Formulieren

Voorkennis

Deze opgave gaat over het werken met formulieren in PHP. De PHP basissyntax zou je reeds onder de knie moeten hebben. De superglobale arrays $_POST en $_GET, die de post- en getwaarden bevatten, alsook persistentie inbouwen en omgaan met de in de theoriesessie vermelde pitfalls zouden je niet onbekend mogen zijn.

Uitwerking

Voor het uitwerken van de labo's maken we vanaf nu gebruik van **MAMP of Wampserver**, in tegenstelling tot vorig labo waar alles op de shell diende uitgevoerd te worden.

De opdrachten 1 en 2 gebruiken de get methode. Vanaf opdracht 3 wordt post gebruikt.

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. De opdrachten hoeven niet per se in volgorde opgelost worden.

Upload één .zip bestand met daarin je .php bestanden.

Doelstellingen

- Het communicatiemechanisme met formulieren tussen client en server via de $_GET en $_POST arrays begrijpen.
- Persistentie en formchecking kunnen integreren in formulieren en omgaan met vaak voorkomende pitfalls bij het werken met formulieren.

Op schijf

- De locatie waar dit project opgeslagen wordt is <webroot>/voornaam.achternaam/labo02
- Sla elk bestand als opgave_x.php op.

Aandachtspunten

Let er op dat tijdens het ontwikkelen foutmeldingen ingeschakeld zijn. In de labo's is dit zo. Indien dit niet het geval is, voeg dan deze twee regels helemaal bovenaan jouw PHP bestand in:

```
error_reporting(E_ALL);
ini_set('display_errors', 'on');
```

Besteed verder de nodige aandacht aan hetgeen tijdens de theorie besproken werd:

- Het volgen van de algemene structuur van een PHP pagina: bovenaan halen we waarden op en manipuleren we deze en onderaan (van zodra het doctype start) gebruiken we enkel nog PHP om zaken weer te geven.
- De correcte verwerking van formuliergegevens in PHP staat en valt bij het correct opbouwen van het formulier in HTML. Brakke HTML kan er voor zorgen dat jouw data niet correct doorgestuurd wordt.
- Detecteer of het desbetreffende formulier werd verzonden a.h.v. het (verborgen) moduleAction veld (bij voorkeur) of a.h.v. de ingedrukte knop.
- Het ontwijken van de pitfalls (undefined index en XSS)
- Werk steeds met ruwe waarden. Pas bij de weergave bouw je de beveiliging tegen XSS in.

Vergeet verder niet om aan de start van dit labo het cursusmateriaal te updaten (via git) zodat je over de laatste versie beschikt. In de submap assets/03/examples/ vind je alle codevoorbeelden. De codevoorbeelden persistency_11.php en formchecking_multipleforms.php vormen een goede basis om van te starten. **Neem gerust de layout over.** We willen ons immers focussen op het schrijven van PHP, niet op het schrijven van HTML.

Opgaven met de `get` methode

1. Maak een PHP-pagina met een formulier en daarin één tekstveld waar je om een naam vraagt. Via de submitknop moet het formulier naar zichzelf verstuurd worden en moet de waarde ervan gepersisteerd worden.

Indien niks werd ingevuld komt de foutmelding "Gelieve jouw naam in te vullen" op het scherm.

2. Maak een PHP-pagina met twee tekstvelden die elk een random natuurlijk getal omvatten (bij een refresh zonder parameters komt er telkens een andere waarde in). Via een knop kan het formulier verstuurd worden. Na versturen moet onderaan het formulier de som van beiden weergegeven worden, en worden de oorspronkelijke twee getallen gepersisteerd.

Let op: de som mag enkel berekend worden indien het formulier verstuurd werd én indien er

twee getalwaarden doorgestuurd werden. Maak gebruik van de filter_var(..., FILTER_VALIDATE_INT) functie in PHP om te bepalen of de waarde van een variabele een integer is.

Opgaven met de `post` methode

3. Maak een formulier met drie radiobuttons voor drie soorten geheugenmodules: 4GB, 8GB, en 16GB van respectievelijk 45, 54 en 109 Euro. Bij het klikken op de submitknop moet het formulier naar zichzelf verstuurd worden en vervolgens in een tekst onderaan de prijs ingevoegd worden, bijvoorbeeld “De prijs is: xx Euro”. Persisteer uiteraard de gekozen radiobutton.

Bemerk dat het in geen geval veilig is om de values van de radiobuttons in te stellen op de prijs van de gekozen geheugenmodule; dit kan men immers via de DevTools/Firebug manipuleren! Opteer voor het gebruik van een associatieve array waar je het aantal GB van de geheugenmodule koppelt aan de prijs ervan.

4. Maak een formulier met de vraag “Computerprobleem melden” met de velden naam, e-mailadres, lokaal (gebruik hiervoor een dropdown met de keuze “LInfo 1” t.e.m. 6), computernummer, de keuze tussen software of hardware (radiobuttons), een titelveld en een tekstvak voor de omschrijving. Vergeet uiteraard de submitknop niet. Na het versturen van het formulier moet de pagina herladen worden mét persistentie van de aangeduide opties en ingevulde waarden.

5. Breid de vorige opgave uit met PHP formchecking en bijhorende foutboodschappen. Indien het formulier correct ingevuld werd, verwijs je door naar de bedankingspagina met als URL opgave_5b.php. Zorg er voor dat de ingevulde naam en e-mailadres op de bedankingspagina weergegeven worden (tip: querystring).