Persistentie

Voorkennis en uitwerking

Deze opgave bouwt verder op de oplossing van LABO 05. Doel is om de website van persistentie te voorzien.

Het is sterk aangeraden alle code en de theoretische achtergrond die deze opgave vereist grondig te bestuderen alvorens je naar het labo komt. Bestudeer eveneens de reeds voorziene code.

De opgaves dienen in volgorde afgewerkte te worden. Het huishoudelijk reglement is zoals steeds van toepassing.

Timing en upload

Dit labo loopt over één sessie (3u) en dient aan het einde van dit labo via Toledo geüpload te worden. Thuis werk je het labo verder af. Upload één .zip bestand met daarin je .php bestanden.

Doelstellingen

- Vlot werken met Twig (herhaling)
- Met sessions en cookies kunnen werken

Op schijf

- De locatie waar dit project opgeslagen wordt is <webroot>/voornaam.achternaam/labo06
- Bewerk de reeds voorziene bestanden

Opgaves

1. Afschermen website met login

- Creëer een nieuwe pagina login.php waar een gebruiker kan aanmelden. Pas na aanmelden mag hij toegang tot alle andere pagina's verkrijgen.
    - Gebruik de bij deze opgave toegevoegde .sql dump. Deze is reeds uitgebreid met een extra users tabel (paswoorden: Azerty123), alsook met de koppeling tussen de todos en de users (veld: user_id).
    - Op elke pagina zal je een stukje code moeten invoegen dat verifieert of een gebruiker al dan niet aangemeld is. Indien niet aangemeld, wordt men naar het aanmeldformulier doorverwezen. Indien men aangemeld login.php bezoekt, wordt men doorverwezen naar browse.php .
    - Bijkomende tips & tricks:
        - Haal bij de verificatie van een gebruiker enkel de info over één bepaalde user uit de databank op.
        - Plaats `session_start()` altijd bovenaan (zo net onder de includes), en niet ergens in het midden van de code.
        - Vergeet niet om na `header('location: ...')` ook een `exit()` te plaatsen. Doe je dit niet, dan kan je alsnog (zij het via een sniffer) de HTML-code van de rest van de website zien!
- Creëer eveneens een logout.php script waarmee men kan afmelden. Ingelogde gebruikers krijgen steeds een link naar dit script te zien onder de footer.

2. Ondersteunen meerdere gebruikers

- Pas de database-queries aan zodat de user_id mee bij een todo-item opgeslagen wordt: enkel de todo-items van de aangemelde gebruiker mogen getoond worden.
- Vergeet niet om naast de overzichtspagina ook de andere pagina's aan te passen zodat deze de user_id mee in rekening nemen in de queries. **Het mag geenszins mogelijk zijn om todo-items van andere gebruikers te kunnen raadplegen**; controleer dit ook bij de bewerk- en verwijderpagina's. **Doe je dit niet dan is je site vatbaar voor parameter tampering.**

3. Onthouden laatste succesvolle login

- Zorg ervoor dat het tijdstip en de gebruikersnaam van de laatst succesvolle login steeds in cookies worden bijgehouden. Indien men na afmelden nogmaals de login-pagina bezoekt, moet onderaan het formulier een melding staan van deze gegevens bv. “Laatste login door jefke op 18/12/2017 om 12:35 ”

4. (optioneel) Registratie gebruikers toelaten

- Laat gebruikers toe om te registreren. Na registratie kunnen ze meteen aanmelden.

Wat onthouden we voor het labo-examen?

1. Beveiligen tegen SQL Injectie (kost je erg veel punten op het examen indien niet)

- Geen parameters rechtstreeks in queries kleven maar steeds prepared statements gebruiken.

2. Beveiligen tegen parameter tampering

- Uitbreiden van de ophaalqueries met een `... AND user_id = ?` clausule.
- `INSERT` en `UPDATE` queries eveneens uitbreiden met die `user_id`.

3. Gebruik parameters steeds in hun ruwe vorm

- Sla username en wachtwoord ruw in de DB op: géén htmlentities() rond draaien!
- Wachtwoorden encrypteren met `password_hash()`. `crypt()` is verouderd; `sha1()` en `md5()` voldoen niet!
- Query: escapen (SQL Injection), automatisch via goed gebruik van prepared statements.
- Weergave op scherm: htmlentities (XSS), automatisch via Twig.