ΟΙΚΟΝΟΜΙΚΟ ΠΑΝΕΠΙΣΤΗΜΙΟ ΑΘΗΝΩΝ

ATHENS UNIVERSITY OF ECONOMICS AND BUSINESS

ΚΕΝΤΡΟ ΕΠΙΜΟΡΦΩΣΗΣ ΚΑΙ ΔΙΑ ΒΙΟΥ ΜΑΘΗΣΗΣ

Spring Security

Αθ. Ανδρούτσος

Spring Security (1)

- Με τον όρο Security εννοούμε δύο πράγματα:

1. **Authentication**: Τον έλεγχο πρόσβασης χρηστών στην εφαρμογή μας (username/password, Multi-factor, token-based, certificates, κλπ.)
2. **Authorization (access-control)**: Τον έλεγχο πρόσβασης στους πόρους (resources) της εφαρμογής μας με βάση δικαιώματα και ρόλους

Spring Security (2)

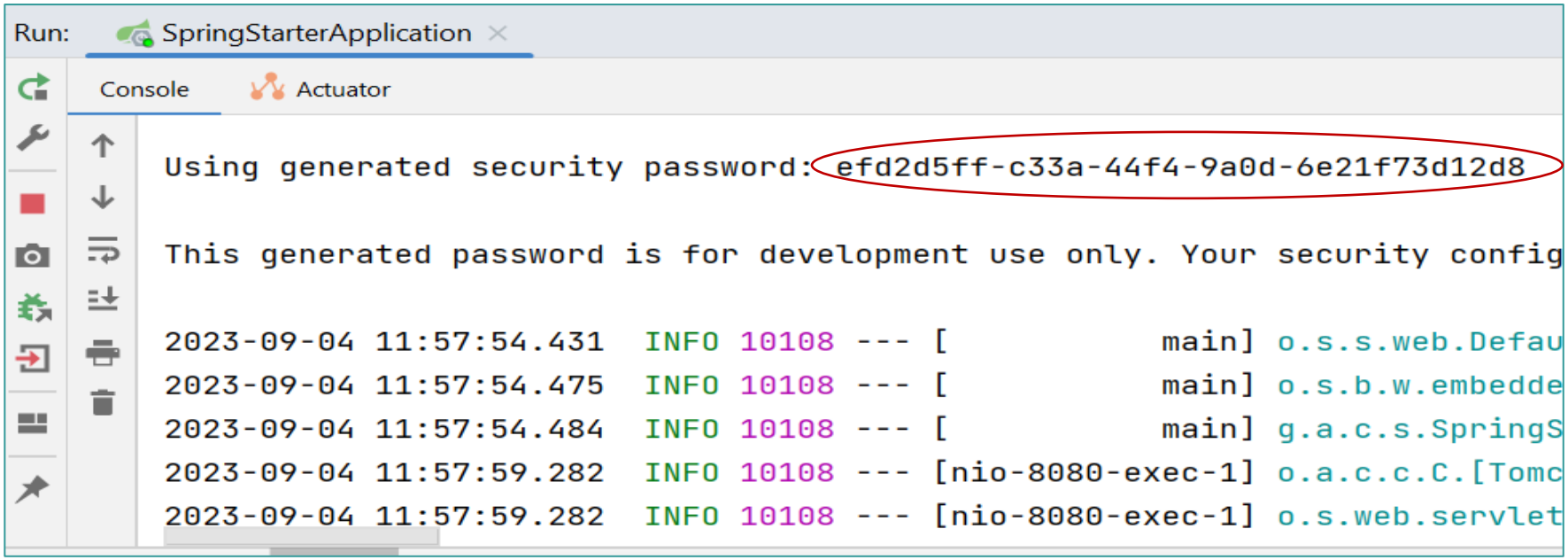
- Το **Spring Security** είναι το Spring module που μας βοηθάει να υλοποιήσουμε τα χαρακτηριστικά ασφαλείας της εφαρμογής μας με σχετικά λίγο configuration
- Μπορούμε να εισάγουμε το Spring Security module στο project μας με το starter-security dependency (Gradle & Maven)

```
implementation 'org.springframework.boot:spring-boot-starter-security'
```

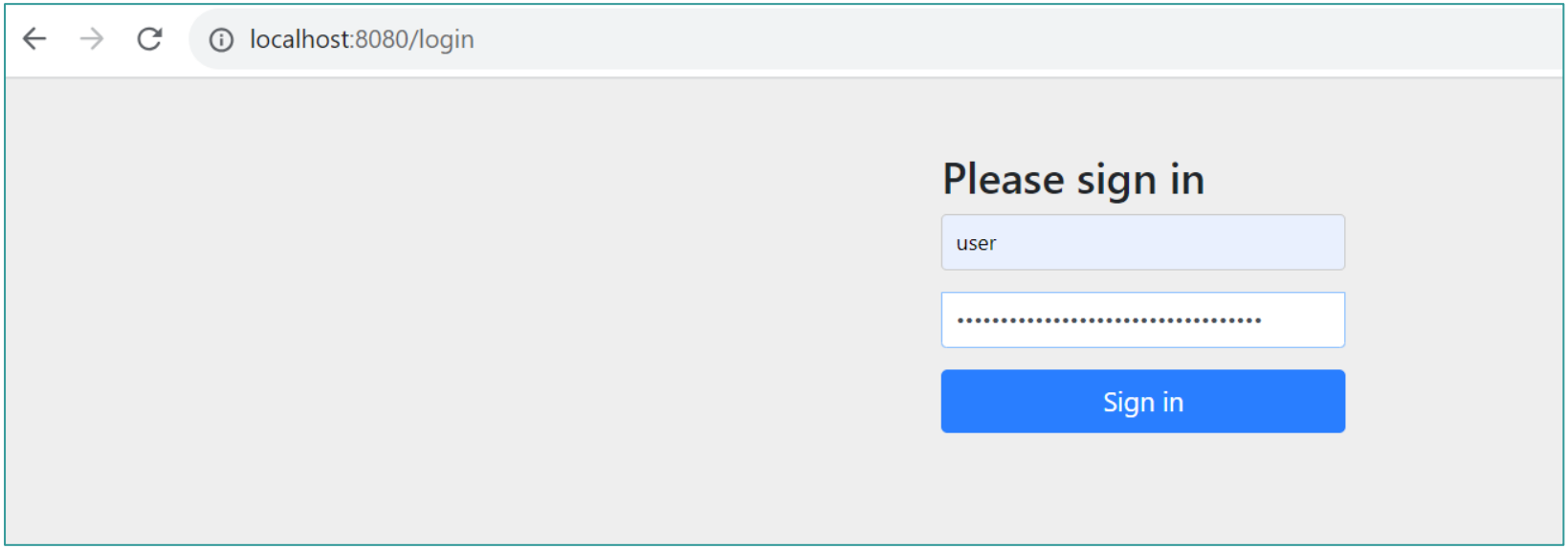
```
<!-- https://mvnrepository.com/artifact/org.springframework.boot/spring-boot-starter-security -->
<dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-security</artifactId>
    <version>2.4.5</version>
</dependency>
```

Login page (1)

- Εισάγοντας το starter-security στα dependencies αυτόματα το Spring παρέχει μία login page με username **user** και ένα password που μας δίνεται στην κονσόλα



Login page (2)



- Μπορούμε να αλλάξουμε το password στο application.properties

```
spring.security.user.password=1234
```

Authentication Mechanism

- **SecurityContextHolder**. Είναι ο χώρος στον οποίο τον Spring αποθηκεύει το ποιος είναι authenticated. Είναι singleton στο πλαίσιο μίας εφαρμογής
- **SecurityContext.** Είναι container για το Authentication object το οποίο περιέχει τα username, password και authorities κάθε user. Το SecurityContext υλοποιείται με ThreadLocal και είναι thread-safe
- **Authentication**. Το Authentication interface σχετίζεται με κάθε user και έχει δύο ρόλους: 1) Πριν το authentication από τον AuthenticationManager περιέχει το username, password και authorities όπως γίνονται map από request. Στη συνέχεια μπαίνει ως input στον Authentication Manager. Σε αυτή τη φάση το isAuthenticated () που παρέχεται από το Authentication object είναι false. 2) Μετά το authentication από τον AuthenticationManager περιέχει τα στοιχεία (username, password και authorities) του authenticated πλέον user. Μπορούμε σε αυτή τη φάση να λάβουμε το authentication object από το SecurityContext

Authentication (1)

- Ένα αντικείμενο **principal** είναι ένας χρήστης που έχει γίνει authenticate. Είναι δηλαδή ένας current logged-in user
- Το **Authentication** object περιέχει τα στοιχεία του *Principal* δηλαδή του κάθε authenticated user
- Με την μέθοδο **getName()** του Authentication επιστρέφεται το identity (π.χ. username) του authenticated principal

Authentication (2)

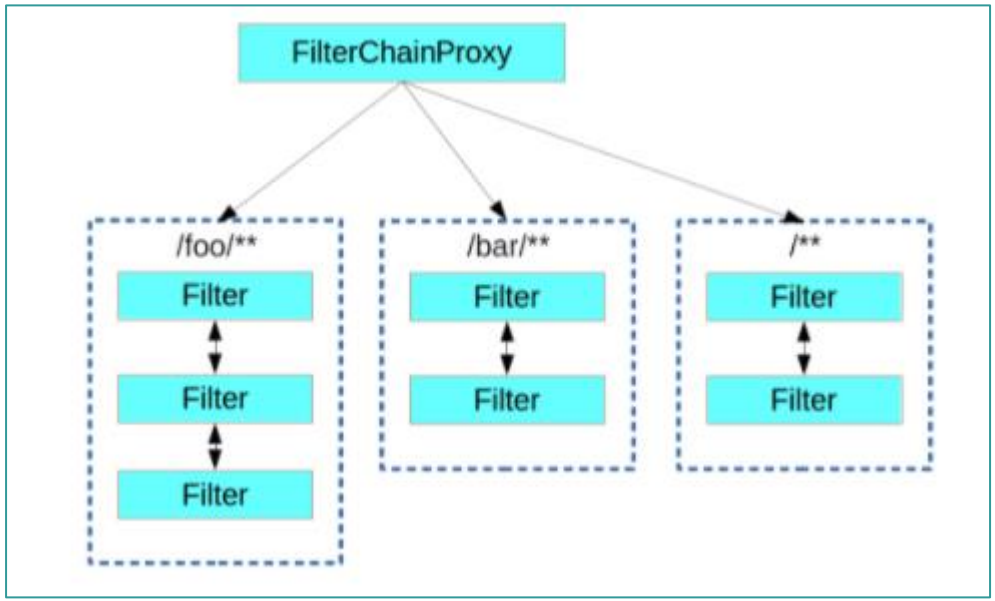
- Όταν ένας χρήστης γίνει authenticate από τον Authentication Manager που **είναι ένα φίλτρο πριν το request φτάσει στους Controllers** η Spring εφαρμογή κάνει attach ένα αντικείμενο *principal* στο authentication object
- Αν ο χρήστης γίνει authenticate ο Principal περιέχει το username του logged-in user, αλλιώς ο principal είναι null
- Αν ο χρήστης γίνει authenticate, το Spring εισάγει το *Authentication* object στο **SecurityContext**
- Το SecurityContect Είναι **thread-safe** αφού είναι μέσα σε ένα ThreadLocal instance

Authorization

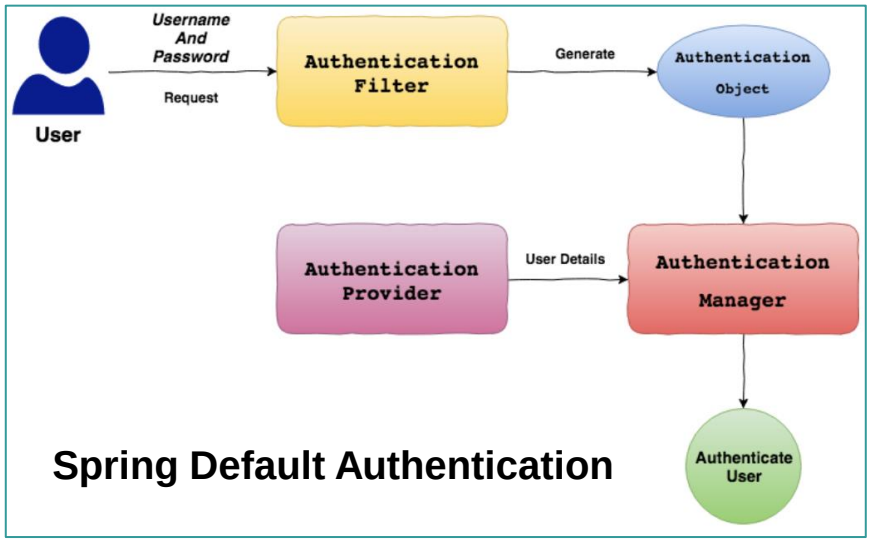
- Είναι η διαδικασία εκχώρησης δικαιωμάτων σε ένα χρήστη (ή εφαρμογή) που του επιτρέπει να έχει πρόσβαση σε πόρους όπως endpoints
- Κατά τη διαδικασία του authorization θα πρέπει είτε όλοι οι χρήστες να επιτρέπεται να έχουν πρόσβαση σε ένα endpoint (όπως π.χ. στο /login) ή με βάση τον ρόλο τους να έχουν πρόσβαση σε συγκεκριμένα endpoints
- Τα authorities (δικαιώματα) αποθηκεύονται στο Collection **Granted Authorities.** Τα authorities μπορούν να είναι και ο ρόλος ή ρόλοι ενός χρήστη

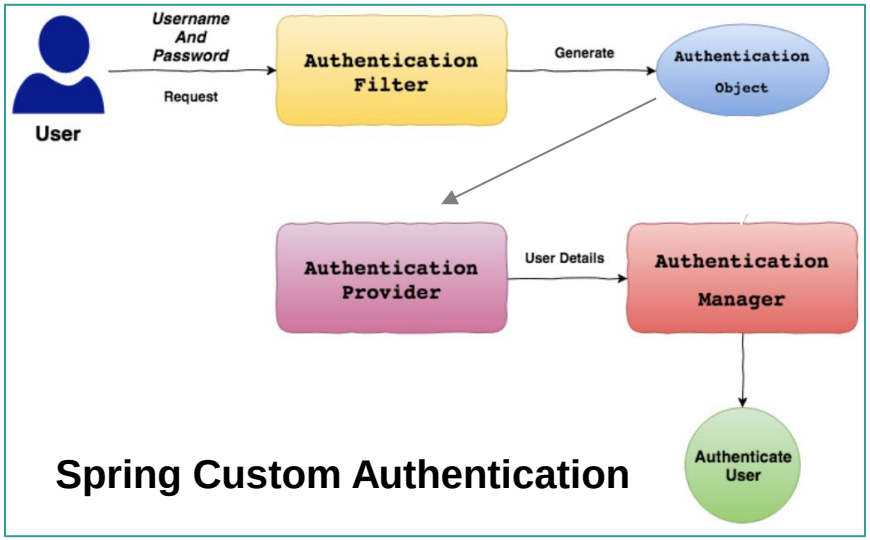
Αρχιτεκτονική Spring Security

- Το **Spring Security** δημιουργεί μία στοίβα ή αλυσίδα (chain) από **Servlet Filters** μεταξύ του client και των controllers της εφαρμογής μας, όπου κάθε φίλτρο ελέγχει κάθε request με βάση τον default Authentication Provider ή με βάση το δικό μας Custom Authentication provider αν έχουμε δημιουργήσει



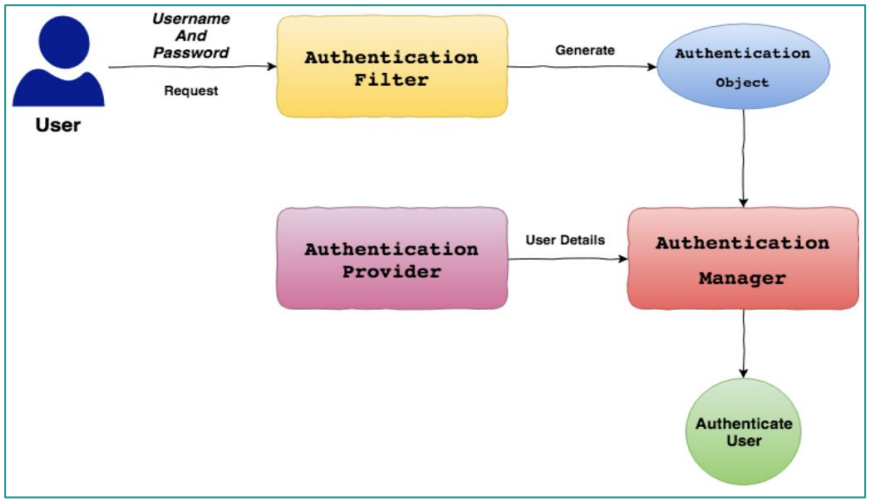
Αρχιτεκτονική Authentication

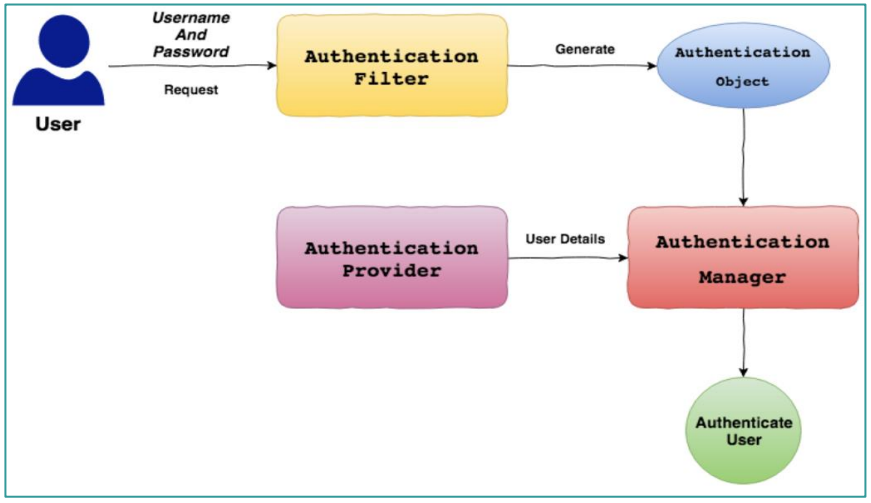




- Ένα παράδειγμα είναι το ***UsernamePassword AuthenticationFilter*** που λαμβάνει το request με τα credentials του χρήστη και δημιουργεί το **Authentication object** με τα credentials του χρήστη
- Η Αρχιτεκτονική του Spring για το Authentication είναι ότι το **AuthenticationFilter** παρέχει το **Authentication object**.

Authentication Provider





- Ο **AuthenticationProvider** παρέχει τα στοιχεία από τη ΒΔ και συγκρίνει με βάση τη μέθοδο authenticate και επιστρέφει ένα UsernamePasswordAuthenticationToken που είναι τύπου Authentication object ή μία εξαίρεση BadCredentialsException αν τα credentials δεν είναι valid. Μπορούμε να κάνουμε **custom υλοποίηση του AuthenticationProvider** ορίζοντας εμείς το έλεγχο των credentials
- Ο **Authentication Manager** είναι ο συντονιστής της όλης διαδικασίας και τον κάνουμε configure με ένα η περισσότερους Authentication providers