//æøå

import { Cookie } from "./cookie";

import { NavMenu } from '../components/NavMenu';

import { Http } from "../utilities/http";

//a homemade libary to keep track of the users credentials

export class User {

private static \_current: User | null = null;

public static get current(): User | null {

return this.\_current;

}

public static tryLogin(username: string, password: string, onSucces: (n: User) => void, onFailure: (n: string) => void): void {

Http.post<CheckResult>('api/UserGateway/Check', {

username: username,

password: password,

}).then(data => {

if (data.type == "authorized") {

Cookie.setCookie("key", data.cookieKey, 60 \* 60);

User.\_current = new User(username, data.cookieKey);

onSucces(User.\_current);

}

else {

onFailure(data.message);

}

})

}

public static check(redirect: boolean = true, onDone: ((n: User | null) => void) | null = null): void {

let \_key: string | null = Cookie.getCookie("key");

if (\_key == null) {

if (onDone != null) {

(onDone as (n: User | null) => void)(null);

}

if (redirect) {

window.location.href = "/";

}

return;

}

let key: string = \_key as string;

Http.post<CheckUserKeyResult>('api/UserGateway/CheckKey', {

key: key,

}).then(data => {

if (data.type == "authorized") {

User.\_current = new User(data.username as string, key);

}

else {

User.\_current = null;

if (redirect) {

window.location.href = "/";

}

}

if (onDone != null) {

(onDone as (n: User | null) => void)(User.current);

}

});

}

public username: string;

public key: string | null;

constructor(username: string, key: string | null = null) {

this.username = username;

this.key = key;

}

}

interface CheckResult {

type: string;

message: string;

authorized: boolean;

cookieKey: string | null;

}

interface CheckUserKeyResult {

type: string;

message: string;

authorized: boolean;

username: string | null;

}