PageTop (desarrollo) - Funcionalidad #604

Autenticación y autorización en PageTop

13/07/2023 01:44 - Manuel Cillero

Estado:	Nueva	**Fecha de inicio:**	12/07/2023
Prioridad:	Normal	**Fecha fin:**	
Asignado a:		**% Realizado:**	0%
Categoría:		**Tiempo estimado:**	0.00 hora
Versión prevista:	Versión mayor		

Descripción

PageTop debe estar preparado de serie para:

- Gestionar **usuarios** (en el core sólo el usuario *anónimo* y un *administrador* —si se configura así en config–).
- Gestionar **roles** (en el core sólo el anonymous_user y authenticated_user para el administrador).
- Gestionar **permisos** habilitados a los roles para autorizar a los usuarios.

Enlaces de interés:

- actix-web-httpauth, sin usar el *middleware* es posible ejecutarlo en la URL de login deseada. No requiere componentes porque la autenticación se realiza desde el navegador.
- Redirect On Success (a dónde enviar al usuario una vez que sus credenciales hayan sido autenticadas).

Histórico

#1 - 13/07/2023 01:51 - Manuel Cillero

Un primer intento exitoso probando actix-web-httpauth sólo ha requerido los siguientes cambios en Cargo.toml:

```
actix-web = "4"
actix-web-httpauth = "0.8.0"
actix-session = { version = "0.7.2", features = ["cookie-session"] }
```

y en src/app.rs:

```
// ...

#[cfg(feature = "database")]
use crate::db;

use actix_web_httpauth::extractors::basic::BasicAuth;

use actix_session::config::{BrowserSession, PersistentSession, SessionLifecycle};

// ...
                .configure(module::all::configure_services)

                // Cuando esté funcionando, cambiar /login por /user/login. Luego podrá ser
                // sobrecargado por otros módulos como p.e. pagetop-user.
                .service(service::web::resource("/login").route(service::web::get().to(login)))

                .default_service(service::web::route().to(service_not_found))
// ...

async fn login(request: service::HttpRequest, _auth: BasicAuth) -> ResultPage<Markup, FatalError> {
    // Si se pulsa cancelar en el diálogo que se ofrece al usuario en el navegador para introducir
    // username y password, entonces se produce un error "HTTP ERROR 401" en el propio navegador y
    // no muestra nada devuelto por el servidor web.
    // Pero si el usuario introduce un username o una password no válidas, entonces desde aquí se
    // puede devolver un error de acceso denegado.

    // format!("Hello, {}!", auth.user_id())
    Err(FatalError::AccessDenied(request))


    // En caso de éxito:
    //
    // 1. Crear una nueva sesión que guarde los datos del usuario.
```

```
    //    - En principio se guardará en memoria, por un tiempo definido (¿cron para liberar las
    //      sesiones caducadas usadas?, ¿cache para no tener todas en memoria?).
    //    - Deberá guardar los roles del usuario (por defecto sólo "authenticated user").
    // 2. Redirigir a una página "frontpage" que podría definirse en el config. Por defecto
    //    sería "/". Pero para que eso funcione hay que usar una respuesta diferente.
    //    - También podría ser un tema de "destination", que habrá que pensar.
    //
    // Aquí está la forma de hacer una redirección:
    // https://www.lpalmieri.com/posts/session-based-authentication-in-rust/#2-2-redirect-on-success
    //
    // Lo primero que tengo que pensar es en la infraestructura para sesiones, usuarios y roles.
    // Se puede crear una carpeta "auth" y dentro se definirían las estructuras para estas
    // sesiones, usuarios, roles y permisos. Esta misma función login podría estar ahí. Y luego
    // poner los mecanismos para que esta estructura sepueda sobrecargar.
}
```

#2 - 21/11/2023 19:09 - Manuel Cillero

- Versión prevista cambiado de 0.1.x a Versión mayor