Endpoint security configuration Herdenk V1.0

All endpoints, except login and register, require a user to be registered and logged in, i.e. authenticated. This allows the application to check authorization at every request made, as the current user is set by the authentication process, encrypted in a JWT token.

Overall design goal is that grave owners have full authority over their grave, who has access to it and reactions posted to the grave can be authorized, removed or even changed by the grave owner. However users should also retain full access to any reactions they posted.

In case of issues, the site administrator has full access and can mitigate or fix these issues if needed.

Apart from the standard, self-explanatory access methods like fullyAuthenticated and permitAll, some authorizations are executed by special Beans:

isSelfOrIsAdmin

returns true if the current user has an ADMIN role or if the current user wants access to his own data

hasGraveAccessOrIsAdmin

return true if the current user has at least READ access or access due to PUBLIC access to the grave.
also returns true if the user has role ADMIN

hasAtLeastOwnerAccess

returns true is the user has role ADMIN or if the current user is OWNER of the grave.

hasAtLeastWriteAccess

returns true is the user has role ADMIN or if the current user is OWNER of the grave or the current user has been granted WRITE access.

isGraveOwnerOrAuthor

returns true if a reaction is written by the current user, or if current user is OWNER of the grave the reaction applies to, or if the current user has role ADMIN

Below the simplified security configuration:

```
      .authorizeRequests()

     "/api/v1/authorities**" fullyAuthenticated()

     "/api/v1/graves**" fullyAuthenticated()

     "/api/v1/users**" fullyAuthenticated()

     "/api/v1/reactions**" fullyAuthenticated()

     "/media**" fullyAuthenticated()

     "/api/v1/login" permitAll()

     "/api/v1/register" permitAll()

GET,   "/api/v1/users/all" hasRole(“ aDMIN")

DELETE,   "/api/v1/users/{userId}" hasRole(“ aDMIN")
```

```
GET,   "/api/v1/users/{userId}" access("@AccessBeans.isSelfOrIsAdmin( #userId )")

PUT,   "/api/v1/users/{userId}" access("@AccessBeans.isSelfOrIsAdmin( #userId )")

GET,   "/api/v1/graves/all" hasRole( “ aDMIN" )

GET,   "/api/v1/graves/summary" permitAll()

GET,   "/api/v1/graves/{graveId}” access(  @AccessBeans.hasGraveAccessOrIsAdmin( #graveId )")

PUT,   "/api/v1/graves/{graveId}” access( "@AccessBeans.hasAtLeastOwnerAccess( #graveId )")

DELETE,"/api/v1/graves/{graveId}” access( "@AccessBeans.hasAtLeastOwnerAccess( #graveId )")

GET,   "/api/v1/authorities/all"hasRole(“ aDMIN")

GET,   "/api/v1/authorities/user/{userId}” access("@AccessBeans.isSelfOrIsAdmin( #userId )")

GET,   "/api/v1/authorities/grave/{graveId}” access("@AccessBeans.hasAtLeastOwnerAccess( #graveId )")

DELETE,"/api/v1/authorities/{UserId}/{graveId}” access("@AccessBeans.hasAtLeastOwnerAccess( #graveId )")

POST,  "/api/v1/authorities/grave/{graveId}/**” access("@AccessBeans.hasAtLeastOwnerAccess( #graveId )")

PUT,   "/api/v1/authorities/grave/{graveId}/**” access("@AccessBeans.hasAtLeastOwnerAccess( #graveId )")

GET,   "/api/v1/reactions/all"hasRole(“ ADMIN")

GET,   "/api/v1/reactions/grave/{graveId}” access("@AccessBeans.hasGraveAccessOrIsAdmin( #graveId )")

GET,   "/api/v1/reactions/user/{userId}” access("@AccessBeans.isSelfOrIsAdmin( #userId )")

POST,  "/api/v1/reactions/grave/{graveId}” access("@AccessBeans.hasAtLeastWriteAccess( #graveId )")

PUT,   "/api/v1/reactions/{reactionId}” access("@AccessBeans.isGraveOwnerOrAuthor( #reactionId )")

GET,   "/api/v1/reactions/permission/{graveId}” access("@AccessBeans.hasAtLeastOwnerAccess( #graveId )")

POST,  "/api/v1/reactions/permission/{graveId}/{permission}"permitAll()

GET,   "/api/v1/reactions/token/{graveId}/{token}” access("@AccessBeans.hasGraveAccessOrIsAdmin( #graveId )")

POST,  "/api/v1/reactions/token/{graveId}/{token}” access("@AccessBeans.hasGraveAccessOrIsAdmin( #graveId )")

DELETE,"/api/v1/reactions/{reactionId}” access("@AccessBeans.isGraveOwnerOrAuthor( #reactionId )")

       "/media/{graveId}/**” access("@AccessBeans.hasGraveAccessOrIsAdmin( #graveId )")

.anyRequest(denyAll()
```