

## ICIN FRONTEND SOURCE CODE

DEVELOPER NAME – RAJA KUMAR

User-Portal>userportal-angular>e2e>src>app.e2e-spec.ts

```
import { AppPage } from './app.po';
import { browser, logging } from 'protractor';

describe('workspace-project App', () => {
  let page: AppPage;

  beforeEach(() => {
    page = new AppPage();
  });

  it('should display welcome message', () => {
    page.navigateTo();
    expect(page.getTitleText()).toEqual('angular-app app is running!');
  });

  afterEach(async () => {
    // Assert that there are no errors emitted from the browser
    const logs = await browser.manage().logs().get(logging.Type.BROWSER);
    expect(logs).not.toContain(jasmine.objectContaining({
      level: logging.Level.SEVERE,
    } as logging.Entry));
  });
});
```

App.po.ts

```
import { browser, by, element } from 'protractor';

export class AppPage {
  navigateTo(): Promise<unknown> {
    return browser.get(browser.baseUrl) as Promise<unknown>;
  }

  getTitleText(): Promise<string> {
    return element(by.css('app-root .content span')).getText() as
    Promise<string>;
  }
}
```

Protractor.conf.js

```
// @ts-check
// Protractor configuration file, see link for more information
// https://github.com/angular/protractor/blob/master/lib/config.ts

const { SpecReporter, StacktraceOption } = require('jasmine-spec-reporter');

/**
 * @type { import("protractor").Config }
 */
exports.config = {
  allScriptsTimeout: 11000,
  specs: [
    './src/**/*.e2e-spec.ts'
  ],
  capabilities: {
    browserName: 'chrome'
  },
  directConnect: true,
  baseUrl: 'http://localhost:4200/',
  framework: 'jasmine',
  jasmineNodeOpts: {
    showColors: true,
    defaultTimeoutInterval: 30000,
    print: function() {}
  },
  onPrepare() {
    require('ts-node').register({
      project: require('path').join(__dirname, './tsconfig.json')
    });
    jasmine.getEnv().addReporter(new SpecReporter({
      spec: {
        displayStacktrace: StacktraceOption.PRETTY
      }
    }));
  }
};

/* To learn more about this file see: https://angular.io/config/tsconfig. */
{
  "extends": "../tsconfig.base.json",
  "compilerOptions": {
    "outDir": "../out-tsc/e2e",
    "module": "commonjs",
    "target": "es2018",
    "types": [
      "jasmine",
    ]
  }
}
```

```
    "jasminewd2",  
    "node"  
  ]  
}  
}
```

User-portal>userportal-angular>app>models

```
//checkbookrequest.ts  
export class CheckBookRequest{  
  
    accNo:string;  
    pages:string;  
  
}  
//Checkbooksresponce.ts  
export class ChequebookResponse{  
    status:boolean;  
    responseMessage:String;  
    account:number;  
}  
//Index.ts  
export * from './user';  
//savingaccount.ts  
export * from './user';  
  
//teansactiopn.ts  
export class Transaction{  
  
    date>Date;  
    id:number;  
    action:String;  
    amount:number;  
}  
  
//transferhistory.ts  
export class TransferHistory{  
    id:number;  
    saccount:string;  
    raccount:string;  
    amount:number;  
    date>Date;  
}
```

```

//user.ts
export class User {
    firstName: string;
    lastName: string;
    userName:string;
    password: string;
    dob:Date;
    phone: number;
    address:string;
    type:string;
    identity:string;
    email: string;
}

export class Login {
    username:string;
    password: string;
}

//userdisplay.ts
export class UserDisplay{

    primaryAccno:number;
    savingsAccno:number;
    primaryBalance:number;
    savingsBalance:number;
}

//userupdate.ts
export class UserUpdate {
    userName:string;
    prevpassword:string;
    password: string;
    phone: number;
    address:string;
    email: string;
}

```

**User-portal>userportal-angular>app>cheque-book-request**

```

//cheque-book-request.component.ts
.container{

```

```
margin: 10rem auto;
width:20em;
}
```

```
.input{
margin-bottom: 2rem ;
margin-top: 2rem;
}
```

```
//cheque-book-request.component.html
```

```
<nav class="navbar navbar-expand-sm navbar-dark bg-dark">
  <a class="navbar-brand" routerLink="/home">ICIN Bank</a>
  <button class="navbar-toggler" type="button" data-toggle="collapse" data-
target="#navbarsExample04" aria-controls="navbarsExample04" aria-
expanded="false" aria-label="Toggle navigation">
    <span class="navbar-toggler-icon"></span>
  </button>

  <div class="collapse navbar-collapse" id="navbarsExample04">
    <ul class="navbar-nav mr-auto">

      <li class="nav-item">
        <a class="nav-link" routerLink="/transactionHistory">Transaction
History<span class="sr-only"></span></a>
      </li>

      <li class="nav-item">
        <a class="nav-link" routerLink="/transferHistory">Transfer
History<span class="sr-only"></span></a>
      </li>

      <li class="nav-item">
```

```

        <a class="nav-link" routerLink="/transfer">Transfer Money<span
class="sr-only"></span></a>

    </li>

    <li class="nav-item">

        <a class="nav-link" routerLink="/chequebookRequest">Request
Checkbook<span class="sr-only"></span></a>

    </li>

    <li class="nav-item dropdown" >

        <a class="nav-link dropdown-toggle" href="#" id="dropdown04" data-
toggle="dropdown" aria-haspopup="true" aria-expanded="false">User Session</a>

        <div class="dropdown-menu" aria-labelledby="dropdown04">

            <a class="dropdown-item" routerLink="/editProfile" >Change Profile
Settings</a>

            <a class="dropdown-item" routerLink="/login">Logout</a>

        </div>

    </li>

</ul>

</div>
</nav>

<div class="container col-xs-4">

    <h3 style="text-align: center;">Request Cheque Book</h3>

    <div class="form-group">

        <div style="text-align: center;" class="form-group">

            <label>Account Number :&nbsp;  </label>
<label>{{requestingAccNo}}</label>

        </div>

    </div>

    <div class="form-group">

```

```

        <label>ChequeBook pages</label>

        <br>

        <select class="form-control" (change) =
"filterSelected($event.target.value)" >

            <option *ngFor="let p of pages" [value]='p.value'>

                {{p.name}}

            </option>

        </select>

        <br>

        <button (click) = "setOption()" class="btn btn-info form-
control"><span *ngIf="loading" class="spinner-border spinner-border-sm mr-
1"></span>Request Chequebook</button>

        <br>

    </div>
</div>

```

```
//cheque-book-request.component.ts
```

```

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

import { ChequeBookRequestComponent } from './cheque-book-request.component';

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

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

```

```

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

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

//cheque-book-request.component.ts

import { Component, OnInit } from '@angular/core';
import { FormBuilder, FormGroup, Validators } from '@angular/forms';
import { Router } from '@angular/router';
import { RequestService } from '../request.service';
import Swal from 'sweetalert2';

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

    requestingAccNo:number=+localStorage.getItem("savingAccNo");

    selectedValue: number;

```



```
loading:boolean=false;

pages = [

  { name: "20", value: 20 },
  { name: "50", value: 50 },
  { name: "75", value: 75 }
]

filterSelected(value){
  this.selectedValue = value;
}

constructor(
  private formBuilder: FormBuilder,
  private router: Router,
  private requestService : RequestService,
)
{}

ngOnInit(): void {

}

setOption() {
  this.loading = true;
  if(this.selectedValue==null){
    this.selectedValue=20;
  }
  try{
    this.requestService.insertRequest(this.requestingAccNo,+this.selectedValue).subscribe((res:any)=>{
```

```

        console.log(res);

        this.loading = false;

        if(res.status===true){
            Swal.fire(
                {
                    icon: 'success',
                    title: 'Chequebook request placed!',
                    text:res.responseMessage
                }
            )
        }else{
            Swal.fire({
                icon: 'error',
                title: 'Oops...',
                text: res.responseMessage,
            })
        }
    });
}

catch{
    this.loading = false;
}

}

}

```

App.edit-profile

```

.container{
    margin: 3rem auto;
    width:20em;

```

```
}
```

```
.input{
```

```
margin-bottom: 4rem ;
```

```
margin-top: 2rem;
```

```
}
```

```
<meta name="viewport" content="width=device-width, initial-scale=1.0">
```

```
<nav class="navbar navbar-expand-sm navbar-dark bg-dark">
```

```
  <a class="navbar-brand" routerLink="/home">ICIN Bank</a>
```

```
  <button class="navbar-toggler" type="button" data-toggle="collapse" data-  
target="#navbarsExample04" aria-controls="navbarsExample04" aria-  
expanded="false" aria-label="Toggle navigation">
```

```
    <span class="navbar-toggler-icon"></span>
```

```
  </button>
```

```
<div class="collapse navbar-collapse" id="navbarsExample04">
```

```
  <ul class="navbar-nav mr-auto">
```

```
    <li class="nav-item">
```

```
      <a class="nav-link" routerLink="/transactionHistory">Transaction  
History<span class="sr-only"></span></a>
```

```
    </li>
```

```
    <li class="nav-item">
```

```
      <a class="nav-link" routerLink="/transferHistory">Transfer  
History<span class="sr-only"></span></a>
```

```
    </li>
```

```
    <li class="nav-item">
```

```

        <a class="nav-link" routerLink="/transfer">Transfer Money<span
class="sr-only"></span></a>

    </li>

    <li class="nav-item">

        <a class="nav-link" routerLink="/chequebookRequest">Request
Checkbook<span class="sr-only"></span></a>

    </li>

    <li class="nav-item dropdown" >

        <a class="nav-link dropdown-toggle" href="#" id="dropdown04" data-
toggle="dropdown" aria-haspopup="true" aria-expanded="false">User Session</a>

        <div class="dropdown-menu" aria-labelledby="dropdown04">

            <a class="dropdown-item" routerLink="/editProfile" >Change Profile
Settings</a>

            <a class="dropdown-item" routerLink="/login">Logout</a>

        </div>

    </li>

</ul>

</div>
</nav>

<div class="container">

    <h3>Edit Profile</h3>

    <br>

    <form [formGroup]="updateForm" (ngSubmit)="update()">

        <div class="col-xs-4" style="left:40%;">

            <div class="form-group">

                <label for="phone">Mobile no</label>

```

```

        <input type="text" formControlName="phone" class="form-control"
[ngClass]="{ 'is-invalid': submitted && fval.phone.errors }"
placeholder="Enter Phone here"/>

        <div *ngIf="submitted && fval.phone.errors" class="invalid-feedback">

            <div *ngIf="fval.phone.errors.required">Mobile no is
required</div>

        </div>
    </div>

    <div class="form-group">

        <label for="address">Address</label>

        <input type="text" formControlName="address" class="form-control"
[ngClass]="{ 'is-invalid': submitted && fval.address.errors }"
placeholder="Enter your Address"/>

        <div *ngIf="submitted && fval.address.errors" class="invalid-
feedback">

            <div *ngIf="fval.address.errors.required">Address is
required</div>

        </div>

    <br>

    <div class="form-group">

        <label for="email">Email</label>

        <input type="text" formControlName="email" class="form-control"
[ngClass]="{ 'is-invalid': submitted && fval.email.errors }"
placeholder="Enter email here"/>

        <div *ngIf="submitted && fval.email.errors" class="invalid-feedback">

            <div *ngIf="fval.email.errors.required">Email is required</div>

            <div *ngIf="fval.email.errors.email">Enter valid email address</div>

        </div>
    </div>

    <div class="form-group">
        <label for="password"> Previous Password</label>

```



```
</form>
```

```
</div>
```

```
<div>
```

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

```
</div>
```

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

```
import { EditProfileComponent } from './edit-profile.component';
```

```
describe('EditProfileComponent', () => {
```

```
  let component: EditProfileComponent;
```

```
  let fixture: ComponentFixture<EditProfileComponent>;
```

```
  beforeEach(async(() => {
```

```
    TestBed.configureTestingModule({
```

```
      declarations: [ EditProfileComponent ]
```

```
    })
```

```
    .compileComponents();
```

```
  }));
```

```
  beforeEach(() => {
```

```
    fixture = TestBed.createComponent(EditProfileComponent);
```

```
    component = fixture.componentInstance;
```

```
    fixture.detectChanges();
```

```
  });
```

```

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

import { Component, OnInit } from '@angular/core';
import { FormBuilder, FormGroup, Validators, } from '@angular/forms';
import { Router } from '@angular/router';
import { UpdateService } from '../update.service';
import { UserUpdate } from '../_models/userupdate';
import Swal from 'sweetalert2';

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

    username:string=localStorage.getItem("username");

    constructor(
        private formBuilder: FormBuilder,
        private router: Router,
        private updateService: UpdateService
    ) { }

    updateForm: FormGroup;
    loading = false;
    submitted = false;

```



```

ngOnInit() {
  this.updateForm = this.formBuilder.group({
    phone: ['',],
    address:[''],
    email: ['', [Validators.email]],
    prevpassword: ['', [Validators.minLength(6)]],
    password:['',[Validators.minLength(6)]]

  });
}

get fval() { return this.updateForm.controls; }

update(){
  this.submitted = true;

  // return for here if form is invalid
  if (this.updateForm.invalid) {
    return;
  }

  this.loading = true;

  const result:UserUpdate= Object.assign({}, this.updateForm.value);

  try{
    this.updateService.update(this.username,result.phone,result.email,result.address,result.prevpassword,result.password).subscribe(

      (data : any) =>{

        // localStorage.clear();

        // localStorage.setItem('user',JSON.stringify(data));

        console.log(data);

        this.loading=false;

        if(data.flag==true){

          Swal.fire(

            {

              icon: 'success',

```

```

        title: 'Profile updated successfully!',
        text: data.message
    }
)
}else{
    Swal.fire({
        icon: 'error',
        title: 'Oops...',
        text: data.message,
    })
}

this.router.navigate(['/editProfile']);

}
);
}catch{
    this.loading=false;
    Swal.fire({
        icon: 'error',
        title: 'Oops...',
        text: "Something went wrong!"
    })
}

}
}

```

App>home

```
body {
    font-family: Cambria, Cochin, Georgia, Times, 'Times New Roman', serif;
}

#userSession{
    float: right;
}

.center {
    left: 50%;
    margin-left: auto;
    margin-right: auto;
}

#userSession{
    float: right;
}

.center {
    left: 50%;
    margin-left: auto;
    margin-right: auto;
}

body .infobox{
    font-family: Cambria, Cochin, Georgia, Times, 'Times New Roman', serif;
    font-weight: 400;
    font-size: 20px;
    color: #323232;
    height: auto;
```

```

padding: 2%;
background-color: #f1ecec;
border: 1px solid #f5f5f5;
border-radius: 15px;
-webkit-box-shadow: 0px 0px 10px 0px #f5f5f5;
box-shadow: 0px 0px 10px 0px #f5f5f5;
margin: 10rem auto;
width: 35em;
}

<nav class="navbar navbar-expand-sm navbar-dark bg-dark">
  <a class="navbar-brand" routerLink="/home">ICIN Bank</a>
  <button class="navbar-toggler" type="button" data-toggle="collapse" data-
target="#navbarsExample04" aria-controls="navbarsExample04" aria-
expanded="false" aria-label="Toggle navigation">
    <span class="navbar-toggler-icon"></span>
  </button>

  <div class="collapse navbar-collapse" id="navbarsExample04">
    <ul class="navbar-nav mr-auto">

      <li class="nav-item">
        <a class="nav-link" routerLink="/transactionHistory">Transaction
History<span class="sr-only"></span></a>
      </li>

      <li class="nav-item">
        <a class="nav-link" routerLink="/transferHistory">Transfer
History<span class="sr-only"></span></a>
      </li>

      <li class="nav-item">

```

```

        <a class="nav-link" routerLink="/transfer">Transfer Money<span
class="sr-only"></span></a>

    </li>

    <li class="nav-item">

        <a class="nav-link" routerLink="/chequebookRequest">Request
Checkbook<span class="sr-only"></span></a>

    </li>

    <li class="nav-item dropdown" >

        <a class="nav-link dropdown-toggle" href="#" id="dropdown04" data-
toggle="dropdown" aria-haspopup="true" aria-expanded="false">User Session</a>

        <div class="dropdown-menu" aria-labelledby="dropdown04">

            <a class="dropdown-item" routerLink="/editProfile" >Change Profile
Settings</a>

            <a class="dropdown-item" routerLink="/login">Logout</a>

        </div>

    </li>

</ul>

</div>
</nav>

<body class="landing" style="background-image:url('background.jpg');">

    <div class="infobox" style="text-align: center;">

        <br>

        <h2>Welcome to ICIN Bank </h2>

        <br>

        <h5>We offer you actions like Transfer of money, Checkbook request, Deposit,
Withdraw</h5>

        <br>

        <h4>Happy Banking !!!</h4>

        <br><br>

```

```

<table class="center">

    <tr>

        <td><h4>Your Savings Account number: </h4></td>

        <td><h4>{{savingAcc}}</h4></td>

    </tr>

    <tr>

        <td><h4>Your Savings Account balance : </h4></td>

        <td><h4>{{savingBalanceLocal}}</h4></td>

    </tr>

</table>

</div>
</body>

<!-- <h3>Select the operation to be performed:</h3>
Deposit:<input type="checkbox" ng-model="depositClicked">
Withdraw:<input type="checkbox" ng-model="withdrawClicked"> -->

    <!-- <label>The account where you need to deposit: </label>
<br>

    <select id="dropDownAccountType" required>
    <option value="Primary">Primary</option>
    <option value="Secondary">Secondary</option>

    </select> -->

```

```

<!-- <table class="center" style="text-indent: 1cm; margin-right: auto;
margin-left: auto;">

    <tr>

        <td><button class="btn btn-info"><a
routerLink="deposit">Deposit</a></button></td>

        <td><button class="btn btn-info"><a
routerLink="withdraw">Withdraw</a></button></td>

    </tr>
</table> -->

    <!-- <label>The account from where you need to withdraw: </label>
<br>

    <select id="dropDownAccountType" required>

        <option value="Primary">Primary</option>

        <option value="Secondary">Secondary</option>

    </select>

    //(click)={onWithdraw}

-->

<!--
<a href="\src\deposit.html">Deposit Money</a>
<br>
<br>
<a href="\src\withdraw.html">Withdraw Money</a> -->

<!--Admin Login
<button (onclick)="adminLogin()">Admin Test</button>-->
<div>

    <router-outlet></router-outlet>

```

</div>

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

```
import { HomeComponent } from '../home.component';
```

```
describe('HomeComponent', () => {
```

```
  let component: HomeComponent;
```

```
  let fixture: ComponentFixture<HomeComponent>;
```

```
  beforeEach(async(() => {
```

```
    TestBed.configureTestingModule({
```

```
      declarations: [ HomeComponent ]
```

```
    })
```

```
    .compileComponents();
```

```
  }));
```

```
  beforeEach(() => {
```

```
    fixture = TestBed.createComponent(HomeComponent);
```

```
    component = fixture.componentInstance;
```

```
    fixture.detectChanges();
```

```
  });
```

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

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

```
  });
```

```
});
```

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

```
import { AuthService } from '../auth.service';
```

```
import { UserService } from '../user.service';
```



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

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

  username:String = localStorage.getItem("username");
  savingAcc:number;
  primaryAcc:number;
  savingBalanceLocal:number;
  primaryBalanceLocal:number;

  constructor(public authService:AuthService, public
userService:UserService,private router: Router) {
    console.log(this.username);
  }

  ngOnInit(): void {

    this.userService.getUser(this.username).subscribe(res=>{
      this.savingAcc = res.savingsAccno;
      this.primaryAcc = res.primaryAccno;
      this.savingBalanceLocal = res.savingsBalance;
      this.primaryBalanceLocal = res.primaryBalance;
      localStorage.setItem("savingAccNo",this.savingAcc.toString());
    });

  }
}
```

```

displayuserdetails() {
    this.userService.getUser(this.username).subscribe(res=>this.ngOnInit());
}
}

```

## App>login

```

.container{
    margin: 5rem auto;
    width:20rem;
}

.input{
    margin-bottom: 2rem ;
    margin-top: 2rem;
}

<div class="container">
<form [formGroup]="loginForm" (ngSubmit)="onFormSubmit()">
    <br>
    <br>
    <h3 style="text-align: center;">User Portal</h3>
    <br>
    <div class="form-group">
        <label for="username">Username</label>
        <input type="text" formControlName="username" class="form-control"
[ngClass]="{ 'is-invalid': submitted && fval.username.errors }"
placeholder="Enter Username"/>
        <div *ngIf="submitted && fval.username.errors" class="invalid-
feedback">

```

```

        <div *ngIf="fval.username.errors.required">Username is
required</div>

    </div>

</div>

<div class="form-group">

    <label for="password">Password</label>

    <input type="password" formControlName="password" class="form-control"
[ngClass]="{ 'is-invalid': submitted && fval.password.errors }"
placeholder="Enter password here" />

    <div *ngIf="submitted && fval.password.errors" class="invalid-
feedback">

        <div *ngIf="fval.password.errors.required">Please enter a
Password</div>

    </div>

</div>

<div class="form-group" style="text-align:center;">

    <button [disabled]="loading" class="btn btn-primary">

        <span *ngIf="loading" class="spinner-border spinner-border-sm mr-
1"></span>

        Login

    </button>

    <a routerLink="/register" class="btn btn-link">Register</a>

</div>
</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();
});
});

import { Component, OnInit } from '@angular/core';
import { Router, ActivatedRoute } from '@angular/router';
import { FormBuilder, FormGroup, Validators } from '@angular/forms';
import { Login } from '../_models';
import { LoginService } from '../login.service';
import { HttpResponse } from '@angular/common/http';
import { AuthService } from '../auth.service';
import Swal from 'sweetalert2';

@Component({

```

```

    selector: 'app-login',
    templateUrl: './login.component.html',
    styleUrls: ['./login.component.css']
  })
}

export class LoginComponent implements OnInit {
  loginForm: FormGroup;
  loading = false;
  submitted = false;
  returnUrl: string;
  isLoginError : boolean = false;

  constructor(
    private formBuilder: FormBuilder,
    private route: ActivatedRoute,
    private router: Router,
    private loginService : LoginService,
    private authService : AuthService
  ) { }

  ngOnInit() {
    this.loginForm = this.formBuilder.group({
      username: ['', Validators.required],
      password: ['', Validators.required]
    });
  }

  // for accessing the form fields
  get fval() { return this.loginForm.controls; }
}

```

```
onFormSubmit() {  
  this.submitted = true;  
  
  if (this.loginForm.invalid) {  
    return;  
  }  
  
  this.loading = true;  
  
  const result:Login= Object.assign({}, this.loginForm.value);  
  this.loginService.loginUser(result.username,result.password).subscribe(  
    (data : any) =>{  
      localStorage.setItem('login',data.loginStatus);  
      localStorage.setItem('username',data.username);  
      if(data.loginStatus==true){  
        Swal.fire({  
          icon: 'success',  
          title: 'Login successful',  
          showConfirmButton: false,  
          timer: 2000  
        })  
        this.router.navigate(['/home']);  
      }  
      else{  
        Swal.fire({  
          icon: 'error',  
          title: 'Oops...',  
          text: data.responseMessage,  
        })  
        this.router.navigate(['/login']);  
        this.loading = false;  
      }  
    },  
  },
```

```

(err : HttpErrorResponse)=>{
    this.isLoginError = true;
});

const logindata = new Login();
this.authService.authenticate(this.isLoginError);

}
}

```

## App>register

```

<div class="container">

<form [formGroup]="registerForm" (ngSubmit)="onFormSubmit()">
    <div class="col-xs-4" style="left:40%;">
        <br>
        <h3 style="text-align: center;">User Registration</h3>
        <br>

        <div class="form-group">
            <label for="firstName">First Name</label>
            <input type="text" formControlName="firstName" class="form-control"
[ngClass]="{ 'is-invalid': submitted && fval.firstName.errors }"
placeholder="Enter First Name here"/>
            <div *ngIf="submitted && fval.firstName.errors" class="invalid-
feedback">
                <div *ngIf="fval.firstName.errors.required">First Name is
required</div>
            </div>
        </div>
    </div>

```

```

<div class="form-group">
  <label for="lastName">Last Name</label>

  <input type="text" formControlName="lastName" class="form-
control" [ngClass]="{ 'is-invalid': submitted && fval.lastName.errors }"
placeholder="Enter Last Name"/>

  <div *ngIf="submitted && fval.lastName.errors" class="invalid-
feedback">

    <div *ngIf="fval.lastName.errors.required">LastName is
required</div>

  </div>
</div>

<div class="form-group">
  <label for="userName">Username</label>

  <input type="text" formControlName="userName" class="form-
control" [ngClass]="{ 'is-invalid': submitted && fval.userName.errors }"
placeholder="Enter User Name"/>

  <div *ngIf="submitted && fval.userName.errors" class="invalid-
feedback">

    <div *ngIf="fval.userName.errors.required">UserName is
required</div>

  </div>
</div>

<div class="form-group">
  <label for="password"> Password</label>

  <input type="password" formControlName="password" class="form-control"
[ngClass]="{ 'is-invalid': submitted && fval.password.errors }"
placeholder="Enter Password here"/>

  <div *ngIf="submitted && fval.password.errors" class="invalid-feedback">

    <div *ngIf="fval.password.errors.required">Password is required</div>

    <div *ngIf="fval.password.errors.minlength">Password must be at least
8 scharacters</div>

```



```

    </div>
</div>

<div class="form-group">
    <label for="dob">DOB</label>

    <input type="date" formControlName="dob" class="form-control"
[ngClass]="{ 'is-invalid': submitted && fval.dob.errors }"/>

    <div *ngIf="submitted && fval.dob.errors" class="invalid-feedback">
        <div *ngIf="fval.dob.errors.required">Date of Birth is
required</div>
    </div>
</div>

<div class="form-group">
    <label for="phone">Mobile No</label>

    <input type="text" formControlName="phone" class="form-control"
[ngClass]="{ 'is-invalid': submitted && fval.phone.errors }"
placeholder="Enter Phone here"/>

    <div *ngIf="submitted && fval.phone.errors" class="invalid-feedback">
        <div *ngIf="fval.phone.errors.required">Phone is required</div>
    </div>
</div>

<div class="form-group">
    <label for="address">Address</label>

    <input type="text" formControlName="address" class="form-control"
[ngClass]="{ 'is-invalid': submitted && fval.address.errors }"
placeholder="Enter your Address"/>

    <div *ngIf="submitted && fval.address.errors" class="invalid-feedback">
        <div *ngIf="fval.address.errors.required">Address is required</div>
    </div>
</div>

```

```

<div class="form-group">
  <label for="type">Choose ID Type:</label>

  <select class="form-control" formControlName="identityType" class="form-
control" [ngClass]="{ 'is-invalid': submitted &&
fval.identityType.errors.required }" (change) =
"filterSelected($event.target.value)" >

    <option *ngFor="let t of identityType" [value]='t.value'>
      {{t.name}}
    </option>
  </select>
</div>

<div class="form-group">
  <label for="file">Upload your Id proof copy</label>

  <input type="file" formControlName="file" class="form-control"
[ngClass]="{ 'is-invalid': submitted && fval.file.errors }" />

  <div *ngIf="submitted && fval.file.errors" class="invalid-feedback">

    <div *ngIf="fval.file.errors.required">Required</div>
  </div>
</div>

<div class="form-group">
  <label for="identity">Enter Id Proof no:</label>

  <input type="text" formControlName="identity" class="form-control"
[ngClass]="{ 'is-invalid': submitted && fval.identity.errors }"
placeholder="Enter Id Proof no: " />

  <div *ngIf="submitted && fval.identity.errors" class="invalid-
feedback">

    <div *ngIf="fval.identity.errors.required">Required</div>
  </div>
</div>

<div class="form-group">
  <label for="email">Email</label>

```

```

    <input type="text" formControlName="email" class="form-control"
[ngClass]="{ 'is-invalid': submitted && fval.email.errors }"
placeholder="Enter email here"/>

    <div *ngIf="submitted && fval.email.errors" class="invalid-feedback">
        <div *ngIf="fval.email.errors.required">Email is required</div>
        <div *ngIf="fval.email.errors.email">Enter valid email address</div>
    </div>
</div>

<div class="center1" style="text-align:center;">
    <button class="btn btn-primary">Register</button>
</div>

</div>
</form>
</div>

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

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

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

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

    beforeEach(() => {

```

```

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

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

import { Component, OnInit } from '@angular/core';
import { FormBuilder, FormGroup, Validators } from '@angular/forms';
import { Router } from '@angular/router';
import { User } from '../_models';
import { RegisterService } from '../register.service';
import Swal from 'sweetalert2';

@Component({
    selector: 'app-register',
    templateUrl: './register.component.html'
})
export class RegisterComponent implements OnInit {

    constructor(
        private formBuilder: FormBuilder,
        private router: Router,
        private registerService : RegisterService,

    ) { }

    identityType = [
        { name: "Aadhar Card", value:"aadhar"},

```

```

        { name: "PAN card", value:"pancard"},
        { name: "Passport", value:"passport"},
        { name: "Voter Id Card", value: "voter" }
    ]
    registerForm: FormGroup;
    loading = false;
    submitted = false;
    selectedOption:string;
    ngOnInit() {
        this.registerForm = this.formBuilder.group({
            firstName: ['', Validators.required],
            lastName: ['', Validators.required],
            userName:['',Validators.required],
            password: ['', [Validators.required, Validators.minLength(6)]],
            dob:['',Validators.required],
            phone: ['', Validators.required],
            address:['',Validators.required],
            identityType:['',Validators.required],
            file:['',Validators.required],
            identity:['',Validators.required],
            email:['',[Validators.required,Validators.email]],

        });
    }

    get fval() { return this.registerForm.controls; }

    onFormSubmit(){
        this.submitted = true;
    }

```

```

// return for here if form is invalid
if (this.registerForm.invalid) {
    return;
}

this.loading = true;

    const result:User= Object.assign({}, this.registerForm.value);


// Do useful stuff with the gathered data
console.log(result.firstName);


    this.registerService.insertUser(result.firstName,
result.lastName,result.userName,result.password,result.dob,result.phone,
result.address, this.selectedOption,result.identity,result.email).subscribe(
    (data : any) =>{
        this.loading=false;
        localStorage.clear();
        localStorage.setItem('user',JSON.stringify(data));
        if(data.registrationStatus==true){
            Swal.fire(
                {
                    icon: 'success',
                    title: 'User registered succesfully!',
                    text:"Please wait for an email for account activation!"
                }
            )
        }else{
            Swal.fire({
                icon: 'error',
                title: 'Oops...',
                text: data.responseMessage,

```

```

        })
    }
    this.router.navigate(['/login']);
}
);

}

filterSelected(selectedOption){
    this.selectedOption= selectedOption;
    console.log('selected value= '+selectedOption);
}

}

```

## App>transaction-history

```

@media
only screen and (max-width: 1000px) {
    table, thead, tbody, th, td, tr,nav {
        display: block;
    }

    thead tr {
        position: absolute;
        top: -9999px;
        left: -9999px;
        text-align: center;
    }
}

```

```

tr { border: 1px solid #ccc; text-align: center;}
td {
    border: none;
    border-bottom: 1px solid #eee;
    position: relative;
    padding-left: 100px;
    margin-left: 150px;
    text-align: center;
}
td:before {
    position: absolute;
    top: 12px;
    left: 6px;
    width: 200px;
    padding-right: 40px;
    white-space: nowrap;
    margin-left: -150px;
    text-align: center;
}
td:nth-of-type(1):before { content: "Date"; }
td:nth-of-type(2):before { content: "Transaction Id"; }
td:nth-of-type(3):before { content: "Action Type"; }
td:nth-of-type(4):before { content: "Amount"; }
}

<meta name="viewport" content="width=device-width, initial-scale=1.0">
<nav class="navbar navbar-expand-sm navbar-dark bg-dark">
    <a class="navbar-brand" routerLink="/home">ICIN Bank</a>

    <button class="navbar-toggler" type="button" data-toggle="collapse" data-
target="#navbarsExample04" aria-controls="navbarsExample04" aria-
expanded="false" aria-label="Toggle navigation">

```



```

    <span class="navbar-toggler-icon"></span>
  </button>

  <div class="collapse navbar-collapse" id="navbarsExample04">
    <ul class="navbar-nav mr-auto">

      <li class="nav-item">
        <a class="nav-link" routerLink="/transactionHistory">Transaction
History<span class="sr-only"></span></a>
      </li>

      <li class="nav-item">
        <a class="nav-link" routerLink="/transferHistory">Transfer
History<span class="sr-only"></span></a>
      </li>

      <li class="nav-item">
        <a class="nav-link" routerLink="/transfer">Transfer Money<span
class="sr-only"></span></a>
      </li>

      <li class="nav-item">
        <a class="nav-link" routerLink="/chequebookRequest">Request
Checkbook<span class="sr-only"></span></a>
      </li>

      <li class="nav-item dropdown" >
        <a class="nav-link dropdown-toggle" href="#" id="dropdown04" data-
toggle="dropdown" aria-haspopup="true" aria-expanded="false">User Session</a>
        <div class="dropdown-menu" aria-labelledby="dropdown04">
          <a class="dropdown-item" routerLink="/editProfile" >Change Profile
Settings</a>

```

```

        <a class="dropdown-item" routerLink="/login">Logout</a>
      </div>
    </li>
  </ul>

</div>
</nav>
<br>
<h3 style="text-align: center;">Transaction History</h3>
<br>

<h4 style="padding-left: 1rem;">Type of account: Saving </h4><br>
<h4 style="padding-left: 1rem;">Saving balance : {{savingBalance}} </h4><br>
<h4 style="padding-left: 1rem;">Account number :{{accNo}} </h4><br>

<table class="table table-bordered">
  <thead>
    <tr>
      <th style="text-align: center;">Date</th>
      <th style="text-align: center;">Transaction Id</th>
      <th style="text-align: center;">Action Type</th>
      <th style="text-align: center;">Amount </th>
    </tr>
  </thead>
  <tbody>
    <tr *ngFor="let transaction of transactionList">
      <td style="text-align: center;">{{transaction.date}}</td>
      <td style="text-align: center;">{{transaction.id}}</td>
      <td style="text-align: center;">{{transaction.action}}</td>
      <td style="text-align: center;">{{transaction.amount}}</td>
    </tr>
  </tbody>
</table>

```

```

    </tr>

    </tbody>
</table>

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

import { TransactionHistoryComponent } from './transaction-history.component';

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

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

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

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

```

```
import { Component, OnInit } from '@angular/core';
import { TransactionService } from '../transaction.service';
import { Transaction } from '../_models/transaction';

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

  username:String=localStorage.getItem("username");
  accNo:number=+localStorage.getItem("savingAccNo");
  public transactionList:Array<Transaction>;
  public savingBalance:number;

  constructor(private transactionService:TransactionService) {}

  ngOnInit(): void {
    this.transactionService.getTransactions(this.accNo).subscribe(res=>{
      this.transactionList = res;
      console.log(this.transactionList);
    });
    this.transactionService.getSavingAccount(this.username).subscribe(res=>{
      this.savingBalance = res.balance;
    });
  }
}
```

## App>transaction-between-accounts

```
.container{
    margin: 10rem auto;
    width:30rem;
}

.input{
    margin-bottom: 2rem ;
    margin-top: 2rem;
}

<meta name="viewport" content="width=device-width, initial-scale=1.0">
<nav class="navbar navbar-expand-sm navbar-dark bg-dark">
    <a class="navbar-brand" routerLink="/home">ICIN Bank</a>
    <button class="navbar-toggler" type="button" data-toggle="collapse" data-
target="#navbarsExample04" aria-controls="navbarsExample04" aria-
expanded="false" aria-label="Toggle navigation">
        <span class="navbar-toggler-icon"></span>
    </button>

    <div class="collapse navbar-collapse" id="navbarsExample04">
        <ul class="navbar-nav mr-auto">

            <li class="nav-item">
                <a class="nav-link" routerLink="/transactionHistory">Transaction
History<span class="sr-only"></span></a>
            </li>

            <li class="nav-item">
```

```

        <a class="nav-link" routerLink="/transferHistory">Transfer
History<span class="sr-only"></span></a>

    </li>

    <li class="nav-item">

        <a class="nav-link" routerLink="/transfer">Transfer Money<span
class="sr-only"></span></a>

    </li>

    <li class="nav-item">

        <a class="nav-link" routerLink="/chequebookRequest">Request
Checkbook<span class="sr-only"></span></a>

    </li>

    <li class="nav-item dropdown" >

        <a class="nav-link dropdown-toggle" href="#" id="dropdown04" data-
toggle="dropdown" aria-haspopup="true" aria-expanded="false">User Session</a>

        <div class="dropdown-menu" aria-labelledby="dropdown04">

            <a class="dropdown-item" routerLink="/editProfile" >Change Profile
Settings</a>

            <a class="dropdown-item" routerLink="/login">Logout</a>

        </div>

    </li>
</ul>

</div>
</nav>

<div class="container" >

    <form [formGroup]="transferForm" (ngSubmit)="transfer()"
method="post">

        <div class="col-xs-4" style="left:35%;">

            <h3 style="text-align: center;">Transfer Money</h3>

        </div>

```



```

        <label>Enter Amount:</label>

        <br>

        <input type="text" formControlName="amount" class="form-
control" [ngClass]="{ 'is-invalid': submitted && fval.amount.errors }"
placeholder="Enter amount"/>

        <div *ngIf="submitted && fval.amount.errors" class="invalid-
feedback">

            <div *ngIf="fval.amount.errors.required">Amount is
required</div>

        </div>
    </div>

    <div class="center1">

        <button [disabled]="loading" class="btn btn-primary" class="btn
btn-info form-control">

            <span *ngIf="loading" class="spinner-border spinner-border-sm
mr-1"></span>

            <a (click)="transfer()" style="color:white;">Transfer</a>

        </button>

    </div>

</form>

</div>

```

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

```
import { TransferBetweenAccountsComponent } from './transfer-between-
accounts.component';
```

```
describe('TransferBetweenAccountsComponent', () => {
```

```
    let component: TransferBetweenAccountsComponent;
```

```
    let fixture: ComponentFixture<TransferBetweenAccountsComponent>;
```

```
    beforeEach(async(() => {
```



```

    TestBed.configureTestingModule({
      declarations: [ TransferBetweenAccountsComponent ]
    })
    .compileComponents();
  }));

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

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

import { Component, OnInit } from '@angular/core';
import { FormBuilder, FormGroup, Validators, } from '@angular/forms';
import { Router } from '@angular/router';
import { TransferService } from '../transfer.service';
import Swal from 'sweetalert2';

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

  constructor( private formBuilder: FormBuilder,

```

```

    private router: Router,
    private transferService: TransferService) { }

transferForm: FormGroup;

loading = false;

submitted = false;

ngOnInit() {

    var username:String=localStorage.getItem("username");
    var accNo=+localStorage.getItem("savingAccNo");
    console.log(accNo)
    console.log(username)
    this.transferForm = this.formBuilder.group({
        username : username,
        saccountNo: accNo,
        ifscNo: ['', [Validators.required, Validators.minLength(8)]],
        raccountNo: ['', [Validators.required]],
        amount:['',[Validators.required]]

    });

}

get saccountno(): any {
    return localStorage.getItem('savingAccNo');
}

get fval() { return this.transferForm.controls; }

transfer(){
    this.submitted = true;

    if (this.transferForm.invalid) {
        return;
    }
}

```

```
}

this.loading = true;

const result:any = Object.assign({}, this.transferForm.value);


// Do useful stuff with the gathered data

try{
    this.transferService.insertEntry(result.username,result.saccountNo,result.
t.ifscNo,result.raccountNo,result.amount).subscribe(
    (data : any) =>{
        this.loading=false;
        if(data.transferStatus==true){
            Swal.fire({
                icon: 'success',
                title: 'Transaction successful',
                text:data.responseMessage
            })
        }
        else{
            Swal.fire({
                icon: 'error',
                title: 'Oops...',
                text: data.responseMessage,
            })
        }
    }
);
}catch{
    this.loading=false;
}
```

```
}  
}
```

## App>transfer-history

```
@media  
only screen and (max-width: 1000px) {  
  table, thead, tbody, th, td, tr, nav {  
    display: block;  
  }  
  
  thead tr {  
    position: absolute;  
    top: -9999px;  
    left: -9999px;  
  }  
  
  tr { border: 1px solid #ccc; }  
  td {
```

```

border: none;
border-bottom: 1px solid #eee;
position: relative;
padding-left: 100px;
margin-left: 150px;
}
td:before {
  position: absolute;
  top: 12px;
  left: 6px;
  width: 200px;
  padding-right: 40px;
  white-space: nowrap;
  margin-left: -150px;
}
td:nth-of-type(1):before { content: "Date"; }
td:nth-of-type(2):before { content: "Transaction Id"; }
td:nth-of-type(3):before { content: "Amount"; }
td:nth-of-type(4):before { content: "Sender Account Number"; }
td:nth-of-type(5):before { content: "Receiver Account Number"; }
}

```

```

<nav class="navbar navbar-expand-sm navbar-dark bg-dark">
  <a class="navbar-brand" routerLink="/home">ICIN Bank</a>
  <button class="navbar-toggler" type="button" data-toggle="collapse" data-
target="#navbarsExample04" aria-controls="navbarsExample04" aria-
expanded="false" aria-label="Toggle navigation">
    <span class="navbar-toggler-icon"></span>

```

```

</button>

<div class="collapse navbar-collapse" id="navbarsExample04">
  <ul class="navbar-nav mr-auto">

    <li class="nav-item">
      <a class="nav-link" routerLink="/transactionHistory">Transaction
History<span class="sr-only"></span></a>
    </li>

    <li class="nav-item">
      <a class="nav-link" routerLink="/transferHistory">Transfer
History<span class="sr-only"></span></a>
    </li>

    <li class="nav-item">
      <a class="nav-link" routerLink="/transfer">Transfer Money<span
class="sr-only"></span></a>
    </li>

    <li class="nav-item">
      <a class="nav-link" routerLink="/chequebookRequest">Request
Checkbook<span class="sr-only"></span></a>
    </li>

    <li class="nav-item dropdown" >
      <a class="nav-link dropdown-toggle" href="#" id="dropdown04" data-
toggle="dropdown" aria-haspopup="true" aria-expanded="false">User Session</a>
      <div class="dropdown-menu" aria-labelledby="dropdown04">
        <a class="dropdown-item" routerLink="/editProfile" >Change Profile
Settings</a>
        <a class="dropdown-item" routerLink="/login">Logout</a>

```



```

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

import { TransferHistoryComponent } from '../transfer-history.component';

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

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

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

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

import { Component, OnInit } from '@angular/core';
import { TransferHistory } from '../_models/transferhistory'
import { TransferHistoryService } from '../transferhistory.service';

@Component({

```



```

    selector: 'app-transfer-history',
    templateUrl: './transfer-history.component.html',
    styleUrls: ['./transfer-history.component.css']
  })
}

export class TransferHistoryComponent implements OnInit {

  private accNo:number =+localStorage.getItem("savingAccNo");
  public transferList:Array<TransferHistory>;

  constructor(private transferService:TransferhistoryService) { }

  ngOnInit(): void {
    this.transferService.getTransferHistory(this.accNo).subscribe(res=>{
      this.transferList=res;
    });
  }
}

```

## Src>app>app-routing.modules.ts

```

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

/**Componenets */
import { LoginComponent } from './login/login.component';
import { RegisterComponent } from './register/register.component';
import { HomeComponent } from './home/home.component';
import { ChequeBookRequestComponent } from './cheque-book-request/cheque-book-request.component';
import { TransferBetweenAccountsComponent } from './transfer-between-accounts/transfer-between-accounts.component';

```

```

import { TransactionHistoryComponent } from './transaction-
history/transaction-history.component';

import { EditProfileComponent } from './edit-profile/edit-profile.component';

import {AuthGuard} from './auth.guard';

import { TransferHistoryComponent } from './transfer-history/transfer-
history.component';

const routes: Routes = [
  {path: 'login', component: LoginComponent},
  {path: 'register', component: RegisterComponent},
  {path: 'transfer', component: TransferBetweenAccountsComponent, canActivate: [AuthGuard]},
  {path: 'transactionHistory', component: TransactionHistoryComponent, canActivate: [AuthGuard]},
  {path: 'chequebookRequest', component: ChequeBookRequestComponent, canActivate: [AuthGuard]},
  {path: 'home', component: HomeComponent, canActivate: [AuthGuard]},
  {path: 'transferHistory', component: TransferHistoryComponent, canActivate: [AuthGuard]},
  {path: 'editProfile', component: EditProfileComponent, canActivate: [AuthGuard]},
  { path: '**', redirectTo: '/login' }
];

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

```

**src>app>app-components.ts**

```

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

```

```

/**Componenets */
import { LoginComponent } from './login/login.component';
import { RegisterComponent } from './register/register.component';
import { HomeComponent } from './home/home.component';
import { ChequeBookRequestComponent } from './cheque-book-request/cheque-book-request.component';
import { TransferBetweenAccountsComponent } from './transfer-between-accounts/transfer-between-accounts.component';
import { TransactionHistoryComponent } from './transaction-history/transaction-history.component';
import { EditProfileComponent } from './edit-profile/edit-profile.component';
import {AuthGuard} from './auth.guard';
import { TransferHistoryComponent } from './transfer-history/transfer-history.component';

const routes: Routes = [
  {path: 'login', component: LoginComponent},
  {path: 'register', component: RegisterComponent},
  {path: 'transfer', component: TransferBetweenAccountsComponent, canActivate: [AuthGuard]},
  {path: 'transactionHistory', component: TransactionHistoryComponent, canActivate: [AuthGuard]},
  {path: 'chequebookRequest', component: ChequeBookRequestComponent, canActivate: [AuthGuard]},
  {path: 'home', component: HomeComponent, canActivate: [AuthGuard]},
  {path: 'transferHistory', component: TransferHistoryComponent, canActivate: [AuthGuard]},
  {path: 'editProfile', component: EditProfileComponent, canActivate: [AuthGuard]},
  { path: '**', redirectTo: '/login' }
];

@NgModule({

```

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

## src>app>app-component.html

```
<router-outlet></router-outlet>  
  
import { TestBed, async } from '@angular/core/testing';  
import { RouterTestingModule } from '@angular/router/testing';  
import { AppComponent } from './app.component';  
  
describe('AppComponent', () => {  
  beforeEach(async(() => {  
    TestBed.configureTestingModule({  
      imports: [  
        RouterTestingModule  
      ],  
      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 'angular-user-registration-and-login-example`, ()
=> {
        const fixture = TestBed.createComponent(AppComponent);
        const app = fixture.debugElement.componentInstance;
        expect(app.title).toEqual('angular-user-registration-and-login-example');
    });

    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
angular-user-registration-and-login-example!');
    });
});

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

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

  constructor(public authService:AuthService,private router: Router) {

  }

```

```

}

import { BrowserModule } from '@angular/platform-browser';
import { NgModule } from '@angular/core';
import { ReactiveFormsModule } from '@angular/forms';
import { HttpClientModule } from '@angular/common/http';
import { BrowserModuleAnimationsModule } from '@angular/platform-browser/animations';
import { AuthService } from '../auth.service';
import { LoginService } from '../login.service';
import { RegisterService } from '../register.service';
import { AppRoutingModuleModule } from '../app-routing.module';
import { AppComponent } from '../app.component';
import { LoginComponent } from '../login/login.component';
import { RegisterComponent } from '../register/register.component';
import { HomeComponent } from '../home/home.component';
import { ChequeBookRequestComponent } from '../cheque-book-request/cheque-book-request.component';
import { TransactionHistoryComponent } from '../transaction-history/transaction-history.component';
import { TransferBetweenAccountsComponent } from '../transfer-between-accounts/transfer-between-accounts.component';
import { EditProfileComponent } from '../edit-profile/edit-profile.component';
import { TransferHistoryComponent } from '../transfer-history/transfer-history.component';

@NgModule({
  declarations: [
    AppComponent,
    LoginComponent,
    RegisterComponent,
    HomeComponent,
    ChequeBookRequestComponent,

```

```

        TransactionHistoryComponent,
        TransferBetweenAccountsComponent,
        EditProfileComponent,
        TransferHistoryComponent,

    ],
    imports: [
        BrowserModule,
        AppRoutingModule,
        ReactiveFormsModule,
        HttpClientModule,
        BrowserAnimationsModule, // required animations module
    ],
    providers: [RegisterService, LoginService, AuthService], //
    bootstrap: [AppComponent]
  })
}

export class AppModule { }

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

import { AuthGuard } from './auth.guard';

describe('AuthGuard', () => {
  let guard: AuthGuard;

  beforeEach(() => {
    TestBed.configureTestingModule({});
    guard = TestBed.inject(AuthGuard);
  });

  it('should be created', () => {

```

```

        expect(guard).toBeTruthy();
    });
});

import { Injectable } from '@angular/core';
import { CanActivate, ActivatedRouteSnapshot, RouterStateSnapshot, UrlTree, Router } from '@angular/router';
import { Observable } from 'rxjs';
import { AuthService } from '../auth.service';

@Injectable({
    providedIn: 'root'
})
export class AuthGuard implements CanActivate {
    constructor(private authService: AuthService, private router: Router){

    }

    canActivate(
        next: ActivatedRouteSnapshot,
        state: RouterStateSnapshot): Observable<boolean | UrlTree> | Promise<boolean | UrlTree> | boolean | UrlTree {
        if(this.authService.isAuthenticated){
            return true;
        }
        this.router.navigate(['']);
        return false;
    }
}

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

import { AuthService } from '../auth.service';

```



```
describe('AuthService', () => {  
  let service: AuthService;  
  
  beforeEach(() => {  
    TestBed.configureTestingModule({});  
    service = TestBed.inject(AuthService);  
  });  
  
  it('should be created', () => {  
    expect(service).toBeTruthy();  
  });  
});  
  
import { Injectable } from '@angular/core';  
import { Login } from '../_models';  
import { Router } from '@angular/router';  
  
@Injectable({  
  providedIn: 'root'  
})  
export class AuthService {  
  isAuthenticated=false;  
  
  constructor(private router: Router) { }  
  
  authenticate(isLoginError: boolean):boolean{  
    if(isLoginError == false){  
      this.isAuthenticated=true;  
      return true;  
    }  
  }  
}
```

```

        else{
            this.isAuthenticated = false;
            return false;
        }
    }
}
}

import { Injectable } from '@angular/core';
import { Login } from '../_models';
import { Router } from '@angular/router';

@Injectable({
    providedIn: 'root'
})
export class AuthService {
    isAuthenticated=false;

    constructor(private router: Router) { }

    authenticate(isLoginError: boolean):boolean{
        if(isLoginError == false){
            this.isAuthenticated=true;
            return true;
        }
        else{
            this.isAuthenticated = false;
            return false;
        }
    }
}
}

```

```

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

import { LoginService } from '../login.service';

describe('LoginService', () => {
  let service: LoginService;

  beforeEach(() => {
    TestBed.configureTestingModule({});
    service = TestBed.inject(LoginService);
  });

  it('should be created', () => {
    expect(service).toBeTruthy();
  });
});

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

@Injectable({
  providedIn: 'root'
})
export class LoginService {
  readonly rootUrl = 'http://localhost:8080';

  constructor(private http: HttpClient) { }

  loginUser(userName: string, password: string) {
    var body = {
      username: userName,

```

```
        password: password
    }

    return this.http.post(this.rootUrl + '/login', body);
}
}
```

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

```
import { RegisterService } from './register.service';
```

```
describe('RegisterService', () => {
```

```
    let service: RegisterService;
```

```
    beforeEach(() => {
```

```
        TestBed.configureTestingModule({});
```

```
        service = TestBed.inject(RegisterService);
```

```
    });
```

```
    it('should be created', () => {
```

```
        expect(service).toBeTruthy();
```

```
    });
```

```
});
```

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

```
import { RequestService } from './request.service';
```

```
describe('RequestService', () => {
```

```
    let service: RequestService;
```

```
    beforeEach(() => {
```

```

    TestBed.configureTestingModule({});
    service = TestBed.inject(RequestService);
  });

  it('should be created', () => {
    expect(service).toBeTruthy();
  });
});

import { Injectable } from '@angular/core';
import { HttpClient } from '@angular/common/http';
import {ChequebookResponse} from '../_models/chequebookresponse'

@Injectable({
  providedIn: 'root'
})
export class RequestService {

  readonly rootUrl = 'http://localhost:8080';

  constructor(private http: HttpClient) { }

  insertRequest(accNo: number,pages:number=20) {
    var body = {
      account: accNo,
      no_of_pages: pages,
    }
    console.log(body);
    return this.http.post<ChequebookResponse>(this.rootUrl +
'/cheque/request', body);
  }
}

```

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

import { TransactionService } from '../transaction.service';

describe('TransactionService', () => {
  let service: TransactionService;

  beforeEach(() => {
    TestBed.configureTestingModule({});
    service = TestBed.inject(TransactionService);
  });

  it('should be created', () => {
    expect(service).toBeTruthy();
  });
});

import { Injectable } from '@angular/core';
import { HttpClient } from '@angular/common/http';
import { Transaction } from '../_models/transaction'
import { SavingAccount } from '../_models/savingaccount'
import { Observable } from 'rxjs';

@Injectable({
  providedIn: 'root'
})
export class TransactionService {

  private url:String;
```

```

constructor(private http:HttpClient) {
    this.url="http://localhost:8080"
}

public getTransactions(accNo):Observable<Transaction[]>{
    return
this.http.get<Transaction[]>(this.url+"/account/getHistory/"+accNo);
}

public getSavingAccount(username):Observable<SavingAccount>{
    return
this.http.get<SavingAccount>(this.url+"/account/getsaving/"+username);
}
}

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

import { TransferService } from './transfer.service';

describe('TransferService', () => {
    let service: TransferService;

    beforeEach(() => {
        TestBed.configureTestingModule({});
        service = TestBed.inject(TransferService);
    });

    it('should be created', () => {
        expect(service).toBeTruthy();
    });
});

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

```

```

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

@Injectable({
  providedIn: 'root'
})
export class TransferService {

  readonly rootUrl = 'http://localhost:8080';

  constructor(private http: HttpClient) { }

  insertEntry(username:string,
saccount:string,ifscNo:string,raccount:string,amount:number) {
    var body = {
      username:username,
      saccount: saccount,
      ifsc: ifscNo,
      raccount:raccount,
      amount:amount
    }
    console.log(body);
    return this.http.post(this.rootUrl + '/account/transfer', body);
  }
}

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

import { TransferhistoryService } from './transferhistory.service';

describe('TransferhistoryService', () => {
  let service: TransferhistoryService;

```



```

beforeEach(() => {
    TestBed.configureTestingModule({});
    service = TestBed.inject(TransferhistoryService);
});

it('should be created', () => {
    expect(service).toBeTruthy();
});
});

import { Injectable } from '@angular/core';
import { TransferHistory } from '../_models/transferhistory';
import { HttpClient } from '@angular/common/http';
import { Observable } from 'rxjs';

@Injectable({
    providedIn: 'root'
})
export class TransferhistoryService {

    private url:String;

    constructor(private http:HttpClient) {
        this.url="http://localhost:8080"
    }

    public getTransferHistory(accNo):Observable<TransferHistory[]>{
        return
this.http.get<TransferHistory[]>(this.url+"/account/getTransfers/"+accNo);
    }

    // public getSavingAccount(username):Observable<SavingAccount>{
    //     return
this.http.get<SavingAccount>(this.url+"/account/getsaving/"+username);

```

```

    // }
}

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

import { UpdateService } from '../update.service';

describe('UpdateService', () => {
  let service: UpdateService;

  beforeEach(() => {
    TestBed.configureTestingModule({});
    service = TestBed.inject(UpdateService);
  });

  it('should be created', () => {
    expect(service).toBeTruthy();
  });
});

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

@Injectable({
  providedIn: 'root'
})
export class UpdateService {

  readonly rootUrl = 'http://localhost:8080';

  constructor(private http: HttpClient) { }
}

```

```

    update(username:string,phone: number,email: string,address:
string,prevpassword:string,newpassword:string) {
        var body = {
            username:username,
            phone : phone,
            email: email,
            address : address,
            password: prevpassword,
            newpassword:newpassword
        }
        console.log(body);
        return this.http.put(this.rootUrl + '/profile/update', body);
    }
}

```

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

```
import { UserService } from './user.service';
```

```

describe('UserService', () => {
    beforeEach(() => TestBed.configureTestingModule({}));

    it('should be created', () => {
        const service: UserService = TestBed.get(UserService);
        expect(service).toBeTruthy();
    });
});

```

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

```
import { UserDisplay } from './_models/userdisplay';
```

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

```

import { Observable } from 'rxjs';

@Injectable({
  providedIn: 'root'
})
export class UserService {

  private url:string;

  constructor(private http:HttpClient) {
    this.url="http://localhost:8080"
  }

  public getUser(username):Observable<UserDisplay>{
    return this.http.get<UserDisplay>(this.url+"/home/"+username);
  }
}

import { Userdisplay } from './userdisplay';

describe('Userdisplay', () => {
  it('should create an instance', () => {
    expect(new Userdisplay()).toBeTruthy();
  });
});

```

## Environments

```

export const environment = {
  production: true
};

```

```
// This file can be replaced during build by using the `fileReplacements`
array.

// `ng build --prod` replaces `environment.ts` with `environment.prod.ts`.
// The list of file replacements can be found in `angular.json`.

export const environment = {
  production: false,
  apiUrl : 'http://localhost:4200/'
};

/*
 * 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/dist/zone-error'; // Included with Angular CLI.

<!doctype html>
<html lang="en" class="full-height">
<head>
  <meta charset="utf-8">
  <title>DreamApp</title>
  <base href="/">
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <link rel="icon" type="image/x-icon" href="favicon.ico">
  <script
src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.16.0/umd/popper.min.js
"></script>
</head>
```

```

<body>
  <app-root></app-root>
</body>
</html>

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));

/**
 * 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 Safari >= 10, Chrome >= 55
(including Opera),

* Edge >= 13 on the desktop, and iOS 10 and Chrome on mobile.

*

* Learn more in https://angular.io/guide/browser-support

*/

/*****
*****

* BROWSER POLYFILLS

*/

/** IE10 and IE11 requires the following for NgClass support on SVG elements
*/

// import 'classlist.js'; // Run `npm install --save classlist.js`.

/**

* Web Animations `@angular/platform-browser/animations`

* Only required if AnimationBuilder is used within the application and using
IE/Edge or Safari.

* Standard animation support in Angular DOES NOT require any polyfills (as of
Angular 6.0).

*/

// import 'web-animations-js'; // Run `npm install --save web-animations-js`.

/**

* 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 requestAnimationFrame
* (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 specified eventNames
*
* 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/dist/zone'; // Included with Angular CLI.

/*****
*****

* APPLICATION IMPORTS
*/

```



