Login

We gaan een veilige en goede login maken

We gaan leren hoe in PHP een gebruiker wordt geauthoriseerd via een login en user id.

We kennen drie delen:

1. Login form
2. De control page (php)
3. De database connectie waarmee het user_id en password worden gecontroleerd.

Maak de files

```
login.html
login.php
db.php
```

We beginnen met de db.php. We maken nog geen connectie met de databse maar defeiniëren voorlopig een assiociative array met user-id en met het niet-encrypted wachtwoord. We gaan dit later uiteraard aanpassen, maar om één en ander te testen is dit nu beter en sneller.

```
// vul het array aan met je eigen login
$logins = [ 'Damon' => 'geheim12', 'Su-Yen' => 'secret123', 'Bob' => 'grotegroep3',
□□□'Omar' => 'Laptop12', 'JariF' => 'verweg-42' ];
```

Maak nu een object login met de private property username en één public functie:

```
Login->authenticate($username, $password);
```

Weet je nog hoe he een object maakt? Maak eerst de class, definiëer de property username en maak de methods/functies. Instantieer de class door er een object van te maken.

Als je niet meer weet hoe dat precies zat met classes en objects, dan kun je bijvoorbeeld kijken naar:

https://www.youtube.com/watch?v=pFz180MbCM8

https://www.roc.ovh/books/veilig-programmeren/page/objecten---extra-uitleg

>

> Vraag 1: waarom staat Login hierboven met een hoofdletter?

Deze functie (of eigenlijk method) returned Trueof False; true als de combinatie username en password klopt en anders False.

Maak nu het form en post de login en userid naar login.php. Check eerst of de variabelen uit het form bekend zijn in login.php door deze af te drukken. Als dat het geval is, check dan of de login en userid combinatie klopt.

Nu is het zaak dat de browser onthoud dat de gebruiker is aangelogd. Hoe doen we dat ook alweer?

Inderdaad met een cookie.

```
setcookie('login', 'True', 1200);
```

> Vraag 2: waar staat de 1200 voor?

> Vraag 3: werkt dit?

> Vraag 4: waarom is dit niet veilig?

Session variabele

We gaan een sessie opbouwen. Een sessie is een stukje geheugen op de webserver waarin gegevens van jou worden bijgehouden. We starten een sessie op met:

```
session_start();
```

Nu kunnen we de status van de authenticatie (of we succesvol zijn aangelogd) in een sessie variabele op de server vastleggen.

```
$_SESSION['user'] = $username;
$_SESSION['login'] = 'True';
```

> Vraag 5: Leg uit waarom het bijhouden van of iemand is aangelogd via een sessie variable wel veilig is (in tegenstelling tot het vastleggen van de inlog status via een cookie).

Deze variabelen worden nu op de server opgeslagen en elke keer als we een pagina vragen worden deze variabele mee gestuurd naar de browser zodat je ze kunt uitlezen.

```
echo $_SESSION['user'];
```

Drukt de user name af.

Maak nu een nieuwe pagina, waarin je laat zien of er iemand is aangelogd en als dat zo is wie dat is. Gebruik hiervoor de sessie variabelen.

De output van de pagina ziet er dus als volgt uit:

> `Er ie niemand aangelogd`
>
> of
>
> `JariF is aangelogd`

Bekijk in je browser welke cookies er worden gebruikt, herken je de geheime sessie key?

> Vraag 6: waarom is de deze sessie key zo lang?

> Stuur de file login.php via teams op (opdracht op tijd klaar is 1 punt).

Opdracht

Verander nu het array zodanig dat alle wachtwoorden encypted worden opgeslagen (in het array). Gebruik hiervoor de php functie `crypt()`. Zoek op hoe deze werkt. Je kunt de functie gebuiken zonder 'salt'. Dit gaan we later behandelen.

Maak alle nodige veranderingen zodat je de gebruikers kan authenticeren aan de hand van de encrypted passwords.

Revision #7
Created 2019-09-21 09:42:58 UTC by Admin
Updated 2019-09-26 10:33:38 UTC by Admin