Implementing Authentication with Symfony

This documentation explains how authentication was implemented in our Symfony application using the ***security.yaml***. It is intended for future junior developers who will be joining the team.

I. Introduction

Symfony provides a powerful security system for managing user authentication. This documentation will guide you through the key configurations and explain how to understand and customize authentication in our project.

II. Security Configuration

The security configuration for our Symfony application is located in the security.yaml file. This file contains important parameters for authentication. Here is an explanation of the key sections of this file :

Password Hashers :

This section specifies how user passwords are hashed. We use the automatic hashing algorithm ('auto') for both basic users and those of the App\Entity\User entity.

```yaml
# https://symfony.com/doc/current/security.html#registering-the-user-hashing-passwords
password_hashers:
    Symfony\Component\Security\Core\User\PasswordAuthenticatedUserInterface: 'auto'
    App\Entity\User:
        algorithm: auto
```

Providers :

Defines the source of users. In our case, users are stored in the App\Entity\User entity. The 'username' property will be used as the login identifier.

```yaml
# https://symfony.com/doc/current/security.html#loading-the-user-the-user-provider
providers:
    # used to reload user from session & other features (e.g. switch_user)
    app_user_provider:
        entity:
            class: App\Entity\User
            property: username
```

Firewalls :

Firewalls define security rules for different parts of the application. The 'main' firewall handles primary authentication and uses the 'app_user_provider' provider. It also uses the form login method ('form_login') with specific security parameters.

```yaml
firewalls:
    dev:
        pattern: ^/(_(profiler|wdt)|css|images|js)/
        security: false
    main:
        lazy: true
        provider: app_user_provider
        form_login:
            enable_csrf: true
            login_path: login
            check_path: login
            always_use_default_target_path: true
            default_target_path: /
        logout:
            path: logout
```

Access Control :

This section defines access controls for different parts of the application. For example, only users with the ROLE_ADMIN role have access to the /users route.

```yaml
access_control:
    - { path: '^/login', roles: PUBLIC_ACCESS }
    - { path: '^/users', roles: ROLE_ADMIN }
    - { path: '^/', roles: ROLE_USER }
```

Test Configuration :

This section specifies the security configuration for tests. Password hashing parameters are lowered to improve test performance.

```yaml
when@test:
    security:
        password_hashers:
            # By default, password hashers are resource intensive and take time. This is
            # important to generate secure password hashes. In tests however, secure hashes
            # are not important, waste resources and increase test times. The following
            # reduces the work factor to the lowest possible values.
            Symfony\Component\Security\Core\User\PasswordAuthenticatedUserInterface:
                algorithm: auto
                cost: 4 # Lowest possible value for bcrypt
                time_cost: 3 # Lowest possible value for argon
                memory_cost: 10 # Lowest possible value for argon
```

III. Conclusion

This documentation covers the key aspects of implementing authentication in our Symfony application using the security.yaml file. Make sure to understand how each section of this file contributes to authentication and customize the configurations according to the project's requirements.