



# Formation PHP -Symfony

## D06 - Authentication & Authorization

Rares Serban [rares.serban@pitechplus.com](mailto:rares.serban@pitechplus.com)  
42 Staff [pedago@staff.42.fr](mailto:pedago@staff.42.fr)

*Summary: Following [42](#) formation course, you will learn about the Symfony security system, authentication and authorization.*

# Contents

<b>I</b>	<b>Foreword</b>	<b>2</b>
<b>II</b>	<b>General Rules</b>	<b>3</b>
<b>III</b>	<b>Day-specific rules</b>	<b>4</b>
<b>IV</b>	<b>Exercise 01</b>	<b>5</b>
<b>V</b>	<b>Exercise 02</b>	<b>6</b>
<b>VI</b>	<b>Exercise 03</b>	<b>7</b>
<b>VII</b>	<b>Exercise 04</b>	<b>8</b>
<b>VIII</b>	<b>Exercise 05</b>	<b>9</b>
<b>IX</b>	<b>Exercise 06</b>	<b>10</b>
<b>X</b>	<b>Exercise 07</b>	<b>11</b>

# Chapter I

## Foreword

Today we are going to learn about the Security component of the Symfony framework and how we can use it to create a register/login flow for our application.

We will also learn how we can secure certain parts of our website for logged in users and even for users with certain roles.

# Chapter II

## General Rules

- This subject is the one and only trustable source. Don't trust any rumor.
- This subject can be updated up to one hour before the turn-in deadline.
- The assignments in a subject must be done in the given order. Later assignments won't be rated unless all the previous ones are perfectly executed.
- Be careful about the access rights of your files and folders.
- You must follow the **turn-in process** for each assignment. The url of your **GIT** repository for this day is available on your intranet.
- Your assignments will be evaluated by your Piscine peers.
- In addition to your peers evaluation, a program called the "Moulinette" should also evaluate your assignments. Fully automated, The Moulinette is tough and unforgiving in its evaluations. As a consequence, it is impossible to bargain your grade with it. Uphold the highest level of rigor to avoid unpleasant surprises.
- All shell assignments must run using `/bin/sh`.
- You must not leave in your turn-in repository any file other than the ones explicitly requested By the assignments.
- You have a question? Ask your left neighbor. Otherwise, try your luck with your right neighbor.
- Every technical answer you might need is available in the **mans** or on the Internet.
- Remember to use the Piscine forum of your intranet and also Slack!
- You must read the examples thoroughly. They can reveal requirements that are not obvious in the assignment's description.
- By Thor, by Odin! Use your brain!!!


# Chapter III

## Day-specific rules

- For this day, your repository must contain just one working Symfony application.
- Best practices of the Symfony framework should be respected.
- Other third party bundles or libraries CAN be used.
- Each exercise will have its own bundle name ExxBundle, where xx is the number of the exercise.
- Each exercise will have its own base route in the form of /eXX, where XX is the number of the exercise. The controllers and actions can however have any name you want.

# Chapter IV

## Exercise 01


	Exercise
Exercise 01: Login & Register	
Turn-in directory : <i>ex/</i>	
Files to turn in : <b>Files and folders from your application</b>	
Allowed functions : <b>All methods</b>	
Notes : <b>n/a</b>	

First set up a new Symfony application. Then create a bundle called E01Bundle. All of the requirements of this exercise will be put in the bundle. The following exercises will also be resolved in their respective bundles named accordingly.

Then set up the Symfony security and add a login and a register form. Your users will have to be loaded from a database. Then also add a homepage which should show a welcome message with the name of the currently logged in user and a logout button or a default welcome message and links to the login and register forms if the user is not logged in.

# Chapter V

## Exercise 02


	Exercise
Exercise 02: User Roles & Administration	
Turn-in directory : <i>ex/</i>	
Files to turn in : <b>Files and folders from your application</b>	
Allowed functions : <b>All methods</b>	
Notes : <b>n/a</b>	

Create an Administrator role for your users. Administrators will have access to a page where all the registered users will be shown.

On this page, the administrator has the ability to delete a user. He should not be able to delete his own user.

# Chapter VI

## Exercise 03

	Exercise
Exercise 03: Posts	
Turn-in directory : <i>ex/</i>	
Files to turn in : <b>Files and folders from your application</b>	
Allowed functions : <b>All methods</b>	
Notes : <b>n/a</b>	

Create a new **Post** entity which should have at least the fields: title, content, created and author. The post should be linked to the user who created it.


The homepage should now display a list with all the posts created ordered from newest to oldest. The list should show the title of the post, the name of its author and the creation date.

Each title should have a link to view the posts details page which should only be accessible by logged in users. The homepage should also have a link for logged in users to a form with the ability to create a new post.



# Chapter VII

## Exercise 04


	Exercise
Exercise 04: Anonymous Users	
Turn-in directory : <i>ex/</i>	
Files to turn in : <b>Files and folders from your application</b>	
Allowed functions : <b>All methods</b>	
Notes : <b>n/a</b>	

Anonymous user sessions should last only a minute. When an anonymous user accesses the website, he will be assigned a random name which comes from a list of animals you define, preceded by the text *anonymous*. Eg: Anonymous dog

The name will be displayed on the homepage. Also a message will be displayed showing the time in seconds since the last request was made.

# Chapter VIII

## Exercise 05

	Exercise
Exercise 05: Votes	
Turn-in directory : <i>ex/</i>	
Files to turn in : <b>Files and folders from your application</b>	
Allowed functions : <b>All methods</b>	
Notes : <b>n/a</b>	


Add ability for logged in users to vote on a post by liking or disliking it. Links for doing this can be found on the post detail page. A user can vote on a post only once.

The posts lists on the homepage should also display the amount of likes/dislikes a post has.

Each time a username appears on the website, besides it a *reputation* score should be shown. The reputation of a user is the sum of all the likes minus the sum of all the dislikes his own posts have received.

# Chapter IX

## Exercise 06


	Exercise
Exercise 06: Post Editing	
Turn-in directory : <i>ex/</i>	
Files to turn in : <b>Files and folders from your application</b>	
Allowed functions : <b>All methods</b>	
Notes : <b>n/a</b>	

Logged in users should be able to edit a post. On the detail page of a post an edit button should be shown.

At the bottom of a post's detail page a text should appear saying which user last edited the post and at what date/time, if the post was edited.

# Chapter X

## Exercise 07

	Exercise
Exercise 07: User Privileges	
Turn-in directory : <i>ex/</i>	
Files to turn in : <b>Files and folders from your application</b>	
Allowed functions : <b>All methods</b>	
Notes : <b>n/a</b>	

Depending on the number of reputation points a user has, he will have the following privileges:

- 0 points (new user) - right to create a post and edit their own post only
- 3 points - right to like posts
- 6 points - right to dislike posts
- 9+ points - right to edit any post

An admin has the right to do everything on the site regardless of his reputation points.

Also use Doctrine Fixtures to populate the database with test users, posts and votes to facilitate the testing of your application.