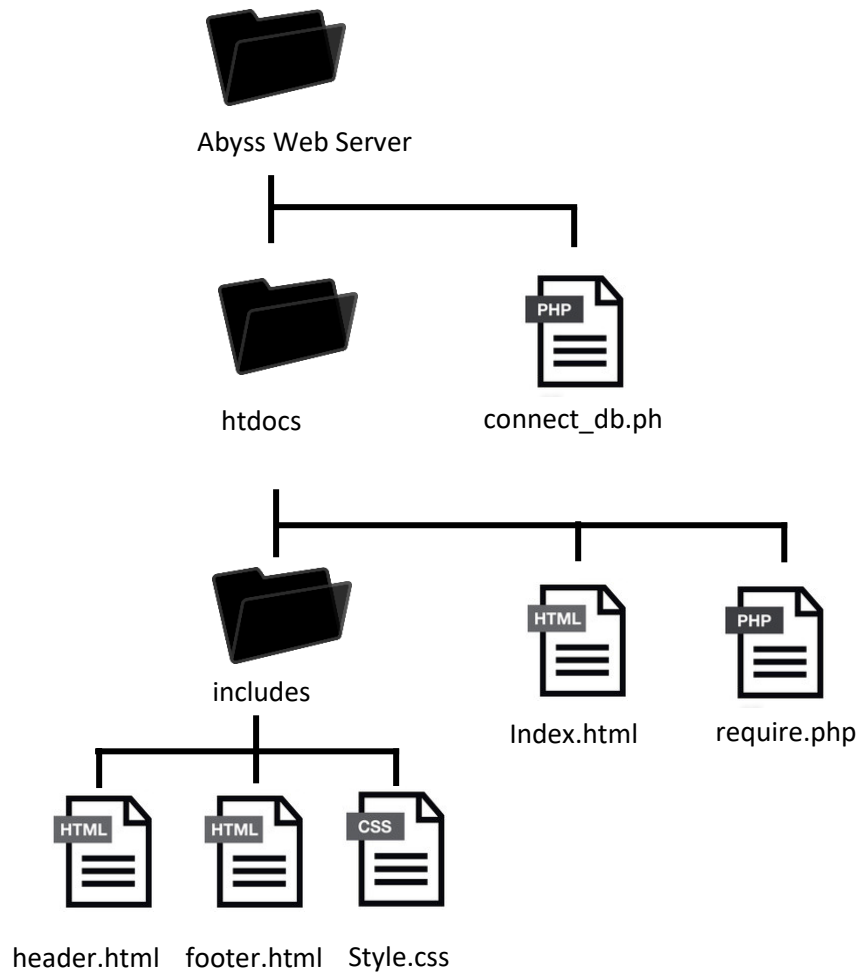
Folder structuur



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.

Data Manipulation Language (DML) with PHP

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.

Result Set Returned OK

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.

You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the right syntax to use near '' at line 1

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.

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.

Data Manipulation Language (DML) with 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 # 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**.

Data Manipulation Language (DML) with PHP

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
```

```
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>
```

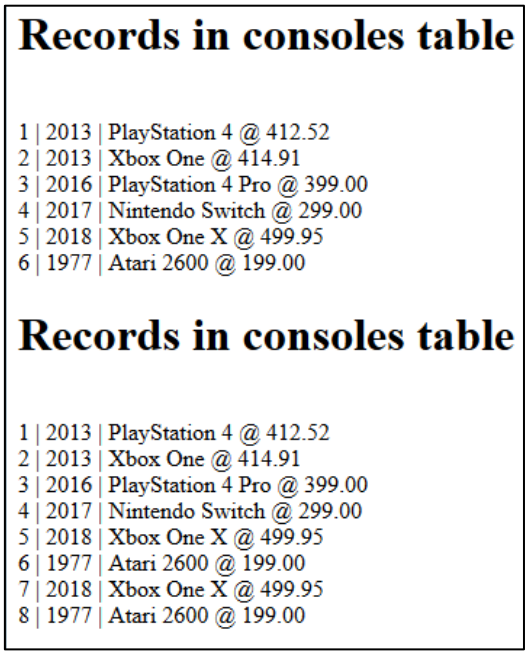
Nu dat we een tabel gemaakt en gevuld hebben met data kan gekeken worden hoe deze bewerkt kan worden doormiddel van PHP.

Data Manipulation Language (DML) with 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

# 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) ;

# 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.

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 )
{
  $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.

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

Data Manipulation Language (DML) with 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

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>
```

Data Manipulation Language (DML) with PHP

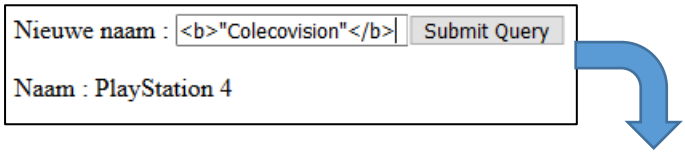
2 Voeg een script toe dat een item uit een bestaande tabel weergeeft.

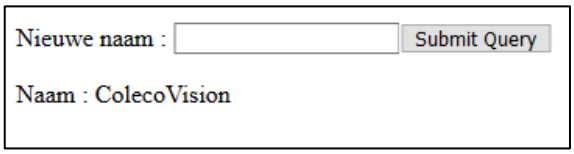
```
<?php
# Connect to MySQL.
require ( '../connect_db.php' ) ;

# 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.

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.