FPT Education

FPT POLYTECHNIC

cdio™

Conceive Design Implement Operate

THỰC HỌC – THỰC NGHIỆP

LẬP TRÌNH FRONT-END FRAMEWORK 1

AUTHENTICATION TRONG ANGULAR

http://www.poly.edu.vn

Mục tiêu

- Hiểu và cài đặt Authencation trong angular
- Signup, Login và xử lý lỗi Login
- Lưu trữ User, thêm Token, cài đặt login, logout tự động

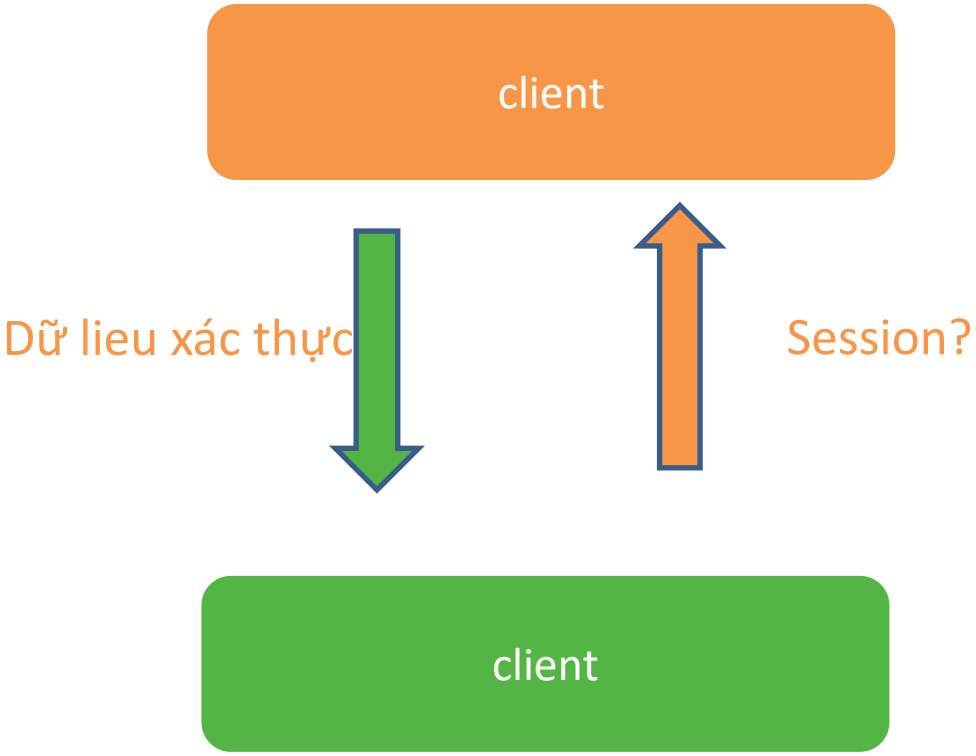
NỘI DUNG

- Authentication hoạt động như thế nào
- Cài đặt trang đăng nhập, đăng ký
- Lưu trữ thông tin đăng nhập
- Sử dụng token
- Bảo vệ route với canActivate

Phần 1: Authentication

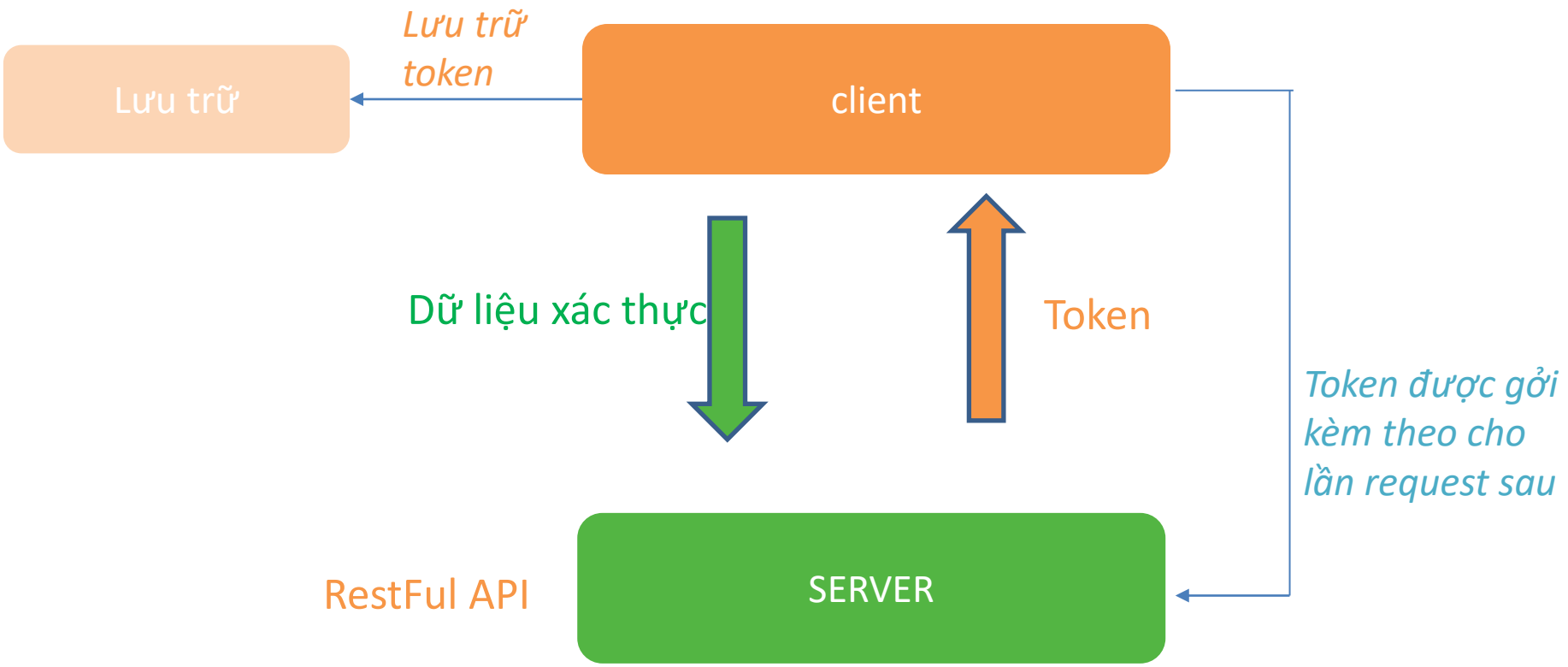
ATHENTICATION TRONG ANGULAR

Cơ chế authencation thông thường, với mô hình client server



ATHENTICATION TRONG ANGULAR

Authencation trong Angular hoạt động như thế nào



ATHENTICATION TRONG ANGULAR

Thiết kế form để xác thực tại client

Tạo component authen và thiết kế giao diện như sau:

```
<form>
<div class="form-group">
<label for="email">E-Mail</label>
<input type="email" id="email" class="form-control" />
</div>
<div class="form-group">
<label for="password">Password</label>
<input type="password" id="password" class="form-control" />
</div>
<div>
<button class="btn btn-primary">Đăng Nhập</button> |
<button class="btn btn-primary">Đăng Kts</button>
</div>
</form>
```

ATHENTICATION TRONG ANGULAR

Thiết kế form để xác thực tại client

Điều khiển form:

```
<form #authForm="ngForm" (ngSubmit)="onSubmit(authForm)">
<div class="form-group">
<label for="email">E-Mail</label>
<input
type="email"
id="email"
class="form-control"
ngModel
name="email"
required
email
/>
</div>
```

```
<div class="form-group">
<label for="password">Password</label>
<input
type="password" id="password"
class="form-control" ngModel name="password"
Required minlength="5"/>
</div>
<div>
<button
class="btn btn-primary"
type="submit" [disabled]="!authForm.valid">Đăng nhập</button>|
<button class="btn btn-primary" type="button">Đăng ký</button>
</div>
```

ATHENTICATION TRONG ANGULAR

Xử lý authencation trong angular

Tạo service phục vụ cho authen

```
export class AuthserviceService {
  url='http://127.0.0.1:3000/auth/';
  constructor(private http:HttpClient) {}
  signUp(email:string,pass:string){
    return this.http.post(this.url+'register',
      {
        "email": email,
        "password": pass,
        "typeUser":0
      }
    );
  }
  signIn(email:string,pass:string){
  }
```

ATHENTICATION TRONG ANGULAR

Xử lý authencation trong angular

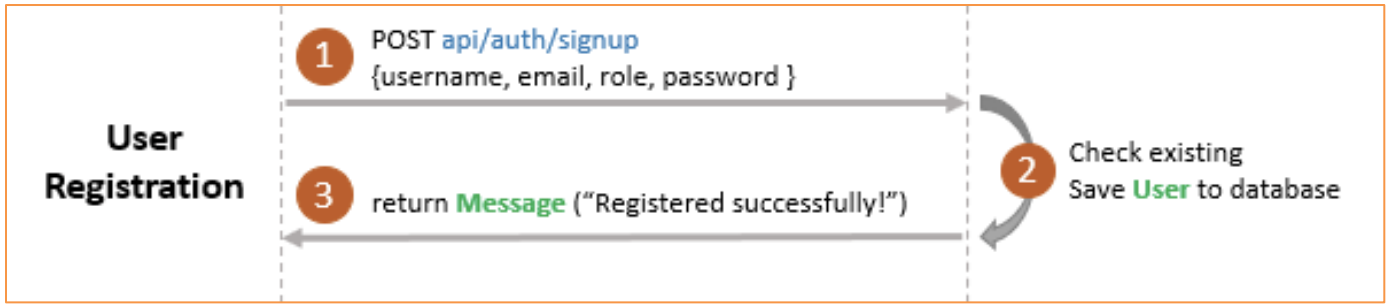
Gởi yêu cầu đăng ký và xử lý lỗi

```
export class AuthserviceService {
  url='http://127.0.0.1:3000/auth/';
  constructor(private http:HttpClient) {}
  signUp(email:string,pass:string){
    return this.http.post(this.url+'register',
      {
        "email": email,
        "password": pass,
        "typeUser":0
      }
    );
  }
  signIn(email:string,
  }
```

ATHENTICATION TRONG ANGULAR

Xử lý authencation trong angular

Gởi yêu cầu đăng ký và xử lý lỗi

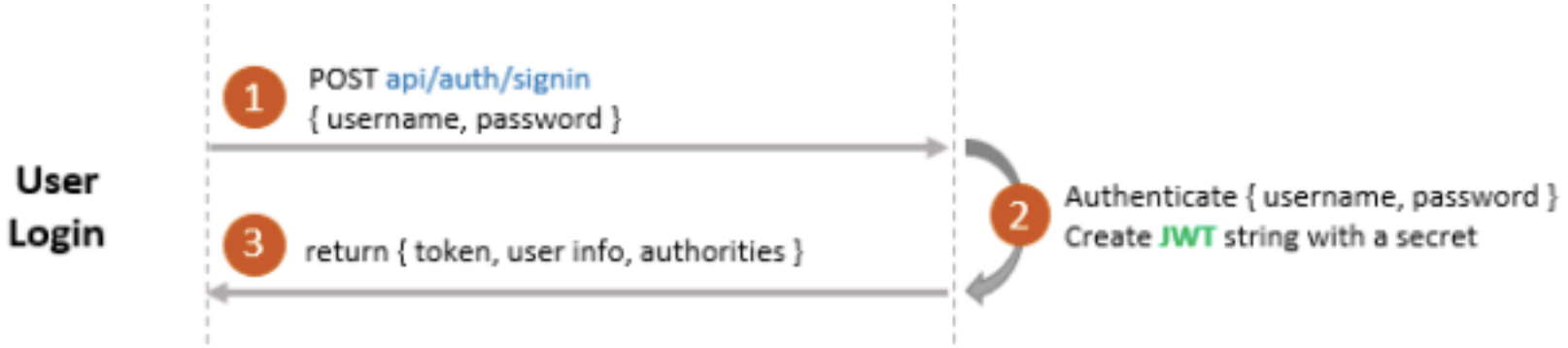



```
onSubmit(form: NgForm) {
let email=form.value.email;
let password=form.value.password;
this.authService.signUp(email,password).subscribe(res=>{
console.log(res);
},
error=>{
console.log(error);
}
)
form.reset();
}
```

ATHENTICATION TRONG ANGULAR

Xử lý authencation trong angular

Gởi yêu cầu đăng nhập và xử lý lỗi



ATHENTICATION TRONG ANGULAR

Xử lý authencation trong angular

Gởi yêu cầu đăng nhập và xử lý lỗi

```
onSubmit(form: NgForm) {
  let email=form.value.email;
  let password=form.value.password;
  this.error=null;
//Dang nhap
  this.authService.signIn(email,password).subscribe(res=> {
    console.log(res);
  },
  error=>{
  console.log(error);
  this.error = error.error.message;
 }
)
```

Xử lý authencation trong angular

Lưu trữ dữ liệu sau khi login, trong đó thường có thông tin tocken để xác thực

```
onSubmit(form: NgForm) {
let email=form.value.email;
let password=form.value.password;
//Dang nhap
this.authService.signIn(email,password).subscribe(res=>
{
  this.authService.setTocken(res.accessToken);
},error=>{
    this.error = error.error.message;
  })
}
```

ATHENTICATION TRONG ANGULAR

Xử lý authencation trong angular

Tự động đăng nhập – auto login

Bước 1: thực hiện lưu trữ thông tin đăng nhập (token) vào localstorage

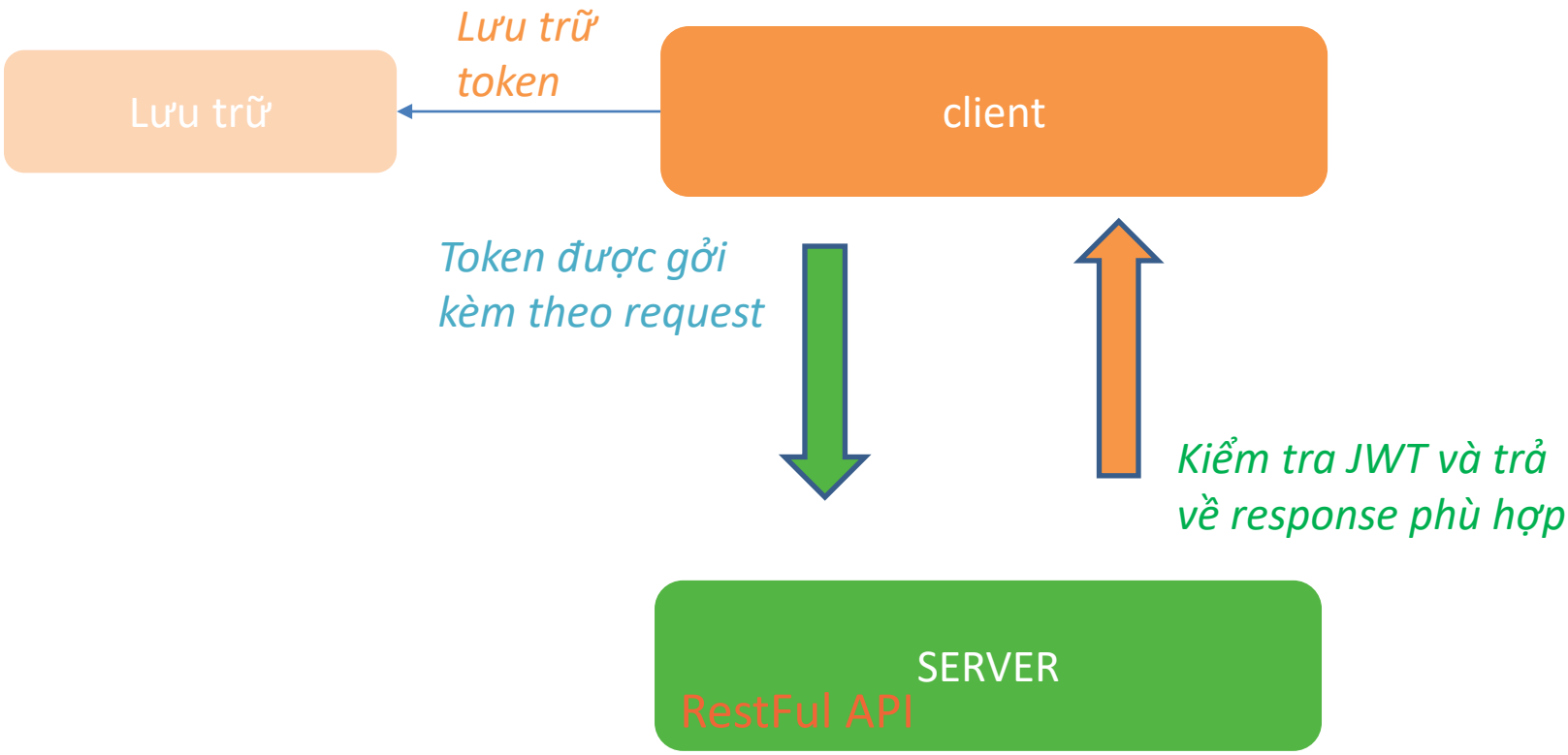
Bước 2: thực hiện có auto login trong auth service

```
autoLogin() {
  const userData: {tocken: string}=JSON.parse(localStorage.getItem('userData'));
 if (!userData) {
    return;
 }
 if (jwtHelper.isTokenExpired(userData.tocken)) {
     this.authService.setTocken(userData.tocken);
 }
}
```

THỰC HIỆN ĐĂNG KÝ VÀ ĐĂNG NHẬP
ĐĂNG NHẬP VÀ ĐĂNG XUẤT TỰ ĐỘNG

Phần 2: Token và CanActive guard

TOCKEN


Lưu trữ
Lưu trữ token
client
Token được gởi kèm theo request
Kiểm tra JWT và trả về response phù hợp
SERVER
RestFul API

Gởi request kèm token

Thêm option header với name trùng với name lấy token phái API

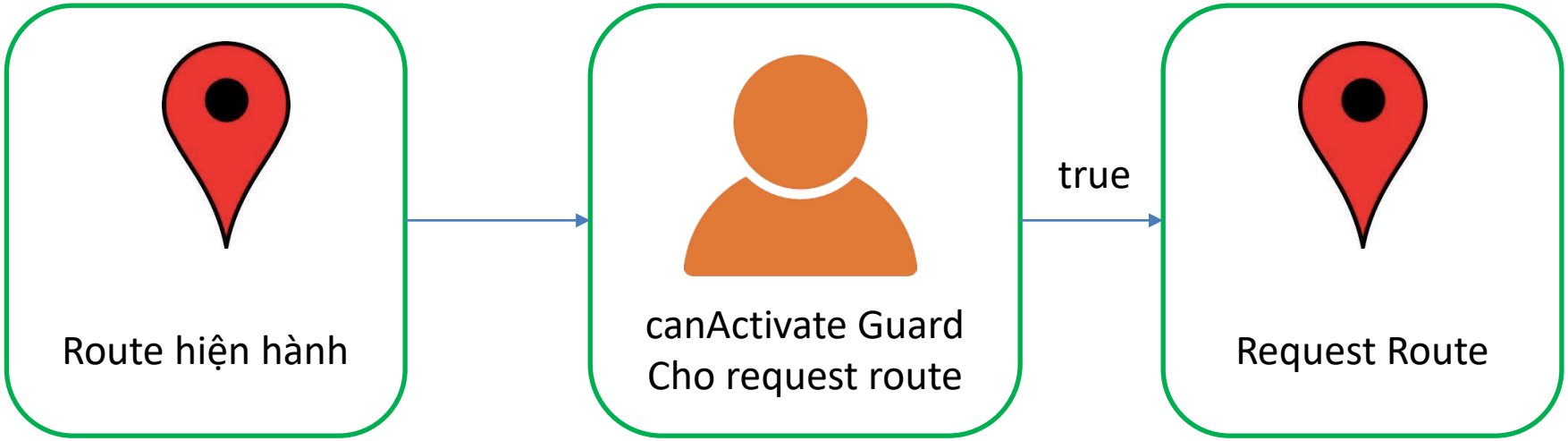
Ví dụ: name lấy tocken xác thực phía server là : x-access-token

```
getAllPost(){
console.log(this.authService.getTocken())
return this.httpService.get(this.url,{headers: new
HttpHeaders()
.set('x-access-token', this.authService.getTocken())
});
}
```

Guard là gì?

- Là dịch vụ (service) dung để xác định user hiện hành có thể điều hướng tới một route nào đó hay là không
- Guard tự động thực thi trước khi đi vào route

Luồng xử lý



Người dùng gởi request
Ví dụ: /post

Kiểm tra xem user có thể có quyền truy cập tới request route hay không, và trả về true hoặc false

Thực hiện

- Quyết định định tuyến dựa trên thời gian hết hạn của token
  - Khi token hết hạn, đây là dấu hiệu cho thấy người dùng đang không được xác thực và không được phép truy cập.
  - Lớp JwtHelperService ở trong angular2-jwt có thể được sử dụng cho việc này.

Thực hiện

- Quyết định định tuyến dựa trên thời gian hết hạn của token

```
npm install --save @auth0/angular-jwt
```

  - Sử dụng angular-jwt ở trong AuthService:

```
export class AuthserviceService {

  tocken:string=null;

  constructor(private http:HttpClient) {}

  public isAuthenticated(): boolean {
    const jwtHelper = new JwtHelperService();
    if(this.tocken===null)
      return false;
    return !jwtHelper.isTokenExpired(this.tocken);
  }
```

Thực hiện

- Quyết định định tuyến dựa trên thời gian hết hạn của token
  - Tạo một service để cài đặt route guard, chẳng hạn như auth-guard.service

```
import { Injectable } from '@angular/core';
import { Router, CanActivate } from '@angular/router';
import { AuthserviceService } from './authservice.service';
@Injectable()
export class AuthGuardService implements CanActivate {
constructor(public auth: AuthserviceService, public router: Router)
canActivate(): boolean {
  if (!this.auth.isAuthenticated()) {
    this.router.navigate(['auth']);
    return false;
  }
  return true;
}
```

Thực hiện

- Quyết định định tuyến dựa trên thời gian hết hạn của token
  - Sử dụng route guard cho các route của người dùng

```
import { AuthGuardService as AuthGuard } from './auth/auth-guard.service';

const routes: Routes = [
{path:'',component:HomeComponent},
{path:'home',component:HomeComponent},
{path:'blogs',component:BlogComponent,canActivate: [AuthGuard] },
```

Thực hiện

- Kiểm tra cho quyền của User
  - Để cấp quyền truy cập với vai trò được đính kèm trong thông tin của user.
  - Để thực hiện, chúng ta có thể sửa đổi guard để tìm kiếm vai trò của user đó ở trong payload của JWT.

Thực hiện

- Kiểm tra cho quyền của User
  - Để đọc được JWT, ta sẽ sử dụng jwt-decode:

```
npm install --save jwt-decode
```

  - Tạo một service để cài đặt role guard, chẳng hạn như roleGuard.service

```
npm g service auth/ roleGuard
```

Thực hiện

- Kiểm tra cho quyền của User
  - Cài đặt roleGuard.service để thực hiện phân quyền

```
canActivate(route: ActivatedRouteSnapshot): boolean {
//lay role duoc truong tai route
  const role = route.data.role;
  const token = this.auth.getTocken();
  // decode the token to get its payload
  var tokenPayload:any;
  if(token){
    tokenPayload = decode(token);
  }
  if (!this.auth.isAuthenticated() || tokenPayload.typeUser !== role) {
    this.router.navigate(['auth']);
    return false;
  }
  return true;
}
```

Thực hiện

- Kiểm tra cho quyền của User
  - Sử dụng roleGuard Service này cho route nào chúng tôi có thể muốn bảo vệ. Ví dụ như route /admin:

```
{
path:'admin', component:DashboardComponent,
 canActivate: [RoleGuard],
 data: {
  role: 1
 }
}
```

GUARD BẢO VỆ ROUTE, PHÂN QUYỀN

SUMARRY

- ☑ Authentication hoạt động như thế nào
- ☑ Cài đặt trang đăng nhập, đăng ký
- ☑ Lưu trữ thông tin đăng nhập - token
- ☑ Sử dụng token
- ☑ Bảo vệ route với canActivate


thank
you!