CODING SHUTTLE

Spring Security

Security Methods Annotations

@Secured

The @Secured annotation is used to define role-based access control at the method level.

`@Secured(”ROLE_USER")`: Ensures the user has the ROLE_USER role.

NOTE: Make sure *secureEnabled* is set to true
`@EnableMethodSecurity(securedEnabled = true)`

@PreAuthorize

The @PreAuthorize annotation is used to secure methods based on expressions. It allows specifying access rules in method-level security.

- @PreAuthorize("hasRole('ADMIN')"): Ensures the user has the ROLE_ADMIN role.
- @PreAuthorize("hasAuthority('READ_WALLET')"): Checks for specific authority.
- @PreAuthorize("@rideSecurity.isWalletOwner(#id)"): calls specific method of a bean

Security Methods vs Request Matchers

@Secured and @PreAuthorize: Apply at the method level, offering fine-grained control over who can execute specific methods based on roles, permissions, and more complex conditions.
Request Matchers: Apply at the URL level, specifying which requests are allowed or require certain roles/permissions.

Use request matchers to secure a REST API endpoint and ensure that only authenticated users can access it. Then, within the API's service layer, you can use @PreAuthorize to enforce specific business rules.