

PROFESSIONAL TRAINING REPORT
at
Sathyabama Institute of Science and Technology
(Deemed to be University)

Submitted in partial fulfillment of the requirements for the award of
Bachelor of Technology in Information Technology

By

K. Sheshasena Reddy

REG. NO. 39120111



DEPARTMENT OF INFORMATION TECHNOLOGY

SCHOOL OF COMPUTING

SATHYABAMA INSTITUTE OF SCIENCE AND TECHNOLOGY
JEPPIAAR NAGAR, RAJIV GANDHI SALAI,
CHENNAI – 600119, TAMILNADU

NOVEMBER 2021



SATHYABAMA
INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)

**Accredited with Grade “A” by
NAAC**

(Established under Section 3 of UGC Act, 1956)
JEPPIAAR NAGAR, RAJIV GANDHI SALAI
CHENNAI– 600119
www.sathyabama.ac.in



DEPARTMENT OF INFORMATION TECHNOLOGY

BONAFIDE CERTIFICATE

This is to certify that this Project Report is the bonafide work of **K.SHESHASENA REDDY (Reg. No: 39120111)** who carried out the project entitled “**Employee Leave Management System** ” under my supervision from June 2021 to November 2021.

Internal Guide

Dr . V. MARIA ANU M.E.,Ph.D.

Head of the Department

Submitted for Viva voce Examination held on _____

Internal Examiner

External Examiner

DECLARATION

I, **K.Sheshasena Reddy** hereby declare that the project report entitled **Employee Management Portal** done by me under the guidance of **Dr V Maria Anu M.E.,Ph.D.** is submitted in partial fulfillment of the requirements for the award of Bachelor of Technology Degree in Information and Technology.

DATE: 8/11/2021

PLACE:Chennai



SIGNATURE OF THE CANDIDATE

Acknowledgement

I am pleased to acknowledge my sincere thanks to **Board of Management of SATHYABAMA** for their kind encouragement in doing this project and for completing it successfully. I am grateful to them.

I convey my thanks to **Dr. T. Sasikala M.E., Ph.D, Dean**, School of Computing, **Dr. R. Subhashini, M.E., Ph.D. Heads of the Department of Information Technology** for providing me necessary support and details at the right time during the progressive reviews.

I would like to express my sincere and deep sense of gratitude to my Project Guide **Dr V.Maria Anu M.E., Ph.D. ,** for her valuable guidance, suggestions and constant encouragement paved way for the successful completion of my project work.

I wish to express my thanks to all Teaching and Non-teaching staff members of the **Department of Information and Technology** who were helpful in many ways for the completion of the project.

TRAINING CERTIFICATE



ABSTRACT

This is a web page made using HTML, CSS, Javascript, and Framework Angular JS.

An Employee portal is very important for a company manager to manage every employee details and leave status. An Employee Portal, sometimes known as an HR Portal, is an online employee engagement platform that puts a range of HR services at the employee's fingertips

An Employee Portal makes available a wide range of relevant HR content, including information on benefits, pay, HR policies, Well-being and careers advice, which means employees can find the answers to most of their questions.

TABLE OF CONTENTS

CHAPTER No.	TITLE	PAGE No
	ABSTRACT	6
1	INTRODUCTION	9
2.	AIM AND SCOPE OF THE PRESENT	11
	2.1 AIM	11
	2.2 SCOPE	11
3.	EXPERIMENTAL OR MATERIALS AND METHODS, USED	14
	3.1 Software Required	14
	3.1.1 Windows 10	
	3.1.2 Node JS	
	3.1.3 Angular CLI	
	3.2 Hardware Required	18
	3.2.1 Hard Disk	
	3.2.2 RAM	
	3.2.3 Processor	
	3.2.4 Mouse	
	3.2.5 Keyboard	
	3.2.6 Speaker	
4.	RESULTS AND DISCUSSION, PERFORMANCE Recommendation	23
	4.1 Result and discussion	23
	4.2 Recommendation	24
	4.3 Screen shots	
	4.3.1 Input	
	4.3.2 Output	

5	SUMMARY AND CONCLUSIONS	26
	REFERENCES	27
	APPENDIX	28
	SOURCE CODE	28
	SAMPLE SCREEN	56

CHAPTER - 1

INTRODUCTION

1.1 ABOUT THE CONCEPT

An Employee Portal, sometimes known as an HR Portal, is an online employee engagement platform that puts a range of HR services at the employee's fingertips. From providing employees with access to helpful HR content and self-service functionality that answers their questions, an Employee Portal enhances the employee service experience, which increases employee satisfaction and engagement levels. It is also a communications and change management tool, enabling HR to effectively engage with their employees and support the management of change and development within the organization. Additional benefits include increased workforce productivity and greater efficiency of HR operations.

Successful businesses strive for long-term sustainable profits. An Employee Portal is a tool that HR can use to support this objective, by improving the employee experience, employee productivity and retaining staff.

An Employee Portal makes available a wide range of relevant HR content, including information on benefits, pay, HR policies, Well-being and careers advice, which means employees can find the answers to most of their questions. In addition, an Instant Search capability means as the employee types in search terms, he or she is 'instantly' presented with potential answers to his or her question, making finding answers to questions easier and more immediate.

CHAPTER - 2

AIM AND SCOPE OF PRESENT INVESTIGATION

2.1 AIM

ESS portals also play an important role in raising HR's stature in the organization, enabling HR professionals to spend more time on strategic efforts in line with the organization's goals and strategy than on focusing on administrative transactions.

2.2 SCOPE

ESS portals allow employees to have quick and easy access to HR-related transactions and services that HR personnel would otherwise have to provide. This can reduce HR's tactical duties and free up time for more strategic pursuits, with realized cost savings in time and efficiency. User-friendly and paperless, ESS portals can also help standardize processes, reducing errors and enhancing reporting capabilities.

Technology has noticeably changed HR management; most firms provide universal access to HR services through technology and web-based applications. These changes often result from the need to cut costs and expand or improve services. Organizations that successfully adopt sophisticated HR technology tools outperform those that do not. But the simple automation of HR processes can no longer guarantee a competitive advantage. Instead, organizations must determine how to use technology to transform their HR practices and market their HR brand.

CHAPTER 3

EXPERIMENTAL OR MATERIALS AND METHODS

3.1 SOFTWARE REQUIREMENT

- Windows 7 or above
- Node JS
- Angular CLI

3.1.1 Windows 10 or above

Windows 10 is a Microsoft operating system for personal computers, tablets, embedded devices and internet of things devices.

Microsoft released Windows 10 in July 2015 as a follow-up to Windows 8. The company has said it will update Windows 10 in perpetuity rather than release a new, full-fledged operating system as a successor.

3.1.2 Node JS

As an asynchronous event-driven JavaScript runtime, Node.js is designed to build scalable network applications. In the following "hello world" example, many connections can be handled concurrently. Upon each connection, the callback is fired, but if there is no work to be done, Node.js will sleep.

This is in contrast to today's more common concurrency model, in which OS threads are employed. Thread-based networking is relatively inefficient and very difficult to use. Furthermore, users of Node.js are free from worries of dead-locking the process, since there are no locks. Almost no function in Node.js directly performs I/O, so the process never blocks except when the I/O is performed using synchronous methods of Node.js standard library. Because nothing blocks, scalable systems are very reasonable to develop in Node.js.

If some of this language is unfamiliar, there is a full article on Blocking vs. Non-Blocking.

Node.js is similar in design to, and influenced by, systems like Ruby's Event Machine and Python's Twisted. Node.js takes the event model a bit further. It presents an event loop as a runtime construct instead of as a library. In other systems, there is always a blocking call to start the event-loop. Typically, behavior is defined through callbacks at the beginning of a script, and at the end a server is started through a blocking call like `EventMachine::run()`. In Node.js, there is no such start-the-event-loop call. Node.js simply enters the event loop after executing the input script. Node.js exits the event loop when there are no more callbacks to perform. This behavior is like browser JavaScript — the event loop is hidden from the user.

HTTP is a first-class citizen in Node.js, designed with streaming and low latency in mind. This makes Node.js well suited for the foundation of a web library or framework.

Node.js being designed without threads doesn't mean you can't take advantage of multiple cores in your environment. Child processes can be spawned by using our `child_process.fork()` API, and are designed to be easy to communicate with. Built upon that same interface is the `cluster` module, which allows you to share sockets between processes to enable load balancing over your cores.



3.1.3 Angular CLI

This topic can help you understand Angular: what Angular is, what advantages it provides, and what you might expect as you start to build your applications.

Angular is a development platform, built on TypeScript. As a platform, Angular includes:

A component-based framework for building scalable web applications

A collection of well-integrated libraries that cover a wide variety of features, including routing, forms management, client-server communication, and more

A suite of developer tools to help you develop, build, test, and update your code

With Angular, you're taking advantage of a platform that can scale from single-developer projects to enterprise-level applications. Angular is designed to make updating as straightforward as possible, so take advantage of the latest developments with a minimum of effort. Best of all, the Angular ecosystem consists of a diverse group of over 1.7 million developers, library authors, and content creators.

This section explains the core ideas behind Angular. Understanding these ideas can help you design and build your applications more effectively.

Components

Components are the building blocks that compose an application. A component includes a TypeScript class with a `@Component()` decorator, an HTML template, and styles. The `@Component()` decorator specifies the following Angular-specific information:

A CSS selector that defines how the component is used in a template. HTML elements in your template that match this selector become instances of the component.

An HTML template that instructs Angular how to render the component.

An optional set of CSS styles that define the appearance of the template's HTML elements.

Templates

Every component has an HTML template that declares how that component renders. You define this template either inline or by file path.

Angular extends HTML with additional syntax that lets you insert dynamic values from your component. Angular automatically updates the rendered DOM when your component's state changes. One application of this feature is inserting dynamic text, as shown in the following example.

```
C:\Windows\system32\cmd.exe
Microsoft Windows [Version 10.0.19043.1288]
(c) Microsoft Corporation. All rights reserved.

C:\Users\ASUS>node -v
v14.18.0

C:\Users\ASUS>npm -v
6.14.15

C:\Users\ASUS>ng --version

Angular CLI
Angular CLI: 12.2.12
Node: 14.18.0
Package Manager: npm 6.14.15
OS: win32 x64

Angular:
...

Package      Version
-----
@angular-devkit/architect 0.1202.12
@angular-devkit/core      12.2.12
@angular-devkit/schematics 12.2.12 (cli-only)
@schematics/angular       12.2.12 (cli-only)
```

3.2 HARDWARE REQUIREMENT

- Hard Disk
- RAM – 1 GB
- Processor – Dual Core or Above
- Mouse
- Keyboard
- Monitor
- Speaker

3.2.1 Hard Disk

A hard disk or fixed disk, is an electro-mechanical data storage device that uses magnetic storage to store and retrieve digital information using one or more rigid rapidly rotating disks (platters) coated with magnetic material.

During the mid-1990s the typical hard disk drive for a PC had a capacity of about 1 gigabyte. As of January 2019, desktop hard disk drives typically had a capacity of 1 to 6 terabytes, with the largest-capacity drives reaching 15 terabytes

3.2.2 RAM

Random access memory (RAM) is a type of data storage used in computers that is generally located on the motherboard. This type of memory is volatile and all information that was stored in RAM is lost when the computer is turned off.

The RAM chip may be individually mounted on the motherboard or in sets of several chips on a small board connected to the motherboard. Older memory types were in the form of chips called dual in-line package (DIP). Although DIP chips are still used today, the majority of memory is in the form of a module, a narrow printed circuit board attached to a connector on the motherboard. The three main memory circuit boards types containing chips are: RIMMs (Rambus in-line memory modules), DIMMs (dual in-line memory modules) and SIMMs (single in-line memory modules). Most motherboards today use DIMMs. Sometimes, people get confused about the difference between memory and storage, in part because both can be measured in megabytes (MB), gigabytes (GB) and terabytes (TB).

By contrast, storage is slower, but it can retain data when the device is powered down. So, for example, if a document has been saved to a hard drive prior to a power outage or system crash, the user will still be able to retrieve it when the system is back up and running.

3.2.3 Processor

A processor is an integrated electronic circuit that performs the calculations that run a computer. A processor performs arithmetical, logical, input/output (I/O) and other basic instructions that are passed from an operating system (OS). Most other processes are dependent on the operations of a processor.

Dual core is a CPU that has two distinct processors that work simultaneously in the same integrated circuit. This type of processor can function as efficiently as a single processor but can perform operations up to twice as quickly.

3.2.4 Mouse

A mouse is a small handheld input device that controls a computer screen's cursor or pointer in conjunction with the way it is moved on a flat surface. The mouse term name originates from its likeness to a small, corded and elliptical shaped device that looks like a mouse tail.

Some mouse devices have integrated features, such as extra buttons that may be programmed and assigned with different commands.

Because the mouse reduces the use of a keyboard, its invention and continuous innovation is considered one of the most important breakthroughs in computer ergonomics.

3.2.5 Keyboard

A computer keyboard is an input device used to enter characters and functions into the computer system by pressing buttons, or keys. It is the primary device used to enter text. A keyboard typically contains keys for individual letters, numbers and

special characters, as well as keys for specific functions. A keyboard is connected to a computer system using a cable or a wireless connection.

A computer keyboard is an input device used to enter characters and functions into the computer system by pressing buttons, or keys. It is the primary device used to enter text. A keyboard typically contains keys for individual letters, numbers and special characters, as well as keys for specific functions.

3.2.6 Monitor

A computer monitor is an electronic device that shows pictures for computers. Monitors often look similar to televisions. The main difference between a monitor and a television is that a monitor does not have a television tuner to change channels. Monitors often have higher display resolution than televisions.

The advent of display technology has paved the way for the continuous evolution of the monitor, whether it's for computers, television, mobile devices or any device that has a display. The current contenders for top-tier technology being used for display devices includes Super LCD 3 (SLCD3) and Super AMOLED. It should be noted that LED displays are actually just a kind of LCD display that use LED lights as backlight illumination.

3.2.7 Speaker

Most Important Speaker is required for this project because when we enter any alphabetical word or paragraph it convert it into sound which will come through the speaker of the system.

CHAPTER 4

RESULT AND DISCUSSION,PERFORMANCE ANALYSIS

4.1 Result & Discussion

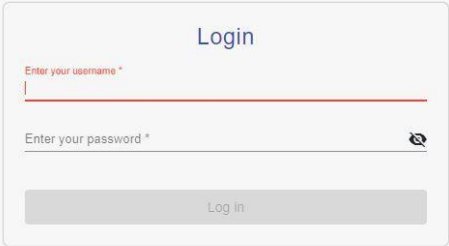
During the course of this research it was realized that building an employee management application is a rigorous task. Consequently, a system that will provide real-time access to employee manager and dates relating to the leaves across to a larger percentage of the employees population was developed. Whatever employee management demands, an application is there to deliver. Without it, the employees would be a list of many employees full of unorganized details. This application can sort the things and help the manager to handle the things and large number of employees in very easier manner.

4.2 RECOMMENDATION

Recommendations The EMS meets a lot of expectations but would perform better if the following recommendations and suggestions are considered; i. System testing and maintenance should be performed regularly to avoid sudden system failures. ii. Updates and other modifications should be introduced with prior notice to users. iii. A administrator should be employed that can manage the interactions between employees and administrators on one side and technicians on the other. This will aid ensure that proper support is provided for the system.

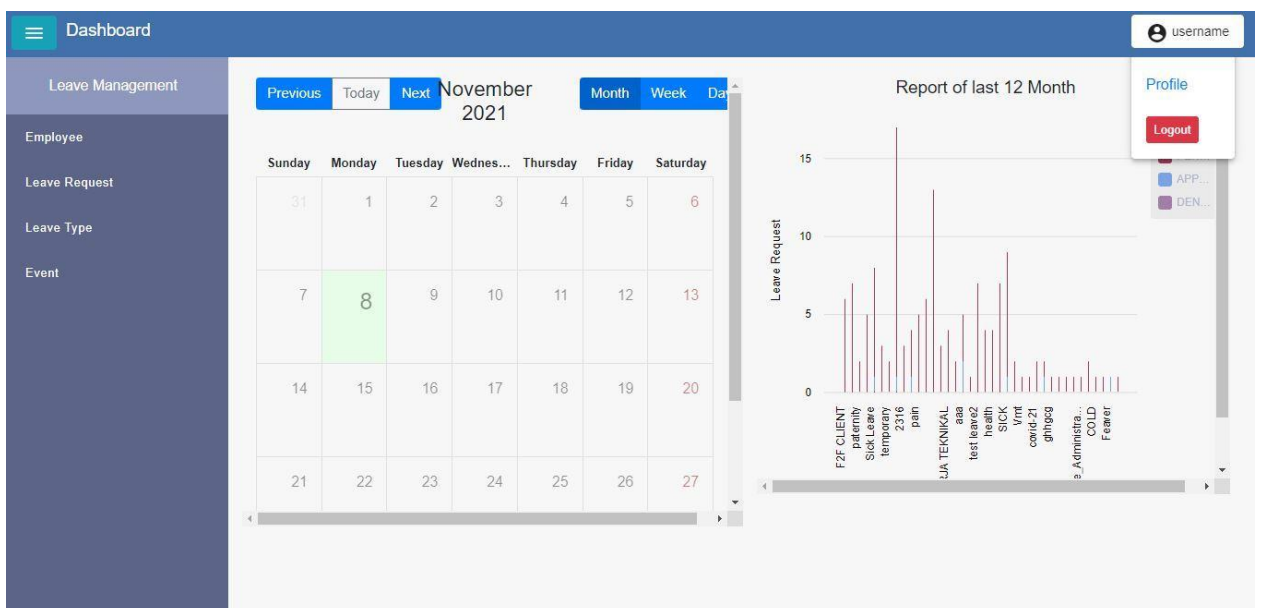
4.3 SCREENSHOTS

4.3.1 Input



A login form titled "Login" with two input fields: "Enter your username *" and "Enter your password *". The password field has a toggle icon. A "Log in" button is at the bottom.

4.3.2 Output



Output for Leave management

Dashboard

username

Leave Management

Employee

Leave Request

Leave Type

Event

All Employees

New Employee

Search

S.N	Name ↑↓	Phone Number ↑↓	Email ↑↓	Supervisor ↑↓	Role ↑↓	Status ↑↓	Options
1	Aa Sheshasena Reddy	6308657589	sheshasena@gmail.com		ROLE_USER	ACTIVE	View
2	abc def	8013996507			ROLE_USER	ACTIVE	View
3	abc adv	9840355935			ROLE_USER	ACTIVE	View
4	abcd abcd abcd	9876543444	dgsdgh	Sheshasena Reddy	ROLE_USER	ACTIVE	View
5	adf afg	3154612101			ROLE_USER	ACTIVE	View
6	adil bin tariq	3126464132			ROLE_USER	ACTIVE	View
7	adil bin adv	1234562878			ROLE_USER	ACTIVE	View

Output for Employee details

CHAPTER 5

SUMMARY AND CONCLUSIONS

EMS is not just a change of technology. It is part of a redefinition of how we as a species transmit knowledge, skills, and values to younger generations of workers and students. Employees will have access to millions or billions of facilities. Some will be Web pages with simple text and graphics. Others may include multimedia simulations. In many fields, EMS has become the default way to request anything online.

REFERENCES

1. www.google.com
2. www.geeksforgeeks.org
3. www.wikipedia.org/html
4. www.wikipedia.org/css
5. www.wikipedia.org/javascript
6. www.wikipedia.org/angular
7. www.tutorialspointsg.com
8. www.javaworld.com
9. www.angularjs.com
10. www.nodejs.com
11. www.github.com

APPENDIX

SOURCE CODE

1 Index.html

```
<!doctype html>
<html lang="en">
<head>
  <meta charset="utf-8">
  <title>EmployeeManagementAngular</title>
  <base href="/">

  <meta name="viewport" content="width=device-width, initial-scale=1">
  <link rel="icon" type="image/x-icon" href="favicon.ico">
  <link href="https://fonts.googleapis.com/icon?family=Material+Icons"
rel="stylesheet">
</head>
<body>
  <app-root></app-root>
</body>
</html>
```

2. Navside Wrapper component.ts

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

import { NavsideWrapperComponent } from './navside-wrapper.component';

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

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

  beforeEach(() => {
    fixture = TestBed.createComponent(NavsideWrapperComponent);
    component = fixture.componentInstance;
```

```

    fixture.detectChanges();
  });

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

```

3.Sidebar.html

```

<nav id="sidebar" [ngClass]="isOpen ? 'active' : ''">

  <ul class="list-unstyled sidebar-items">
    <p class="sidebar-header">Leave Management</p>
    <li>
      <a routerLink="/home/employees" routerLinkActive="active">Employee</a>
    </li>
    <li>
      <a routerLink="/home/leaverequests" routerLinkActive="active">Leave
Request</a>
    </li>
    <li>
      <a routerLink="/home/leavetypes" routerLinkActive="active">Leave
Type</a>
    </li>
    <li>
      <a routerLink="/home/event" routerLinkActive="active">Event</a>
    </li>
  </ul>

</nav>

```

4.Sidebar.ts

```

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

import { SidebarComponent } from './sidebar.component';

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

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

  beforeEach(() => {
    fixture = TestBed.createComponent(SidebarComponent);
    component = fixture.componentInstance;
    fixture.detectChanges();
  });

```

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

5.Top Navigation

```
<nav class="navbar fixed-top custom-nav">

  <div>
    <button type="button" class="btn btn-info" (click)="toggleSidebar()">
      <mat-icon>menu</mat-icon>
    </button>

    <a class="navbar-brand text-white" routerLink="/home">&nbsp;   Dashboard</a>
  </div>

  <div>
    <button mat-flat-button [matMenuTriggerFor]="menu">
      <mat-icon>account_circle</mat-icon>&nbsp;  username
    </button>
    <mat-menu #menu="matMenu" class="text-center">
      <mat-list>
        <mat-list-item> <a routerLink="/home/profile"
routerLinkActive="active">Profile</a> </mat-list-item>
        <mat-list-item> <span class="btn btn-sm btn-danger"
(click)="_auth.logout()">Logout</span> </mat-list-item>
      </mat-list>
    </mat-menu>
  </div>

</nav>

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

import { TopNavigationComponent } from './top-navigation.component';

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

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

  beforeEach(() => {
    fixture = TestBed.createComponent(TopNavigationComponent);
    component = fixture.componentInstance;
    fixture.detectChanges();
  });

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

```

    });
  });

import { AuthService } from '../../../feature/dashboard/auth/auth.service';
import { Component } from '@angular/core';
import { SidebarService } from '../service/sidebar.service';

@Component({
  selector: 'app-top-navigation',
  templateUrl: './top-navigation.component.html',
  styleUrls: ['./top-navigation.component.css']
})
export class TopNavigationComponent {

  isLoggedIn: boolean = this._auth.isLoggedIn();

  constructor(private _sideBarService: SidebarService, public _auth:
AuthService) { }

  toggleSidebar() {
    this._sideBarService.toggle();
  }

}

```

6.AUTH

```

import { Injectable } from '@angular/core';
import { CanActivate, ActivatedRouteSnapshot, RouterStateSnapshot, Router }
from '@angular/router';
import { Observable } from 'rxjs';

@Injectable()
export class AdminAuthGuard implements CanActivate {
  constructor(private router: Router) {

  }

  canActivate(
    next: ActivatedRouteSnapshot,
    state: RouterStateSnapshot): Observable<boolean> | Promise<boolean> |
boolean {
    if (localStorage.getItem('token') != null &&
localStorage.getItem('role') === 'ROLE_ADMIN') {
      return true;
    }
    this.router.navigate(['/']);
    return false;
  }
}

```

```

import { Injectable } from '@angular/core';
import { CanActivate, ActivatedRouteSnapshot, RouterStateSnapshot, Router }
from '@angular/router';

```

```

import { Observable } from 'rxjs';

@Injectable()
export class AuthGuard implements CanActivate {
  constructor(private router: Router) {

  }

  canActivate(
    next: ActivatedRouteSnapshot,
    state: RouterStateSnapshot): Observable<boolean> | Promise<boolean> |
boolean {
    if (localStorage.getItem('token') != null) {
      return true;
    }
    this.router.navigate(['/']);
    return false;
  }
}

```

```

import { catchError } from 'rxjs/operators';
import { Observable, throwError, of } from 'rxjs';
import { Constant } from '../constant/constant';
import { Injectable } from '@angular/core';
import { HttpClient, HttpHandler, HttpRequest } from '@angular/common/http';
import { Router } from '@angular/router';

@Injectable()
export class AuthService {

  private _loginUrl = Constant.API_ENDPOINT + '/oauth/token';
  private _client_id = 'demo-client';
  private _client_secret = 'demo-secret';

  private tokenHeader = {
    'Authorization': 'Basic ' + btoa(this._client_id + ':' +
this._client_secret),
    'Content-Type': 'application/x-www-form-urlencoded'
  };

  constructor(private http: HttpClient, private _router: Router) { }

  // convert to form data
  getFormUrlEncoded(toConvert) {
    const formBody = [];
    // tslint:disable-next-line:forin
    for (const property in toConvert) {
      const encodedKey = encodeURIComponent(property);
      const encodedValue = encodeURIComponent(toConvert[property]);
      formBody.push(encodedKey + '=' + encodedValue);
    }
    return formBody.join('&');
  }

  errorHandler(error: any) {

```

```

        console.log('Auth Service api error ', error);
        return throwError(error);
    }

    loginUser(user) {
        localStorage.removeItem('token');
        localStorage.removeItem('access_token');

        const data = {
            'username': user.username,
            'password': user.password,
            'grant_type': 'password'
        };

        return this.http.post<any>(this._loginUrl, this.getFormUrlEncoded(data), {
            headers: this.tokenHeader });
    }

    refreshToken(request: HttpRequest<any>, next: HttpHandler) {

        if (localStorage.getItem('refreshToken') != null) {
            this.callRefreshToken()
                .subscribe(res => {
                    // console.log('Token retrieve successful', res);
                    localStorage.setItem('token', res.access_token);
                    localStorage.setItem('refreshToken', res.refresh_token);
                    next.handle(request);
                    location.reload();
                },
                err => {
                    // console.log('refresh token also results into error ', err);
                    this.logout();
                });
        } else {
            // console.log("Cant use Refresh token");
            this._router.navigate(['/']);
        }
    }

    callRefreshToken(): Observable<any> {
        localStorage.removeItem('token');
        // console.log("refresh token called ",
        localStorage.getItem("refreshToken"));

        const data = {
            'grant_type': 'refresh_token',
            'client_id': this._client_id,
            'client_secret': this._client_secret,
            'refresh_token': localStorage.getItem('refreshToken')
        };

        return this.http.post<any>(this._loginUrl, this.getFormUrlEncoded(data), {
            headers: this.tokenHeader })
            .pipe(catchError(this.errorHandler));
    }

```

```

isLoggedIn() {
  // console.Log("is Logged In ", !!localStorage.getItem("token"));
  return !!localStorage.getItem('token');
}

isAdmin() {
  return localStorage.getItem('role') === 'ROLE_ADMIN';
}

logout() {
  // console.Log("Logged Out called");
  localStorage.removeItem('token');
  localStorage.removeItem('refreshToken');
  localStorage.removeItem('role');
  localStorage.clear();
  this._router.navigate(['/']);
  // location.reload();
}
}

```

7.LOGIN

```

<div class="container">
  <form #loginForm="ngForm" (ngSubmit)="userLogin(loginForm)" class="mx-auto
col-md-6 col-lg-5 offset-md-3 loginform-box">

    <h4 class="text-center mb-3 login-title"> Login </h4>

    <div *ngIf="!!login_user_msg" [ngClass]="has_error ? 'alert alert-danger':
'alert alert-info'">{{login_user_msg}}</div>

    <div class="row">
      <mat-form-field class="col-sm-12">
        <input matInput name="username" placeholder="Enter your username"
ngModel autofocus minlength="2" required>
      </mat-form-field>
    </div>

    <div class="row">
      <mat-form-field class="col-sm-12">
        <input matInput name="password" ngModel placeholder="Enter your
password" [type]="hide ? 'password' : 'text'" required>
        <mat-icon matSuffix (click)="hide = !hide">{{hide ? 'visibility_off' :
'visibility'}}</mat-icon>
      </mat-form-field>
    </div>

    <button class="mt-4 btn-block" type="submit" mat-raised-button
color="primary" [disabled]="loginForm.invalid">Log in</button>

  </form>
</div>

```

```

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

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

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

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

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

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

```

```

/* Login Form Box */
.loginform-box {
  border: 1px solid rgb(206, 204, 204);
  border-radius: 7px;
  /* background-color: #3297a7; */
  padding: 2%;
  margin: 10% 0%;
  /* color: #fff; */
}
.login-title {
  letter-spacing: 1px;
  font-weight: 400;
  color: #3f51b5;
}

```

8.TS FILE

```

import { Component, OnInit } from '@angular/core';
import { Router } from '@angular/router';
import { AuthService } from '../dashboard/auth/auth.service';

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

```

```

login_user_msg: string;
has_error = false;
hide = true;

constructor(private _router: Router, public _auth: AuthService) { }

ngOnInit() {
  if (localStorage.getItem('token') != null) {
    this._router.navigate(['/home']);
  }
}

userLogin(form) {
  // console.log("form", form.value);
  this._auth.loginUser(form.value)
    .subscribe(res => {
      // console.log("Token reterive successful", res)
      this.has_error = false;
      this.login_user_msg = 'Login in, Please wait... !!!';
      localStorage.setItem('token', res.access_token);
      localStorage.setItem('refreshToken', res.refresh_token);
      this._router.navigate(['/home']);
    },
    error => {
      // console.log("user login error", error.error);
      this.has_error = true;
      this.login_user_msg = 'Invalid Username and Password !!!';
    });
}
}

```

9.APP ROUTING MODULE

```

import { AuthGuard } from './feature/dashboard/auth/auth.guard';
import { LoginComponent } from './feature/login/login.component';
import { NgModule } from '@angular/core';
import { Routes, RouterModule } from '@angular/router';
import { PageNotFoundComponent } from './core/page-notfound/page-notfound.component';

const routes: Routes = [
  {path: 'home', loadChildren:
'./feature/dashboard/dashboard.module#DashboardModule', canActivate:
[AuthGuard]},
  { path: '', component: LoginComponent },
  {path: '**', component: PageNotFoundComponent},
];

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

```

10.TS FILE

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

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

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

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

  it('should render title in a h1 tag', () => {
    const fixture = TestBed.createComponent(AppComponent);
    fixture.detectChanges();
    const compiled = fixture.debugElement.nativeElement;
    expect(compiled.querySelector('h1').textContent).toContain('Welcome to leave-management-angular!');
  });
});
```

11.APP.MODULE.TS

```
import { AuthGuard } from './feature/dashboard/auth/auth.guard';
import { RouterModule } from '@angular/router';
import { NgModule } from '@angular/core';
import { AppComponent } from './app.component';

import { AppRoutingModule } from './app-routing.module';
import { CoreModule } from './core/core.module';
// import { BrowserModule } from '@angular/platform-browser';
import { BrowserAnimationsModule } from '@angular/platform-browser/animations';
import { SharedModule } from './shared/shared.module';
import { FeatureModule } from './feature/feature.module';
import 'hammerjs';

@NgModule({
```

```

    declarations: [
      AppComponent,
    ],
    imports: [
      BrowserAnimationsModule,
      AppRoutingModule,
      RouterModule,
      CoreModule,
      FeatureModule,
      SharedModule,
    ],
    providers: [AuthGuard],
    bootstrap: [AppComponent]
  })
export class AppModule { }

```

12.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));

```

13.STYLES.CSS

```

/* You can add global styles to this file, and also import other style files
*/
@import "~@ng-select/ng-select/themes/default.theme.css";
@import "~@angular/material/prebuilt-themes/indigo-pink.css";

body {
  margin: 0;
  background-color: #f7f7f7;
  font-family: sans-serif;
  font-size: .875rem;
  font-weight: 400;
  line-height: 1.5;
}

.outlet-page-content {
  margin-top: 1%;
}

/* Custom Paragraph */
.long-copy {
  line-height: 145%;
  width: 70%;
}

```

```

    text-align: center;
    margin: 0 auto;
    margin-bottom: 1%;
    letter-spacing: 1px;
}
.content-box {
    padding: 0.5%;
}
/* ALL Form Box */
.form-box {
    margin-top: 1%;
    border: 1px solid #dadada;
    padding: 2%;
    background-color: #f7f7f7;
    box-shadow: 0px 0px 12px #dadada;
}
.custom-ngSelect.is-invalid {
    border: 1px solid rgb(248, 20, 20);
    padding: 1%;
}

.form-box .form-group {
    /* border-bottom: 1px solid #dee2e6; */
    padding: 0.5% 0%;
}
.form-box .form-title {
    color: #3f51b5;
    padding: 1% 0%;
    border-bottom: 1px solid #ccc;
}
.custom-tab a.active {
    color: #0000ff;
    border-bottom: 1px solid #0000ff;
}
.table thead {
    cursor: pointer;
}
.mat-icon {
    vertical-align: middle;
}
.table-details {
    font-weight: 100;
    color: #767676;
    font-style: italic;
}
.ngx-pagination li {
    border-radius: 5px !important;
    font-size: 1.05rem;
}
.list-search {
    margin-left: -8%;
}
.cal-month-view .cal-day-badge {
    background-color: #39538c;
}
/* .cal-month-view, .cal-week-view, .cal-day-view {

```

```

    margin: auto;
    zoom: 90%;
} */
/*
.form-box h4 {
    letter-spacing: 1px;
    font-size: 1.7em;
    font-weight: 500;
    font-family: sans-serif;
}
.form-box input {
    height: 45px;
    margin-top: 5px;
}
.form-box .checkbox input {
    height: 20px;
    width: 20px;
} */

/* .selected {
    background-color: #0d597a !important;
    color: white;
} */

.custom-nav {
    background-color: rgb(65, 112, 171);
}

.wrapper {
    display: flex;
    width: 100%;
}

#sidebar {
    width: 250px;
    position: fixed;
    top: 54px;
    left: 0;
    height: 100vh;
    z-index: 999;
    background: rgb(92, 101, 136);
    color: #fff;
    transition: all 0.3s;
}

#sidebar.active {
    margin-left: -250px;
}

#sidebar ul li a {
    padding: 15px 30px;
    font-size: 1em;
    letter-spacing: 0.04em;
    display: block;
    color: #fff;
}

```

```

        cursor: pointer;
    }

#sidebar ul li a:hover {
    text-decoration: none;
    color: #fff;
    background: #7386D5;
}

.sidebar-items .sidebar-header {
    text-align: center;
    font-size: 1.2em;
    margin: 0;
    padding: 8%;
    font-weight: 400;
    background-color: #8e97bd;
}

/* -----
   CONTENT STYLE
   ----- */

#content {
    width: calc(100% - 250px);
    padding: 54px 15px;
    min-height: 100vh;
    transition: all 0.3s;
    position: absolute;
    top: 0;
    right: 0;
}

#content.active {
    width: 100%;
}

.cu-calen-viewdate h3{
    font-size: 1.7em;
}

/* -----
   MEDIAQUERIES
   ----- */

@media (max-width: 768px) {
    #sidebar {
        margin-left: -250px;
    }
    #sidebar.active {
        margin-left: 0;
    }
    #content {
        width: 100%;
    }
    #content.active {
        width: calc(100% - 250px);
    }
}

```

```
}  
}
```

14.TEST.TS

```
// This file is required by karma.conf.js and loads recursively all the .spec  
// and framework files  
  
import 'zone.js/dist/zone-testing';  
import { getTestBed } from '@angular/core/testing';  
import {  
  BrowserDynamicTestingModule,  
  platformBrowserDynamicTesting  
} from '@angular/platform-browser-dynamic/testing';  
  
declare const require: any;  
  
// 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().map(context);
```

15.ANGULAR.JSON

```
{  
  "$schema": "./node_modules/@angular/cli/lib/config/schema.json",  
  "version": 1,  
  "newProjectRoot": "projects",  
  "projects": {  
    "leave-management-angular": {  
      "root": "",  
      "sourceRoot": "src",  
      "projectType": "application",  
      "prefix": "app",  
      "schematics": {},  
      "architect": {  
        "build": {  
          "builder": "@angular-devkit/build-angular:browser",  
          "options": {  
            "outputPath": "dist/leave-management-angular",  
            "index": "src/index.html",  
            "main": "src/main.ts",  
            "polyfills": "src/polyfills.ts",  
            "tsConfig": "src/tsconfig.app.json",  
            "assets": [  
              "src/favicon.ico",  
              "src/assets"  
            ],  
            "styles": [  
              "node_modules/angular-calendar/css/angular-calendar.css",  
              "node_modules/bootstrap/dist/css/bootstrap.min.css",  
            ]  
          }  
        }  
      }  
    }  
  }  
}
```

```

        "node_modules/ngx-bootstrap/datepicker/bs-datepicker.css",
        "src/styles.css"
    ],
    "scripts": [],
  },
  "configurations": {
    "production": {
      "fileReplacements": [
        {
          "replace": "src/environments/environment.ts",
          "with": "src/environments/environment.prod.ts"
        }
      ],
      "optimization": true,
      "outputHashing": "all",
      "sourceMap": false,
      "extractCss": true,
      "namedChunks": false,
      "aot": true,
      "extractLicenses": true,
      "vendorChunk": false,
      "buildOptimizer": true
    }
  },
  "serve": {
    "builder": "@angular-devkit/build-angular:dev-server",
    "options": {
      "browserTarget": "leave-management-angular:build"
    },
    "configurations": {
      "production": {
        "browserTarget": "leave-management-angular:build:production"
      }
    }
  },
  "extract-i18n": {
    "builder": "@angular-devkit/build-angular:extract-i18n",
    "options": {
      "browserTarget": "leave-management-angular:build"
    }
  },
  "test": {
    "builder": "@angular-devkit/build-angular:karma",
    "options": {
      "main": "src/test.ts",
      "polyfills": "src/polyfills.ts",
      "tsConfig": "src/tsconfig.spec.json",
      "karmaConfig": "src/karma.conf.js",
      "styles": [
        "src/styles.css"
      ],
      "scripts": [],
      "assets": [
        "src/favicon.ico",
        "src/assets"
      ]
    }
  }
}

```

```

    }
  },
  "lint": {
    "builder": "@angular-devkit/build-angular:tslint",
    "options": {
      "tsConfig": [
        "src/tsconfig.app.json",
        "src/tsconfig.spec.json"
      ],
      "exclude": [
        "**/node_modules/**"
      ]
    }
  }
},
"leave-management-angular-e2e": {
  "root": "e2e/",
  "projectType": "application",
  "architect": {
    "e2e": {
      "builder": "@angular-devkit/build-angular:protractor",
      "options": {
        "protractorConfig": "e2e/protractor.conf.js",
        "devServerTarget": "leave-management-angular:serve"
      },
      "configurations": {
        "production": {
          "devServerTarget": "leave-management-angular:serve:production"
        }
      }
    }
  },
  "lint": {
    "builder": "@angular-devkit/build-angular:tslint",
    "options": {
      "tsConfig": "e2e/tsconfig.e2e.json",
      "exclude": [
        "**/node_modules/**"
      ]
    }
  }
},
"defaultProject": "leave-management-angular"
}

```

16.PACKAGE.JSON

```

{
  "name": "leave-management-angular",
  "version": "0.0.0",
  "scripts": {
    "ng": "ng",
    "start": "ng serve",
    "build": "ng build",
    "test": "ng test",
    "lint": "ng lint",

```

```

    "e2e": "ng e2e"
  },
  "private": true,
  "dependencies": {
    "@angular/animations": "^6.1.10",
    "@angular/cdk": "^7.0.1",
    "@angular/common": "^6.1.0",
    "@angular/compiler": "^6.1.0",
    "@angular/core": "^6.1.0",
    "@angular/forms": "^6.1.0",
    "@angular/http": "^6.1.0",
    "@angular/material": "^7.0.1",
    "@angular/platform-browser": "^6.1.0",
    "@angular/platform-browser-dynamic": "^6.1.0",
    "@angular/router": "^6.1.0",
    "@ng-select/ng-select": "^2.10.2",
    "@swimlane/ngx-charts": "^10.0.0",
    "angular-calendar": "^0.26.4",
    "bootstrap": "^4.1.3",
    "core-js": "^2.5.4",
    "date-fns": "^1.29.0",
    "hammerjs": "^2.0.8",
    "ngx-bootstrap": "3.0.1",
    "ngx-pagination": "^3.2.1",
    "normalize.css": "^8.0.0",
    "rxjs": "~6.2.0",
    "zone.js": "~0.8.26"
  },
  "devDependencies": {
    "@angular-devkit/build-angular": "~0.8.0",
    "@angular/cli": "~6.2.4",
    "@angular/compiler-cli": "^6.1.0",
    "@angular/language-service": "^6.1.0",
    "@types/jasmine": "~2.8.8",
    "@types/jasminewd2": "~2.0.3",
    "@types/node": "~8.9.4",
    "codelyzer": "~4.3.0",
    "jasmine-core": "~2.99.1",
    "jasmine-spec-reporter": "~4.2.1",
    "karma": "~3.0.0",
    "karma-chrome-launcher": "~2.2.0",
    "karma-coverage-istanbul-reporter": "~2.0.1",
    "karma-jasmine": "~1.1.2",
    "karma-jasmine-html-reporter": "^0.2.2",
    "protractor": "~5.4.0",
    "ts-node": "~7.0.0",
    "tslint": "~5.11.0",
    "typescript": "~2.9.2"
  }
}

```

17.TSLINT

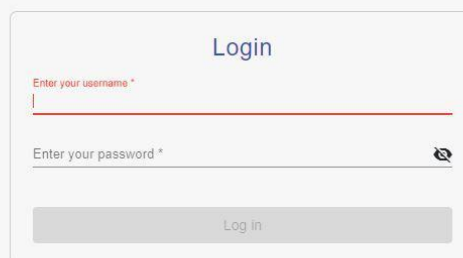
```

{
  "extends": "../tslint.json",

```

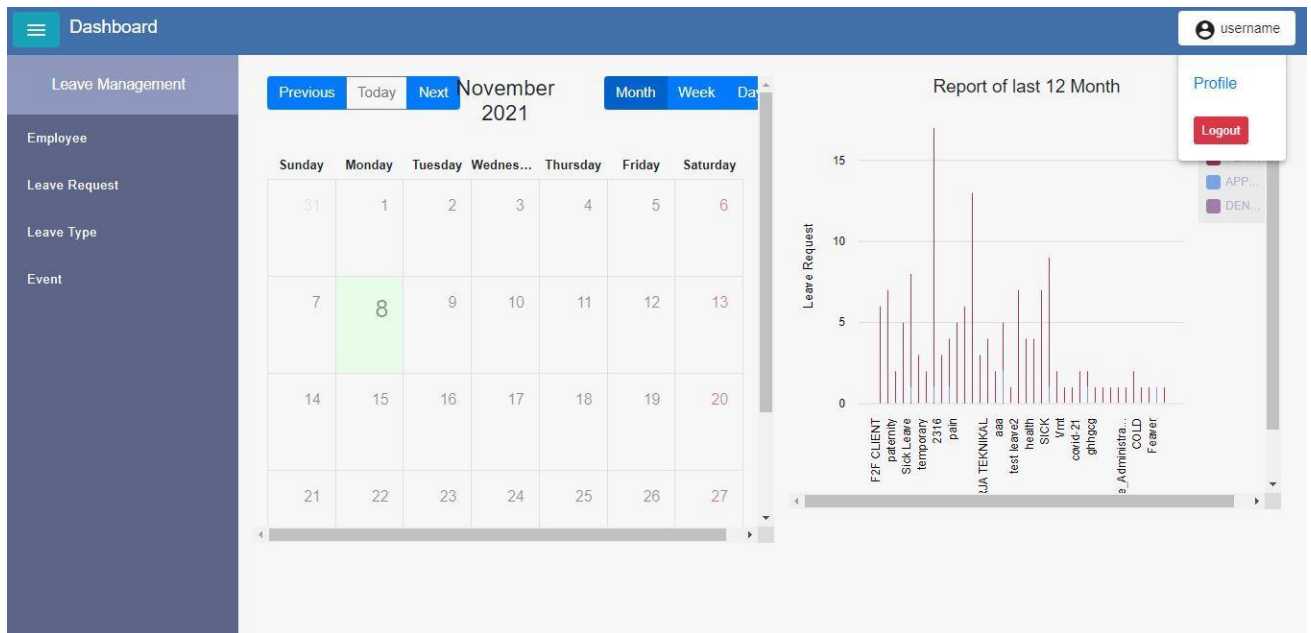
```
"rules": {  
  "directive-selector": [  
    true,  
    "attribute",  
    "app",  
    "camelCase"  
  ],  
  "component-selector": [  
    true,  
    "element",  
    "app",  
    "kebab-case"  
  ]  
}
```

SAMPLE SCREENS



A sample login form with the following elements:

- Title: Login
- Username field: Enter your username *
- Password field: Enter your password * (with a toggle icon)
- Login button: Log in



Dashboard

username

Leave Management

Employee

Leave Request

Leave Type

Event

All Employees New Employee

Search

S.N	Name	Phone Number	Email	Supervisor	Role	Status	Options
1	Aa Sheshasena Reddy	6308657589	sheshasena@gmail.com		ROLE_USER	ACTIVE	View
2	abc def	8013996507			ROLE_USER	ACTIVE	View
3	abc adv	9840355935			ROLE_USER	ACTIVE	View
4	abcd abcd abcd	9876543444	dgsdgh	Sheshasena Reddy	ROLE_USER	ACTIVE	View
5	adf afg	3154612101			ROLE_USER	ACTIVE	View
6	adil bin tariq	3126464132			ROLE_USER	ACTIVE	View
7	adil bin adv	1234562878			ROLE_USER	ACTIVE	View

Dashboard

username

Leave Management

All Employees

New Employee

Employee

Leave Request

Leave Type

Event

Employee Registration

Name *

First Name

Middle Name

Last Name

Username *

Username

Password *

Password

Phone No. *

Phone Number

Email

Email

Supervisor Employee

Select Employee Supervisor

Status *

☒ ACTIVE ☐ INACTIVE

Submit

Dashboard

username

Leave Management

All Leave Types

New Leave Type

Employee

Leave Request

Leave Type

Event

Search

Q

S.N	Leave Type	Options
1	aaaaa	View
2	Conge_Administratif	View
3		View
4		View
5	test leave	View
6	test leave2	View
7	aaa	View

Dashboard username

Leave Management

Employee

Leave Request

Leave Type

Event

All Events

New Event

S.N	Title ↑↓	Start Date ↑↓	End Date ↑↓	Options
1	adfggf	2020-09-30	2020-09-30	<button>View</button>
2	adfggf	2020-09-30	2020-09-30	<button>View</button>
3	adfggf	2020-09-30	2021-02-03	<button>View</button>
4	adfggf	2020-09-30	2020-09-30	<button>View</button>
5	Android Session	2020-07-31	2020-07-31	<button>View</button>
6	Another random session no1	2021-06-29	2021-06-30	<button>View</button>
7	Azure arch	2021-09-08	2021-09-09	<button>View</button>
8	cough	2021-03-18	2021-03-31	<button>View</button>

Dashboard username

Leave Management

Employee

Leave Request

Leave Type

Event

All Events

New Event

Create New Event

Title *

Event Title

Start Date *

Date From

End Date *

Date To

Submit