6.2 Prepared Statements

Ενότητα 6: Ασφάλεια χρηστών

Πολιτόπουλος Νικόλαος – npolitop@csd.auth.gr

Τι θα δούμε

- Prepared Statements

Recap

- Στο προηγούμενο μάθημα είδαμε τι είναι το SQL Injection και πως μπορούμε να το αποφύγουμε με 2 τρόπους:
  - Escaping Quotes
  - Prepared Statements
- Χρησιμοποιώντας την εντολή quote της PDO εισάγουμε εισαγωγικά σε ένα string και προσπερνάμε τους ειδικούς χαρακτήρες
  - `echo $pdo->quote('Smith'); //Prints 'Smith' (with quotes)`

Prepared statements

- Η 2η μέθοδος για να λύσουμε αυτό το πρόβλημα είναι τα **Prepared statements**
- Τα prepared statements είναι ένα ειδικό query το οποίο γίνεται compiled πριν εκτελεστεί.
- Αντί να γράψουμε ένα query χρησιμοποιώντας ένα string με concatenation, γράφουμε ένα query με placeholders στο string τα οποία θα αντικατασταθούν όταν το query εκτελεστεί

Prepared statements

- Για να γίνει αυτό χρησιμοποιούμε τη συνάρτηση **prepare** της PDO
- Όταν ένα query γίνεται *prepared* «αποθηκεύεται» και περιμένει να εκτελεστεί
- Πρέπει να καλέσουμε και τη συνάρτηση **execute** για να εκτελέσουμε το query.

Prepared Statements

- Όταν γράφουμε ένα query που θα γίνει prepared δε χρειάζεται να βάζουμε εισαγωγικά ή την τιμή που θέλουμε να χρησιμοποιήσουμε στο query
- Στη θέση τους ορίζουμε έναν **placeholder**
- Ο placeholder είναι ένα όνομα της επιλογής μας που από μπροστά του έχει την **άνω κάτω τελεία (:)** Π.χ. :testname
- `$pdo->prepare('SELECT * FROM person WHERE surname = :name');`

Prepared statements

- `$pdo->prepare('SELECT * FROM person WHERE surname = :name');`
- Η prepare function επιστρέφει ένα PDOStatement Object
- Αυτό το αντικείμενο είναι ένα prepared query που μπορούμε να εκτελέσουμε αργότερα.
- Πρέπει να αποθηκεύσουμε το αντικείμενο μέσα σε μια μεταβλητή για να μπορούμε να το εκτελέσουμε αργότερα.
- `$stmt = $pdo->prepare('SELECT * FROM person WHERE surname = :name');`
- Συμβατικά καλό είναι να χρησιμοποιείτε τη συνάρτηση **$stmt** (statement)

Prepared statements

- Εφόσον έχετε ετοιμάσει ένα $stmt αντικείμενο μπορεί να κάνετε execute το query
- Όταν εκτελείται το query χρησιμοποιώντας την συνάρτηση execute πρέπει να συμπληρώσετε μαζί και ένα array για την αντικατάσταση των placeholders στο query.
- Για παράδειγμα το παρακάτω query έχει ένα placeholder που ονομάζει “name”

```
$stmt = $pdo->prepare('SELECT * FROM person WHERE surname
= :name');
```

Prepared statements

- Για να παρέχουμε τις αντικαταστάσεις των placeholders δημιουργούμε ένα array με keys τα ονόματά τους και σαν τιμές αυτά που θέλουμε να μπουν μέσα

```
$stmt = $pdo->prepare('SELECT * FROM person WHERE surname = :name’);
     $criteria = [
'name' => $_POST['surname']
];
$stmt->execute($criteria);
```

- Αυτό θα στείλει το query στη ΒΔ
- Όμως, αντιθέτως με τη query, η execute δεν επιστρέφει τις εγγραφές που θέλουμε από το ερώτημα στη ΒΔ

Prepared statements

- Μόλις κάνουμε execute το query μπορούμε να χρησιμοποιήσουμε ένα foreach loop για να προσπελάσουμε όλες τις εγγραφές

Παράδειγμα 1

```
$stmt = $pdo->prepare('SELECT * FROM person WHERE surname
= :name');
$criteria = [
     'name' => $_POST['surname']
];
$stmt->execute($criteria);
foreach ($stmt as $row) {
echo '<p>' . $row['firstname'] . '</p>';
}
```

Prepared statements

- Μπορούμε επίσης να χρησιμοποιήσουμε την εντολή $stmt->fetch() για να μας γυρίσει μόνο μία εγγραφή

Παράδειγμα 2

```
$stmt = $pdo->prepare('SELECT * FROM person WHERE surname
= :name');
$criteria = [
     'name' => $_POST['surname']
];
$stmt->execute($criteria);
$row = $stmt->fetch();
```

Prepared Statements

- Η εντολή fetch() επιστρέφει μία εγγραφή και **τότε** ο δείκτης της μεταφέρεται στην επόμενη. Εάν δεν υπάρχει επιστρέφει **false**.
- Μπορούμε να τη συνδυάσουμε με μια **while** επανάληψη για να προσπελάσουμε όλες τις εγγραφές που επεστράφησαν.

Παράδειγμα 3

```
while ($row =
$stmt->fetch()) {
     echo '<p>’ .
     $row['firstname'] .
     '</p>';
}
```

```
foreach ($stmt as $row) {
     echo '<p>’ .
     $row['surname'] .
     '</p>';
}
```

Prepared Statements

- Τα Prepared Statements απαιτούν περισσότερο κώδικα αλλά είναι:
  - Πολύ πιο ασφαλή και έχουν «ανοσία» στο SQL Injection
  - Είναι πιο ευανάγνωστα γιατί δεν χρειάζεται να γίνει concatenation στα strings
  - Γρηγορότερα στην εκτέλεση

Prepared Statements

- Τα Prepared Statements μπορούν να χρησιμοποιηθούν για οποιοδήποτε query και με όσες παραμέτρους θέλουμε

Παράδειγμα 4α

```
$stmt = $pdo->prepare('INSERT INTO person (email, firstname, surname, birthday)
VALUES (:email, :firstname, :surname, :birthday)
');
$criteria = [
    'firstname' => $_POST['firstname'],
    'surname' => $_POST['surname'],
    'email' => $_POST['email'],
    'birthday' => $_POST['birthday']
];
$stmt->execute($criteria);
```

Παράδειγμα 4β

```
$pdo->query('INSERT INTO person (email, firstname, surname, birthday)
                    VALUES ("' . $_POST['email'] . '", "' . $_POST['firstname'] .
                    '", "' . $_POST['surname'] . '", "' . $_POST['birthday'] . '")
');
```

Prepared Statements

- Ο κώδικας για τα Prepared Statements μπορεί να μειωθεί σημαντικά αρκεί να σχεδιάσετε σωστά τον κώδικά σας.
- Αν για παράδειγμα τα ονόματα των placeholders στο query είναι ίδια με τα πεδία σε μια φόρμα που χρησιμοποιείτε, το array με τα κριτήρια δημιουργείται αυτόματα για εσάς

Παράδειγμα 5α

```
$stmt = $pdo->prepare('INSERT INTO person (email, firstname, surname, birthday)
VALUES (:email, :firstname, :surname, :birthday)
');
$criteria = [
    'firstname' => $_POST['firstname'],
    'surname' => $_POST['surname'],
    'email' => $_POST['email'],
    'birthday' => $_POST['birthday']
];
$stmt->execute($criteria);
```

Παράδειγμα 5β

```
$stmt = $pdo->prepare('INSERT INTO person (email, firstname, surname, birthday)
        VALUES (:email, :firstname, :surname, :birthday)
');
$stmt->execute($_POST);
```

Παράδειγμα 5γ

```
<form action="add.php" method="POST">
  <label>First name:</label>
  <input type="text" name="firstname" />
  <label>Surname:</label>
  <input type="text" name="surname" />
  <label>Email:</label>
  <input type="text" name="email" />
  <label>Birthday:</label>
  <input type="text" name=“birthday" />
  <input type="submit" name="submit" value=”Submit” />
</form>
```

Prepared Statements

- Αυτό θα δημιουργήσει ένα πρόβλημα καθώς γίνεται post και το submit καθώς είναι μέρος της φόρμας
- Το $_POST[‘submit’] έχει οριστεί αλλά δεν υπάρχει placeholder
- Για να το διορθώσουμε αυτό αρκεί να το αφαιρέσουμε από το array

```
$stmt = $pdo->prepare('INSERT INTO person (email, firstname, surname, birthday)
                VALUES (:email, :firstname, :surname, :birthday)
‘);
unset($_POST['submit']);
$stmt->execute($_POST);
```

Σύνοψη

- Είδαμε τι είναι τα prepared statements
- Πως τα ορίζουμε και τα εκτελούμε
- Περισσότερος κώδικας αλλά:
  - Ασφαλέστερα
  - Πιο ευανάγνωστα
  - Γρηγορότερα