Gebruikers registreren

Een gebruikersdatabase maken

Een veel gebruikte databases bij websites is een gebruikersdatabase om informatie van de gebruikers van de website te registreren. In deze les gaan we zo'n database bouwen met de bijbehorende registratie en inlog pagina's.

Wanneer de gebruiker aanmeldingsgegevens invoert die overeenkomen met de gegevens die al in de databasetabel zijn opgeslagen, kan het script toegang tot de website verlenen. Doorgaans kan een databasetabel met gebruikersdetails het gebruikers-ID, de voornaam, de achternaam, het e-mailadres, het wachtwoord en de herhalingsdatum van elke gebruiker in afzonderlijke kolommen opslaan:

Kolom naam:	Type inhoud:	Voorbeeld:
user_id	Number	123
first_name	Text	Janchi
last_name	Text	Ras
email	Text	janchi@voorbeeld.com
pass	Text	Wachtwoord123
reg_date	Date/Time	2018-10-07 10:19:00

Het ontwerp van deze databasetabel vereist **CHAR**- of **VARCHAR**-gegevenstypen die moeten worden gespecificeerd voor kolommen die tekst opslaan. Gebruikers-ID's moeten positieve (**UNSIGNED**) integer-**INT**-waarden zijn die de tabel automatisch toewijst met **AUTO_INCREMENT** en de gebruikers eerste tijdstempel moet worden opgeslagen als een **DATETIME**-waarde. Het e-mailadres van de gebruiker moet **UNIQUE** zijn, dus ze kunnen niet meerdere keren worden geregistreerd en omdat al deze items nodig zijn voor registratie, moet elke kolom een **NOT NULL**-regel hebben.

1. Start een teksteditor en maak een SQL-opdracht die de tabelvereisten beschrijft die in de bovenstaande tekst worden getoond.

```
CREATE TABLE IF NOT EXISTS users (
user_id INT UNSIGNED NOT NULL AUTO_INCREMENT,
first_name VARCHAR(20) NOT NULL,
last_name VARCHAR(40) NOT NULL,
email VARCHAR(60) NOT NULL,
pass CHAR(40) NOT NULL,
reg_date DATETIME NOT NULL,
PRIMARY KEY (user_id),
UNIQUE (email)
);
```

Gebruikers registreren

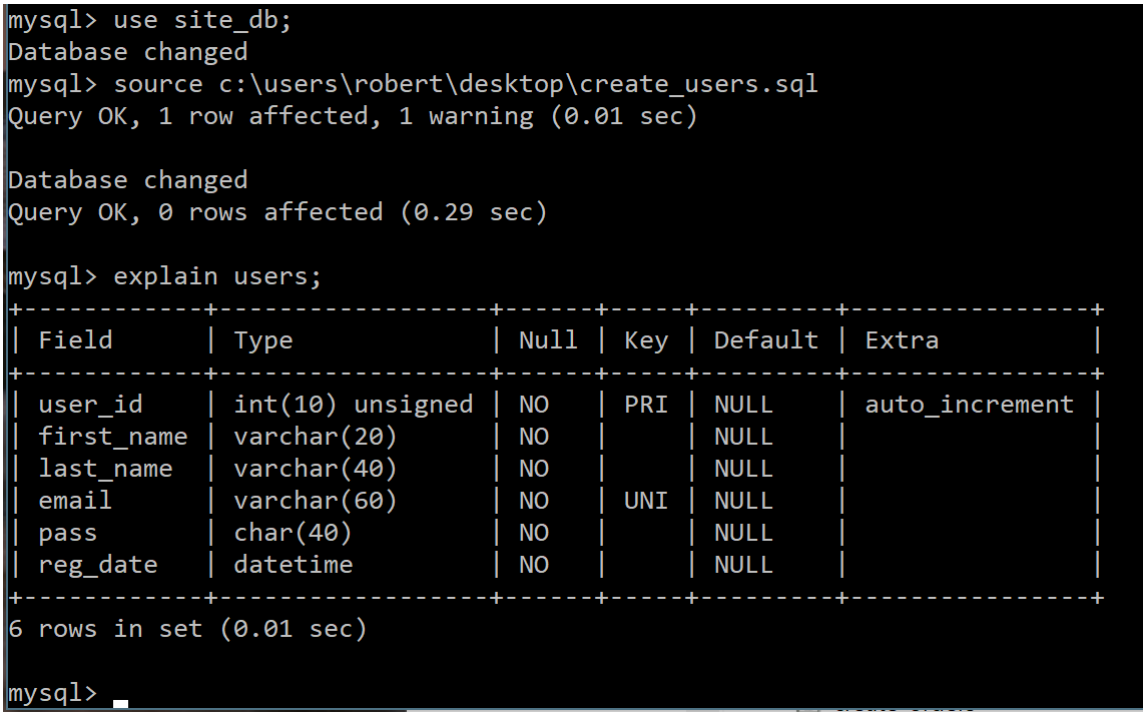
2 Sla het bestand op als **create_user.sql** en start vervolgens de MSCLC en log in als een gebruiker met volledige toegangsrechten, zoals de rootgebruiker.

3 Voer vervolgens een opdracht uit om de database te gebruiken:
USE site_db ;

4 Voer nu de onderstaande SQL-opdracht door het bron pad van het SQL-bestand op uw systeem te vermelden, zoals:

SOURCE C:\Users\<Gebruikersnaam>\Desktop\create_users.sql

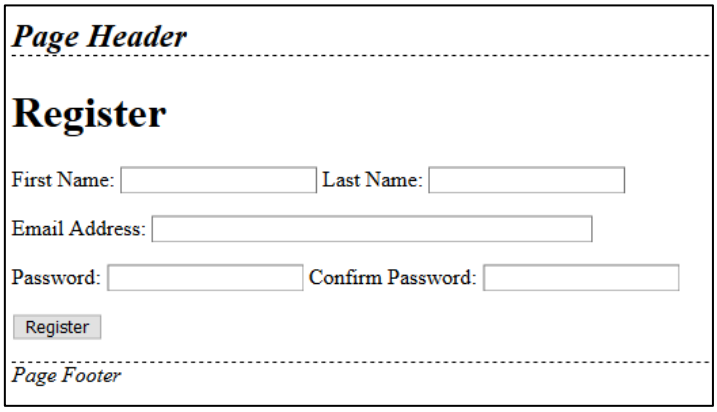
5 Als laatste, geef een opdracht om het tabelformaat te bevestigen:
EXPLAIN users ;



Gebruikers registreren

Het maken van een registerpagina

Een gebruikersregistratieformulier bevat invoervelden zoals de gebruiker zijn voornaam, achternaam, e-mailadres en een uniek wachtwoord. Dit kan op een webpagina worden weergegeven door een PHP-script dat ook de kop- en voetteksten van de pagina bevat.



Om het formulier te verwerken, moet het attribuut van HTML **<form>** tag ingediend worden met de **POST**-methode en de **action** methode moet de naam van hetzelfde PHP-script opgeven dat het formulier weergeeft. Hierdoor kan elke invoer door de gebruiker worden behouden door de invoervelden in het geval dat een item niet wordt gevalideerd.

Wanneer het formulier wordt ingediend na invoer door de gebruiker, zal het PHP-script berichten weergeven om de gebruiker te informeren over elke validatiefout of een bevestiging weergeven dat registratie is gelukt en een hyperlink naar een inlogpagina tonen.

1 Start een teksteditor en begin een PHP-script met instructies om de paginatitel in te stellen en een paginakoptekst op te nemen.

```
<?php # DISPLAY COMPLETE REGISTRATION PAGE

# Set page title and display header section.
$page_title = 'Register' ;
include ( 'includes/header.html' ) ;
```

Gebruikers registreren

2 Voeg nu een voorwaardelijke test toe die alleen de instructies uitvoert die het bevat als het formulier is ingediend.

```
# Check form submitted.
if ( $_SERVER[ 'REQUEST_METHOD' ] == 'POST' )
{
	# Instructies van stap 3 – 10 dienen hier te worden toegevoegd
}
?>
```

3 Voeg vervolgens instructies toe om de databaseverbinding te openen en initialiseer een array voor foutberichten:

```
# Connect to the database.
 require ('../connect_db.php');

# Initialize an error array.
$errors = array();
```

4 Voeg instructies in om een foutmelding op te slaan als het veld voor de voornaam leeg blijft, of sla de waarde ervan op in een variabele.

```
# Check for a first name.
if ( empty( $_POST[ 'first_name' ] ) )
{
	$errors[] = 'Enter your first name.' ;
}
else
{
	$fn = mysqli_real_escape_string( $dbc, trim( $_POST[ 'first_name' ] ) ) ;
}
```

5 Voeg instructies in om een foutmelding op te slaan als het veld voor de achternaam leeg blijft, of sla de waarde ervan op in een variabele.

```
# Check for a last name.
if (empty( $_POST[ 'last_name' ] ) )
{
	$errors[] = 'Enter your last name.' ;
	}
else
{
	$ln = mysqli_real_escape_string( $dbc, trim( $_POST[ 'last_name' ] ) ) ;
}
```

Gebruikers registreren

6 Voeg instructies toe om een foutmelding op te slaan als het veld voor het e-mailadres leeg blijft, of sla de waarde ervan op in een variabele.

```
# Check for an email address:
if ( empty( $_POST[ 'email' ] ) )
{
	$errors[] = 'Enter your email address.';
}
else
{
	$e = mysqli_real_escape_string( $dbc, trim( $_POST[ 'email' ] ) ) ;
}
```

7 Voeg instructies toe om de wachtwoord in een variabele op te slaan als de wachtwoorden een geldige overeenkomst hebben, of sla een foutmelding op als ze niet overeenkomen of als het eerste veld leeg is.

```
# Check for a password and matching input passwords.
if ( !empty($_POST[ 'pass1' ] ) )
{
	if ( $_POST[ 'pass1' ] != $_POST[ 'pass2' ] )
	{
		$errors[] = 'Passwords do not match.' ;
	}
	else
	{
		$p = mysqli_real_escape_string( $dbc, trim( $_POST[ 'pass1' ] ) ) ;
	}
}
else
{
	$errors[] = 'Enter your password.' ;
}
```

8 Voeg instructies toe om een foutmelding op te slaan als het e-mailadres al is geregistreerd in de databasetabel 'users'.

```
# Check if email address already registered.
if ( empty( $errors ) )
{
	$q = "SELECT user_id FROM users WHERE email='$e'" ;
	$r = @mysqli_query ( $dbc, $q ) ;
	if ( mysqli_num_rows( $r ) != 0 ) $errors[] = 'Email address already
registered. <a href="login.php">Login</a>' ;
}
```

Gebruikers registreren

9 Voeg nu instructies in om de gebruikersgegevens in de databasetabel op te slaan, een bevestiging weer te geven wanneer de registratie slaagt, de databaseverbinding te sluiten, een voettekst toe te voegen en het script te sluiten.

```
# On success register user inserting into 'users' database table.
if ( empty( $errors ) )
{
	$q = "INSERT INTO users (first_name, last_name, email, pass, reg_date)
	VALUES ('$fn', '$ln', '$e', SHA1('$p'), NOW() )";
	$r = @mysqli_query ( $dbc, $q ) ;
	if ($r)
	{
		echo '<h1>Registered!</h1><p>You are now
	registered.</p><p><a href="login.php">Login</a></p>';
	}

	# Close database connection.
	mysqli_close($dbc);

	# Display footer section and quit script:
	include ('includes/footer.html');
	exit();
}
```

10 Voeg alternatieve instructies toe om alle opgeslagen foutmeldingen weer te geven wanneer registratie niet kan worden voltooid en sluit de databaseverbinding.

```
# Or report errors.
else
{
	echo '<h1>Error!</h1><p id="err_msg">The following error(s)
occurred:<br>' ;
	foreach ( $errors as $msg )
	{
		echo " - $msg<br>" ;
	}
	echo 'Please try again.</p>';

	# Close database connection.
	mysqli_close( $dbc );
}
```

11 Voeg nu het HTML-sticky-formulier toe dat dit script gebruikt.

```
<!-- Display body section with sticky form. -->
<h1>Register</h1>
<form action="register.php" method="post">
<p>
	First Name: <input type="text" name="first_name" size="20"
value="<?php if (isset($_POST['first_name'])) echo
$_POST['first_name']; ?>">
	Last Name: <input type="text" name="last_name" size="20"
value="<?php if (isset($_POST['last_name'])) echo $_POST['last_name'];
?>">
</p>
<p>
	Email Address: <input type="text" name="email" size="50"
value="<?php if (isset($_POST['email'])) echo $_POST['email']; ?>">
</p>
<p>
	Password: <input type="password" name="pass1" size="20"
value="<?php if (isset($_POST['pass1'])) echo $_POST['pass1']; ?>" >
	Confirm Password: <input type="password" name="pass2"
size="20" value="<?php if (isset($_POST['pass2'])) echo
$_POST['pass2']; ?>">
</p>
<p>
	<input type="submit" value="Register">
</p>
</form>
```

12 Voeg een laatste scriptinstructie toe om een voettekst toe te voegen en sla het script vervolgens op in de map **/htdocs** als **register.php**.

```
<?php

# Display footer section.
include ( 'includes/footer.html' ) ;

?>
```

Gebruikers registreren

Verwerking registraties

Test het registratieproces om ervoor te zorgen dat de validatiecontroles correct worden uitgevoerd en het PHP-script de juiste bevestigingsberichten voor de gebruiker bevat.

1 Voert de URL **http://localhost/registrer.php** in het adresveld van een web browser in om Abyss de registratiepagina te tonen in de browser.

2 Klik nu op de knop "Registreren" om berichten te zien verschijnen die adviseren dat veldvalidatie voor elke invoer is mislukt.

3 Vul vervolgens alle invoervelden in en zorg ervoor dat u identieke wachtwoordwaarden invoert. Klik vervolgens op de knop "Registreren" om een bericht te zien waarin wordt geadviseerd dat de registratie is gelukt en een hyperlink naar een inlogpagina geeft.

4 Keer terug naar de registratiepagina en voer identieke gebruikersgegevens in zoals eerder, klik vervolgens op de knop "Registreren" om een bericht te zien verschijnen waarin wordt geadviseerd dat uw e-mailadres is geregistreerd en een hyperlink naar een inlogpagina geeft.

Een inlogpagina maken

Nadat het gebruikersregistratieproces voltooid is, kan nu de inlogpagina gemaakt worden waar de gebruiker nu toegang tot heeft door de hyperlinks te volgen die bij de registratie werden getoond, of door de inlogpagina rechtstreeks aan te spreken.

1 Start een teksteditor en begin een PHP-script met instructies om de paginatitel in te stellen en een paginakop op te nemen.

```
<?php # DISPLAY COMPLETE LOGIN PAGE.

# Set page title and display header section.
$page_title = 'Login' ;
include ( 'includes/header.html' ) ;
```

Gebruikers registreren

2 Voeg nu een voorwaardelijke test toe die eventuele foutmeldingen weergeeft als deze bestaan en geef een hyperlink terug naar de registratiepagina wanneer de inlogpoging mislukt.

```
# Display any error messages if present.
if ( isset( $errors ) && !empty( $errors ) )
{
	echo '<p id="err_msg">Oops! There was a problem:<br>' ;
	foreach ( $errors as $msg )
	{
		echo " - $msg<br>" ;
	}
	echo 'Please try again or <a href="register.php">Register</a>
</p>' ;
}
?>
```

3 Voeg vervolgens het HTML-formulier toe dat de gebruiker zijn of haar aanmeldingsgegevens zal verzenden om te worden verwerkt door een PHP-script.

```
<!-- Display body section. -->
<h1>Login</h1>
<form action="login_action.php" method="post">
	<p>Email Address: <input type="text" name="email"></p>
	<p>Password: <input type="password" name="pass"></p>
	<p><input type="submit" value="Login" ></p>
</form>
```

4 Voeg een laatste scriptinstructie toe om een voettekst toe te voegen en sla het script vervolgens op in de map **/htdocs** als **login.php**.

```
<?php

# Display footer section.
include ( 'includes/footer.html' ) ;

?>
```

5 Klik op de hyperlink op de registratiepagina of voer de locatie **http://localhost/login.php** in het adresveld van de webbrowser in om Abyss de inlogpagina te laten verschijnen.

Gebruikers registreren

Aanmeldgereedschap geven

Nadat u het gebruikersaanmeldingsformulier hebt ingevuld, kunt u nu de aanmeldingshulpprogramma's maken waarmee aanmeldingsaanvallen worden gevalideerd en waar nodig een nieuwe pagina wordt geladen. Deze hulpmiddelen kunnen eenvoudig beschikbaar worden gemaakt voor andere pagina's op de website door ze in een PHP **require** instructie te plaatsen waarin dit script wordt genoemd.

1 Start een teksteditor en begin een PHP-script met een functieblok om een pagina te laden die als het argument is opgegeven.

```
<?php # LOGIN HELPER FUNCTIONS.

# Functie om opgegeven of standaard URL te laden.
function load( $page = 'login.php' )
{
	# Instructies van stappen 2-4 hier invoeren
}
```

2 Voeg in het **load** functie blok een instructie toe om een URL string of protocol, huidig domain en directory samen te stellen.

```
# Begin de URL met het protocol, het domein en de huidige map.
$url='http://' . $_SERVER['HTTP_HOST'] . dirname( $_SERVER['PHP_SELF'] ) ;
```

3 Voeg vervolgens in het **load** functie blok instructies toe om eventuele achterliggende *slashes* uit de URL string te verwijderen en voeg vervolgens een *forward slash* toe en het opgegeven pagina als argument.

```
# Verwijder achterste slashes en voeg de paginanaam toe aan de URL.
$url = rtrim( $url, '/\\' ) ;
$url .= '/' . $page ;
```

4 Ten slotte voeg in het load functie blok instructies toe om de opgegeven pagina te laden en sluit u het script af.

```
# Voer de omleiding uit en sluit af.
header( "Location: $url" ) ;
exit() ;
```

Gebruikers registreren

5 Voeg een functieblok toe om de inloggegevens van de gebruiker te valideren als correct of geef foutmeldingen.

```
# Functie om e-mailadres en wachtwoord te controleren.
function validate( $dbc, $email = '', $pwd = '')
{
	# Instructies van stappen 6-10 hier invoeren
}
```

6 Voeg in het validatie-functieblok een instructie toe om een array te initialiseren voor foutberichten.

```
# Initialiseer fouten array.
$errors = array() ;
```

7 Voeg vervolgens instructies toe om een foutmelding op te slaan als het e-mailveld leeg blijft, of sla de waarde ervan op in een variabele.

```
# Controleer het e-mailveld
if ( empty( $email ) )
{
	$errors[] = 'Enter your email address.' ;
}
else
{
	$e = mysqli_real_escape_string( $dbc, trim( $email ) ) ;
}
```

8 Voeg nu instructies in om een foutmelding op te slaan als het wachtwoordveld leeg blijft, of sla de waarde ervan op in een variabele.

```
# Controleer het wachtwoordveld.
if ( empty( $pwd ) )
{
	$errors[] = 'Enter your password.' ;
}
else
{
	$p = mysqli_real_escape_string( $dbc, trim( $pwd ) ) ;
}
```

9 Voeg instructies toe om een foutmelding op te slaan als de e-mail en het wachtwoord niet in de database tabel worden gevonden, of het bijbehorende *user-id*, *first name* en *last name* aan de *aanroepende* doorgeven - als de inlogpoging slaagt.

```
# Op succes ophalen user_id, first_name en achternaam van 'gebruikers'
database.
if ( empty( $errors ) )
{
	$q = "SELECT user_id, first_name, last_name FROM users WHERE
email='$e' AND pass=SHA1('$p')" ;
	$r = mysqli_query ( $dbc, $q ) ;
	if ( @mysqli_num_rows( $r ) == 1 )
	{
		$row = mysqli_fetch_array ( $r, MYSQLI_ASSOC ) ;
		return array( true, $row ) ;
	}
	# Of bij falen foutmelding toeveoegen.
	else
	{
		$errors[] = 'Email address and password not found.' ;
	}
 }
```

10 Voeg ten slotte een instructie in het validatie-functieblok toe om de lijst met foutmeldingen terug te sturen naar de *aanroepende* - als de inlogpoging mislukt.

```
# Bij falen ophalen van foutmelding / en
return array( false, $errors ) ;
```

11 Zorg ervoor dat dit script nu wordt opgeslagen in de directory **/htdocs** van de webserver als **login_tools.php**.

Aanmeldpogingen verwerken

Nadat u het aanmeldingsformulier en de aanmeldingshulpmiddelen hebt voltooid, maakt u nu het PHP-script om inlogpogingen daadwerkelijk te verwerken.

Wanneer een aanmeldingspoging slaagt, stelt dit script gebruikersgegevens die uit de database zijn opgehaald in als ***sessiegegevens*** - zodat de ingelogde gebruiker eenvoudig opnieuw kan worden herkend terwijl hij/zij tussen andere pagina's van dezelfde website navigeerd.

Gebruikers registreren

1 Start een teksteditor en begin een PHP-script met een testblok om te zien of het inlogformulier is ingediend.

```
<?php # PROCESS LOGIN ATTEMPT.

# Controleer ingediend formulier
if ( $_SERVER[ 'REQUEST_METHOD' ] == 'POST' )
{
	# Instructies van stappen 2-6 hier invoeren
}
```

2 Voeg nu een instructie in om de databaseverbinding te openen.

```
# Open database connectie.
require ( '../connect_db.php' ) ;
```

3 Voeg vervolgens een statement toe om de login-tools beschikbaar te maken.

```
# Maak verbinding, laad en valideer functies.
require ( 'login_tools.php' ) ;
```

4 Voeg een verklaring in om ervoor te zorgen dat aanmelding gelukt is en verkrijg de bijbehorende gebruikersgegevens van id, voornaam en achternaam.

```
# Controleer login.
list ( $check, $data ) = validate ( $dbc, $_POST[ 'email' ], $_POST[ 'pass' ] ) ;
```

5 Voeg nu instructies in om de gebruikersgegevens als sessiegegevens in te stellen en een startpagina te laden of om foutmeldingen toe te wijzen.

```
# Op succes set sessiegegevens en toon ingelogde pagina.
if ( $check )
{
	# Access session.
	session_start();
	$_SESSION[ 'user_id' ] = $data[ 'user_id' ] ;
	$_SESSION[ 'first_name' ] = $data[ 'first_name' ] ;
	$_SESSION[ 'last_name' ] = $data[ 'last_name' ] ;
	load ( 'home.php' ) ;
}
```

Gebruikers registreren

```
# Of bij falen set errors.
else
{
        $errors = $data;
}
```

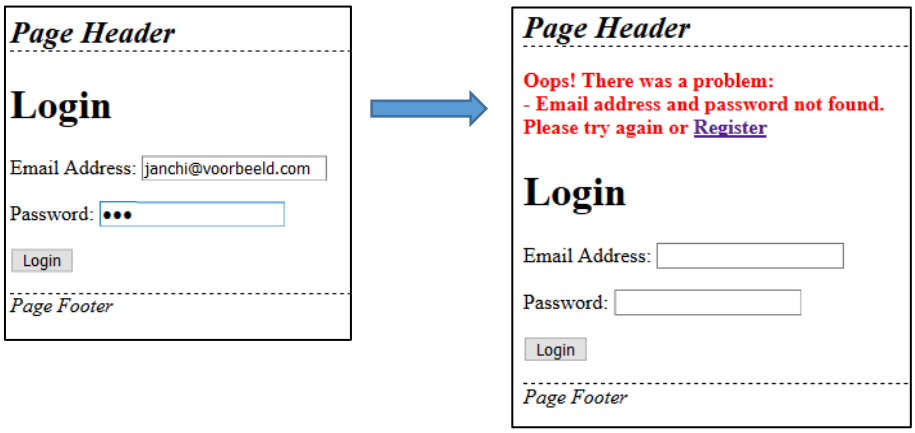
6 Voeg vervolgens een instructie in om de databaseverbinding te sluiten.

```
# Sluit database verbinding af.
mysqli_close( $dbc ) ;
```

7 Voeg ten slotte na het testblok een verklaring toe om door te gaan met het weergeven van de inlogpagina wanneer de inlogpoging mislukt.

```
# Blijf doorgaan met het tonen van de inlogpagina bij een storing.
include ( 'login.php' ) ;

?>
```

8 Bewaar dit script in de map webserver **/htdocs** als **login_action.php** en voer vervolgens de inlogpagina-locatie **http://localhost/login.php** in in het adresveld van uw web browser en probeer in te loggen met onjuiste gebruikersgegevens.



Inlogsucces bevestigen

Nu het PHP-script voltooid is om inlogpogingen daadwerkelijk te verwerken, kan nu de startpagina (*home page*) gemaakt worden die moet worden geladen wanneer de aanmeldingspoging van de gebruiker slaagt. Deze bevat hyperlinks om de gebruiker toe te staan te navigeren tussen andere pagina's van de website.

1 Start een teksteditor en begin een PHP-script met een instructie om toegang tot sessiegegevens toe te staan.

```
<?php # DISPLAY COMPLETE LOGGED IN PAGE.

# Access session.
session_start() ;
```

2 Voeg uitspraken toe om de browser om te leiden naar de inlogpagina als de gebruiker nog niet is ingelogd.

```
# Omleiden indien niet ingelogd.
if ( !isset( $_SESSION[ 'user_id' ] ) ) { require ( 'login_tools.php' ) ; load() ; }
```

3 Voeg vervolgens instructies toe om de paginatitel in te stellen en een paginakop toe te voegen.

```
# Stel een paginatitel en een koptekstgedeelte in.
$page_title = 'Home' ;
include ( 'includes/header.html' ) ;
```

4 Voeg uitspraken toe om hyperlinks naar andere pagina's op de website te maken.

```
# Toon body sectie.
echo "<h1>HOME</h1><p>You are now logged in, {$_SESSION['first_name']}
{$_SESSION['last_name']} </p>";
```

5 Voeg uitspraken toe om hyperlinks naar andere pagina's op de website te maken.

```
# Maak navigatie links.
echo '<p><a href="forum.php">Forum</a> | <a href="shop.php">Shop</a> |
<a href="goodbye.php">Logout</a></p>';
```

6 Voeg een laatste verklaring toe om een paginavoettekst op te nemen.

```
# Toon voettekst sectie.
include ( 'includes/footer.html' ) ;
?>
```

7 Sla dit script op in de directory van uw webserver **/htdocs** als **home.php** en voer vervolgens in de adresbalk van uw webbrowser het adresadres **http://localhost/login.php** in en probeer in te loggen met de juiste gebruikersgegevens.

Folder structuur

De bestanden in de Abyss Web Server moeten op de manier van de afbeelding hierboven worden opgeslagen. Alle volgende voorbeelden gaan uit van deze structuur.

Verbinding maken

Voor het maken van een verbinding met de MySQL Server is er een standaard script.

1 Start een tekst editor, zoals '*notepad*' of 'notepad++' en type daar de onder staande script in.

```
<?php
$dbc = mysqli_connect('<host>', '<user>','<wachtwoord>', '<database>')
Or die ( mysqli_connect_error() );
mysqli_set_charset( $dbc, 'utf-8');
```

2 Sla het bestand op als **connect_db.php** in de hoofd map van de Abyss Web Server zoals in de afbeelding is weergegeven.

3 Start een tekst editor, zoals '*notepad*' of 'notepad++' en type daar de onder staande script in.

```
<?php
require ( '../connect_db.php' ) ;

if ( mysqli_ping( $dbc ) )
{
        echo 'MySQL Server ' . mysqli_get_server_info ($dbc) .' on ' .
        mysqli_get_host_info ($dbc) ;
}
```

4 Sla het bestand op als **require.php** in de hoofd map van de Abyss Web Server zoals in de afbeelding is weergegeven. Voer het script uit met localhost. Het resultaat in de browser is dan als volgt:

MySQL Server 8.0.12 on localhost via TCP/IP

Opmerking:
Bestanden die alleen pure PHP-code bevatten, zoals die hierboven vermeld, behoren de PHP **?>** Afsluitende tag aan het einde van het bestand weg te laten. Dit voorkomt onbedoelde witruimte of nieuwe regels na de PHP-sluitingstag, die ongewenste effecten kan veroorzaken.

Zoekopdrachten uitvoeren

Zodra een PHP-script verbinding heeft gemaakt met een database op de MySQL Server, kunnen query's worden uitgevoerd met behulp van de PHP **mysqli_query() functie**, die deze syntaxis heeft:

```
mysqli_query ( database-connectie-object, sql-query)
```

Wanneer de functie **mysqli_query()** wordt aangeroepen, wordt een "*result set*" of een **TRUE**-waarde geretourneerd waar de aanroep succesvol is, of een **FALSE**-waarde wanneer deze mislukt.

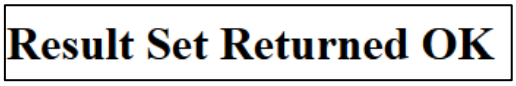
Optioneel kan in een script een beroep gedaan worden op **mysql_close()** om de verbinding expliciet te sluiten nadat de query is uitgevoerd. Dit gebeurt echter automatisch aan het einde van het script, maar het maken van deze functie aanroep is een goede gewoonte.

1 Start een tekst editor, zoals '*notepad*' of 'notepad++' en type daar de onder staande script in.

```
<?php # CONNECT TO MySQL DATABASE.

require( '../connect_db.php' ) ;

# Create a MySQL query.
$q = 'SHOW TABLES' ;

# Execute the query.
$r = mysqli_query( $dbc , $q ) ;

# Show results.
if( $r )
```

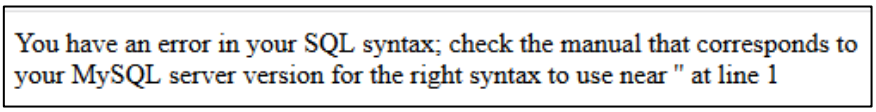
```
{
  echo '<h1>Result Set Returned OK</h1>' ;
}
else
{
  echo '<p>' . mysqli_error( $dbc ) . '</p>' ;
}

# Close the connection.
mysqli_close( $dbc ) ;
```

2 Sla het bestand op als **query.php** in /htdocs. Voer het script uit met localhost in een browser om de bevestiging te zien dat de querry succesvol is uitgevoerd.



3 Bewerk nu de SQL-querytekenreeks om deze foutief te maken: $q = 'SHOW TABLE';

3 Sla het bewerkte PHP-script nogmaals op en open het opnieuw in een browser om het bericht te zien waarin de queryfout wordt beschreven.



Resultaten ophalen

Sommige SQL-query's, zoals **INSERT**, **UPDATE**, **DELETE** of **ALTER**, retourneren geen gegevens uit de database. Dus wanneer deze met succes gemaakt zijn van een PHP-script, retourneert de **mysqli_query()** -functie eenvoudigweg een **TRUE**-waarde.

Andere SQL-query's, zoals **SHOW** of **SELECT**, retourneren een "*result-set*" wanneer ze met succes zijn gemaakt vanuit PHP.

Een *resultaatenset* bevat rijen records die regel-voor-regel kunnen worden gelezen met behulp van een **while**-lus. Elke rij kan als een array worden geretourneerd met behulp van de functie **mysqli_fetch_array()**, waarvoor de *resultatenset*-variabele als eerste argument moet worden opgegeven.

Optioneel, kan het een tweede variabele vragen om op te geven welk type array moet worden geretourneerd: een associatieve array (met behulp van kolomnamen), een numerieke array of beide.

Gebruikers registreren

Handig genoeg kan het arraytype worden opgegeven met behulp van een van de ingebouwde PHP-constante-waarden die hieronder worden vermeld:

Constante:	Voorbeeld:
MYSQLI_ASSOC	$row ['kolom-naam']
MYSQLI_NUM	$row [0]
MYSQLI_BOTH	$row ['kolom-naam'] OR $row [0]

Optioneel kan een script een beroep doen op **mysqli_free_result()** om de '*result set*' geheugen vrij te maken. Dit gebeurt automatisch aan het einde van het script, maar net als met **mysqli_close()** is het maken van deze aanroep een goede gewoonte.

1 Maak een PHP-script dat begint met het opnemen van het standaardscript om een verbinding tot stand te brengen met een database op de MySQL-server.

```
<?php # CONNECT TO MySQL DATABASE.

	require( '../connect_db.php' ) ;

	# Create a MySQL query.
	$q = 'SHOW TABLES' ;

	# Execute the query.
	$r = mysqli_query( $dbc , $q ) ;

	# Show results.
	if( $r )
	{
		echo '<h1>Tables on site_db database</h1> ' ;
		while ( $row = mysqli_fetch_array( $r , MYSQLI_NUM ) )
		{
			echo '<br>'. $row[0] ;
		 }
		mysqli_free_result( $r ) ;
	}
	else
	{
		 echo '<p>' . mysqli_error( $dbc ) . '</p>'  ;
	}

	# Close the connection.
	mysqli_close( $dbc ) ;
```

2 Sla het PHP-script op in de map **/htdocs** als **result.php** en open het in de browser met localhost om de resultaten te bekijken.

Veranderingen doorvoeren

Wijzigingen kunnen worden toegepast op een bestaande MySQL-databasetabel door SQL-query's te maken van **INSERT**, **UPDATE**, **DELETE** of **ALTER**.

1 Start een tekst editor en type de onderstaande query over.

```
CREATE TABLE IF NOT EXISTS consoles
(
  code        INT AUTO_INCREMENT PRIMARY KEY ,
  naam        VARCHAR(32) ,
  prijs       DECIMAL(6,2),
  jaar        INT
) ;

INSERT INTO consoles ( naam , prijs , jaar )
VALUES  ( "PlayStation 4" , 412.52 , 2013) ,
              ( "Xbox One" , 414.91 , 2013 ) ,
              ( "PlayStation 4 Pro" , 399.00 , 2016 ) ,
              ( "Nintendo Switch" , 299.00 , 2017 );

SELECT * FROM consoles ;
```

2 Sla het bestand op het bureaublad op als **insert_data.sql**. Start de MSCLC en log in als een gebruiker met volledige toegangsrechten, zoals root.

3 Voer vervolgens een query uit om de gevestigde database USE site_db; te kunnen gebruiken.

4 Voer nu de SQL-query uit, door het pad naar het SQL-bestand op uw computer te vermelden, zoals:

```
SOURCE C:\Users\<Gebruikersnaam>\Desktop\insert_data.sql
```

Gebruikers registreren

```
mysql> Source C:\Users\Robert\Desktop\insert_data.sql
Query OK, 0 rows affected, 1 warning (0.01 sec)

Query OK, 4 rows affected (0.04 sec)
Records: 4  Duplicates: 0  Warnings: 0

+------+------------------+--------+------+
| code | naam             | prijs  | jaar |
+------+------------------+--------+------+
|    1 | PlayStation 4    | 412.52 | 2013 |
|    2 | Xbox One         | 414.91 | 2013 |
|    3 | PlayStation 4 Pro | 399.00 | 2016 |
|    4 | Nintendo Switch  | 299.00 | 2017 |
+------+------------------+--------+------+
4 rows in set (0.00 sec)

mysql>
```

Nu dat we een tabel gemaakt en gevuld hebben met data kan gekeken worden hoe deze bewerkt kan worden doormiddel van PHP.

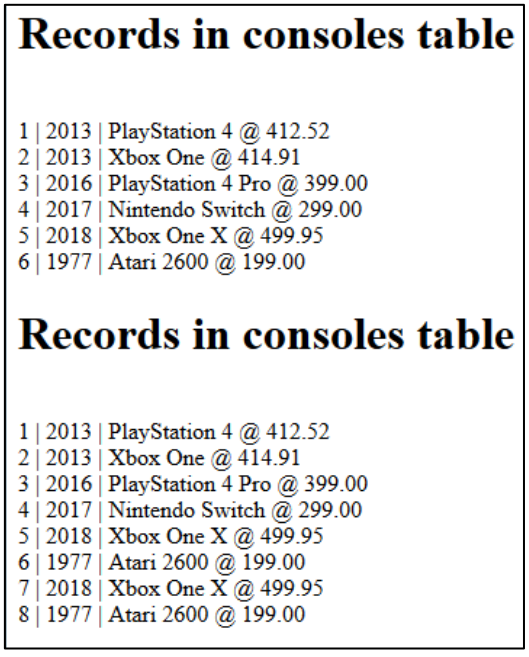
1. Maak een PHP-script dat begint met het opnemen van het standaardscript om een verbinding tot stand te brengen met een database op de MySQL-server.

```php
<?php

# CONNECT TO MySQL DATABASE.
require( '../connect_db.php' ) ;

# Function to create and execute a MySQL query.
function show_records( $dbc )
{
  $q = 'SELECT * FROM consoles' ;
  $r = mysqli_query( $dbc , $q ) ;
   if ( $r )
  {
     echo '<h1>Records in consoles table</h1> ' ;
     while ( $row = mysqli_fetch_array( $r , MYSQLI_ASSOC ) )
    {
      echo '<br>' . $row['jaar'] . $row['code'] . ' | ' . $row[ 'naam' ] .  ' @ ' .
$row[ 'prijs' ] ;
     }
  }
  else
  {
     echo '<p>' . mysqli_error( $dbc ) . '</p>' ;
   }
}

# Call the function.
show_records($dbc) ;
```

Gebruikers registreren

```
# Create and execute another MySQL query.
$q = 'INSERT INTO consoles ( naam , prijs , jaar ) VALUES ( "Xbox One X" ,
499.95 , 2018 ) , ( "Atari 2600" , 199.00 , 1977 ) ' ;
$r = mysqli_query( $dbc , $q ) ;

# Call the function again.
if( $r)
{
  show_records($dbc) ;
}
else
{
  echo '<p>' . mysqli_error( $dbc ) . '</p>' ;
}

# Close the connection.
mysqli_close( $dbc ) ;
```

2 Sla het PHP-script op in de map **/htdocs** als **insert.php** en open het in de browser met localhost om de resultaten te bekijken.



Records in consoles table

1 | 2013 | PlayStation 4 @ 412.52
2 | 2013 | Xbox One @ 414.91
3 | 2016 | PlayStation 4 Pro @ 399.00
4 | 2017 | Nintendo Switch @ 299.00
5 | 2018 | Xbox One X @ 499.95
6 | 1977 | Atari 2600 @ 199.00

Records in consoles table

1 | 2013 | PlayStation 4 @ 412.52
2 | 2013 | Xbox One @ 414.91
3 | 2016 | PlayStation 4 Pro @ 399.00
4 | 2017 | Nintendo Switch @ 299.00
5 | 2018 | Xbox One X @ 499.95
6 | 1977 | Atari 2600 @ 199.00
7 | 2018 | Xbox One X @ 499.95
8 | 1977 | Atari 2600 @ 199.00

Records tellen

Een resultaten-set geretourneerd door de ingebouwde PHP **mysqli_query()** functie kan worden onderzocht om te achterhalen hoeveel rijen het bevat met behulp van de **mysqli_num_rows()** functie.

Gebruikers registreren

De geretourneerde rij-telwaarde kan worden getest voordat de **while**-lus wordt uitgevoerd die records retourneert om ervoor te zorgen dat de resultaten-set niet leeg is.

1. Maak een PHP-script dat begint met het opnemen van het standaardscript om een verbinding tot stand te brengen met een database op de MySQL-server.

```
<?php
# CONNECT TO MySQL DATABASE.
require( '../connect_db.php' ) ;

# Function to create and execute a MySQL query.
function show_records( $dbc )
{
  $q = 'SELECT * FROM consoles' ;
  $r = mysqli_query( $dbc , $q ) ;
  $num = mysqli_num_rows( $r ) ;

   if ( $num > 0 )
  {
     echo '<h1>Records in consoles table</h1> ' ;
     while ( $row = mysqli_fetch_array( $r , MYSQLI_ASSOC ) )
    {
      echo '<br>' . $row['code'] . ' | ' . $row['jaar'] . ' | ' . $row[ 'naam' ] .
' @ ' . $row[ 'prijs' ] ;
     }
     echo "<br>$num Records" ;
  }
  else
  {
     echo '<p>' . mysqli_error( $dbc ) . '</p>'  ;
  }
}
# Call the function.
show_records($dbc) ;

# Create and execute another MySQL query.
$q = 'INSERT INTO consoles ( naam , prijs , jaar ) VALUES ( "Atari 5200"
, 269.99 , 1982 ) ';
$r = mysqli_query( $dbc , $q ) ;

# Call the function again.
if( $r)
{
   show_records($dbc) ;
}
```

```
else
{
   echo '<p>' . mysqli_error( $dbc ) . '</p>' ;
}

# Close the connection.
mysqli_close( $dbc ) ;
```

2 Sla het PHP-script op in de map **/htdocs** als **count.php** en open het in de browser met localhost om de resultaten te bekijken.

Records in consoles table

1 | 2013 | PlayStation 4 @ 412.52
2 | 2013 | Xbox One @ 414.91
3 | 2016 | PlayStation 4 Pro @ 399.00
4 | 2017 | Nintendo Switch @ 299.00
4 Records

Records in consoles table

1 | 2013 | PlayStation 4 @ 412.52
2 | 2013 | Xbox One @ 414.91
3 | 2016 | PlayStation 4 Pro @ 399.00
4 | 2017 | Nintendo Switch @ 299.00
5 | 1982 | Atari 5200 @ 269.99
5 Records

Records bijwerken

De ingebouwde PHP functie **mysqli_affected_rows()** retourneert een geheel getal dat het aantal rijen is dat wordt beïnvloed door een **INSERT**-, **UPDATE**- of **DELETE**-query. In tegenstelling tot de functie **mysqli_num_rows()**, waarbij de *resultatenset* als argument wordt gebruikt, neemt de functie **mysqli_affected_rows()** het *database-verbindingsobject* als argument aan:

mysqli_query (database-connectie-object, sql-query)

De geretourneerde rijwaarde van de aangepaste rijen kan worden getest om te verzekeren dat een **UPDATE**-query is geslaagd voor een verwacht aantal records.

1 Maak een PHP-script dat begint met het opnemen van het standaardscript om een verbinding tot stand te brengen met een database op de MySQL-server.

```
<?php # CONNECT TO MySQL DATABASE.

require( '../connect_db.php' ) ;

# Function to create and execute a MySQL query.
function show_records( $dbc )
```

Gebruikers registreren

```
{
  $q = 'SELECT * FROM consoles' ;
  $r = mysqli_query( $dbc , $q ) ;
  $num = mysqli_num_rows( $r ) ;
   if ( $num > 0 )
  {
     echo '<h1>Records in consoles table</h1> ' ;
     while ( $row = mysqli_fetch_array( $r , MYSQLI_ASSOC ) )
    {
          echo '<br>' . $row['code'] . ' | ' . $row['jaar'] . ' | ' . $row[ 'naam' ] .  '
@ ' . $row[ 'prijs' ] ;
     }
  } else { echo '<p>' . mysqli_error( $dbc ) . '</p>'  ; }
}
# Call the function.
show_records($dbc) ;

# Create and execute another MySQL query.
$q = 'UPDATE consoles SET jaar = 9999 WHERE jaar = 2013  ' ;
$r = mysqli_query( $dbc , $q ) ;

# Call the function again if 2 records updated.
if(  mysqli_affected_rows( $dbc ) == 2 )
{
  echo '<hr>' . mysqli_affected_rows( $dbc ) . ' Records Updated...' ;
  show_records($dbc) ;
 }

# Close the connection.
mysqli_close( $dbc ) ;
```

2 Sla het PHP-script op in de map **/htdocs** als **update.php** en open het in de browser met localhost om de resultaten te bekijken.

Gebruikers registreren

Resultaten valideren

PHP biedt ingebouwde functies waarmee gegevens per type kunnen worden getest voor validatie. Elk van de functies in de onderstaande tabel retourneert **TRUE** als de gegevens van een bepaald type zijn, anders keren ze **FALSE** terug.

Functie:	Validaties:	Voorbeeld:
is_numeric()	Numeriek, zelfs als string	7.0, 70, '70'
is_int()	Integer	50
is_float()	Floating-point getal	3.1415
is_string()	String	'Hallo Wereld'
is_null()	Null	Null
is_scalar()	Variable	$var
is_array()	Array	$dagen
is_bool()	Boolean	TRUE
is_resource()	External Resource	File Stream

1 Maak een PHP-script dat begint met het opnemen van het standaardscript om een verbinding tot stand te brengen met een database op de MySQL-server.

```
<?php

require ( '../connect_db.php' ) ;

$q = 'SELECT prijs FROM consoles WHERE naam = "PlayStation 4 Pro" ' ;
$r =  mysqli_query( $dbc , $q )  ;
while ($row = mysqli_fetch_array( $r , MYSQLI_NUM ) ) { $var = $row[0] ; }

$result= is_numeric( $var ) ? 'numeric' : 'not numeric';
echo "$var is $result<br>" ;

$result= is_int( $var ) ? 'an integer' : 'not an integer';
echo "$var is $result<br>" ;

$result= is_float( $var ) ? 'a float' : 'not a float';
echo "$var is $result<br>" ;

$result= is_string( $var ) ? 'a string' : 'not a string';
echo "$var is $result<br>" ;

$result= is_null( $var ) ? 'NULL' : 'not NULL';
echo "$var is $result<br>" ;

$result= is_scalar( $var ) ? 'a variable' : 'not a variable';
echo "$var is $result<br>" ;
```

```
$result= is_array( $var ) ? 'an array' : 'not an array';
echo "$var is $result<br>" ;

$result= is_bool( $var ) ? 'a boolean' : 'not a boolean';
echo "$var is $result<br>" ;

$result= is_resource( $var) ? 'a resource' : 'not a resource';
echo "$var is $result<br>" ;

mysqli_close( $dbc ) ;
```

2 Sla het PHP-script op in de map **/htdocs** als **validate.php** en open het in de browser met localhost om de resultaten te bekijken.

```
399.00 is numeric
399.00 is not an integer
399.00 is not a float
399.00 is a string
399.00 is not NULL
399.00 is a variable
399.00 is not an array
399.00 is not a boolean
399.00 is not a resource
```

Zorgen voor veiligheid

Bij het programmeren met PHP en MySQL is het belangrijk om deze beveiligingsproblemen te overwegen om te voorkomen dat gebruikers gegevens invoeren die, per ongeluk of op kwade wijze, ongewenste effecten kunnen produceren:

- Sta niet toe dat het databaseverbindingsscript, *connect_db.php,* direct toegankelijk is - plaats het in de bovenliggende map van **/htdocs**.
- Gebruik validatiefuncties om ervoor te zorgen dat gegevens bestaan en van het juiste type zijn - gebruik vaak **isset()**, **!Empty()** en **is_numeric()**.
- Controleer (*escape*) speciale tekens in een string om ze veilig te maken voor query's. Geef ze eerst door aan **mysqli_real_escape_string()**.
- Vermijd HTML-speciale tekens - verander ze in een entiteitsformaat met **htmlentities()** of verwijder ze met **strip_tags()**.

Gebruikers toestaan om gegevens te verzenden, omgeven door aanhalingstekens, kan een scriptfout veroorzaken wanneer die gegevens in string worden ingevoerd, omdat de aanhalingstekens de string voortijdig beëindigen. Bijvoorbeeld, in de opdracht **$str = "Ingediende $data"**, waarbij de **$data** variabele zelf een string tussen aanhalingstekens bevat.

Het doorgeven van de string aan de **mysqli_real_escape_string()** functie retourneert deze in een veilig formaat. Evenzo kan het problematisch zijn om gebruikers toe te staan gegevens te verzenden die

zijn omgeven door HTML-tags, maar door deze door te geven aan **strip_tags()** worden de tags verwijderd of door deze door te geven aan **htmlentities()** worden ze omgezet in een entiteitsindeling.

1 Maak een HTML-document met een formulier met één tekst-invoer-veld voor het indienen van gegevens.

```
<!DOCTYPE HTML>
<html lang="en">
        <head>
                <meta charset="UTF-8">
                <title>PHP Security</title>
        </head>
<body>

<form action="secure.php" method="POST">
        <p>Nieuwe Naam : <input type="text" name="naam"><input
        type="submit"></p>
</form>
```

2 Voeg een script toe dat een item uit een bestaande tabel weergeeft.

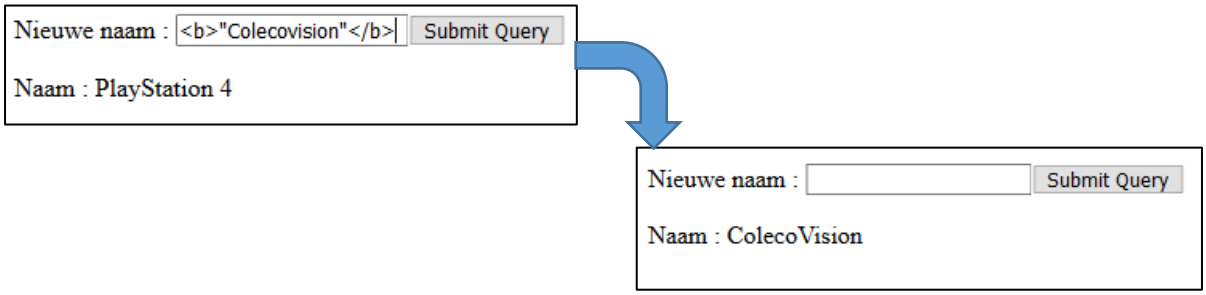
```
<?php
# Connect to MySQL.
require ( '../connect_db.php' ) ;

# Validate field is not empty and not numeric.
if ( !empty( $_POST['naam'] )  && !is_numeric( $_POST[ 'naam'] ) )
{
 $naam = $_POST['naam']  ;

 # Make string safe for queries by allowing quotes.
 $naam = mysqli_real_escape_string( $dbc , $naam ) ;

 # Disallow HTML tags.
$naam = strip_tags( $naam ) ;

 # Execute update.
 $q = 'UPDATE consoles SET naam = "' . $naam . '" WHERE code = 1' ;
 mysqli_query( $dbc , $q )  ;
}
else
{
   echo 'Geen valide nieuwe naam ingevoerd' ;
}

$q = 'SELECT * FROM consoles WHERE code = 1 ' ;
$r =  mysqli_query( $dbc , $q )  ;
while ($row = mysqli_fetch_array( $r , MYSQLI_NUM) )
{
   echo  "<p>Naam : $row[1] </p>" ;
```

```
}
mysqli_close( $dbc ) ;
?>
```

3 Eindig het script met de eind tags van HTML

```
        </body>
</html>
```

3 Sla het PHP-script op in de map **/htdocs** als **secure.php** en open het in de browser met localhost om de resultaten te bekijken.



Omgaan met fouten

PHP biedt nuttige foutmeldingen wanneer het problemen ondervindt met een script - beschrijft zowel runtime- als syntax fouten.
Bovendien kunt u uw eigen aangepaste fouten afhandelingsprogramma maken om berichten te tonen voor scriptproblemen die kunnen optreden.

Elke fout die optreedt, kan worden gedefinieerd door een ingebouwde PHP-constante die een unieke numerieke waarde heeft, zoals die in de onderstaande tabel:

Waarde:	Constanten:	Omschrijving:
1	E_ERROR	Fatale runtime-fout, waardoor de uitvoering van het script wordt gestopt.
2	E_WARNING	Niet-fatale runtime-fout, die het script niet stopt.
4	E_PARSE	Syntax fout, die voorkomt dat de Interpreter het script uitvoert.
8	E_NOTICE	Mogelijke runtime-fout, die eenvoudig een kennisgevingstip biedt.
256	E_USER_ERROR	Fatale user-generated error ingesteld door **trigger_error()** functie.
512	E_USER_WARNING	Niet-fatale door de gebruiker gegenereerde fout, ingesteld door **trigger_error()** functie.
1024	E_USER_NOTICE	Door de gebruiker gegenereerde kennisgeving, ingesteld door de **trigger_error()** functie.

Gebruikers registreren

Door de gebruiker gegenereerde kennisgeving, ingesteld door de **trigger_error()** functie van PHP. Fouten genereren vijf beschrijvende componenten die als argumenten kunnen worden doorgegeven aan de fouten behandelaar – *error level* (foutenniveau) , *message* (bericht), *file* (bestand), *line* (regel) en *context*. Het *error level* en *message* zijn verplicht.

1 Maak een HTML-document met een PHP-script dat begint met het benoemen van een aangepaste fouten behandelaar functie.

```
<!DOCTYPE HTML>
<html lang="en">
	<head>
		<meta charset="UTF-8">
		<title>PHP Error Handling</title>
	</head>
<body>

<?php

	set_error_handler( 'error_handler' ) ;

	function error_handler ( $level , $message , $file , $line )
	{
		echo "Error [$level] : $message <br>(Check line $line in $file
		)"  ;
	}

	$num = 500 ;
	if ( $num > 100 )
	{
		trigger_error( 'Nummer moet kleiner zijn dan 100' ,
		E_USER_ERROR ) ;
	}
?>
</body>
</html>
```

2 Sla het PHP-script op in de map **/htdocs** als **error.php** en open het in de browser met localhost om de resultaten te bekijken.