Facebook</a><br/>
      <a href="#" aria-label="Twitter"><i class="fab fa-twitter"></i>Twitter</a><br/>
      <a href="#" aria-label="LinkedIn"><i class="fab fa-linkedin"></i>LinkedIn</a>
    </div>
  </div>
</footer>
```

src\app\shared\footer\footer.component.spec.ts

```
import { ComponentFixture, TestBed } from '@angular/core/testing';

import { FooterComponent } from './footer.component';

describe('FooterComponent', () => {
  let component: FooterComponent;
  let fixture: ComponentFixture<FooterComponent>;

  beforeEach(async () => {
    await TestBed.configureTestingModule({
      declarations: [ FooterComponent ]
    })
    .compileComponents();

    fixture = TestBed.createComponent(FooterComponent);
    component = fixture.componentInstance;
    fixture.detectChanges();
  });

  it('should create', () => {
    expect(component).toBeTruthy();
  });
});
```

src\app\shared\footer\footer.component.ts

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

@Component({
  selector: 'app-footer',
  templateUrl: './footer.component.html',
  styleUrls: ['./footer.component.css']
})
export class FooterComponent implements OnInit {

  constructor() { }

  ngOnInit(): void {
  }
```

```
}
```

src\app\shared\navbar\navbar.component.css

```
/* Reset default styles */
* {
  margin: 0;
  padding: 0;
  box-sizing: border-box;
}

/* Navbar container */
.navbar {
  background-color: #1a1a1a;
  padding: 1rem 2rem;
  font-family: Arial, sans-serif;
  display: flex;
  justify-content: center; /* Center the navbar horizontally */
  width: 100%;
}

/* Main navigation list */
.navbar ul {
  list-style: none;
  display: flex;
  gap: 2rem;
  align-items: center;
  justify-content: center; /* Center list items */
  max-width: 1200px;       /* Limit maximum width */
  width: 100%;
}

/* Main navigation items */
.navbar > ul > li {
  position: relative;
  color: #ffffff;
  cursor: pointer;
  padding: 0.5rem 1rem;
  font-size: 1rem;
  text-align: center;      /* Center text within items */
}

/* All links in navbar */
.navbar a {
  color: #ffffff;
  text-decoration: none;
  transition: color 0.3s ease;
  display: block;          /* Make entire area clickable */
  text-align: center;      /* Center link text */
}

/* Hover effect for links */
.navbar a:hover {
  color: #00b4d8;
```

```
}

/* Dropdown menus */
.dropdown {
  position: absolute;
  top: 100%;
  left: 50%;              /* Center dropdown under parent */
  transform: translateX(-50%) translateY(-10px);  /* Center and add initial offset */
  background-color: #2a2a2a;
  min-width: 200px;
  border-radius: 8px;
  box-shadow: 0 6px 12px rgba(0, 0, 0, 0.3);
  opacity: 0;
  visibility: hidden;
  transition: all 0.3s ease;
  z-index: 1000;
}

/* Show dropdown on hover */
.navbar > ul > li:hover .dropdown {
  opacity: 1;
  visibility: visible;
  transform: translateX(-50%) translateY(0);  /* Center and remove offset on hover */
}

/* Dropdown items */
.dropdown li {
  padding: 0.75rem 1rem;
  border-bottom: 1px solid #3a3a3a;
  text-align: center;     /* Center dropdown text */
}

.dropdown li:last-child {
  border-bottom: none;
}

/* Dropdown hover effect */
.dropdown li:hover {
  background-color: #3a3a3a;
}

/* Sign Out button special styling */
.navbar > ul > li:last-child a {
  background-color: #333;
  padding: 0.5rem 1rem;
  border-radius: 8px;
  transition: background-color 0.3s ease;
}

.navbar > ul > li:last-child a:hover {
  background-color: #444;
  color: #00b4d8;
}

/* Responsive design */
@media (max-width: 768px) {
  .navbar {
```

```
      padding: 1rem;
  }

  .navbar ul {
      flex-direction: column;
      gap: 1rem;
      width: 100%;
      max-width: 100%;
  }

  .dropdown {
      position: static;
      width: 100%;
      opacity: 1;
      visibility: visible;
      transform: none;
      display: none;
  }

  .navbar > ul > li:hover .dropdown {
      display: block;
      transform: none;
  }

  .navbar > ul > li {
      width: 100%;
  }
}
```

src\app\shared\navbar\navbar.component.html

```
<nav class="navbar">
    <ul>
      <li><a routerLink="/dash/dashboard">Dashboard</a></li>
      <li>
        Client
        <ul class="dropdown">
          <li><a routerLink="/client/client-list">Client List</a></li>
        </ul>
      </li>
      <li>
        Projects
        <ul class="dropdown">
          <li><a routerLink="/project/project-list">Project List</a></li>
        </ul>
      </li>
      <li>
        Meetings
        <ul class="dropdown">
            <li><a routerLink="/meetings/meeting-list">Meeting list</a></li>
            <li><a routerLink="/meetings/meeting-calender">Meeting Calender</a></li>
            <li><a routerLink="/meetings/meeting-details">Current Meeting</a></li>
        </ul>
    </li>
```

```
      <li><a routerLink="/auth/login">Sign Out</a></li>
    </ul>
  </nav>
```

src\app\shared\navbar\navbar.component.spec.ts

```
import { ComponentFixture, TestBed } from '@angular/core/testing';

import { NavbarComponent } from './navbar.component';

describe('NavbarComponent', () => {
  let component: NavbarComponent;
  let fixture: ComponentFixture<NavbarComponent>;

  beforeEach(async () => {
    await TestBed.configureTestingModule({
      declarations: [ NavbarComponent ]
    })
    .compileComponents();

    fixture = TestBed.createComponent(NavbarComponent);
    component = fixture.componentInstance;
    fixture.detectChanges();
  });

  it('should create', () => {
    expect(component).toBeTruthy();
  });
});
```

src\app\shared\navbar\navbar.component.ts

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

@Component({
  selector: 'app-navbar',
  templateUrl: './navbar.component.html',
  styleUrls: ['./navbar.component.css']
})
export class NavbarComponent implements OnInit {

  constructor() { }

  ngOnInit(): void {
  }

}
```

src\environments\environment.prod.ts

```
export const environment = {
  production: true
};
```

src\environments\environment.ts

```
// This file can be replaced during build by using the `fileReplacements` array.
// `ng build` replaces `environment.ts` with `environment.prod.ts`.
// The list of file replacements can be found in `angular.json`.

export const environment = {
  production: false
};

/*
 * For easier debugging in development mode, you can import the following file
 * to ignore zone related error stack frames such as `zone.run`, `zoneDelegate.invokeTask`.
 *
 * This import should be commented out in production mode because it will have a negative impact
 * on performance if an error is thrown.
 */
// import 'zone.js/plugins/zone-error';  // Included with Angular CLI.
```