Lecture 3
2025/2026

Databases, Web Programming and Interfacing

DWPI

- Databases, Web Programming and Interfacing
  - An VI IT4T
    - 1C/1L/1P
- Timetable
  - Friday, every week (fiecare saptamana) 1C + 2L (17-20)
  - Test + Project Presentation: **last week** (Weekend)

Grade

- 10% - Test/Exam – last week – 1h
- 40% - Project
  - Personal(80%)/Team(20%)
- Previous database project
  - receives a **web user interface**
  - **split** in individual assignments
  - bonus (individual) for **supplemental embedded systems access!**
    - admin/user – web user interface
    - data supplier – embedded system (even partial)

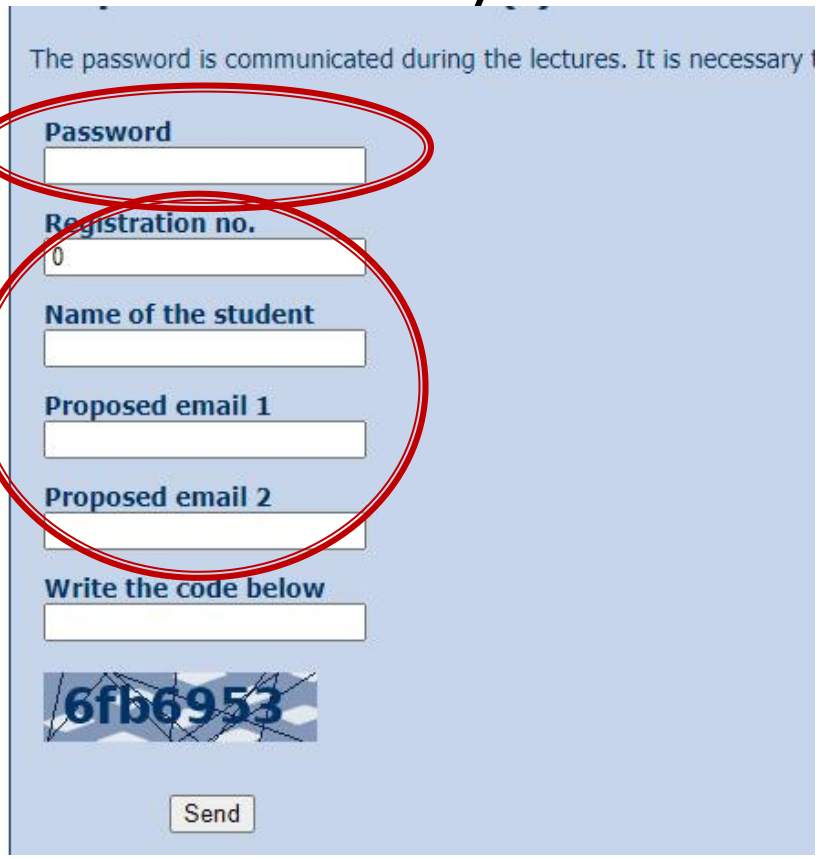
Info

- https://rf-opto.etti.tuiasi.ro/master_it.php



Online – Registration no.

- access to **online exams** requires the **password** received by email

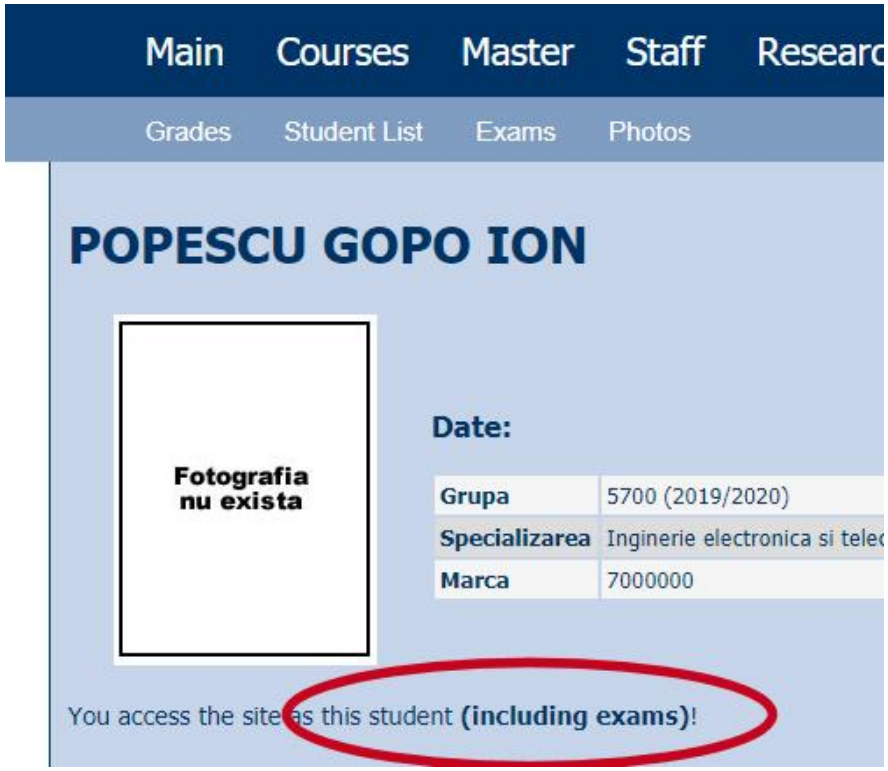




Online

- access email/password

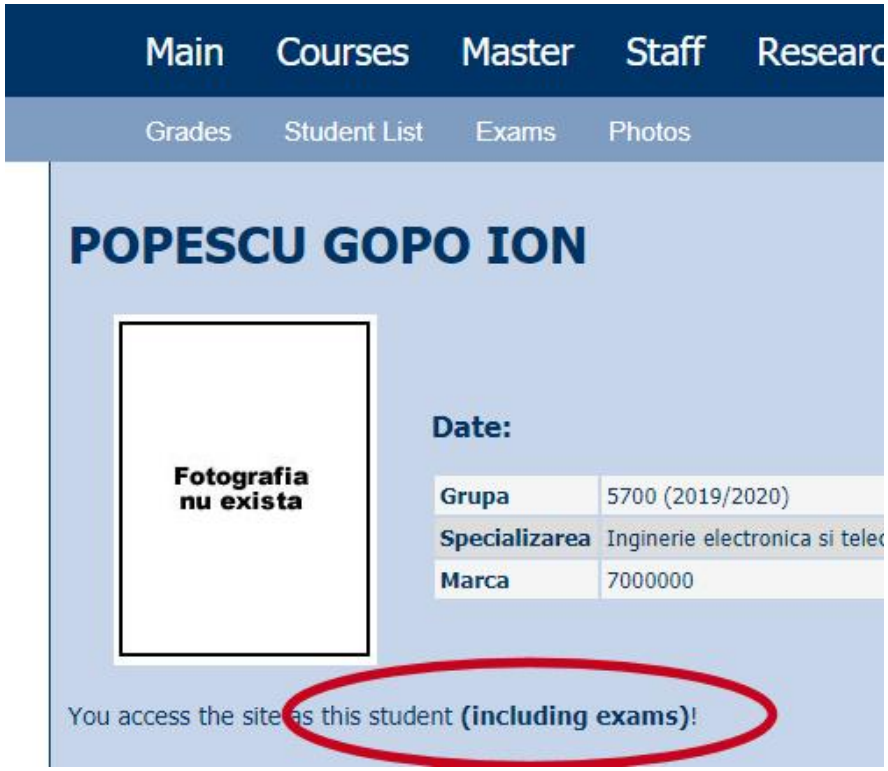




Online

- access email/password





Password

- received by email

Important message from RF-OPTO Inbox ×

Radu-Florin Damian
to me, POPESCU

Romanian › English Translate message

Laboratorul de Microunde si Optoelectronica
Facultatea de Electronica, Telecomunicatii si Tehnologia Informatiei
Universitatea Tehnica "Gh. Asachi" Iasi

In atentia: POPESCU GOPO ION

Parola pentru a accesa examenele pe server-ul **rf-opto** este
Parola: [redacted]

Identificati-va pe server, cu parola, cat mai rapid, pentru confirmare.

Memorati acest mesaj intr-un loc sigur, pentru utilizare ulterioara

---

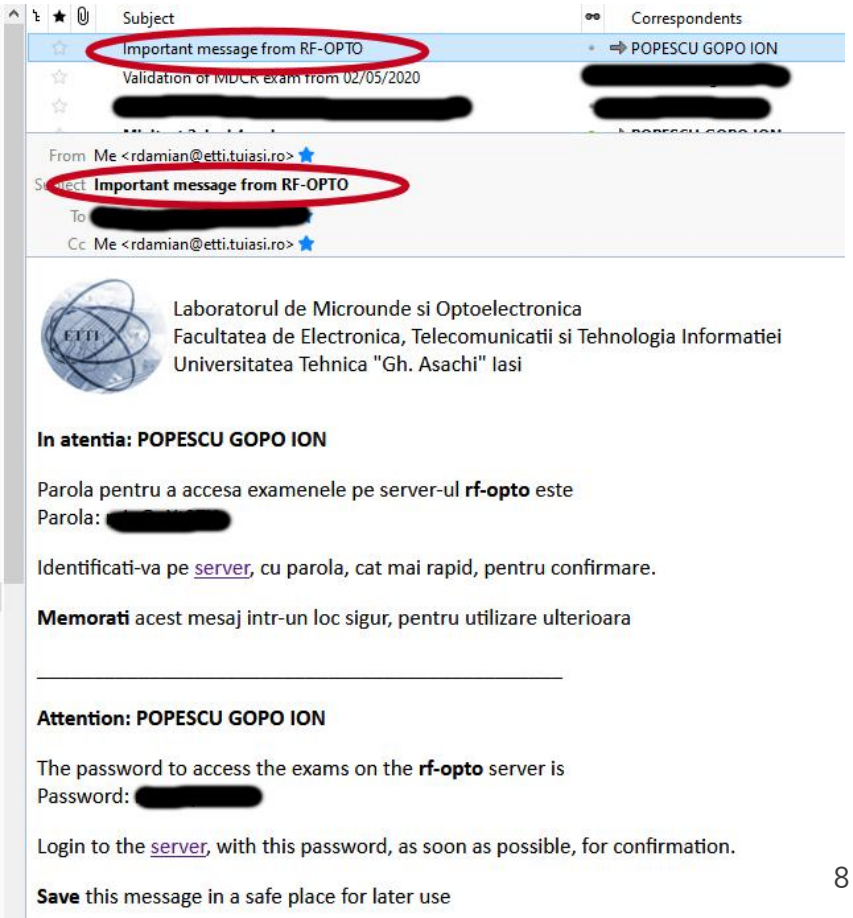
Attention: POPESCU GOPO ION

The password to access the exams on the **rf-opto** server is
Password: [redacted]

Login to the server, with this password, as soon as possible, for confirmation.

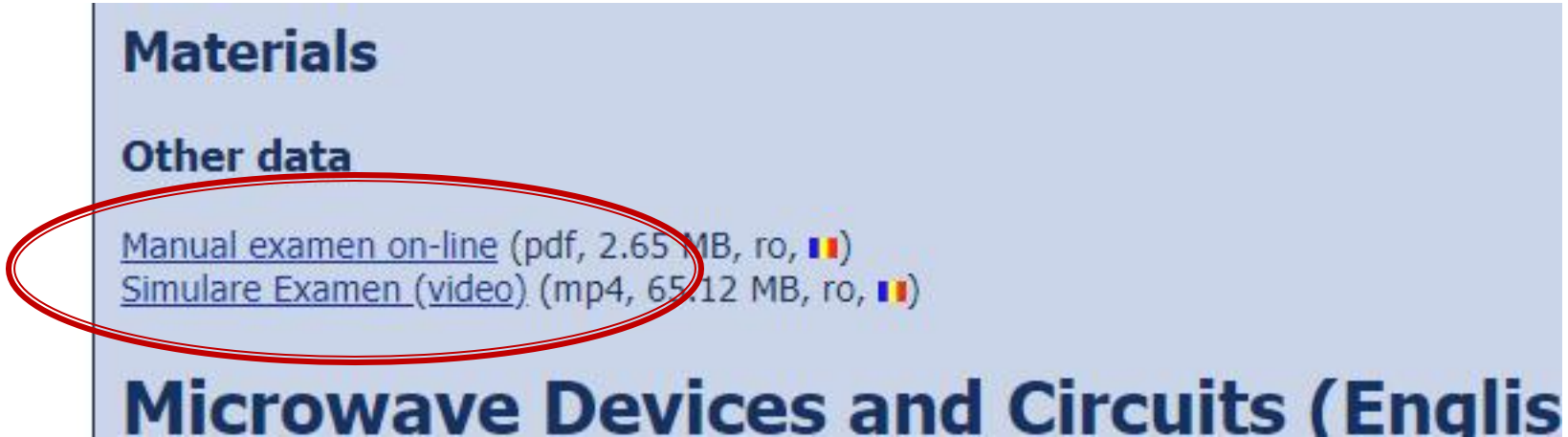
Save this message in a safe place for later use

Reply Reply all Forward



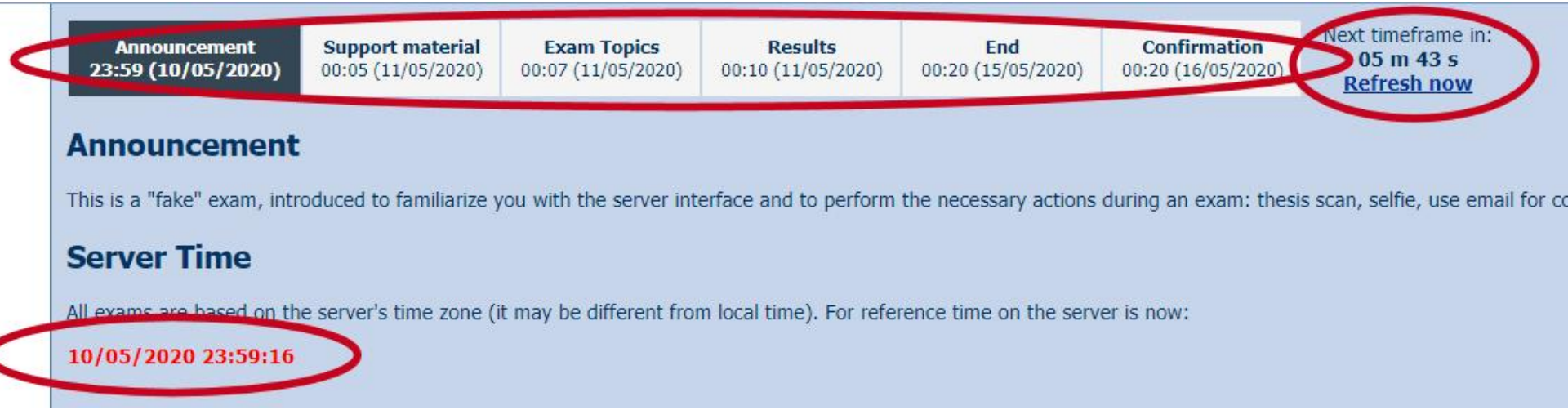
Manual examen online

- The online exam app used for:
  - lectures (attendance)
  - laboratory
  - project
  - examinations



Examen online

- always against a **timetable**
  - long period (project submission/laboratory results)
  - short period (tests: 15min, exam: 2h)



Introduction

Necessity

```
        <td><?php echo  $row['Documente'];?> </td>
        <td><?php echo $row['user_creat'];?> </td>
        <td class="smaller"><a href="control_lot.php?id=-<?php echo $row['ID_LOT'];?>">dezactiveaza</a><br /><a href=
"control_lot.php?id=<?php echo $row['ID_LOT'];?>">modifica</a></td>
    </tr>
  <?php $index++;
    } while ( $row = mysql_fetch_assoc($result));?>
</table><?php
}
else
    {
    echo "<p>Nu exista loturi active</p>";
    }
?>

<p class="title">Loturi inactive</p>
<?php
$query = "SELECT l.*, c.`nume_user` AS `user_creat`
        FROM `lot` AS l
        LEFT JOIN `users` AS c ON (l.`User`=c.`id_user`)
        WHERE l.`Activ` = 0 ORDER BY l.`ID_LOT` DESC";
$result = mysql_query($query);
$total=0;
if ($result && (mysql_num_rows($result) > 0))
    {
    $total=mysql_num_rows($result);
    $row = mysql_fetch_assoc($result);
    }
if ($total>0)
{?>
<table align="center">
  <tr class="lista_titlu">
```

HTML

E:\Documents\Pagini Web\Safir dep\lista_lot.php - Notepad++

File Edit Search View Encoding Language Settings Tools Macro Run Plugins Win

logfile.php | lista_lot.php

```
539 <?php }?>
540
541 <div id="maincontent"><!-- InstanceBeginEditable name="Conti
542 <p><span class="title">Adauga</span><br/>
543   <a href="control_lot.php?id=0">Lot nou</a></p>
544 <p class="title">Loturi active</p>
545 <?php
546 $query = "SELECT l.*, c.`nume_user` AS `user_creat`
547         FROM `lot` AS l
548         LEFT JOIN `users` AS c ON (l.`User`=c.`id_user`)
549         WHERE l.`Activ` <> 0 ORDER BY l.`ID_LOT` DESC";
550 $result = mysql_query($query);
551 $total=0;
552 if ($result && (mysql_num_rows($result) > 0))
553     {
554     $total=mysql_num_rows($result);
555     $row = mysql_fetch_assoc($result);
556     }
557 if ($total>0)
558 {?>
559 <table align="center">
560   <tr class="lista_titlu">
561     <td align="center">Nr.</td>
562       <td>Numar</td>
563       <td>Data</td>
564       <td>Nr. pui</td>
565       <td>Gr. pui</td>
566       <td>Nr. pui morti</td>
567       <td>Gr. pui morti</td>
568       <td>Ferma</td>
569       <td>Documente</td>
570       <td>Creat</td>
571       <td>Comenzi</td>
572     </tr>
573   <?php $index=1;
574   do { ?>
575   <tr class="<?php if ($index%2) echo "lista_in
576     <td align="center"><?php echo $index; ?> </td>
577       <td><?php echo $row['Numar'];?> </td>
578       <td><?php echo date("d/m/Y", strtotime($row['Data'])
579       <td><?php echo  $row['Pui'];?> </td>
580       <td><?php echo  $row['Greutate'];?> </td>
581       <td><?php echo  $row['Pui_Morti'];?> </td>
582       <td><?php echo  $row['Greutate_Morti'];?> </td>
583       <td><?php echo  $row['Ferma'];?> </td>
584       <td><?php echo  $row['Documente'];?> </td>
```

```
<table align="center">
  <tr class="lista_titlu">
    <td align="center">Nr.</td>
      <td>Numar</td>
      <td>Data</td>
      <td>Nr. pui</td>
      <td>Gr. pui</td>
```

HTML + PHP

E:\Documents\Pagini Web\Safir dep\lista_lot.php - Notepad++

File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?

logfile.php | lista_lot.php

```
539  <?php }?>
540
541  <div id="maincontent"><!-- InstanceBeginEditable name="Continut" -->
542  <p><span class="title">Adauga</span><br/>
543    <a href="control_lot.php?id=0">Lot nou</a></p>
544  <p class="title">Loturi active</p>
545  <?php
546  $query = "SELECT l.*, c.`nume_user` AS `user_creat`
547          FROM `lot` AS l
548          LEFT JOIN `users` AS c ON (l.`User`=c.`id_user`)
549          WHERE l.`Activ` <> 0 ORDER BY l.`ID_LOT` DESC";
550  $result = mysql_query($query);
551  $total=0;
552  if ($result && (mysql_num_rows($result) > 0))
553      {
554      $total=mysql_num_rows($result);
555      $row = mysql_fetch_assoc($result);
556      }
557  if ($total>0)
558  {?>
559  <table align="center">
560    <tr class="lista_titlu">
561      <td align="center">Nr.</td>
562        <td>Numar</td>
563        <td>Data</td>
564        <td>Nr. pui</td>
565        <td>Gr. pui</td>
566        <td>Nr. pui morti</td>
567        <td>Gr. pui morti</td>
568        <td>Ferma</td>
569        <td>Documente</td>
570        <td>Creat</td>
571        <td>Comenzi</td>
572      </tr>
573    <?php $index=1;
574    do { ?>
575    <tr class="<?php if ($index%2) echo "lista_impar"; else echo "lista_par";
576      <td align="center"><?php echo $index; ?> </td>
577          <td><?php echo $row['Numar'];?> </td>
578          <td><?php echo date("d/m/Y", strtotime($row['Data']));?> </td>
579          <td><?php echo  $row['Pui'];?> </td>
580          <td><?php echo  $row['Greutate'];?> </td>
581          <td><?php echo  $row['Pui_Morti'];?> </td>
582          <td><?php echo  $row['Greutate_Morti'];?> </td>
583          <td><?php echo  $row['Ferma'];?> </td>
584          <td><?php echo  $row['Documente'];?> </td>
```

```
<tr class="<?php if ($index%2) echo "lista_impar"; else echo "lista_p
  <td align="center"><?php echo $index; ?> </td>
      <td><?php echo $row['Numar'];?> </td>
      <td><?php echo date("d/m/Y", strtotime($row['Data']));?> <
      <td><?php echo  $row['Pui'];?> </td>
      <td><?php echo  $row['Greutate'];?> </td>
```

HTML + PHP + SQL

E:\Documents\Pagini Web\Safir dep\lista_lot.php - Notepad++

File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?

logfile.php | lista_lot.php

```
539  <?php }?>
540
541  <div id="maincontent"><!-- InstanceBeginEditable name="Continut" -->
542  <p><span class="title">Adauga</span><br/>
543    <a href="control_lot.php?id=0">Lot nou</a></p>
544  <p class="title">Loturi active</p>
545  <?php
546  $query = "SELECT l.*, c.`nume_user` AS `user_creat`
547          FROM `lot` AS l
548          LEFT JOIN `users` AS c ON (l.`User`=c.`id_user`)
549          WHERE l.`Activ` <> 0 ORDER BY l.`ID_LOT` DESC";
550  $result = mysql_query($query);
551  $total=0;
552  if ($result && (mysql_num_rows($result) > 0))
553      {
554      $total=mysql_num_rows($result);
555      $row = mysql_fetch_assoc($result);
556      }
557  if ($total>0)
558  {?>
559  <table align="center">
560    <tr class="lista_titlu">
561      <td align="center">Nr.</td>
562        <td>Numar</td>
563        <td>Data</td>
564        <td>Nr. pui</td>
565        <td>Gr. pui</td>
566        <td>Nr. pui morti</td>
567        <td>Gr. pui morti</td>
568        <td>Ferma</td>
569        <td>Documente</td>
570        <td>Creat</td>
571        <td>Comenzi</td>
572      </tr>
573    <?php $index=1;
574    do { ?>
575    <tr class="<?php if ($index%2) echo "lista_impar"; else echo "lista_par";?>">
576      <td align="center"><?php echo $index; ?> </td>
577          <td><?php echo $row['Numar'];?> </td>
578          <td><?php echo date("d/m/Y", strtotime($row['Data']));?> </td>
579          <td><?php echo  $row['Pui'];?> </td>
580          <td><?php echo  $row['Greutate'];?> </td>
581          <td><?php echo  $row['Pui_Morti'];?> </td>
582          <td><?php echo  $row['Greutate_Morti'];?> </td>
583          <td><?php echo  $row['Ferma'];?> </td>
584          <td><?php echo  $row['Documente'];?> </td>
```

```php
<?php
$query = "SELECT l.*, c.`nume_user` AS `user_creat`
        FROM `lot` AS l
        LEFT JOIN `users` AS c ON (l.`User`=c.`id_user`)
        WHERE l.`Activ` <> 0 ORDER BY l.`ID_LOT` DESC";
$result = mysql_query($query);
$total=0;
if ($result && (mysql_num_rows($result) > 0))
    {
    $total=mysql_num_rows($result);
    $row = mysql_fetch_assoc($result);
    }
```

HTML

- Offers **the logical structure** of the document
- Required
  - the **final result** of any type of programming for a distributed application web is almost exclusively HTML code (HTML structure eventually manipulated by "client-side scripting"/Javascript)
- **rf-opto Previous years** Curs 1 PAW 2021-2022 – Recapitulare HTML (RO + video)
- Course: Web Technologies (2012-2013) (EN)
  - https://rf-opto.etti.tuiasi.ro/internet.php
  - Web Technologies, Lecture 5
  - Web Technologies, Lecture 6

Learn more

- Web Hypertext Application Technology Working Group (WHATWG) HTML Living Standard
  - https://html.spec.whatwg.org/
- W3Schools Online Web Tutorials
  - https://www.w3schools.com/html/
  - https://www.w3schools.com/css/
  - + JavaScript, PHP, SQL, Python

Hypertext PreProcessor

PHP

Escaping from HTML

- **<?php ... ?>**
  - XML type – default, always available, recommended
- <? ... ?>
  - short, usually deactivated
  - absent in PHP 7,8
- <script language="php"> ... </script>
  - script type, available
- <% ... %>
  - ASP type, usually deactivated
  - absent in PHP 7,8

Escaping from HTML

- **echo** …. a PHP language construct: output one or more strings (equivalent with puts() in C)
- can process data
  - echo $a + $b;
- in most cases "output" is the data sent to the client by the web server
- "output" can usually be considered:
  - the current document
  - that particular script position inside the document

Escaping possibilities

- All versions offer the same HTML source for the browser
- The one that leaves the HTML structure outside the escaping tags and only the dynamic data results from PHP processing is **recommended**
- HTML + PHP code is interpreted much more elegantly in WYSIWYG editors

```
<h2>Transaction result</h2>
<?php echo '<p>Command received</p>';?>
```

```
<h2>Transaction result</h2>
<p><?php echo 'Command received';?></p>
```

```
<?php echo '<h1>Online Shop XXX SRL</h1>';?>
<?php echo '<h2>Transaction result</h2>';?>
<?php echo '<p>Command received</p>';?>
```

```
<?php
echo '<h1>Online Shop XXX SRL</h1>';
echo '<h2>Transaction result</h2>';
echo '<p>Command received</p>';
?>
```

PHP – variables

- variable – **$** character followed by the variable name
- the variable name is "case sensitive"
- a frequent mistake is forgetting the $ char
  - PHP Notice: Use of undefined constant an – assumed $an (or 'an') in D:\\Server\\
- Data types
  - scalar
  - composite
  - special

PHP – data types

- scalar
  - Boolean
  - integer
  - float (double)
  - **string**
- composite
  - array
  - object
- special
  - resource
  - NULL

PHP – Variables

$var = **expression**

- Control of variables is automatic, "on the fly"
  - A variable starts with the **$** sign, followed by the name of the variable
  - PHP has no command for declaring a variable. It is created the moment you first assign a value to it
  - PHP automatically associates a data type to the variable, depending on its value
  - Variable names are case-sensitive ($age and $AGE are two different variables)

PHP – data types

- PHP automatically associates a data type to the variable, depending on its value
- automatic conversions are often **not** numerical but "**human**"
- In PHP 7, type declarations were added for function arguments, return values and class properties

```
<?php
echo $variable ;  //  type Null, not initialized – value NULL (only)
$variable  = "0";  // $ variable   type  string (ASCII 48)
$variable  += 2;   // $ variable   type  integer (2)
$variable  = $ variable  + 1.3;  // $variable   type  float (3.3)
$variable  = 5 + "10 obiects"; // $variable   type  integer (15)
$var2=5; // $var2   type  integer (5)
$variable =$var2."10 objects";    //  $variable   type  string  "510 objects"
?>
```

PHP – operators

- Operators
  - Arithmetic
  - Assignment
  - Bitwise
  - Comparison
  - Incrementing/Decrementing
  - Logical
  - **String**

PHP – operators

- Arithmetic
  - -$a – Negation
  - $a + $b – Addition
  - $a - $b – Subtraction
  - $a * $b – Multiplication
  - $a / $b – Division
  - $a % $b – Modulo (remainder)
  - $a ** $b – Exponentiation
- **String**
  - **$a.$b – Concatenation String a with String b**

PHP – operators

- Assignment
  - $a=$b
  - $a+=$b (a=a+b)
  - $a-=$b (a=a-b)
  - $a/=$b (a=a/b)
  - $a*=$b (a=a*b)
  - $a%=$b (a=a%b)
  - $a**=$b ($a=a^b$)
  - **$a.=$b (a=a concatenate b - strings)**

Control Structures

- while
- do-while
- for
- switch
- return
- break
- goto

- similar to C/C++ equivalents

```
$i = 1;
while ($i <= 10) {
    echo $i++;
}
```

```
$i = 10;
do {
    echo $i--;
} while ($i > 0);
```

```
for ($i = 1; $i <= 10; $i++) {
    echo $i;
}
```

```
switch ($i) {
    case 0:
        echo "i este 0";
        break;
    case 1:
        echo "i  este  1";
        break;
    default:
        echo "i nici 1 nici 0";
        break;
}
```

Control Structures

- include()
- require()
- include_once()
- require_once()

- includes **and** evaluates the specified file
- used to avoid multiple evaluations of common code sections
- require will halt the script if the specified file is not found
- …_once() checks if the code from a file has already been included and it will **not** be included **again**

String Variables

PHP – data types

- scalar
  - Boolean
  - integer
  - float (double)
  - **string**
- composite
  - array
  - object
- special
  - resource
  - NULL

String Variables

- The ultimate goal of PHP is to populate the existing fields in an HTML skeleton with data (in the form of **text**)
- As a result, string data is treated more complex than the C/C++ counterpart
  - more ways of defining strings
  - more ways of interpreting strings
  - **much** more string related functions

String Variables

- string definition
  - single quoted: apostrophe ‘ ’
  - double quoted: quotation marks “ ”
  - block definition
    - heredoc <<< X … X;
    - nowdoc <<<‘X’ … X;(PHP>5.3.0)

String Variables ' '

- single quotes ' ' are used for the definition of classic basic strings
  - a set of characters is defined
  - processing inside the string is reduced
    - \' is a literal single quote
    - \\ and \ are a literal backslash
    - **only!!!**

String Variables " "

- double-quotes " " are used for the definition of complex strings
  - complex processing inside the string more than the C/C++ equivalent
    - special ASCII characters, similar to C++: \n, \r, \t, \\, \v, \e, \f, \x, \u
    - \" character double-quote
    - \$ character $
    - **variable names** inside the string will be expanded !!!

String Variables " "

- unescaped character dollar sign $ means a variable name will follow
  - the parser will take as many tokens as possible to form a valid variable name ($x, $x->y, $x[y])
  - if more complex variables are required (2 indexes array x[y][z] or non-integer indexes) enclose the variable name in curly braces { } to explicitly specify the end of the name

String Variables " "

- **simple** syntax for variable parsing

```
<?php
$juice = "apple";

echo "He drank some $juice juice.";
//  He drank some apple juice.
echo "He drank some juice made of $juices.";
//  He drank some juice made of .  //s caracter valid pentru variabile

?>
```

String Variables " "

- **simple** syntax for variable parsing

```
<?php
 $juices = array("apple", "orange", "koolaid1" => "purple");
 class people {
   public $john = "John Smith";
}

$people = new people();
echo "$people->john drank some $juices[0] juice.";
// John Smith drank some apple juice.
?>
```

String Variables " "

- **complex** syntax for variable parsing { }

```
<?php
$juice = "apple";

echo "He drank some juice made of $juices.";
//  He drank some juice made of .  //s caracter valid pentru variabile
echo "He drank some juice made of ${juice}s."
//  He drank some juice made of apples.  // {} arata unde se incheie
numele variabilei
?>
```

String Variables " "

- **complex** syntax for variable parsing { }

```
<?php
 $juices = array(array("apple", "orange"), "koolaid1" => "purple");
 class people {
    public $name = "John Smith";
}

$obj->values[3] = new people();
echo "$obj->values[3]->name drank some $juices[0][1] juice.";
 //  drank some  juice.
echo "{$obj->values[3]->name} drank some {$juices[0][1]} juice.";
// John Smith drank some apple juice.
?>
```

Heredoc syntax

```
<?php
// no indentation
echo <<<END
        a
       b
      c
\n
END;

// 4 spaces of indentation
echo <<<END
       a
      b
     c
     END;
?>
```

Nowdoc syntax

- "Nowdocs are to single-quoted strings what heredocs are to double-quoted strings"
- **Multiline** string with minimal (**no**) processing

```
<?php
 echo <<<'EOD'
Example of string spanning multiple lines
using nowdoc syntax. Backslashes are always treated
literally,
e.g. \\ and \'.
EOD;
?>
```

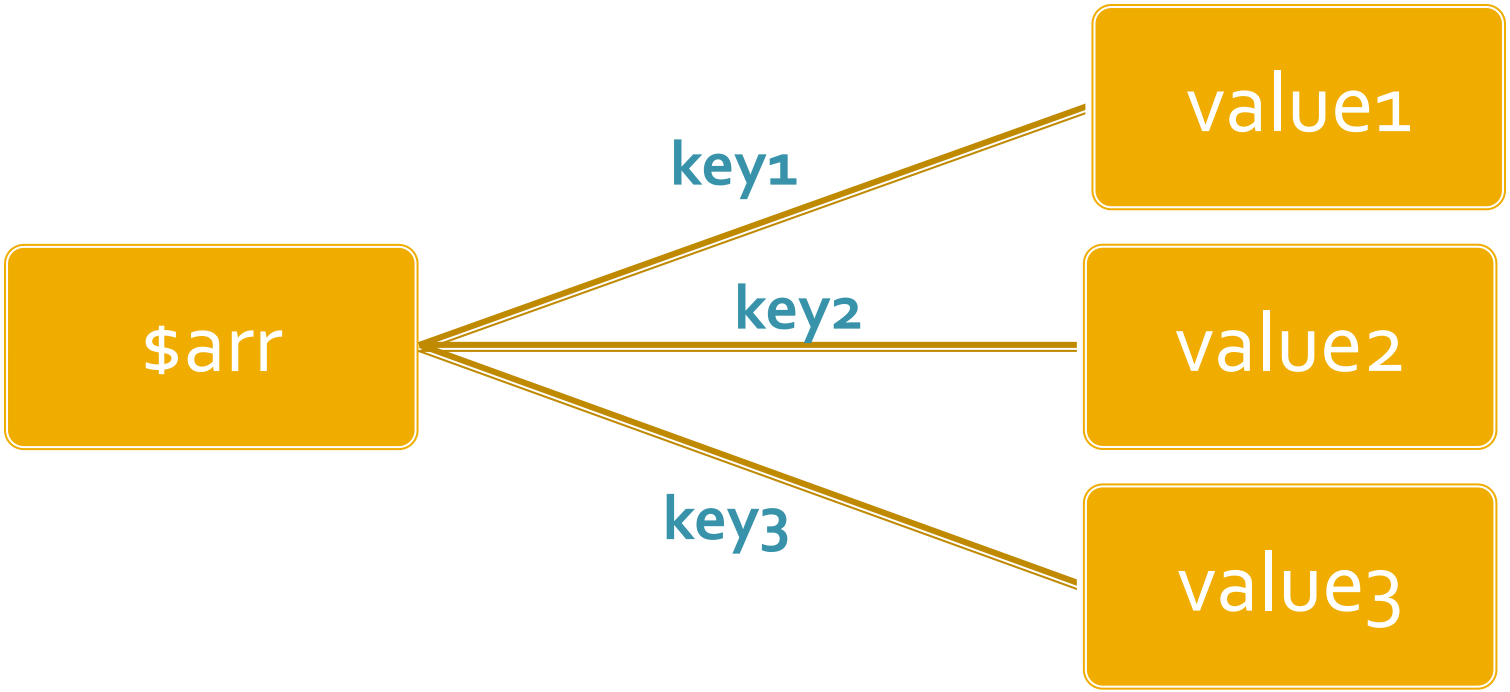
Arrays

Arrays in PHP

- An array in PHP is actually an ordered map. A map is a type that associates **values** to **keys**
- unlike C, Basic, **keys** are **not** required to be **integers**, can be **strings**
- default keys (if not otherwise specified) are consecutive integers with first key 0 (C syntax).
- defining a key / value pair
  - key **=>** value
- create an array
  - $arr = **array**("definition of key / value pairs")
  - pairs: key **=>** value, key **=>** value, …

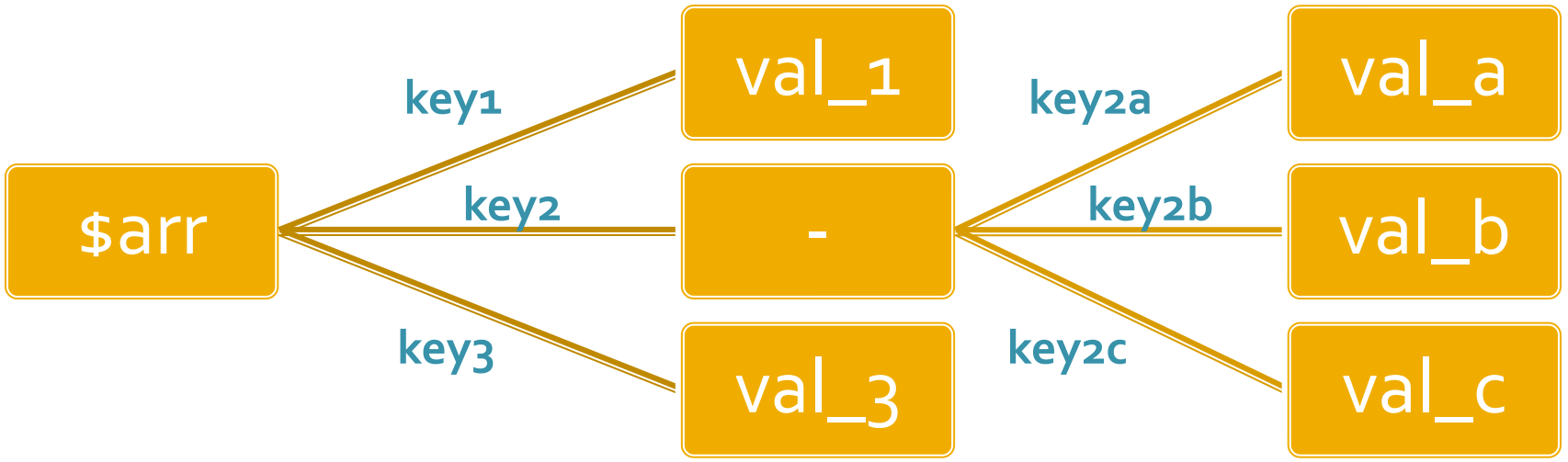
Array = Logical tree

- $arr = array(key1 => value1, key2 => value2, key3 => value3)



Array = Logical tree

- In particular, one or more of the values can in turn be an array, leading to **branching** of the tree
- $arr = array(key1 => val_1, key2 => array(key2a => val_a, key2b => val_b, key2c => val_c), key3 => val_3)



Arrays in PHP

```
$matr = array(1, 2, 3, 4, 5);

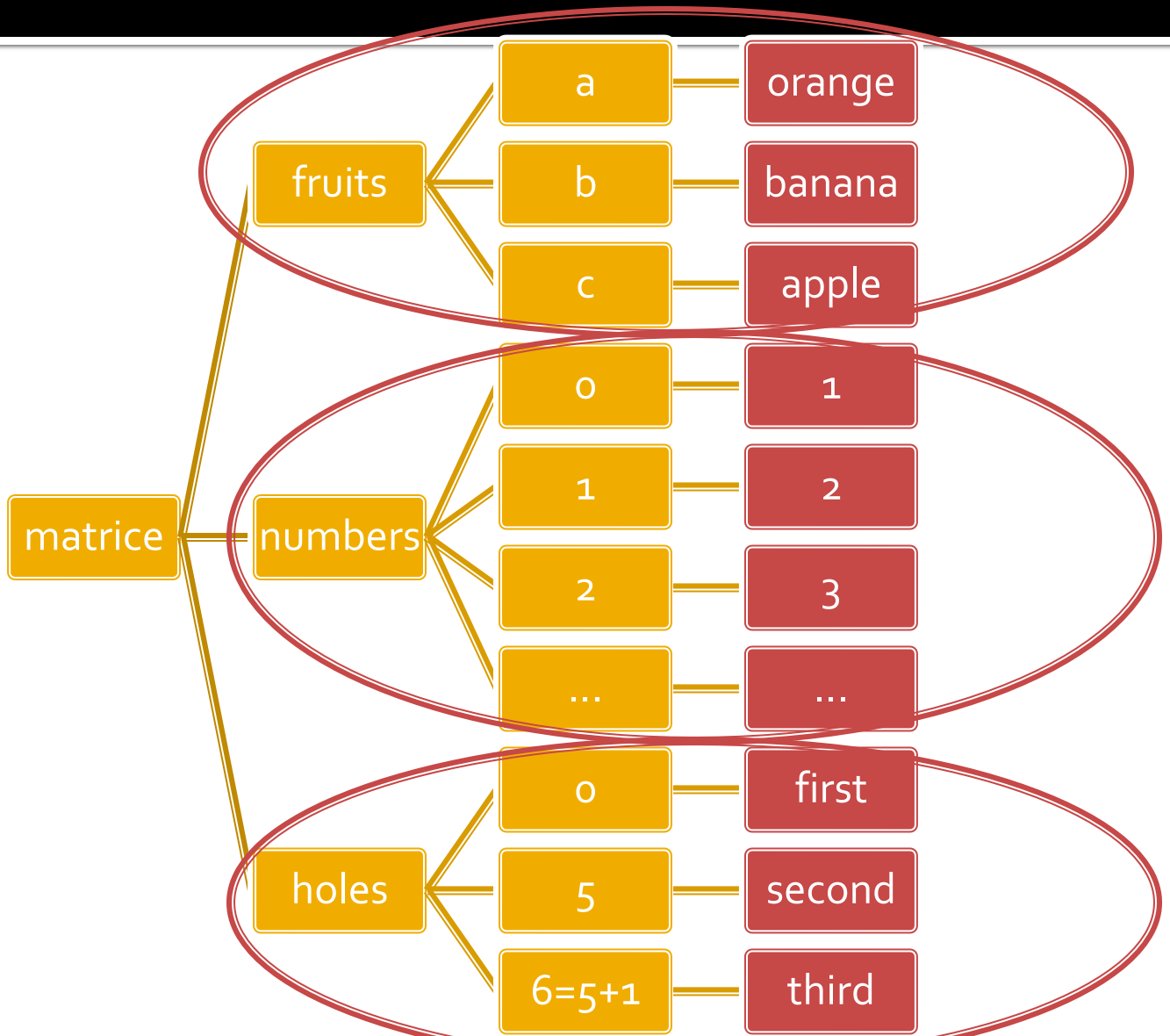
$matr[0]=1
$matr[1]=2
$matr[2]=3
$matr[3]=4
$matr[4]=5
```

```
$matr = array('a' => 1, 'b'  => 2, 3, 4, 5);

$matr['a']=1
$matr['b']=2
$matr[0]=3
$matr[1]=4
$matr[2]=5
```

```
$matrice= array (
        "fruits"  => array("a" => "orange", "b" => "banana", "c" => "apple"),
        "numbers" => array(1, 2, 3, 4, 5, 6),
        "holes"   => array("first", 5 => "second", "third")
);
```

Arrays in PHP



View array content (debug)

```
echo "<pre>";
print_r ($matr);
echo "</pre>";
```

```
$matr= array (
"fruits"  =>
array("a" => "orange", "b" => "banana", "c" => "apple" ,
"ultim"),
"numbers" =>
array(1, 2, 3, 4, 5, 6),
"holes"   =>
array("first", 5 => "second", "third")
);
echo $matr;
echo "<pre>";
print_r ($matr);
echo "</pre>";
```

```
Array

Array
(
    [fruits] => Array
        (
            [a] => orange
            [b] => banana
            [c] => apple
            [0] => ultim
        )

    [numbers] => Array
        (
            [0] => 1
            [1] => 2
            [2] => 3
            [3] => 4
            [4] => 5
            [5] => 6
        )

    [holes] => Array
        (
            [0] => first
            [5] => second
            [6] => third
        )

)
```

Foreach loop

- foreach (array_expression **as** $key => $value)
  statement
- foreach (array_expression **as** $value) statement
- foreach construct is used to loop through each key/value pair in an array
- On each iteration assign the current element's key to the local variable **$key** and the value of the current element is assigned to the local variable **$value** (scope: statement)
- foreach() works with a **copy** of the array, you cannot change the original array inside the statement
  - foreach ($arr as $key => $value) {
    $value = 'other value'; //**doesn't** work
    $arr[$key] = 'other value'; //works
    }

Example – foreach

```
$matr = array (
  "fruits"  => array("a" => "orange", "b" => "banana", "c" => "apple", "ultim"),
  "numbers" => "in loc de numere",
  "holes"   => "in loc de ce era"
);
foreach ($matr as $cheie => $continut)
            echo "matr[".$cheie."]=".$continut."<br />";
```

```
matr[fruits]=Array
matr[numbers]=in loc de numere
matr[holes]=in loc de ce era
```

PHP Global Variables - Superglobals

PHP Global Variables - Superglobals

- PHP Global Variables - Superglobals (predefined variables)
  - are always accessible, regardless of scope
  - Examples:
    - **$_SERVER** — Server and execution environment information
    - **$_GET** — HTTP GET variables
    - **$_POST** — HTTP POST variables
    - **$_FILES** — HTTP File Upload variables
    - **$_REQUEST** — HTTP Request variables
    - $_SESSION — Session variables
    - $_ENV — Environment variables
    - $_COOKIE — HTTP Cookies

Forms in HTML

- required for the user to send data to the server
- <form>…</form>
- Specific attributes:
  - action: address of the script which receives the data
    - <form **action**="<?php echo $_SERVER['PHP_SELF'];?>">
    - <form **action**="actiuni/file2.html">
  - method: method for transmitting data: post or get
    - <form **method**="post" action= … >

Methods

- **post** : data is transmitted as a block (inside the body of the HTTP request)
- **get** : appends form-data into the URL : results.php?prob=81&an=2009
- **get** must be used only for "idempotent" data,
  - no collateral effects
  - no change in server status (databases, etc)
- we can emulate a form (**get**) by writing links appropriately

Form elements

- inside <form>…</form>
  - input
  - select/option
  - textarea
  - button
- **all** elements will have a name
  - attribute: **name**=“”
  - the name will be directly present in the URI generated by get, or will be a variable sent by post
- data is effectively sent by a **type="submit"** button

Examples

```
<input name="textfield" type="text" value="ceva" />

<input name="Ok" type="submit" value="Trimite" />

<label><input name="check" type="checkbox" value="5" checked />check1</label>

<label><input name="RG1" type="radio" value="a" checked="checked" />but1</label>
<label><input name="RG1" type="radio" value="b" />but2</label>

<input name="hid" type="hidden" value="6" />
```

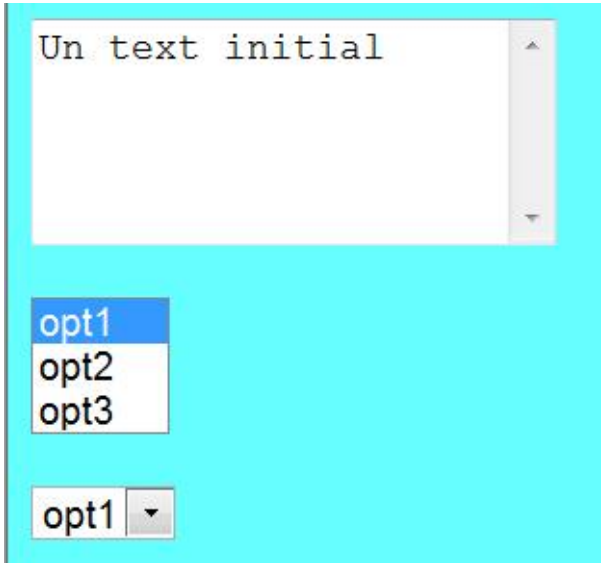


TEXTAREA/SELECT

```
<textarea name="textarea" cols="20" rows="5">Un text initial</textarea><br /><br />

<select name="select." size="3">
 <option value="1" selected="selected">opt1</option>
 <option value="2">opt2</option>
 <option value="3">opt3</option>
</select><br /><br />

<select name="select...">
 <option value="1" selected="selected">opt1</option>
 <option value="2">opt2</option>
 <option value="3">opt3</option>
</select>
```



Getting user submitted data

- When a user submits the data by clicking on "Submit", the form data is found in the file specified in the **action** attribute of the <form> tag in one of the superglobal variables:
  - $_POST – method="post"
  - $_GET – method="get"
  - $_REQUEST – both methods
- the superglobal variables are **arrays** with **string keys** controlled by the **name** attribute of the input element
  - <input type="text" name="**books_quant**" size="3" maxlength="3" />
  - $_POST['**books_quant**'] contains the user input in the receiving script

Organizing $_POST

- **name** attributes in the form inputs become **keys** in the superglobal array $_POST
  - <input type="text" name="**books_quant**" size="3" maxlength="3" />
  - $_POST['**books_quant**'] contains the user input
- creating **name** "array like", we can control branching of $_POST grouping input elements in the form as required
  - <input type="text" name="**quant[books]**" size="3" maxlength="3" />
  - $_POST ['**quant**'] ['**books**'] contains the user input

Accessing a MySQL server from PHP

Accessing a MySQL server from PHP

- PHP has two extensions in order to interact with a MySql server (local or remote), these must be activated in php.ini.
  - mysql
  - mysqli (improved: functions for MySql > 4.1)
- A database can be accessed if the PHP script knows a MySql server user with access rights
  - usually every application has its specific MySql user with specific access rights
- A database can also be created from PHP, but it is not the recommended method if it is not necessary
  - the code is difficult to implement and used only once

Accessing a MySQL server from PHP

- mysql_connect
  - Open a connection to a MySQL Server
  - resource **mysql_connect** (string server , string user, string password)
  - returns a MySQL link identifier on success or false on failure
- mysql_query
  - Send a MySQL query
  - resource **mysql_query** (string query [, resource link_identifier])
  - result
    - SELECT, SHOW, DESCRIBE or EXPLAIN: returns a resource or false
    - UPDATE, DELETE, DROP, etc: returns true/false

Accessing a MySQL server from PHP

- mysql_fetch_assoc
  - Returns an **associative array** that corresponds to the fetched row and moves the internal data pointer ahead, or **false** if there are no more rows. The **string** keys of the array are the field names (columns) in the DB table
  - array **mysql_fetch_assoc** (resource result)
- mysql_fetch_row
  - Returns an **numerical** array that corresponds to the fetched row, or false
  - array mysql_fetch_row (resource result)

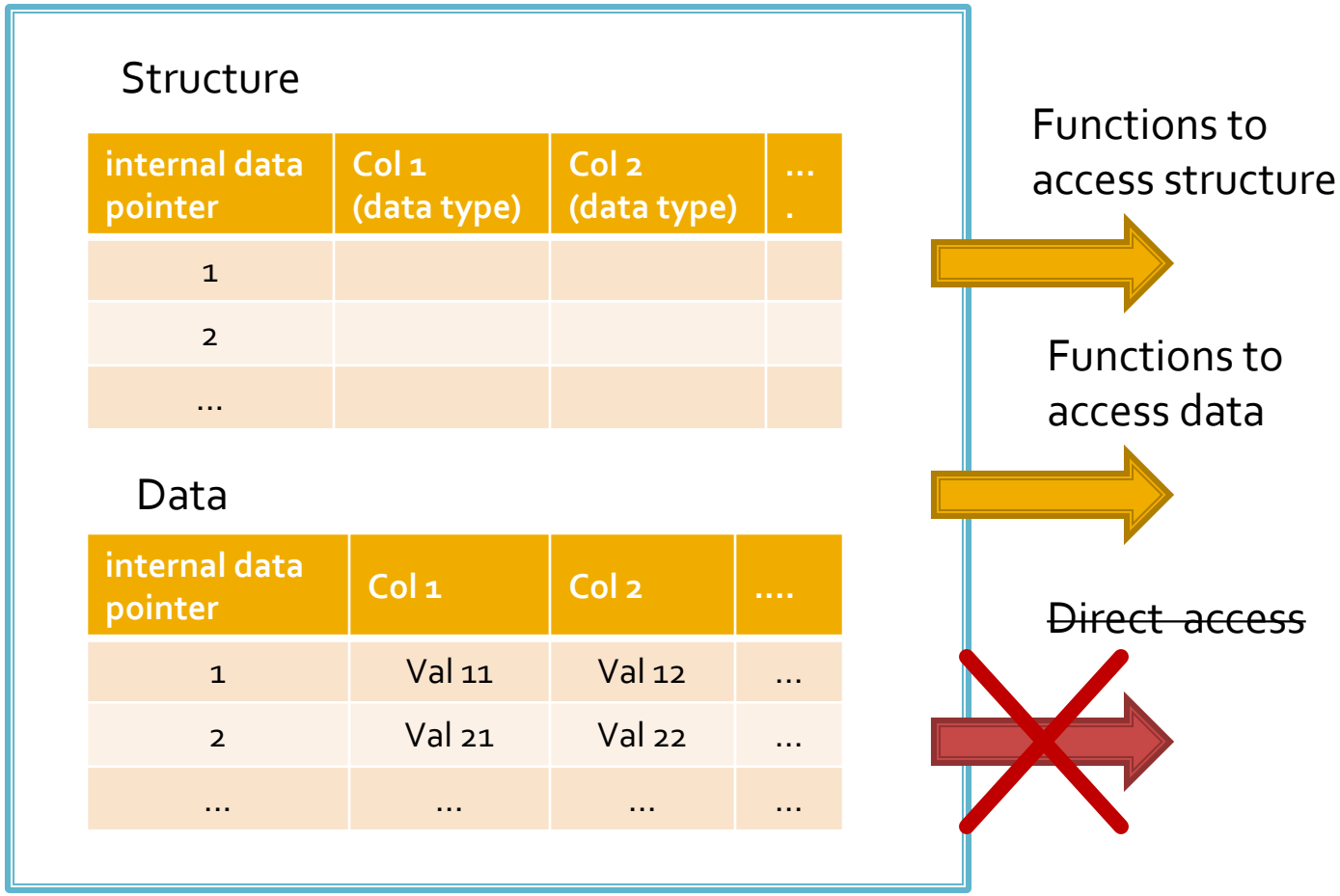
Accessing a MySQL server from PHP

- mysql_fetch_array
  - groups functionality of mysql_fetch_assoc and mysql_fetch_row
  - array **mysql_fetch_array** (resource result [, int result_type])
  - MYSQL_ASSOC, MYSQL_NUM, MYSQL_BOTH (default)
- mysql_data_seek
  - moves the internal row pointer of the MySQL result associated with the specified result identifier to point to the specified row number. The next call to a MySQL fetch function would return that row
  - bool **mysql_data_seek** (resource result, int row_number)

MySQL resources

- Resources are a combination between
  - Structured data (values + structure) resulted from a SQL query
  - functions to access those values/structure
- Analogy with OOP
  - a special "class" created following a SQL query with predefined procedures to access the results of that query

MySQL resources



MySQL resources

- Structure access functions are rarely used
  - most applications are designed on a fixed DB structure, and the structure of the received data is known
  - exception: general DB applications, eg: PhpMyAdmin
- Most data access functions are characterized by sequential access
  - the data is read line by line
  - simultaneously, internal data pointer advances to the next position, preparing the next read

MySQL resources

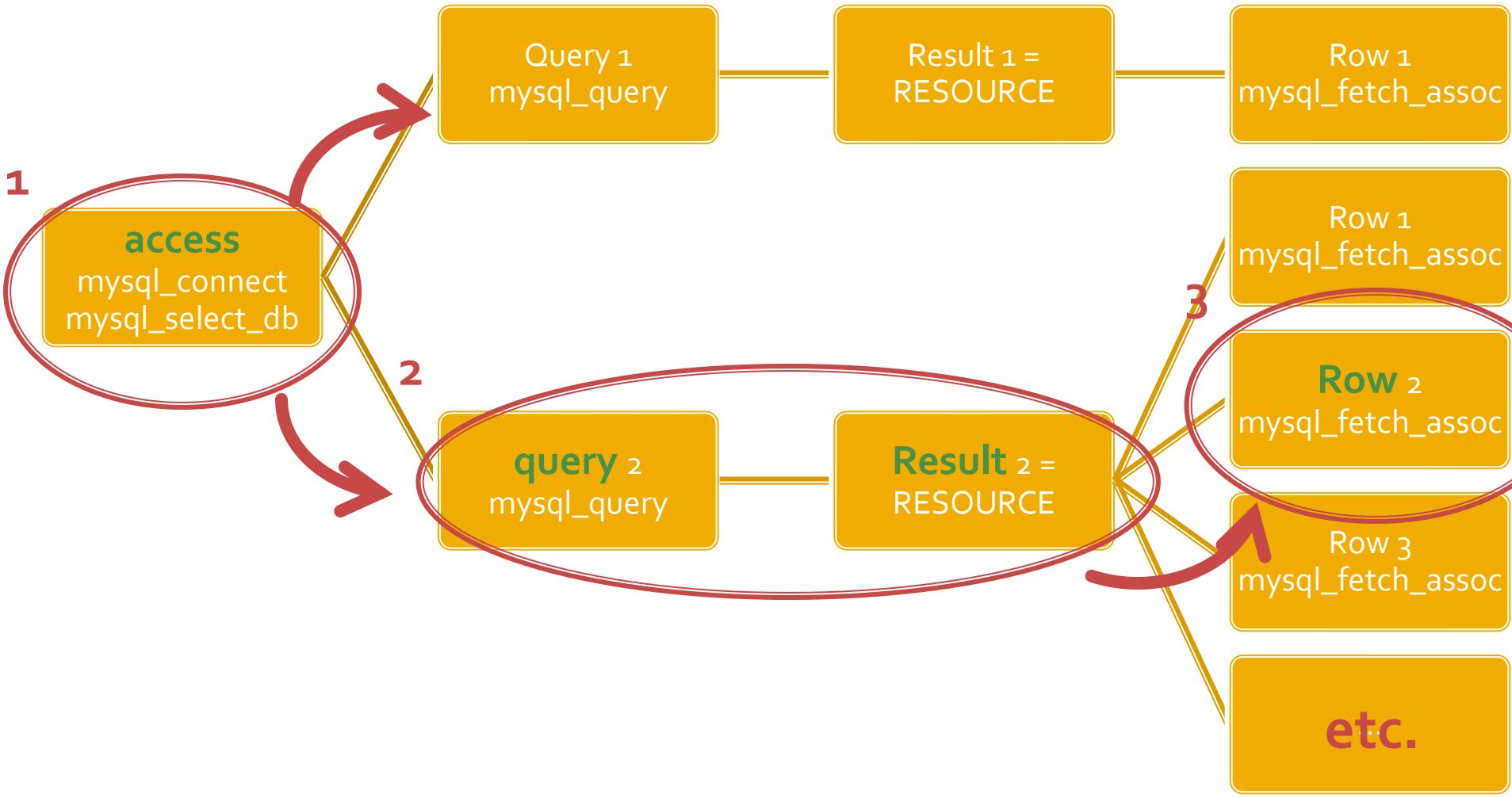
- Functions are optimized for use in a **do { } while()** loop or a **while() { }** loop
  - Returns **false** if there are no more rows
- typically we fetch a single row (mysql_fetch_assoc) followed by a **do { } while()** loop
  - to allow a "problem detection" code to run only once
  - or generate "single steps" for displaying a successful result (eg: table head)

Example

```
$hostname = "localhost";
$database = "world";
$username = "web";
$password = “ceva";
$conex= mysql_connect($hostname, $username, $password);
mysql_select_db($database, $ conex);
```

```
$query = "SELECT `Code`,`Name`,`Population` FROM `country` AS c ";
$result = mysql_query($ query, $ conex) or die(mysql_error());
$row_result = mysql_fetch_assoc($ result );
$totalRows_ result = mysql_num_rows($ result );
```

Accessing a MySQL server from PHP



!! IMPORTANT

PHP > 5.5

PHP 5.5, 7, 8

- Starting with PHP 5.5.0 the mysql extension was **deprecated**
  - any function from this extension generates an **E_DEPRECATED** error/warning
  - Starting with PHP 7.0.0 the mysql extension was **removed**
- Instead we must use:
  - mysqli extension (MySQL Improved)
  - PDO extension (PHP Data Objects)

mysqli extension

- Other than enhanced security offers access to newer facilities of the DB server:
  - Prepared Statements (speed, security)
    - server side
    - client side
  - server Stored Procedures (speed, security)
  - Multiple Statements
  - Transactions (integrity)

mysqli extension

- Supports two interfaces
  - procedural interfaces (similar to mysql)
  - OOP (similar to PDO)
- Procedural interface (almost) identical to the original mysql extension
  - easy transition
  - small differences (parameter)

mysqli – Procedural

```php
<?php
$mysqli = mysqli_connect("example.com", "user", "password", "database");
$res = mysqli_query($mysqli, "SELECT 'Please do not use the mysql extension ' AS _msg FROM DUAL");
$row = mysqli_fetch_assoc($res);
echo $row['_msg'];

$mysql = mysql_connect("example.com", "user", "password");
mysql_select_db("test");
$res = mysql_query("SELECT ' for new developments.' AS _msg FROM DUAL", $mysql);
$row = mysql_fetch_assoc($res);
echo $row['_msg'];
?>
```

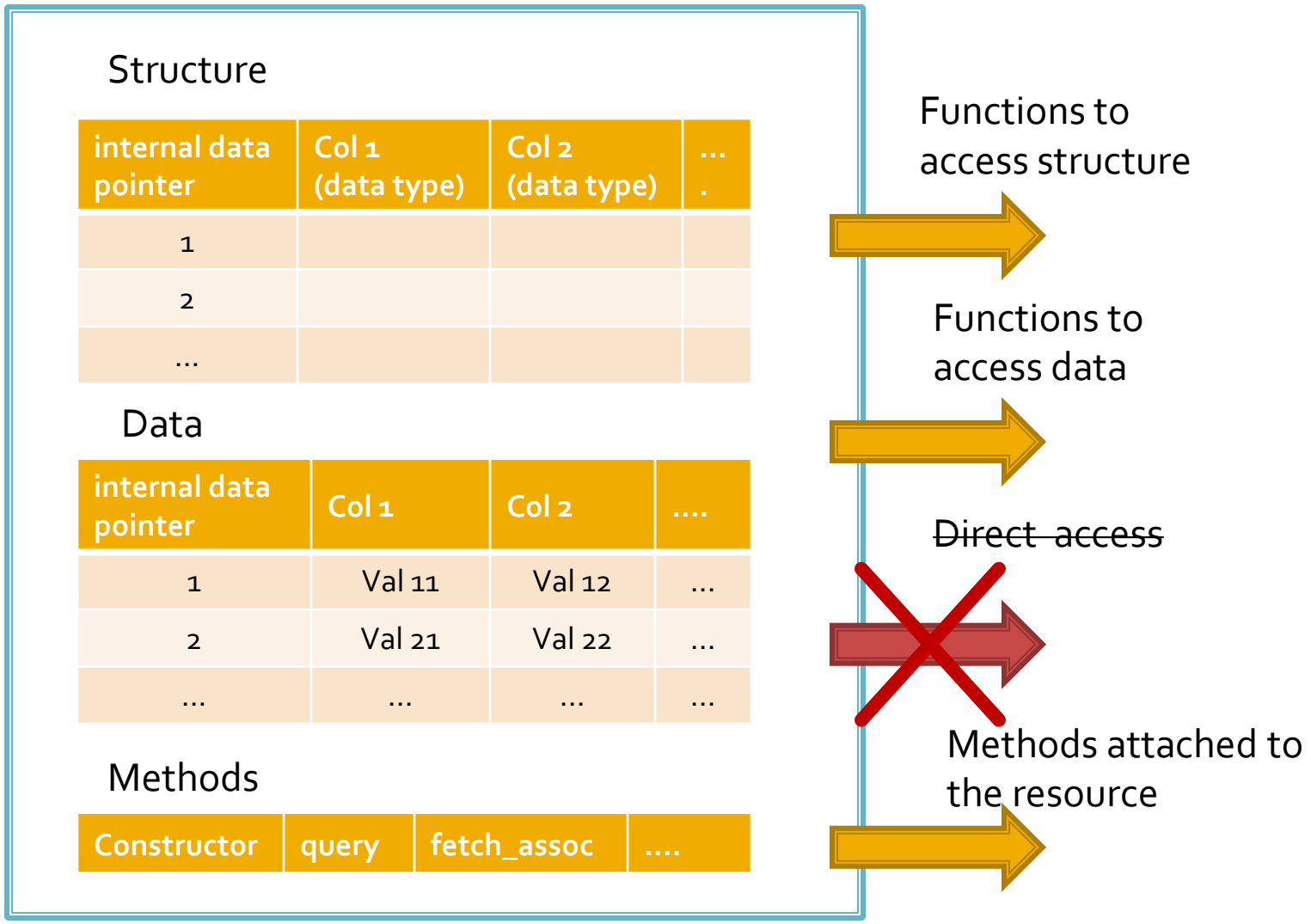
- all mysql functions have a mysqli equivalent
- most functions have the same parameters in the same order
- there are functions with small differences (Ex: **mysqli_connect**, **mysqli_query**)

mysqli – OOP

```php
<?php
$var = new mysqli("example.com", "user", "password", "database");
$res = $var->query ($mysqli, "SELECT 'Please do not use the mysql extension ' AS _msg FROM DUAL");
$row = $res->fetch_assoc();
echo $row['_msg'];

$mysql = mysql_connect("example.com", "user", "password");
mysql_select_db("test");
$res = mysql_query("SELECT ' for new developments.' AS _msg FROM DUAL", $mysql);
$row = mysql_fetch_assoc($res);
echo $row['_msg'];
?>
```

MySQL resources – mysqli



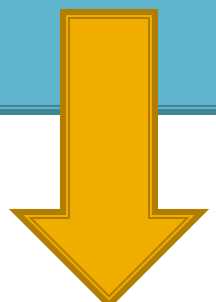
Conversion to mysqli (mandatory)

- example

```
$hostname = "localhost";
$database = "dbwpi";
$username = "web";
$password = "test";
$conex= mysql_connect($hostname, $username, $password);
mysql_select_db($database, $conex);

$query = "SELECT p.*, c.`nume` AS `nume_categ` FROM `produse` AS p
                LEFT JOIN `categorii` AS c ON (c.`id_categ` = p.`id_categ`)";
$result = mysql_query($query, $conex) or die(mysql_error());
$row_result = mysql_fetch_assoc($result);
$totalRows_result = mysql_num_rows($result);

do {
                $produse[$row_result['nume_categ']][$row_result['nume']]=array ("descr" => $row_result['detalii'], "pret"
=> $row_result['pret'], "cant" => $row_result['cant']);
                }
while ($row_result = mysql_fetch_assoc($result));
```



mysqli (Procedural)

```
//$conex= mysql_connect($hostname, $username, $password);
//mysql_select_db($database, $conex);
$conex = mysqli_connect($hostname, $username, $password, $database);

$query = "SELECT p.*, c.`nume` AS `nume_categ` FROM `produse` AS p
                LEFT JOIN `categorii` AS c ON (c.`id_categ` = p.`id_categ`)";
//$result = mysql_query($query, $conex) or die(mysql_error());
$result = mysqli_query($conex, $query);

//$row_result = mysql_fetch_assoc($result);
$row_result = mysqli_fetch_assoc($result);

//$totalRows_result = mysql_num_rows($result);
$totalRows_result = mysqli_num_rows($result);

do {
                $produse[$row_result['nume_categ']][$row_result['nume']]=array ("descr" => $row_result['detalii'], "pret"
=> $row_result['pret'], "cant" => $row_result['cant']);
                }
//while ($row_result = mysql_fetch_assoc($result));
while ($row_result = mysqli_fetch_assoc($result));
```

mysqli (OOP)

```
//$conex= mysql_connect($hostname, $username, $password);
//mysql_select_db($database, $conex);
//$conex = mysqli_connect($hostname, $username, $password, $database);
$conex = new mysqli($hostname, $username, $password, $database);

$query = "SELECT p.*, c.`nume` AS `nume_categ` FROM `produse` AS p
                LEFT JOIN `categorii` AS c ON (c.`id_categ` = p.`id_categ`)";
//$result = mysql_query($query, $conex) or die(mysql_error());
//$result = mysqli_query($conex, $query);
$result = $conex->query( $query );

//$row_result = mysql_fetch_assoc($result);
//$row_result = mysqli_fetch_assoc($result);
$row_result = $result->fetch_assoc();

//$totalRows_result = mysql_num_rows($result);
//$totalRows_result = mysqli_num_rows($result);
$totalRows_result = $result->num_rows;

do {
                $produse[$row_result['nume_categ']][$row_result['nume']]=array ("descr" => $row_result['detalii'], "pret"
=> $row_result['pret'], "cant" => $row_result['cant']);
                }
//while ($row_result = mysql_fetch_assoc($result));
while ($row_result = $result->fetch_assoc());
```

Textbooks

- https://www.php.net/
- https://rf-opto.etti.tuiasi.ro/master_it.php
  - **3 X PHP and MySQL Bible !!**

Template

Template

- simultaneous control of the esthetic and functional design for all pages in the site
- separation the application from the esthetic design

Example



Magazin online Firma X SRL

Lista Produse

Nr.	Produs	Pret
1	Carti	100
2	Caiete	50
3	Penare	150
4	Stilouri	125
5	Creioane	25
Comanda		

```
<html>
<head>
<title>Magazin online Firma X SRL</title>
</head>
<body bgcolor="#CCFFFF">
<table width="600" border="0" align="center">
<tr><td><img src="images/antet.gif" width="600" height="100" /></td></tr>
<tr><td height="600" valign="top" bgcolor="#FFFFCC">
Continut
</td></tr>
</table>
</body>
</html>
```

Control statements

- include()
- require()
- include_once()
- require_once()

- to insert the content of one PHP file (used as parameter) into another PHP file (that uses the include/require statement) before the server executes it
- **require** stops the execution of the current script if the parameter file is **not** found
- …**_once()** checks if the respective file has been included before and does not include it again

Example 2

- repeated sections can be moved to a separate file and inserted with require()
- first step: common areas are identified

```
<html>
<head>
<title>Magazin online Firma X SRL</title>
</head>
<body bgcolor="#CCFFFF">
<table width="600" border="0" align="center">
<tr><td><img src="images/antet.gif"
width="600" height="100" /></td></tr>
<tr><td height="600" valign="top"
bgcolor="#FFFFCC">
Continut
</td></tr>
</table>
</body>
</html>
```

Example 3

```
<html>
<head>
<title>Magazin online Firma X
SRL</title>
</head>
<body bgcolor="#CCFFFF">
<table width="600" border="0"
align="center">
<tr><td><img src="images/antet.gif"
width="600" height="100" /></td></tr>
<tr><td height="600" valign="top"
bgcolor="#FFFFCC">
Continut
</td></tr>
</table>
</body>
</html>
```

header.php

```
<html>
<head>
<title>Magazin online Firma X
SRL</title>
</head>
<body bgcolor="#CCFFFF"><?php
//orice cod comun PHP
?><table width="600" border="0"
align="center">
<tr><td><img src="images/antet.gif"
width="600" height="100" /></td></tr>
<tr><td height="600" valign="top"
bgcolor="#FFFFCC">
<h1>Magazin online Firma X SRL</h1>
```

footer.php

```
</td></tr>
</table>
</body>
</html>
```

Using a template

- header.php
  - any common structure (HTML) code
  - any common application code (PHP) – almost all pages in an application need:
    - data access
    - check access rights
    - constant definitions
    - define/load data **from** session ($_SESSION)
- footer.php
  - any common structure (HTML) code
  - any common application code (PHP) – usually less:
    - save data **into** the session ($_SESSION)

Template

- Any php file in my application:
  - <?php require('header.php');?>
  - <?php require('footer.php');?>
- and automatically that file has the same esthetic and functional design

*.php

```
<?php require('header.php');?>
<h2>Lista Produse</h2>
<table border="1">
...
</table>
<?php require('footer.php');?>
```

Advantages working with template

- speed of application development
- clear separation of the application from the form
- unitary form
  - "don't make me think"
- simultaneous control of the esthetic and functional design for all pages in the site
- defining common data in a single file
  - define('BOOK_PRICE',100);

Active Links

Methods

- **post** : data is transmitted as a block (inside the body of the HTTP request)
- **get** : appends form-data into the URL : results.php?prob=81&an=2009
- **get** must be used only for "idempotent" data,
  - no collateral effects
  - no change in server status (databases, etc)
- we can emulate a form (**get**) by writing links appropriately

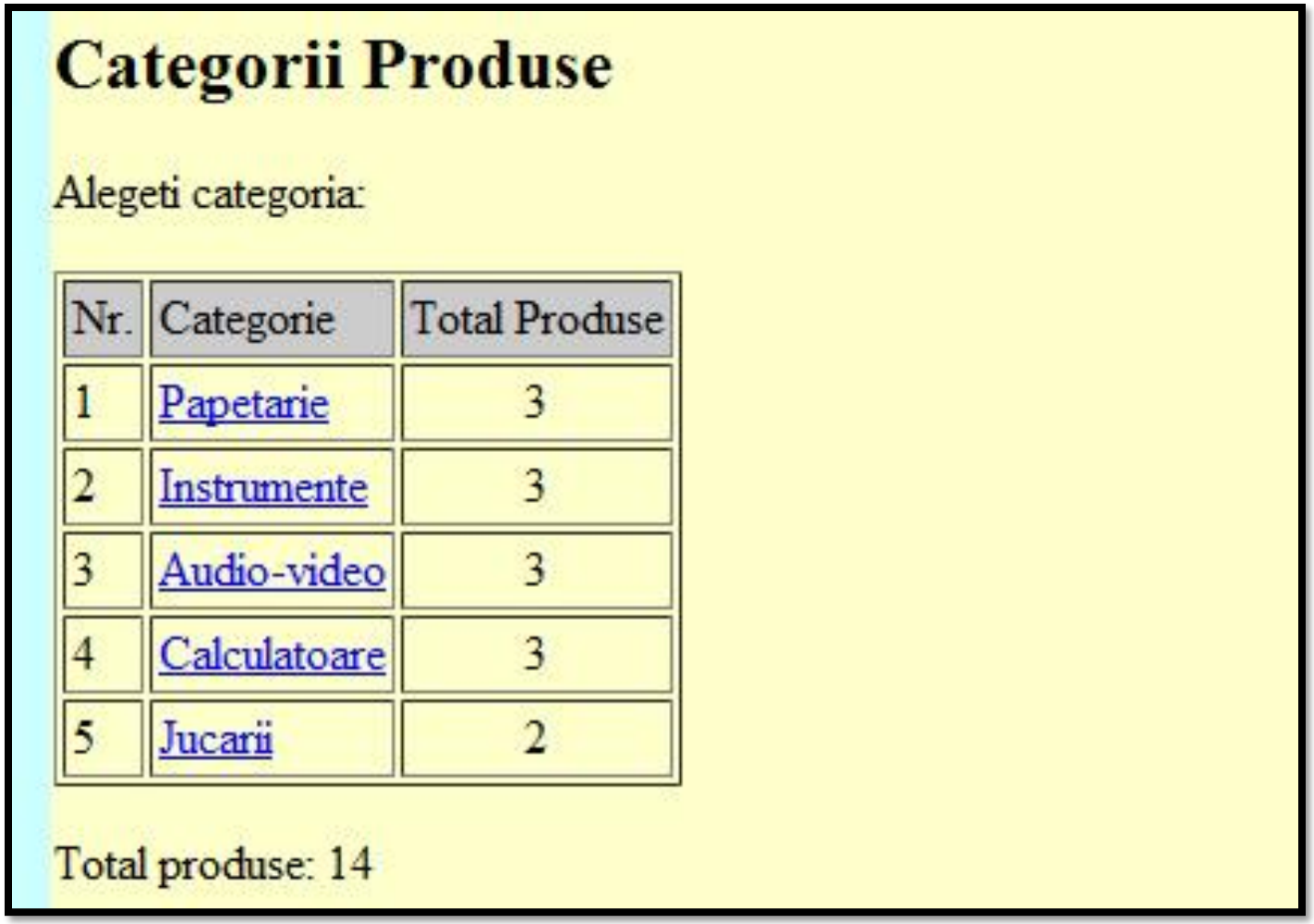
Active Links

- used to send specific **information** to the target file
- in file1.php
  - <a href="file2.php?categ=<?php echo $cat;?>"> <?php echo $cat;?>
- in file2.php
  - $_GET['categ']="value $cat associated to that specific link"

$cat – $_GET



Active Links



Categorii Produse

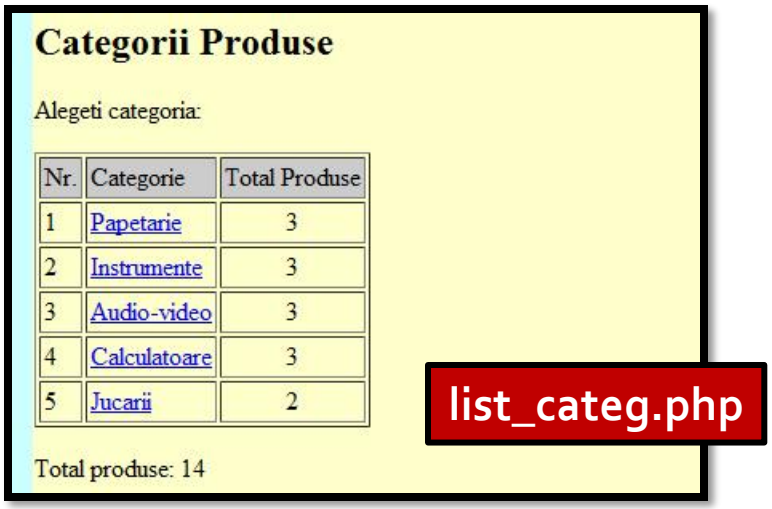
Alegeti categoria:

Nr.	Categorie	Total Produse
1	Papetarie	3
2	Instrumente	3
3	Audio-video	3
4	Calculatoare	3
5	Jucarii	2

Total produse: 14

Application flowchart

Rudimentary online shop app



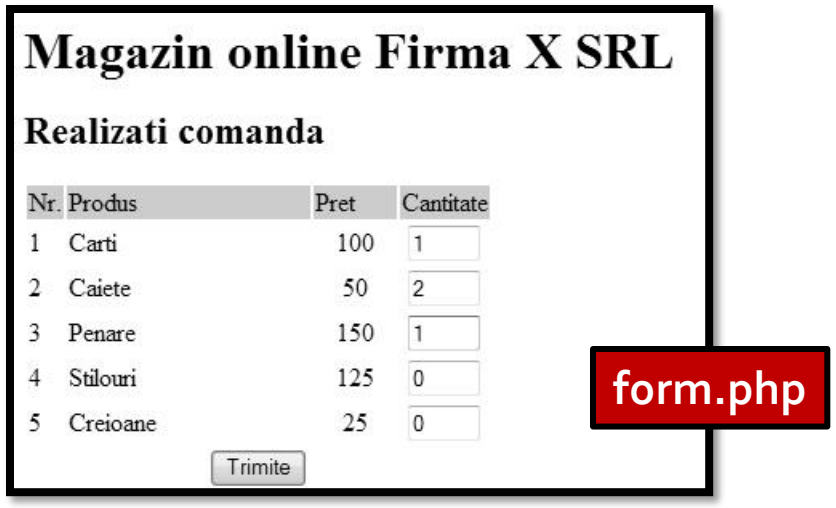
Categorii Produse

Alegeti categoria:

Nr.	Categorie	Total Produse
1	Papetarie	3
2	Instrumente	3
3	Audio-video	3
4	Calculatoare	3
5	Jucarii	2

Total produse: 14

list_categ.php



Magazin online Firma X SRL

Realizati comanda

Nr.	Produs	Pret	Cantitate
1	Carti	100	1
2	Caiete	50	2
3	Penare	150	1
4	Stilouri	125	0
5	Creioane	25	0

Trimite

form.php



Magazin online Firma X SRL

Rezultate comanda

Pret total (fara TVA): 350

Pret total (cu TVA): 416.5

Comanda receptionata la data: 17/03/2010 ora 08:24

result.php

Application flowchart – Buyer

- As the application leaves a linear thread of execution, it is necessary to introduce a flowchart (tree) of the application
- Buyer
  - reading the required data (database access) is done in header.php, common for all files



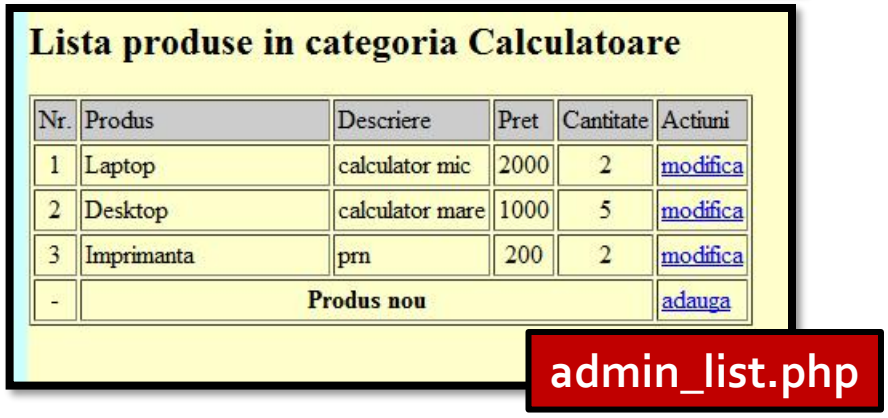
Application flowchart – Seller

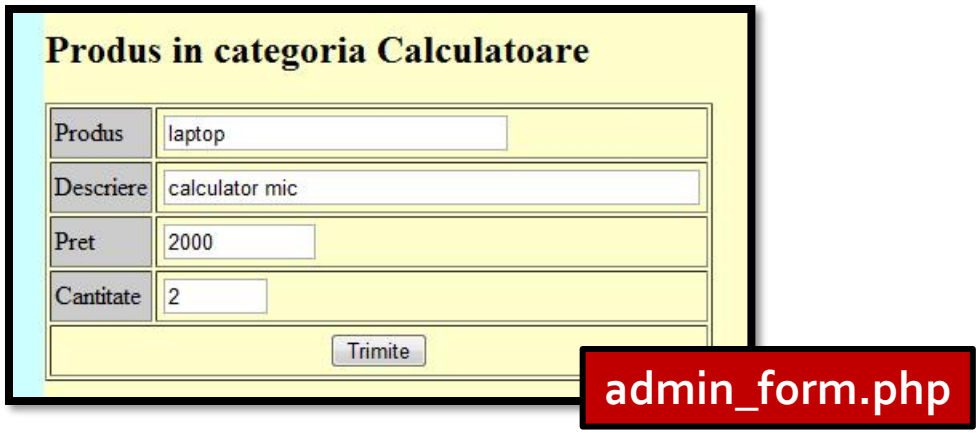
- The appearance of the application for the seller
  - introduces a parallel thread of execution with the necessity of the initial choice: buyer/seller
  - brings the possibility of writing in the database
  - various writing operations
    - insert new product category
    - insert new product in an existing category
    - modify existing product
  - writing in the database involves 2 actions:
    - collection of raw data from the user
    - data processing

Seller thread for online shop app

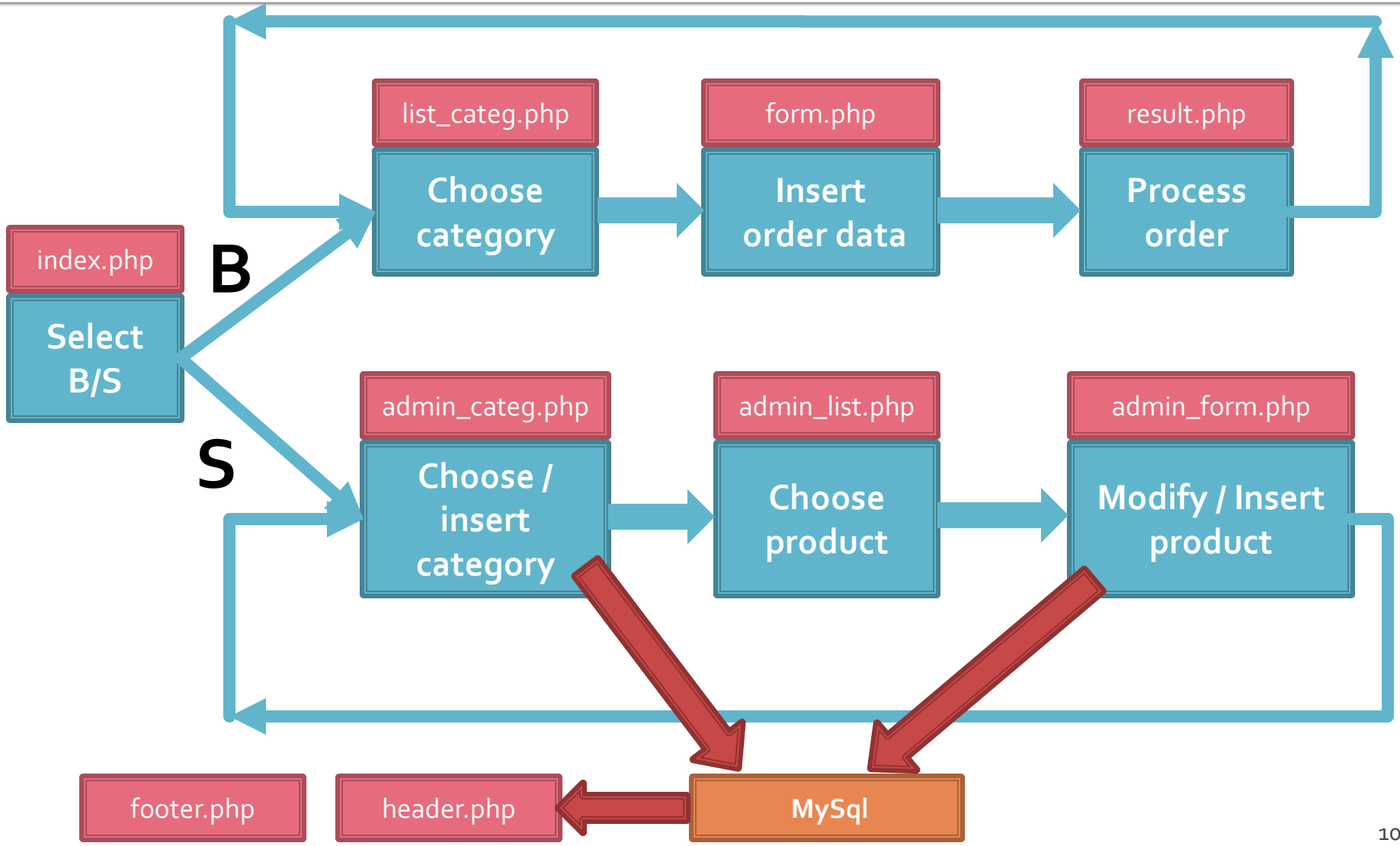






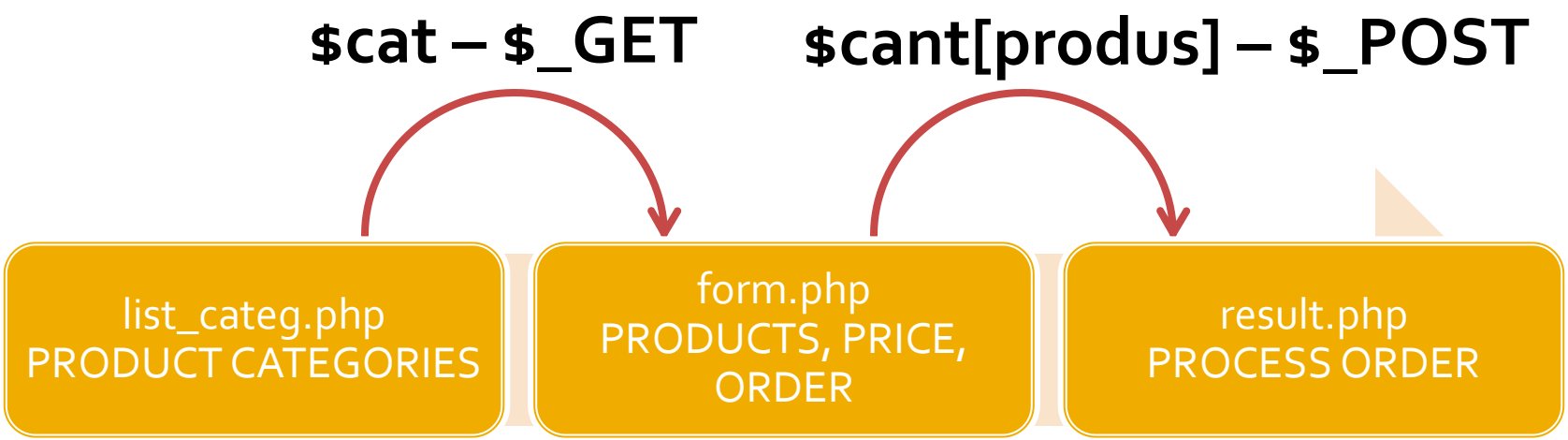


Application flowchart



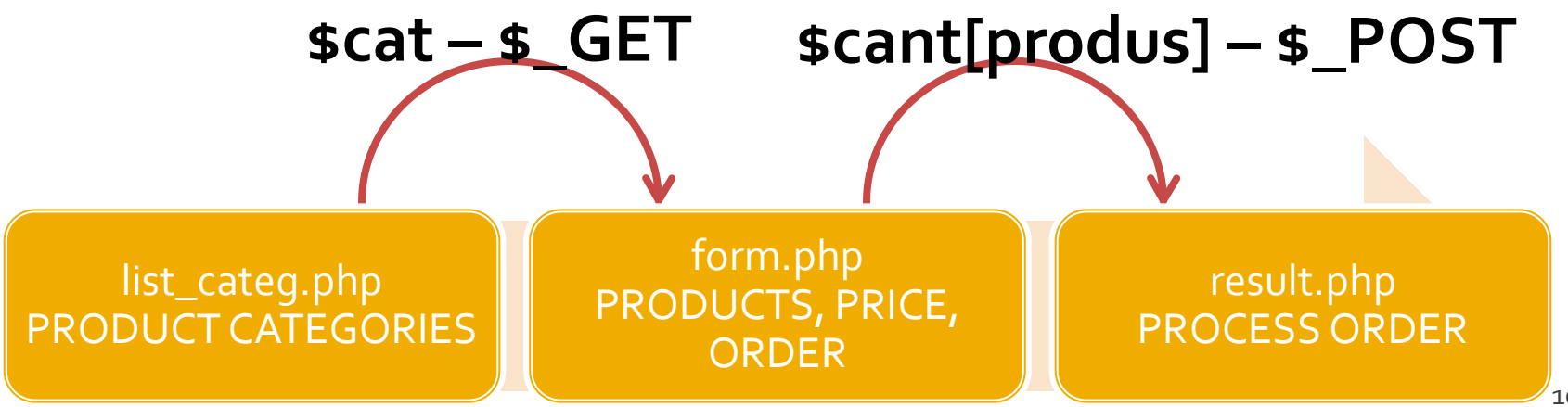
Application flowchart

- The application flowchart must also include information related to :
  - **what** data is transmitted between the different pages
  - **how** data is transmitted between pages

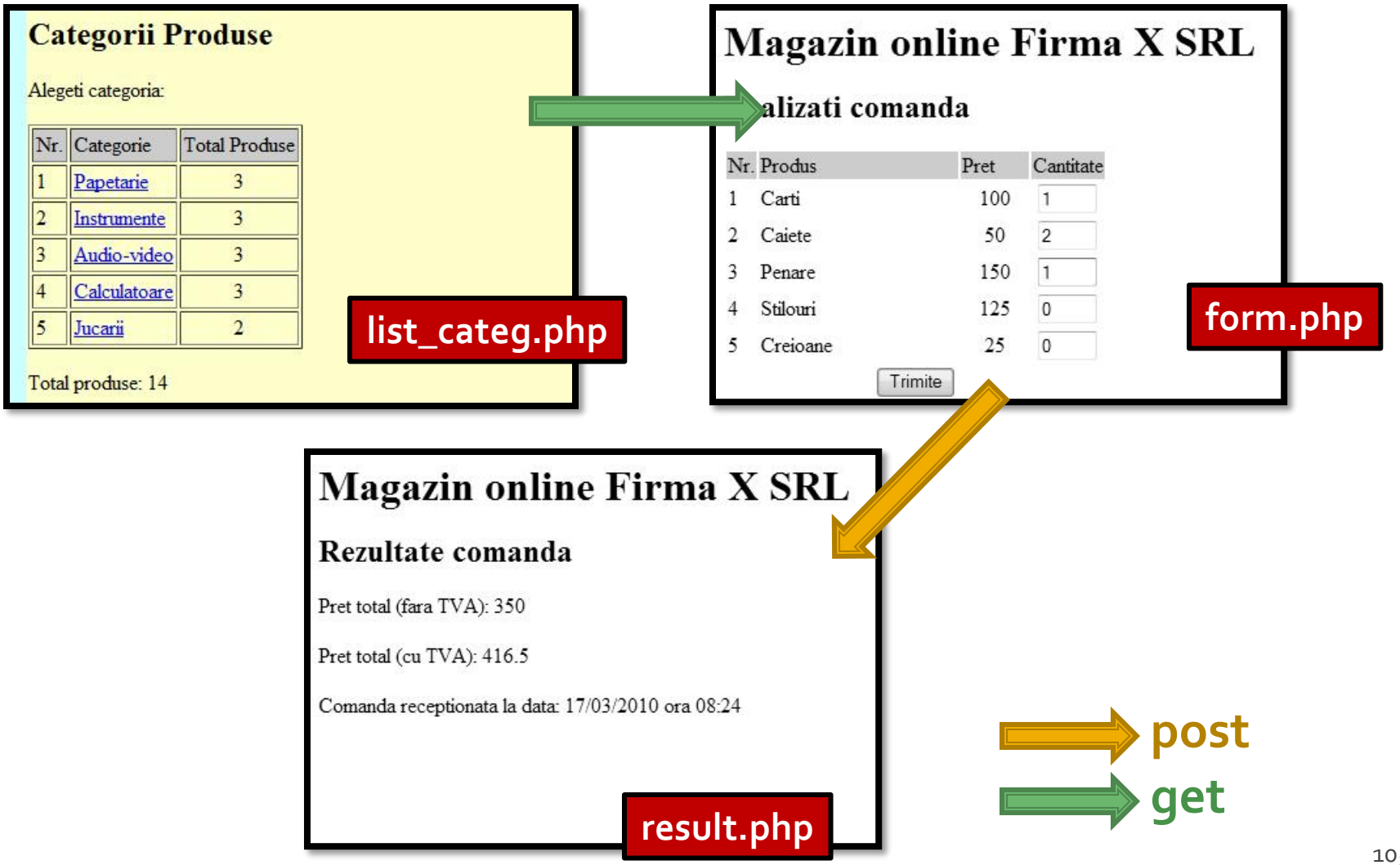


Application flowchart

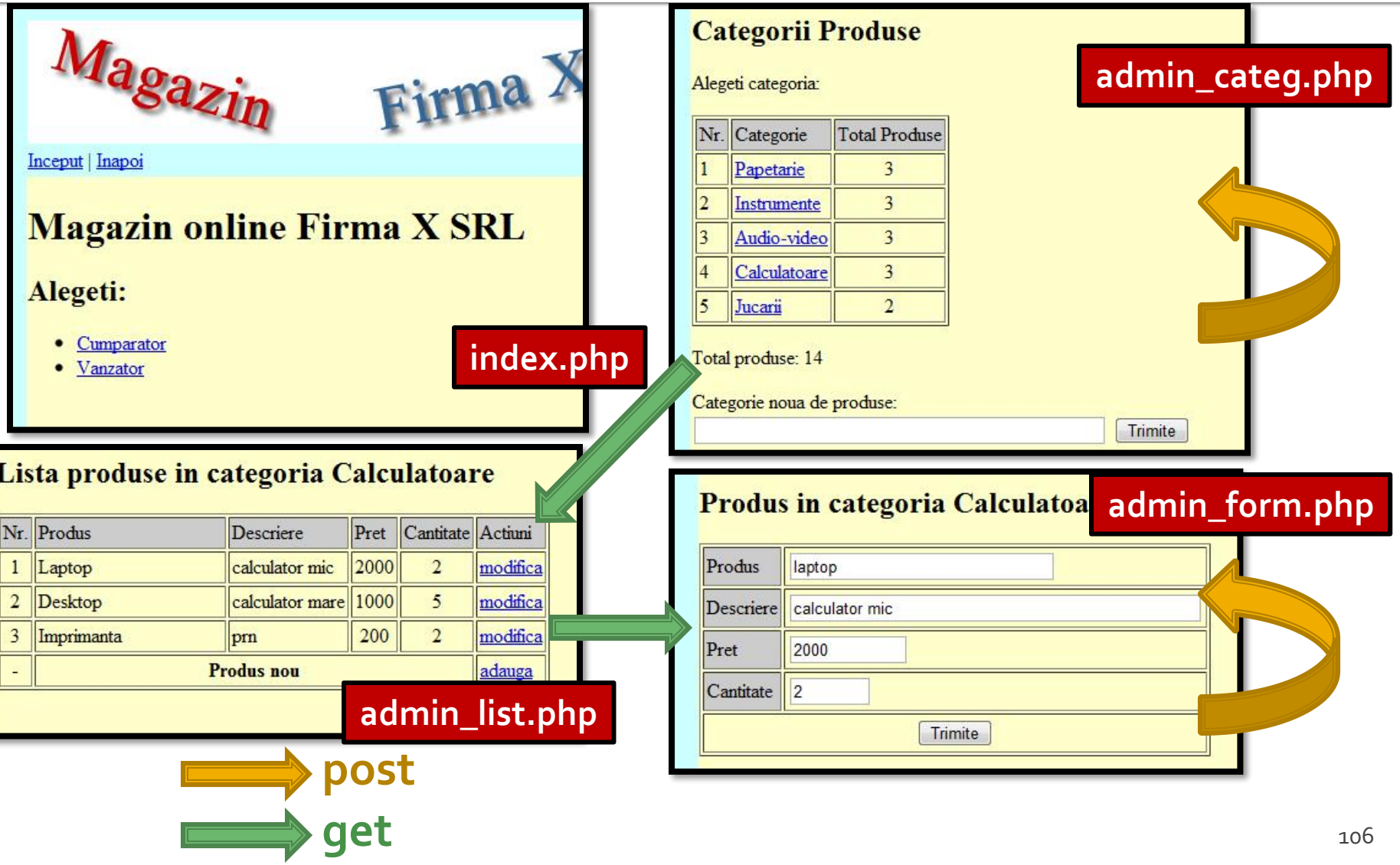
- Application flowchart – Example
  - the list of categories will transmit a single variable to the next file so we can use "active links", get method, **$_GET** in next file
  - the order form transmits multiple data included in a form, so the transmission is done with post method, **$_POST** in next file
- The choice of $_GET/$_POST has implications both in:
  - the page that transmits the data
  - as well as on the page that receives them



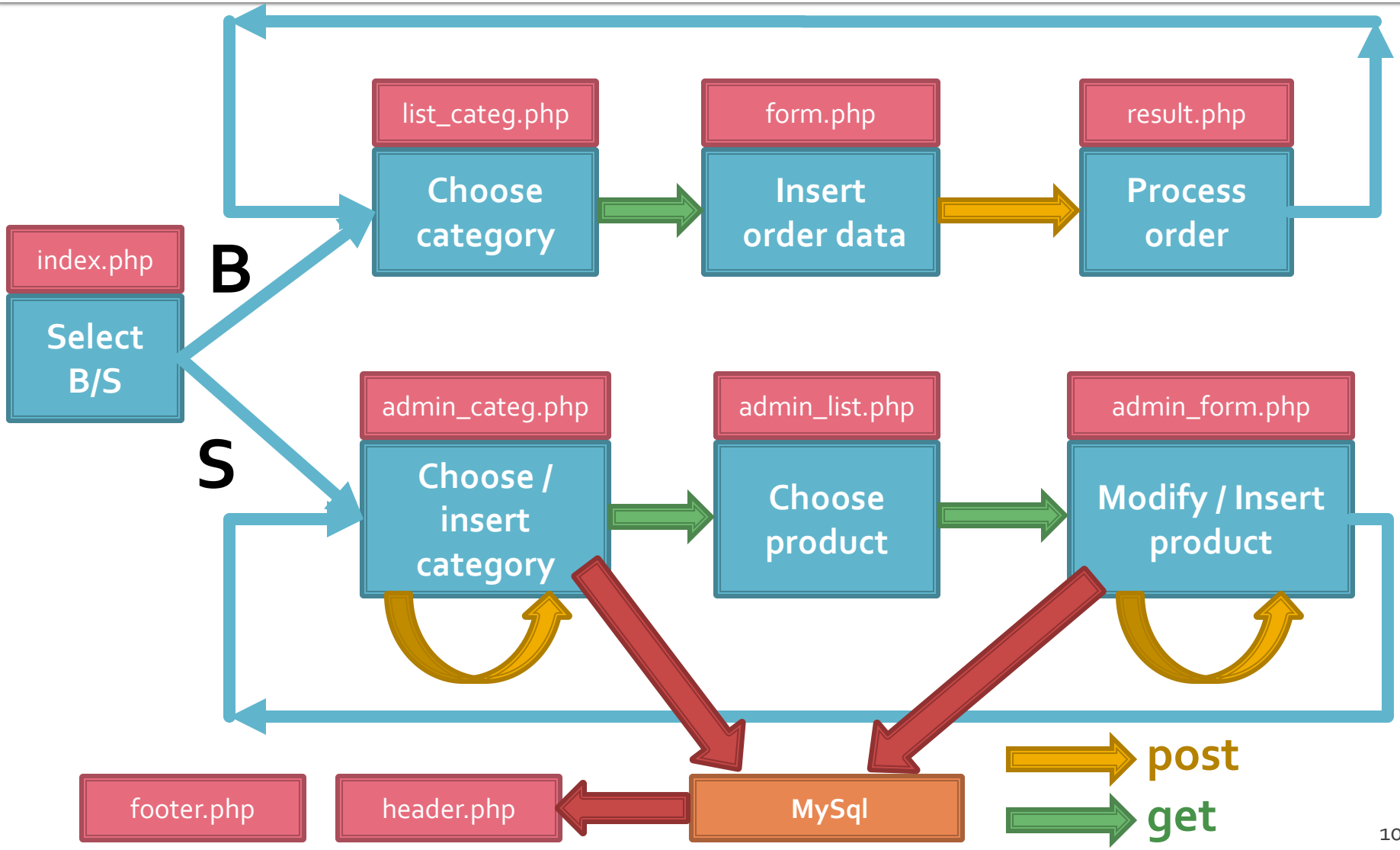
Flowchart (Buyer)



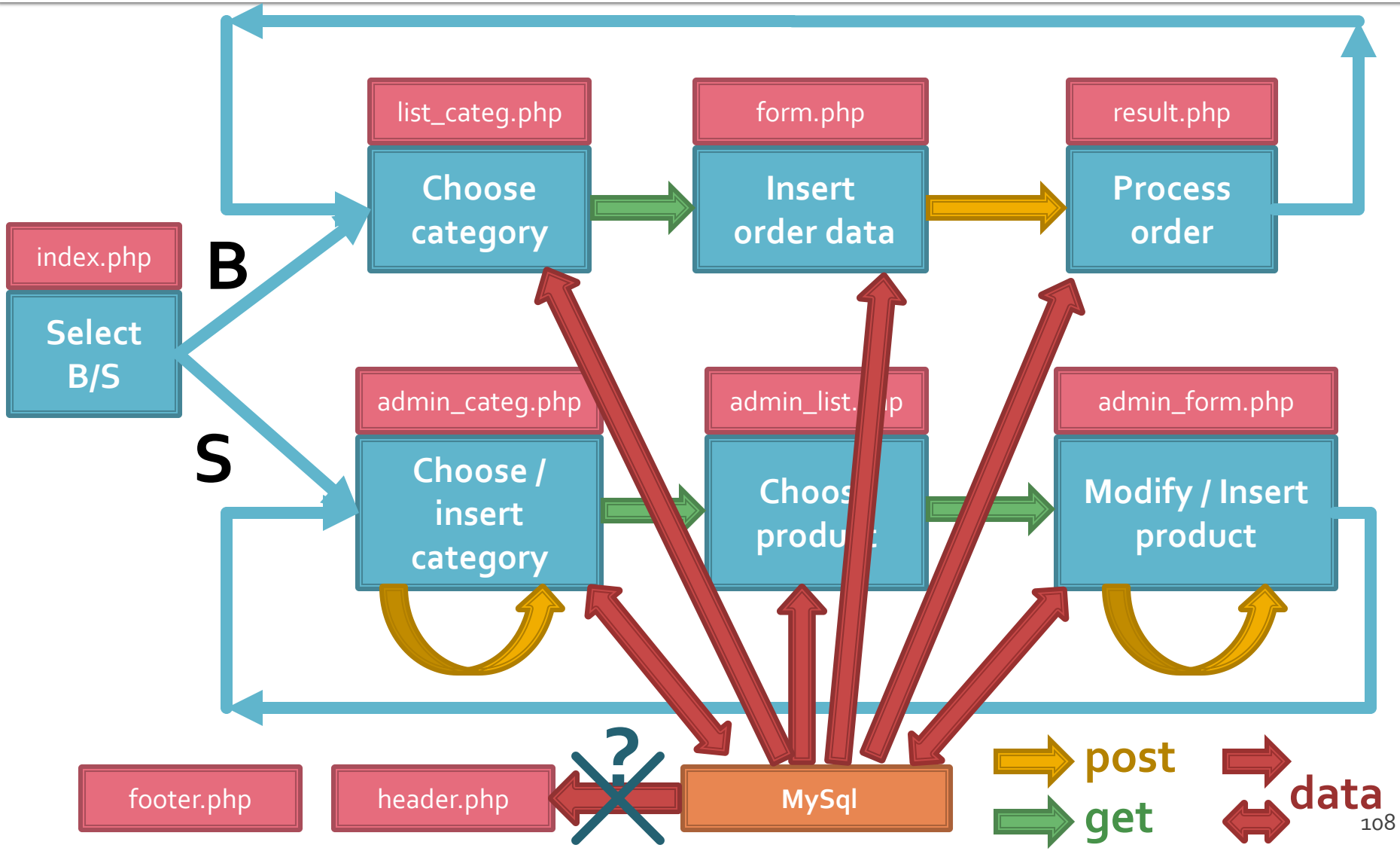
Flowchart (Seller)



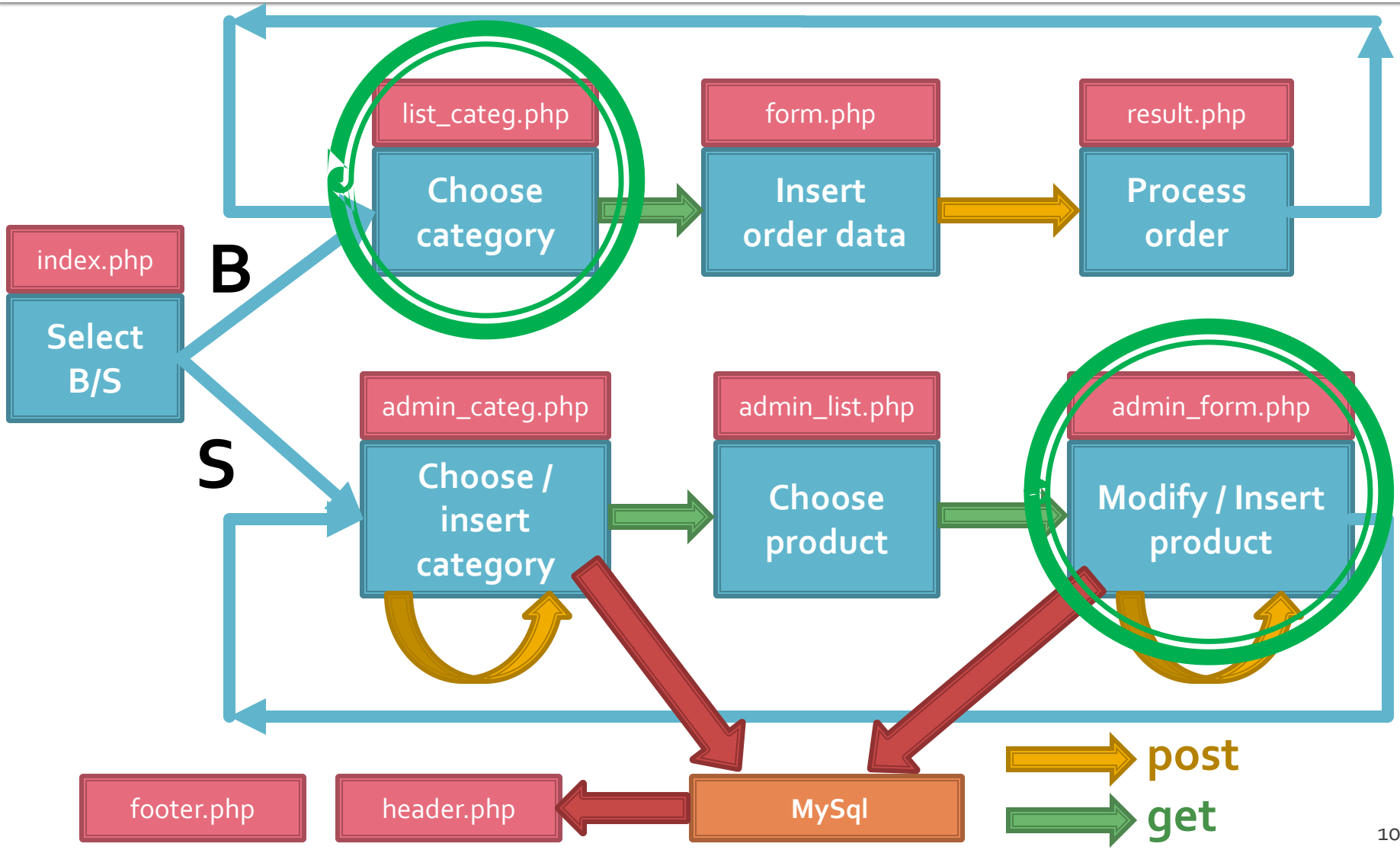
Complete application flowchart



Optimal application flowchart

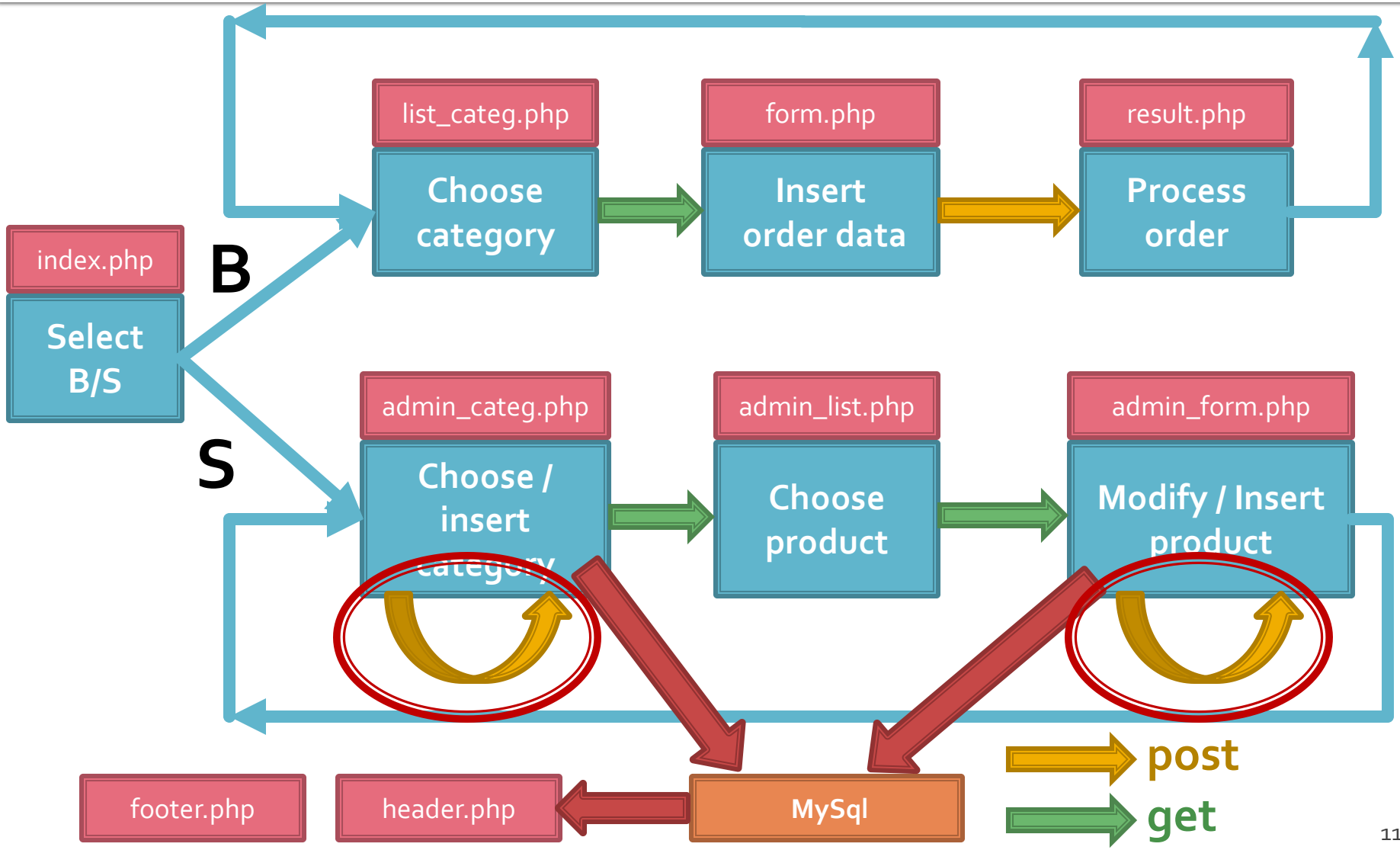


Individual assignment



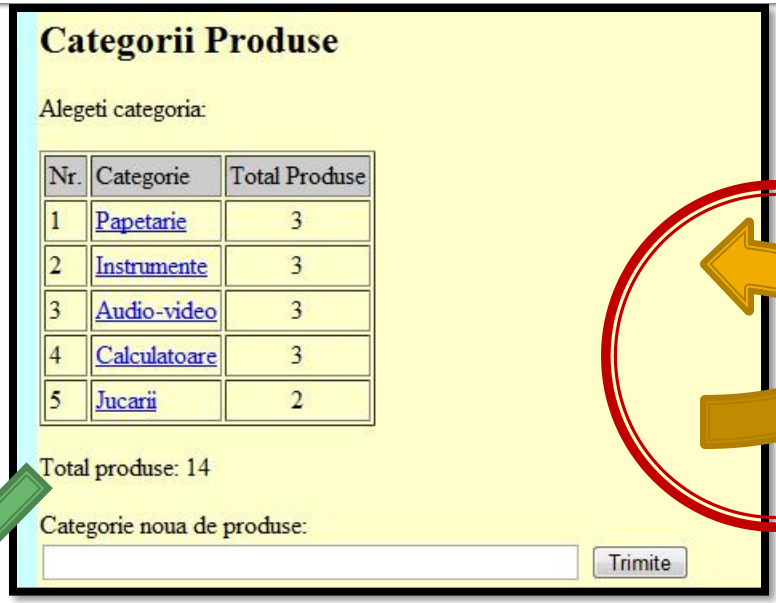
Single file for data collection AND processing

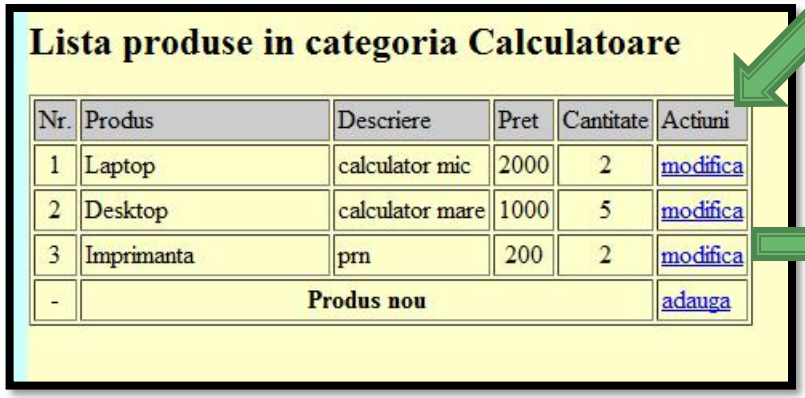
Complete application flowchart

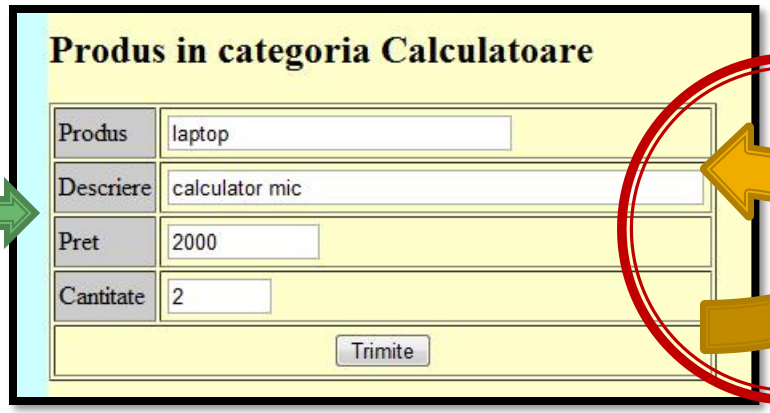


Flowchart (Seller)









post

get

Single file for data collection AND processing

- This option is often preferred
- It allows the unitary preservation of all operations for the performance of an action
  - easier access
  - ease of programming
  - avoiding errors: File does not exist: /Server/...
- The same file is initially used to collect data and then, if their presence is detected, for their processing

Single file for data collection AND processing

- The "action" file for <form> will be the current file
- it is recommended to use the global variable $_SERVER['SCRIPT_NAME']
  - flexibility when renaming files
- alternatively $_SERVER['PHP_SELF'] is not recommended
  - security issues
- The data collection section is displayed only in the absence of data

```
<form action="<?php echo $_SERVER['SCRIPT_NAME '];?>" method="post">
<p><input name="date_ok" type="submit" value="Trimite" /></p>
</form>
```

Single file for data collection AND processing

- The detection of the existence of the data is done by checking the existence (**isset**($variable)) of the user inserted values
  - for extra protection, their content can also be checked

```
if (isset($_POST[" date_ok "]))
        { //date trimise
        if ($_POST[" date_ok "]=="Trimite" )
                { // data sent by the current file
                // data processing
                }
        }
else
        {
        // data collection
<form action="<?php echo $_SERVER['SCRIPT_NAME '];?>" method="post">
<p><input name="date_ok" type="submit" value="Trimite" /></p></form>
        }
```

PHP Debug

View array content (debug)

```
echo "<pre>";
print_r ($matr);
echo "</pre>";
```

```
$matr= array (
"fruits"  =>
array("a" => "orange", "b" => "banana", "c" => "apple" ,
"ultim"),
"numbers" =>
array(1, 2, 3, 4, 5, 6),
"holes"   =>
array("first", 5 => "second", "third")
);
echo $matr;
echo "<pre>";
print_r ($matr);
echo "</pre>";
```

```
Array

Array
(
    [fruits] => Array
        (
            [a] => orange
            [b] => banana
            [c] => apple
            [0] => ultim
        )

    [numbers] => Array
        (
            [0] => 1
            [1] => 2
            [2] => 3
            [3] => 4
            [4] => 5
            [5] => 6
        )

    [holes] => Array
        (
            [0] => first
            [5] => second
            [6] => third
        )

)
```

Verify/debug PHP code

- It is recommended to use the array visualization option
  - In the file that receives the data
  - temporarily until the final version of the code
- the use of "verbose" code (manual) in the initial stages of writing PHP code can be extended to other types of data
  - the only (almost) debugging method in PHP
  - temp <?php echo "a=";echo $a; ?>

```
echo "<pre>";
print_r ($_POST);
echo "</pre>";
```

Debug

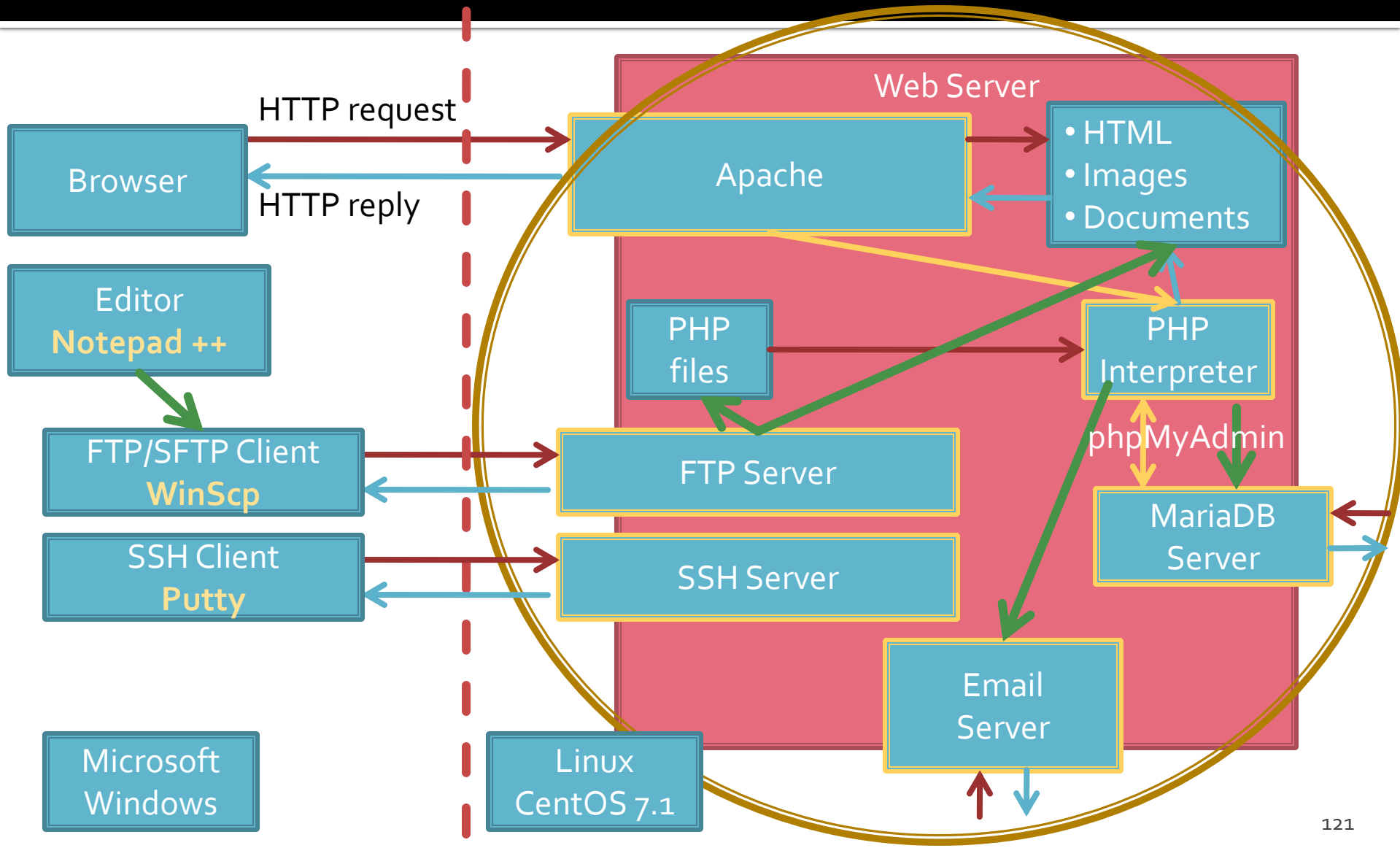
```
echo "<pre>";
print_r ($_POST);
echo "</pre>";
```

```
<p>temp res <?php echo "a=";echo $a; ?> </p>
```

```
echo "<pre>".print_r ($_GET,true)."</pre>";
```

Reference server and applications

Using LAMP


Web Server
HTTP request
HTTP reply
Browser
Apache
• HTML
• Images
• Documents
Editor
Notepad ++
PHP
files
PHP
Interpreter
FTP/SFTP Client
WinScp
FTP Server
phpMyAdmin
MariaDB
Server
SSH Client
Putty
SSH Server
Email
Server
Microsoft
Windows
Linux
CentOS 7.1

Using LAMP

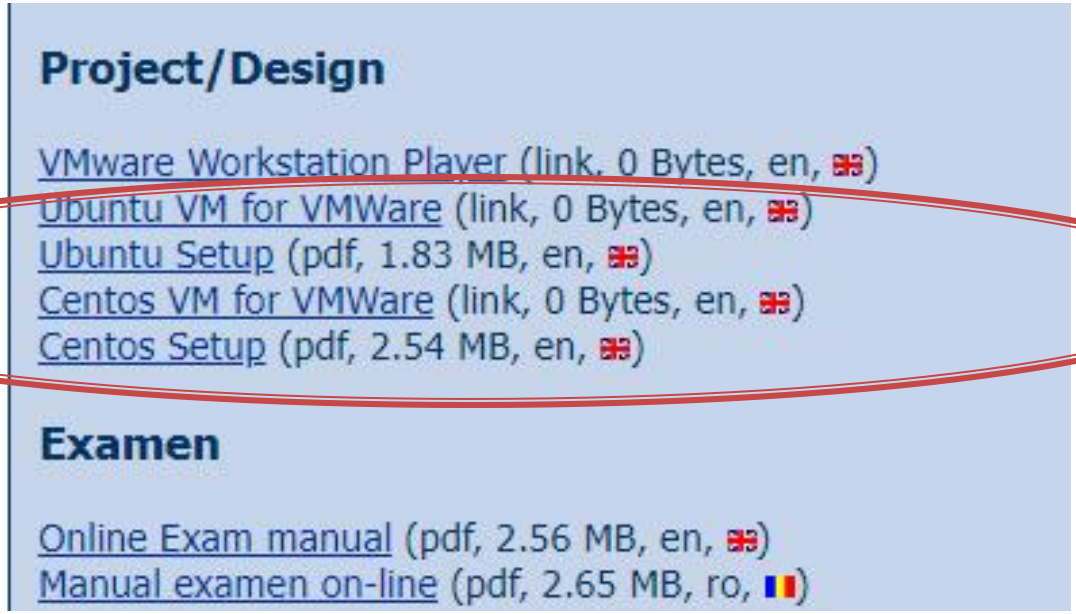
- differences from a Windows computer
  - system commands difficult
    - command prompt, SSH, Putty
  - files submitted by FTP
    - Copy/Paste unavailable
  - administration of MySql server:
    - using PhpMyAdmin (preloaded)
    - using other software on the host computer (eg. MySQL Workbench)

Using LAMP – Advantages

- Advantages
  - Available applications have newer versions (**2020 ~**)
    - CentOS/7.1, Apache/2.4.6, PHP/5.4.16, MariaDB/5.5.44, PhpMyAdmin/4.4.15
    - Ubuntu/20.04, Apache/ 2.4.41, PHP/ 7.4.3, MariaDB/ 10.3.31, PhpMyAdmin/4.9.5
    - Debian/12.5, Apache/ 2.4.57, PHP/ 8.2.7, MariaDB/ 10.11.6, PhpMyAdmin/5.2.1
  - Available applications very similar with real life hosting solutions
    - SSH
    - FTP
    - Email
      - for full functionality the network interface for the VM must be changed **Host-only** -> **Bridged**

Reference Server

- rf-opto.etti.tuiasi.ro > Master > Web Design



Project/Design

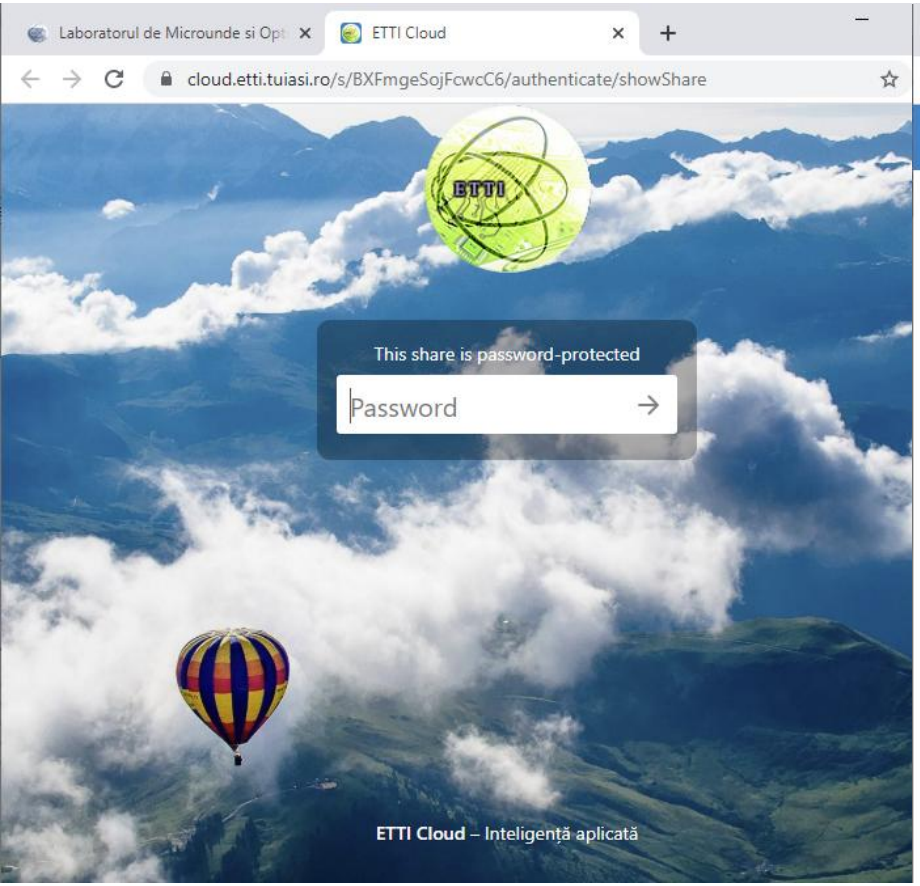
VMware Workstation Player (link, 0 Bytes, en, GB)
Ubuntu VM for VMWare (link, 0 Bytes, en, GB)
Ubuntu Setup (pdf, 1.83 MB, en, GB)
Centos VM for VMWare (link, 0 Bytes, en, GB)
Centos Setup (pdf, 2.54 MB, en, GB)

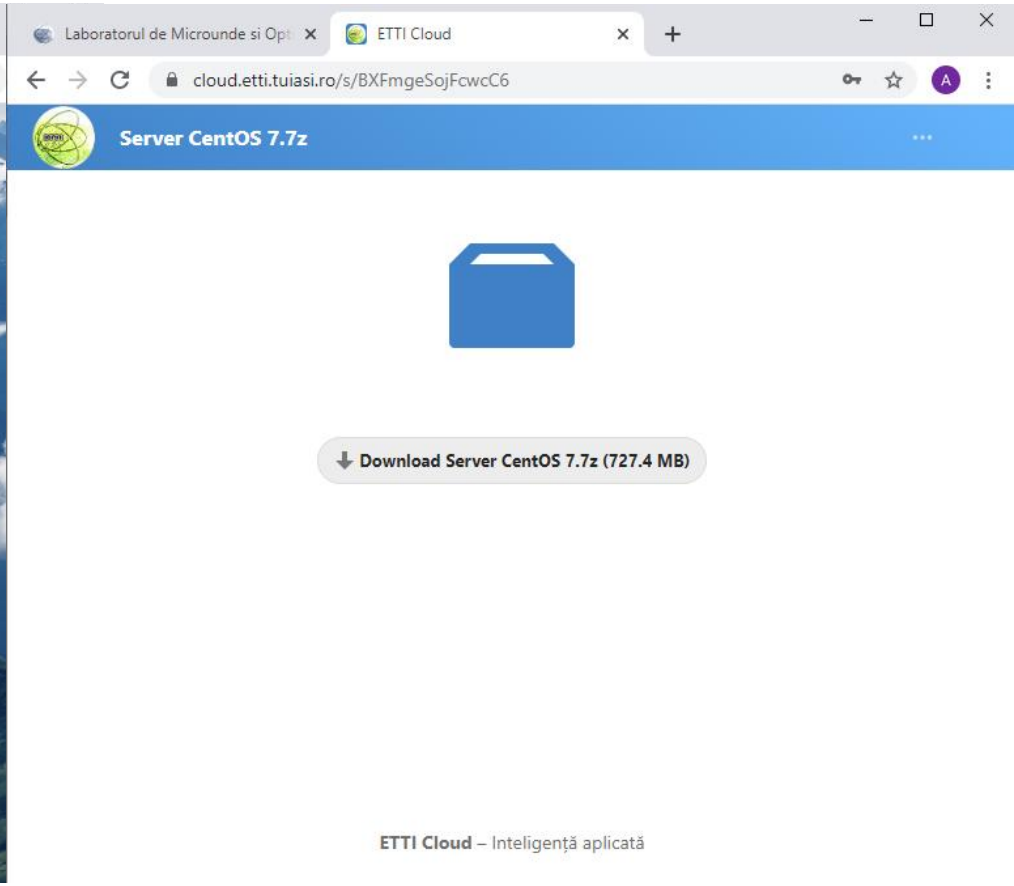
Examen

Online Exam manual (pdf, 2.56 MB, en, GB)
Manual examen on-line (pdf, 2.65 MB, ro, RO)

Reference Server

- Cloud ETTI: **RF-opto3#**





Server referinta

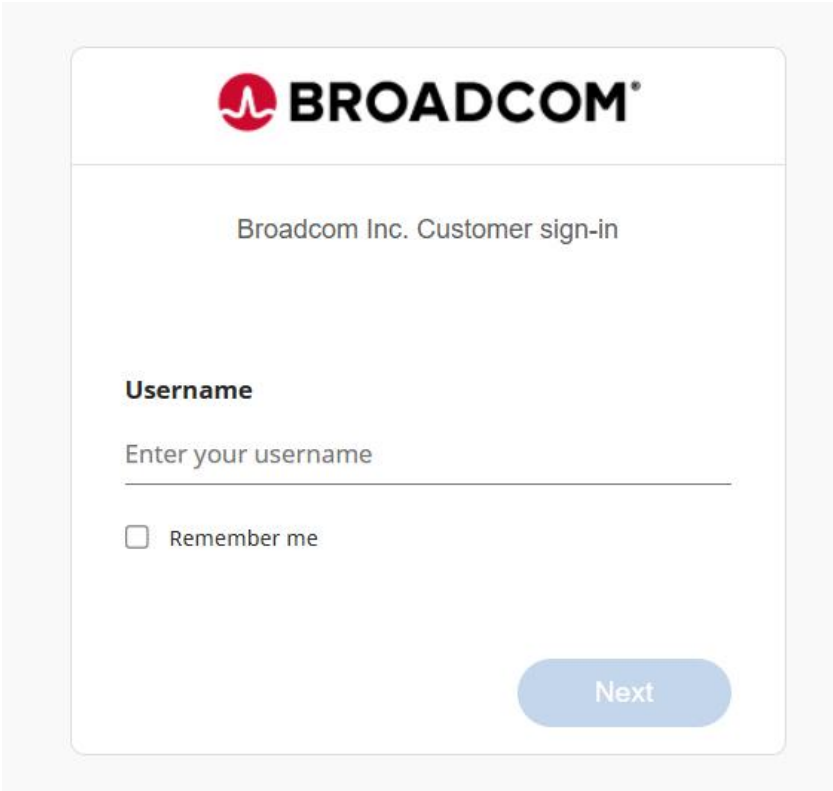
- Masina virtuala
- VMware Workstation Player
  - Gratuit (non-comercial)
  - https://www.vmware.com/products/workstation-player/workstation-player-evaluation.html
- Inlocuit de VMware Workstation Pro



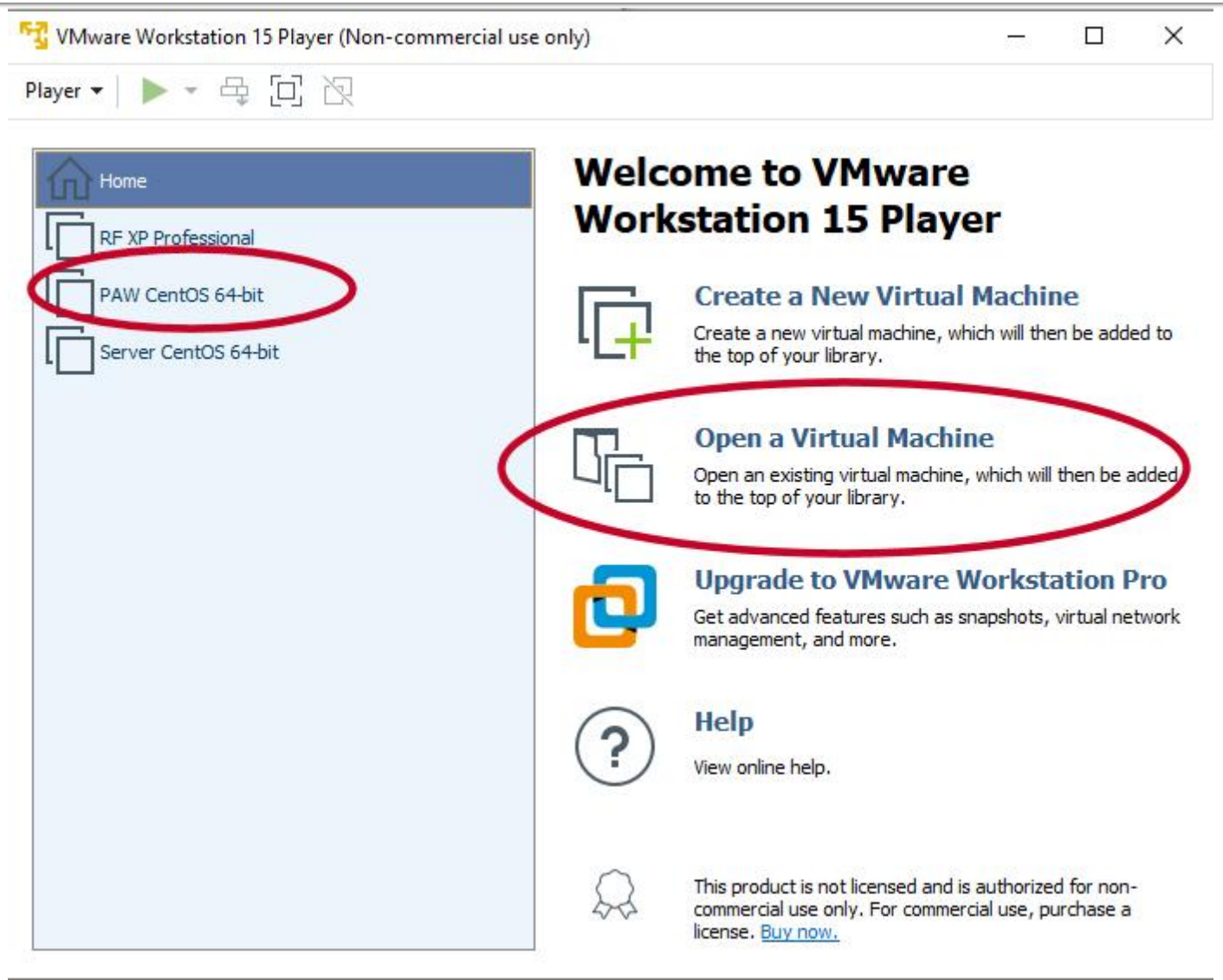
Server referinta

- Masina virtuala
- VMware Workstation Pro (Broadcom)





Reference Server

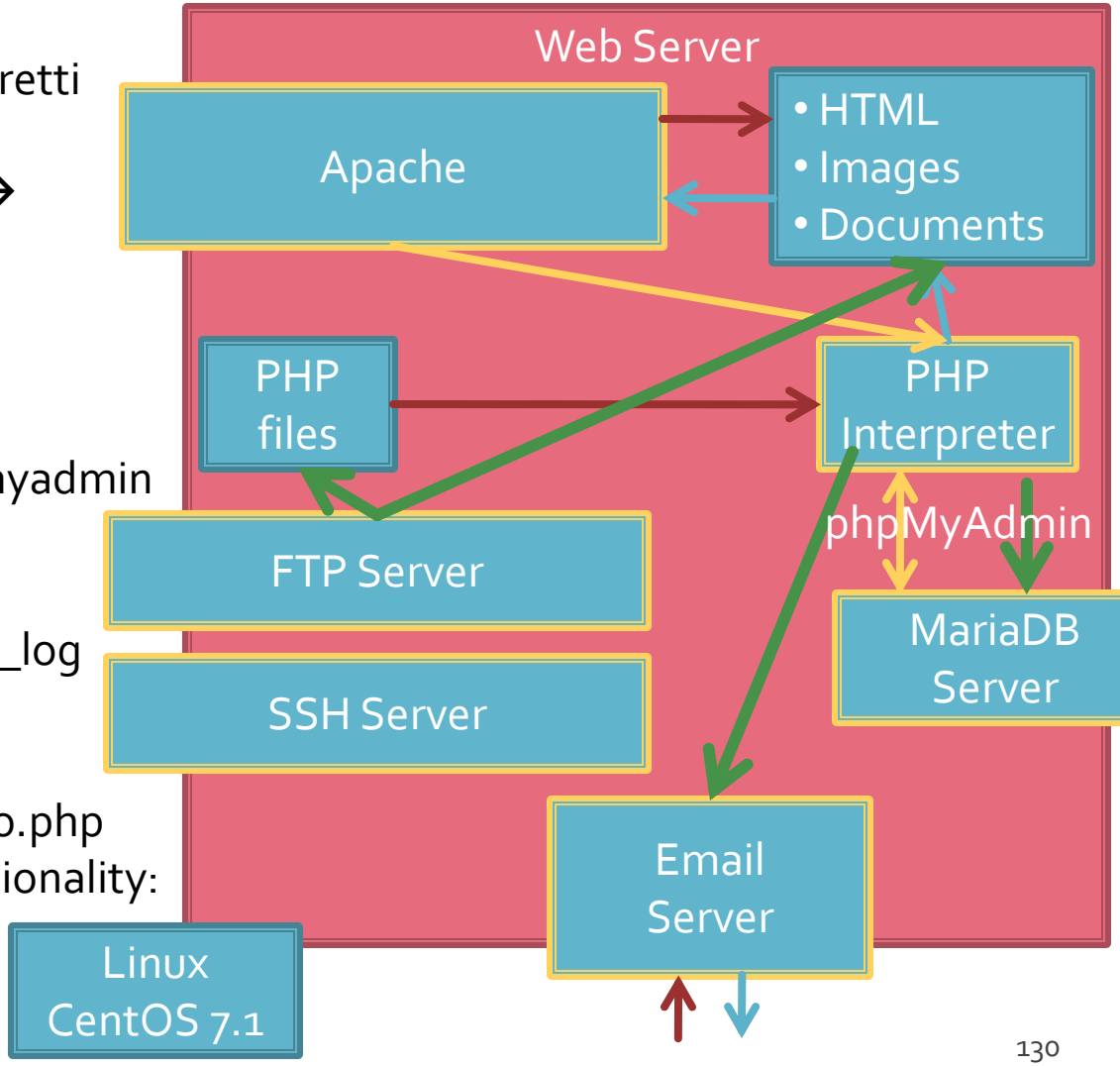


Possible problems

- Current VMWare Player runs **only** on **64bit** operating systems Windows/Linux
  - for 32bit operating systems previous (**certified originals**) can be made available on rf-opto
- The host computer **must** enable **Hardware Virtualization**
  - Hardware Virtualization is enabled in BIOS, depending on the PC manufacturer: Processor, Chipset, Northbridge
  - Options name: VT-x, AMD-V, Vanderpool, Hyper-V, SVM, Intel Virtualization Technology. if available: Intel VT-d, AMD IOMMU
- VM archive requires **7zip** native to the target operating system

Using LAMP

1. login → root:masterrc / paw:masteretti
2. ifconfig → 192.168.30.5
3. putty.exe → 192.168.30.5 → SSH → root:masterrc (remote login)
4. [other linux command line]
5. FTP → Winscp → SFTP → student:masterrc@192.168.30.5
6. MySql → http://192.168.30.5/phpmyadmin → root:masterrc / root:masteretti
7. Apache Error Log →
7a. putty → nano /var/log/httpd/error_log
7b. http://192.168.30.5/logfile.php (nonstandard)
8. PHP info → http://192.168.30.5/info.php
9. if DHCP service stops Apache functionality: service httpd restart



LAMP Reference Server 2023

- Linux, two variants
  - Centos 7.1
    - PHP 5.4.16
    - MariaDB 5.5.44
    - Apache 2.4.6
    - **root**/student:masterrc
  - Ubuntu 20.04 (**recommended**)
    - PHP 7.4.3
    - MariaDB 10.3.31
    - Apache 2.4.41
    - **paw**/student:masteretti
    - correction **paw FTP access**:
      - sudo usermod -a -G upload paw
      - sudo chmod -R 775 /var/www

LAMP Reference Server 2025

- Linux, three proposed servers
  - CentOS 7.1
  - Ubuntu 20.04
  - **Debian 12.5**

LAMP Reference Server

- Centos 7.1
  - PHP 5.4.16
  - MariaDB 5.5.44 / root:masterrc
  - Apache 2.4.6
  - PhpMyAdmin/4.4.15
  - **root**/student:masterrc
  - Python 2.7.5
  - creat: Workstation Player 12.x (**12**)

LAMP Reference Server

- Ubuntu 20.04
  - PHP 7.4.3
  - MariaDB 10.3.31 / root:masteretti
  - Apache 2.4.41
  - **paw**/student:masteretti
  - necesar suplimentar pentru **acces FTP user paw**:
    - sudo usermod -a -G upload paw
    - sudo chmod -R 775 /var/www
  - Python 3.8.10
  - creat: Workstation Player 15.x (**16**)

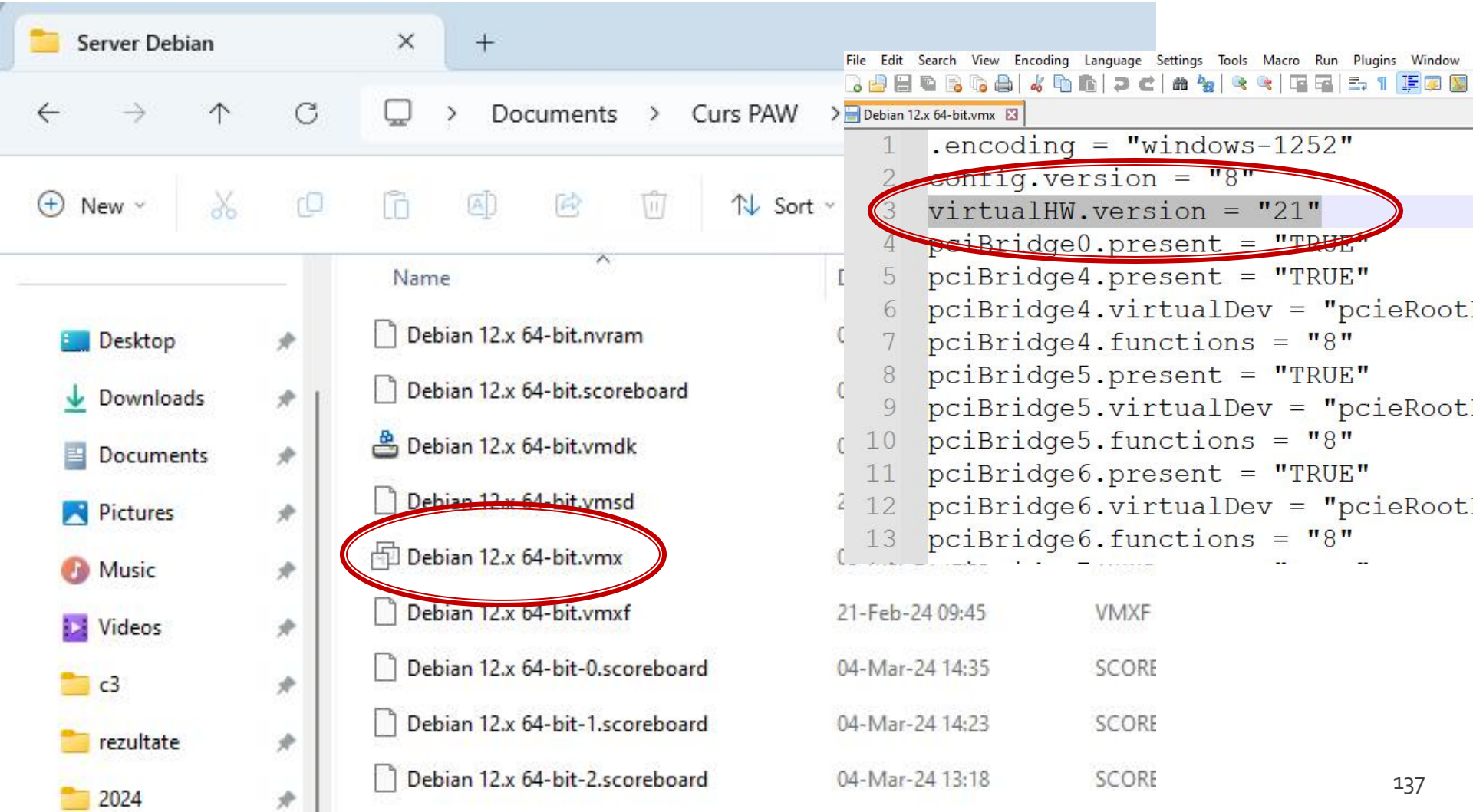
LAMP Reference Server

- Debian 12.5
  - PHP 8.2.7
  - MariaDB 10.11.6 / root:masteretti
  - Apache 2.4.57
  - PhpMyAdmin/5.2.1 deb
  - **root**/paw/student:masteretti
  - Python 3.11.2
  - creat: Workstation Player 17.5 (**21**)

LAMP Reference Server

- To run a server on an earlier version of VMware Player:
  - locate the server's "*.vmx" file
  - change virtualHW.version = "**21**" to a lower (earlier) value
    - in 2.13/1.08 -> **18**

LAMP Reference Server



Support applications

- WinSCP (client FTP, gratuit)
  - https://winscp.net/eng/download.php
- Notepad ++ (editor, avansat, gratuit)
  - https://notepad-plus-plus.org/downloads/
- Putty (remote access)
  - https://www.putty.org/
- MySQL Workbench (gratuit, cont Oracle)
  - https://www.mysql.com/products/workbench/
- Visual Studio Code (gratuit, Microsoft)
  - https://code.visualstudio.com/download

Aplicatii suport

- Variante portabile

Laboratory

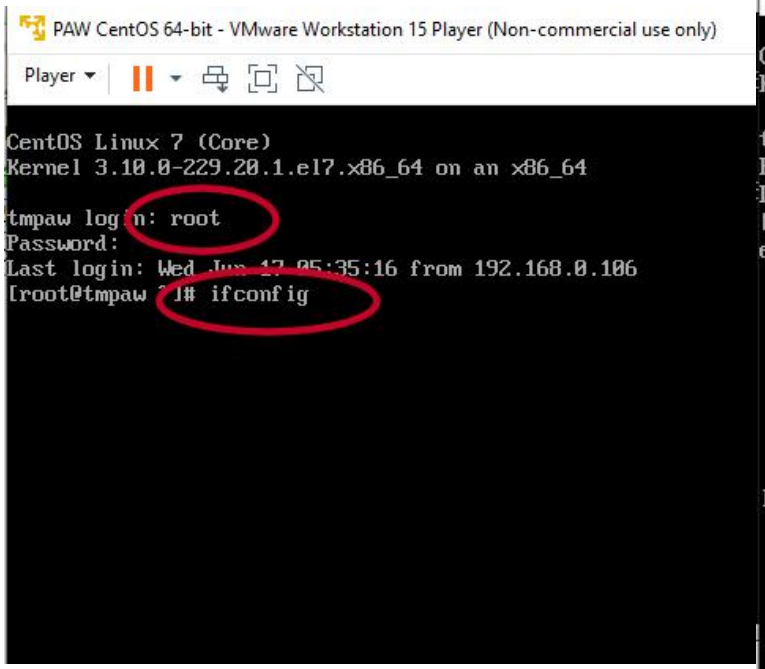
Laborator 1 (pdf, 1.48 MB, ro,)
Server Win2000 pentru VMWare Player - lab 1 (cloud) (link, 0 Bytes, en,)
Accesorii laborator (x32 - partial) (zip, 28.92 MB, en,)
Accesorii laborator (x64 - complet) (zip, 133.58 MB, en,)

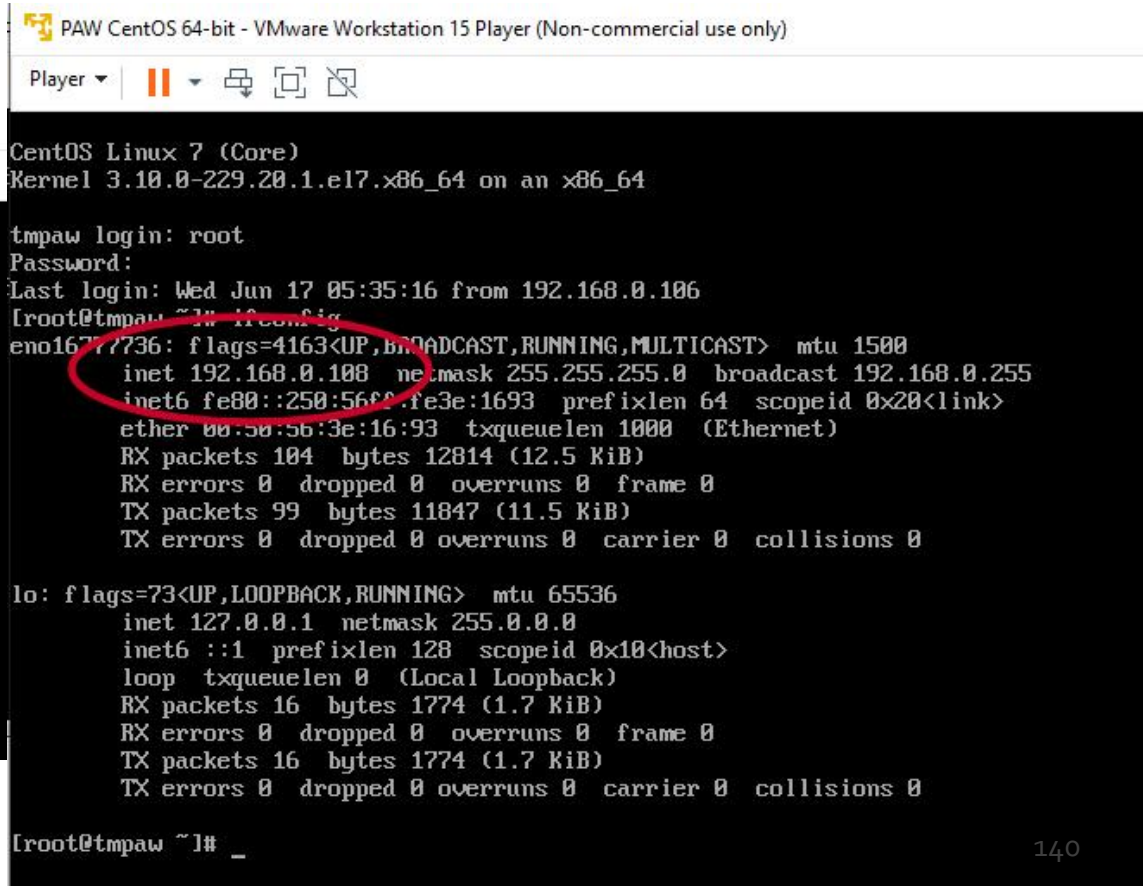
Project/Design

VMware Workstation Player (link, 0 Bytes, en,)
Server CentOS pentru VMWare Player (cloud) (link, 0 Bytes, en,)
Instalare Centos (pdf, 2.54 MB, en,)
Server Ubuntu pentru VMWare Player (cloud) (link, 0 Bytes, en,)
Instalare Ubuntu (pdf, 1.83 MB, en,)

IP address

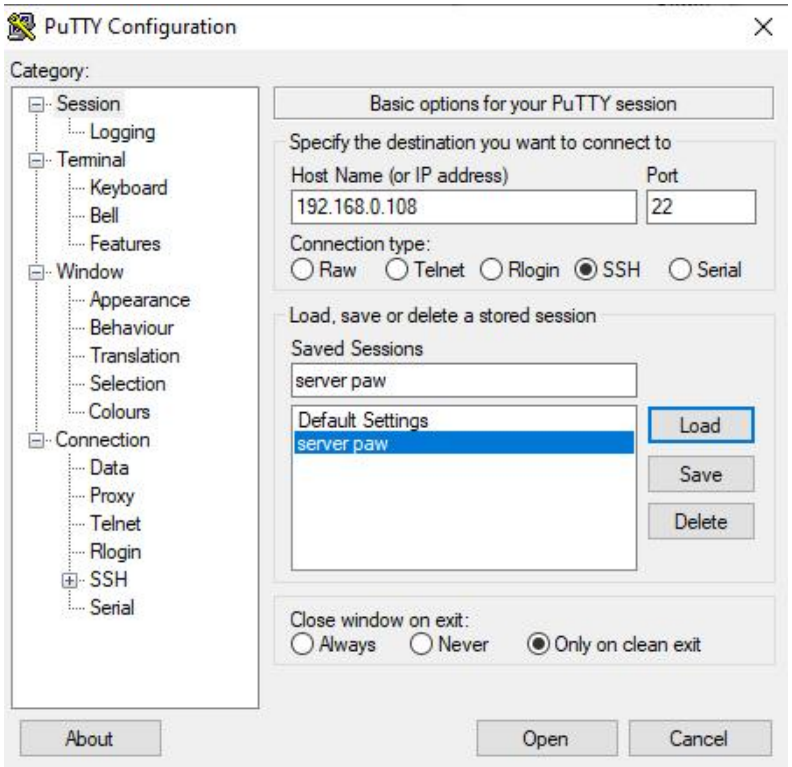
- login, ifconfig
- Ctrl + Alt + mouse

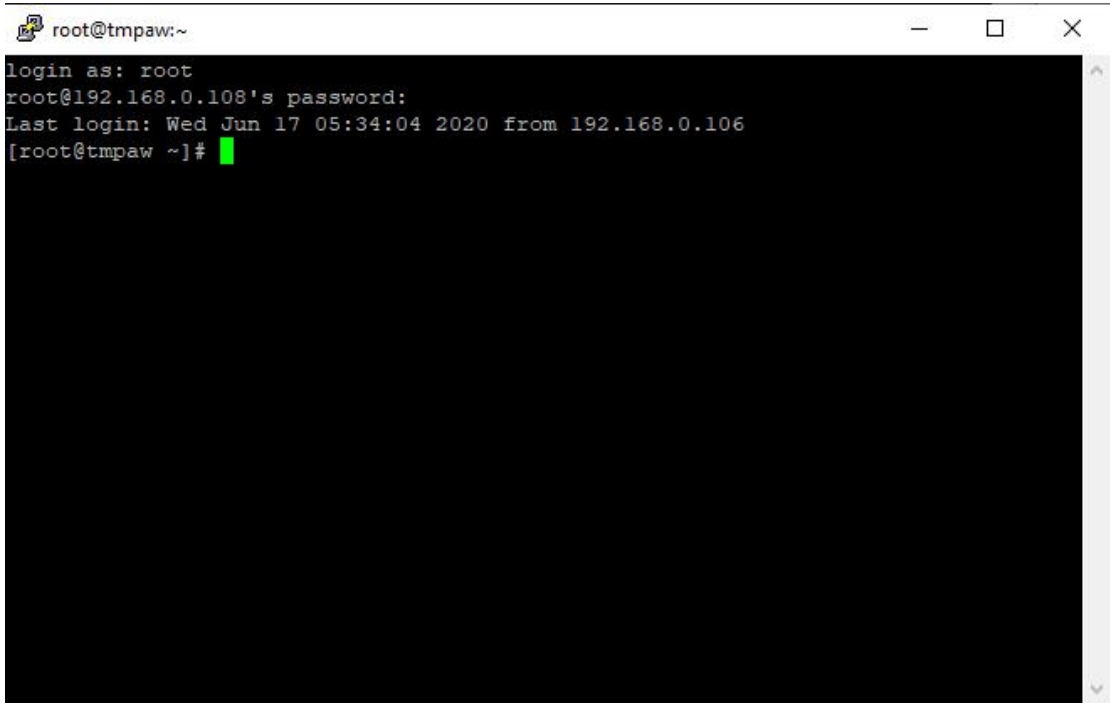




Putty

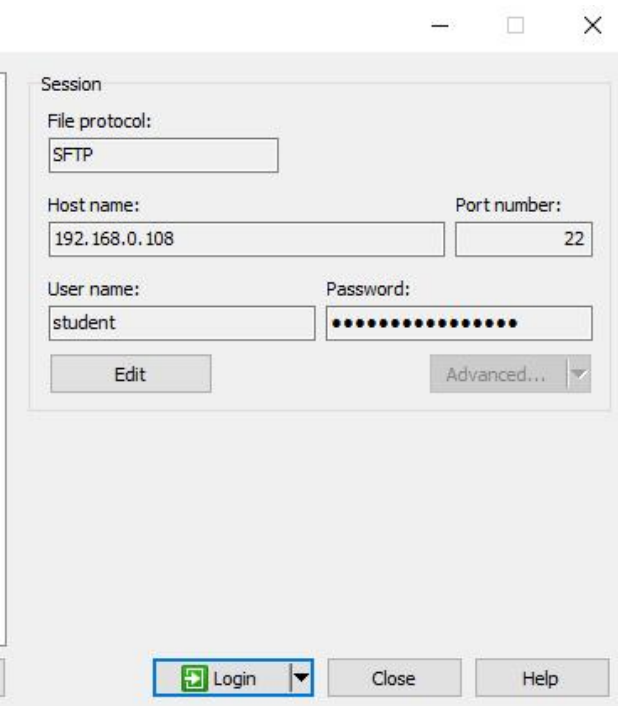
- putty.exe
- avoids mouse capture (CentOS), copy/paste etc.

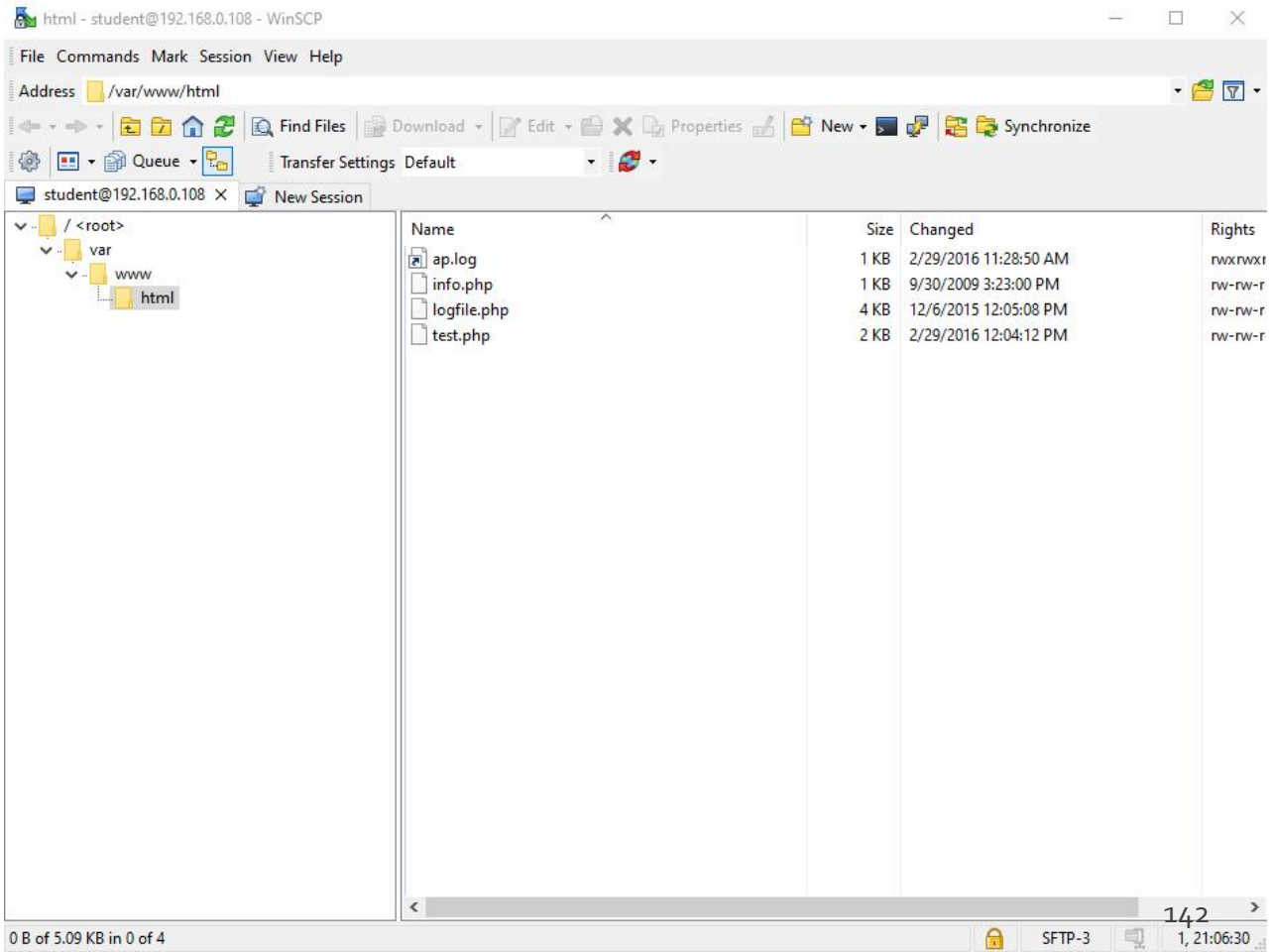




WinSCP

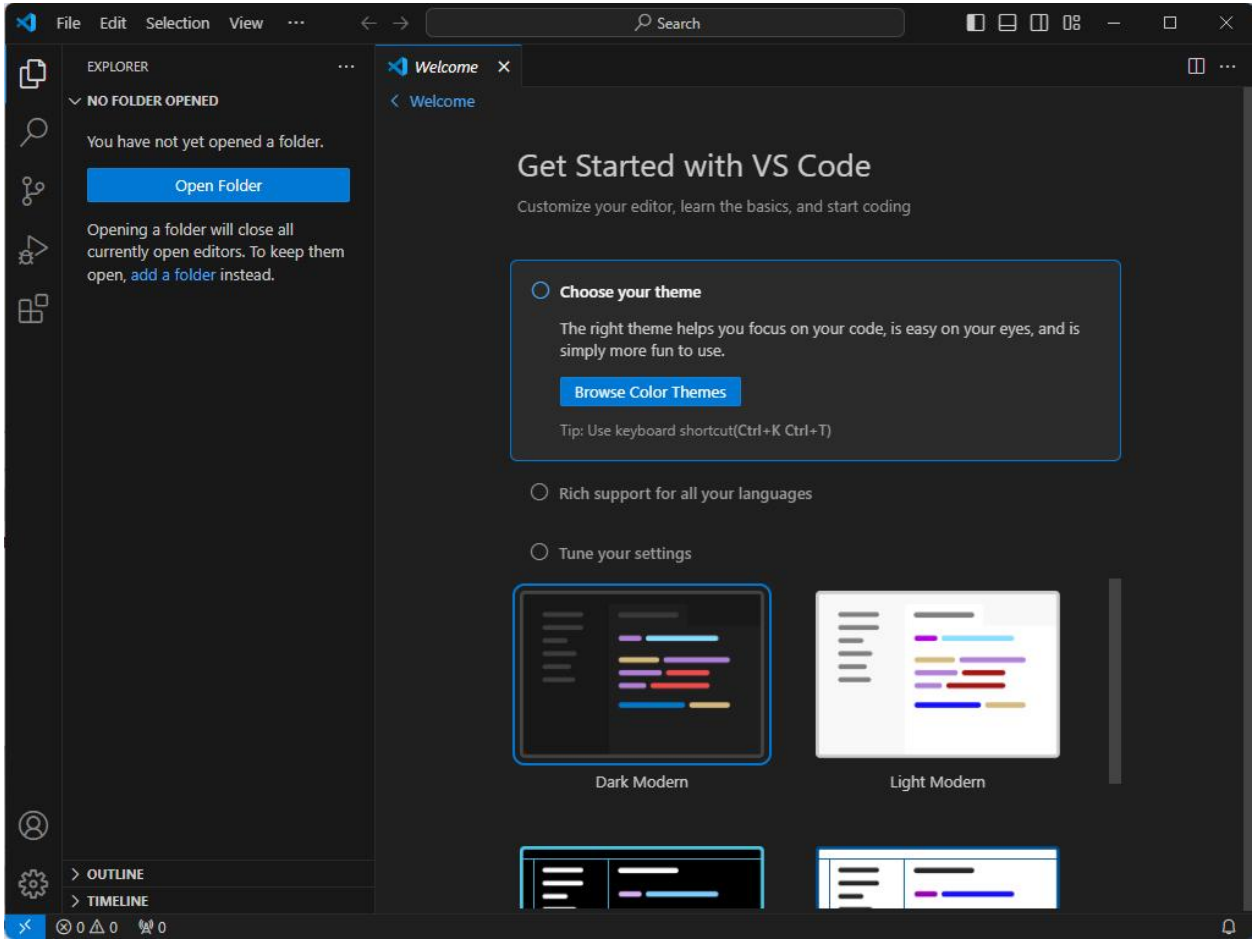
- FTP client
- upload files





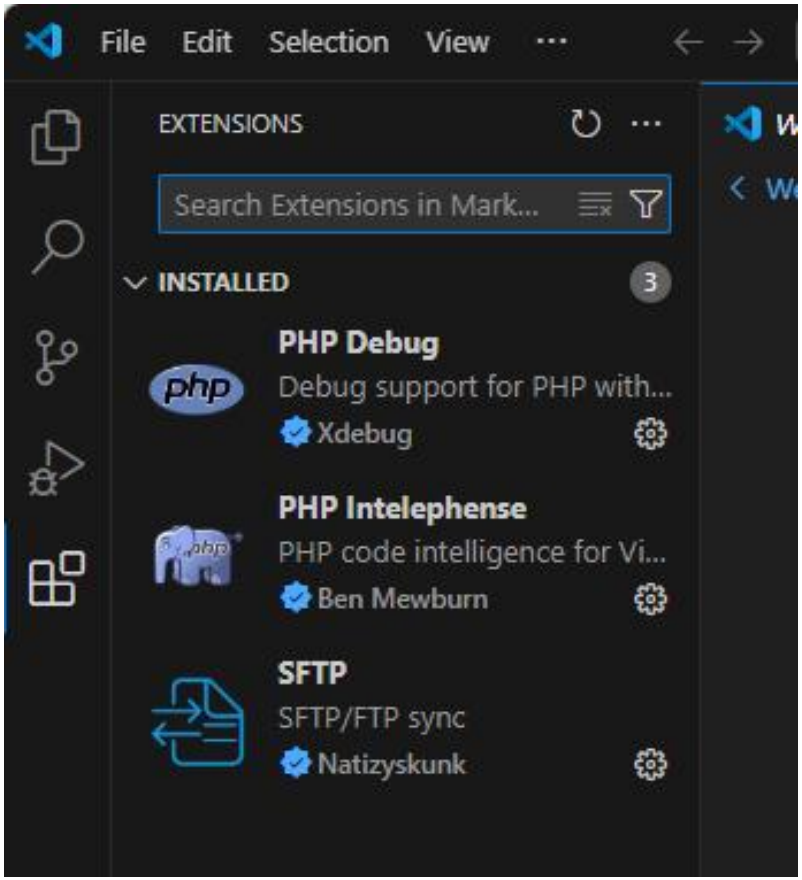
Visual Studio Code

- 1.87 Portable

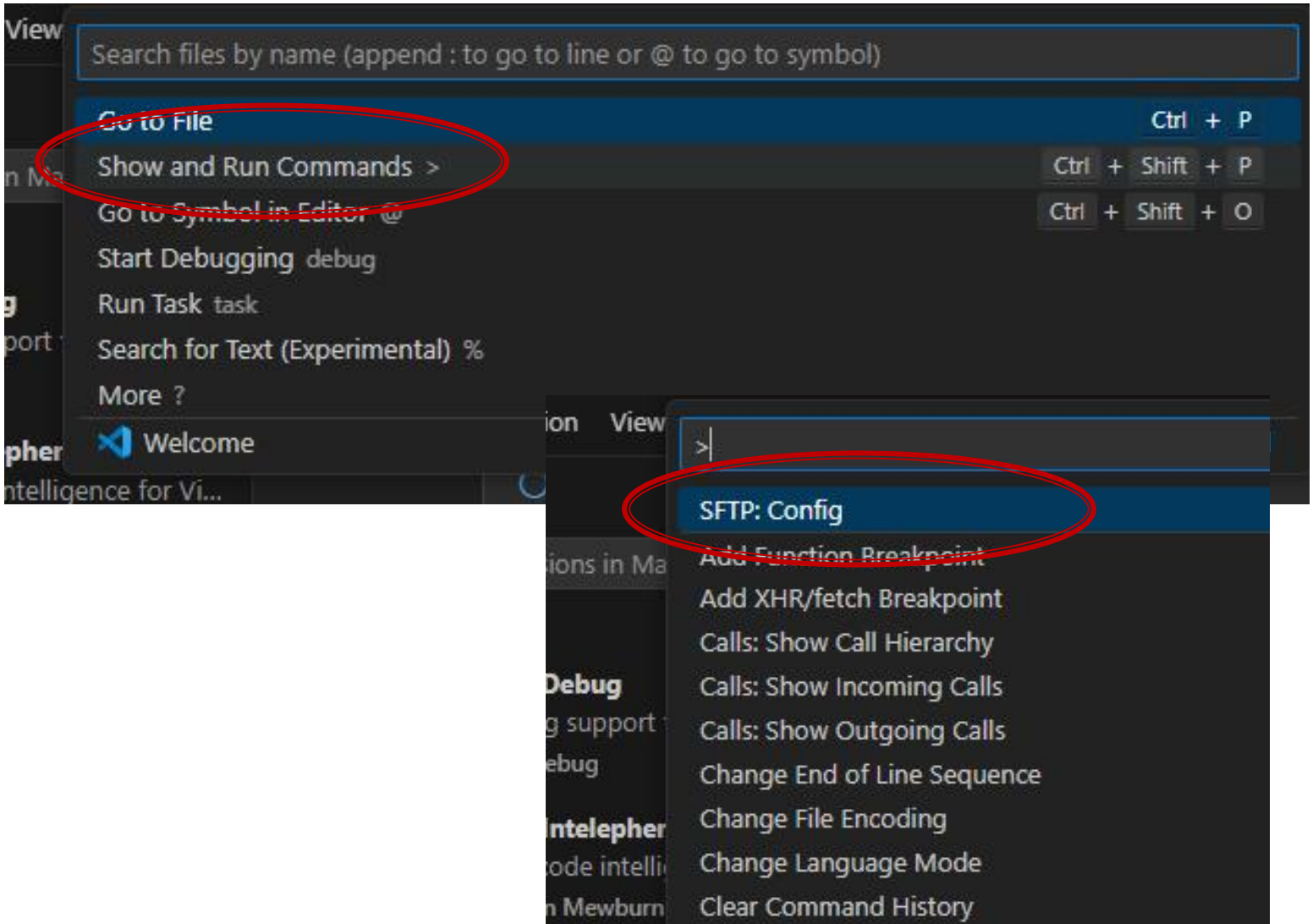


Visual Studio Code

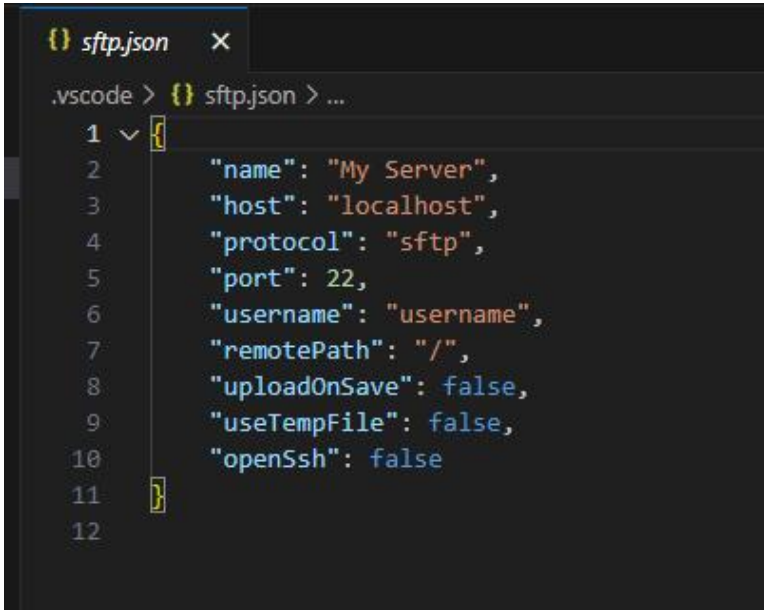
- Installed extensions
  - PHP Intelephense
    - PHP 8 -> Debian
  - PHP Debug (inactive for now)
  - SFTP – automatic save on server

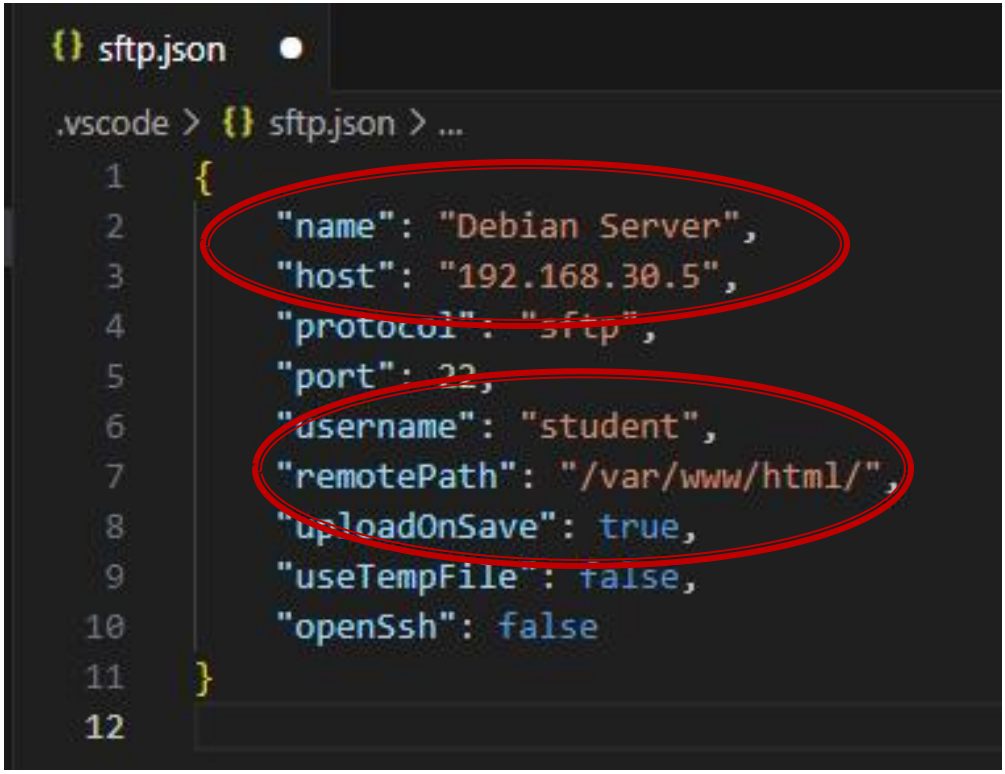


Visual Studio Code

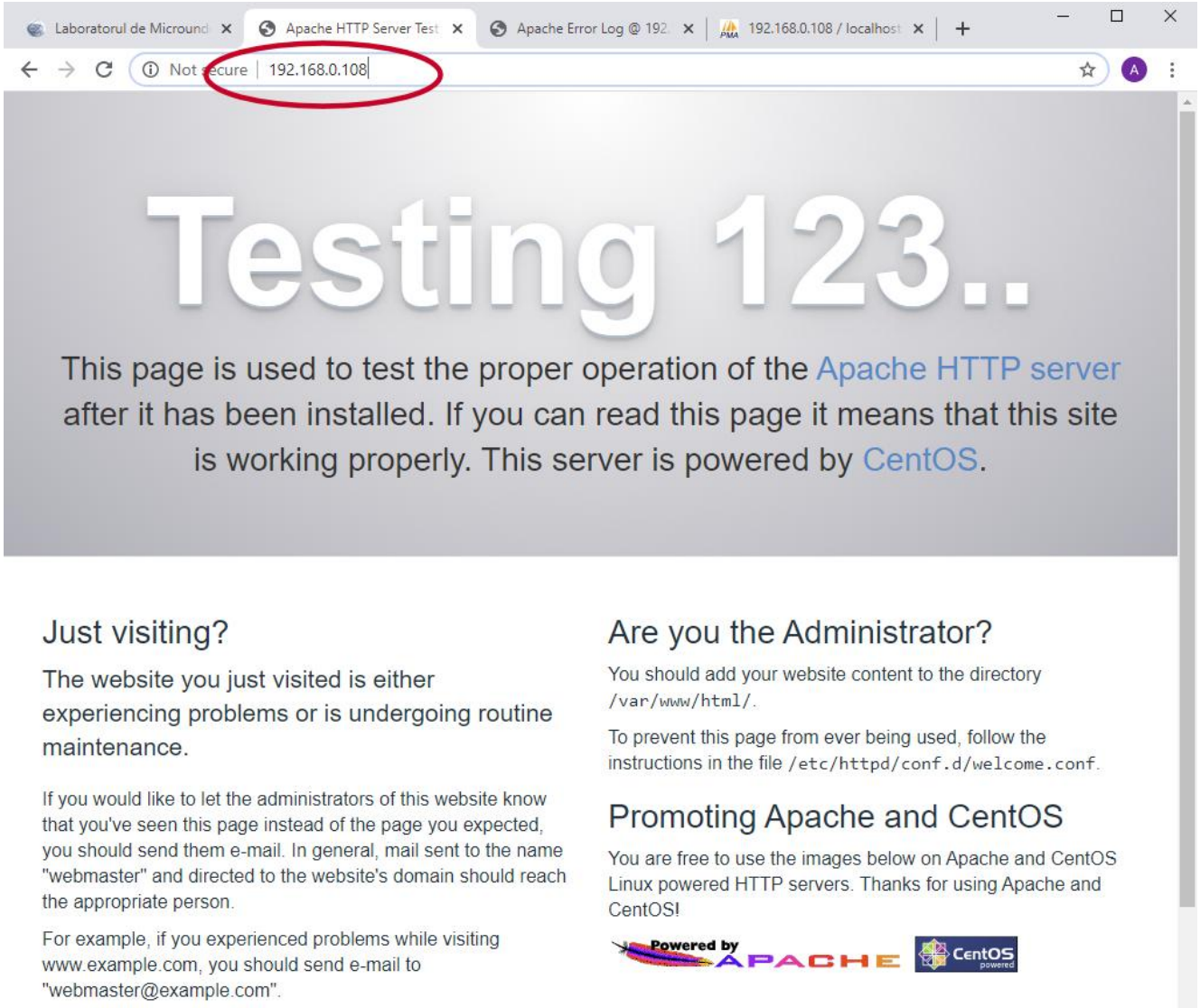


Visual Studio Code

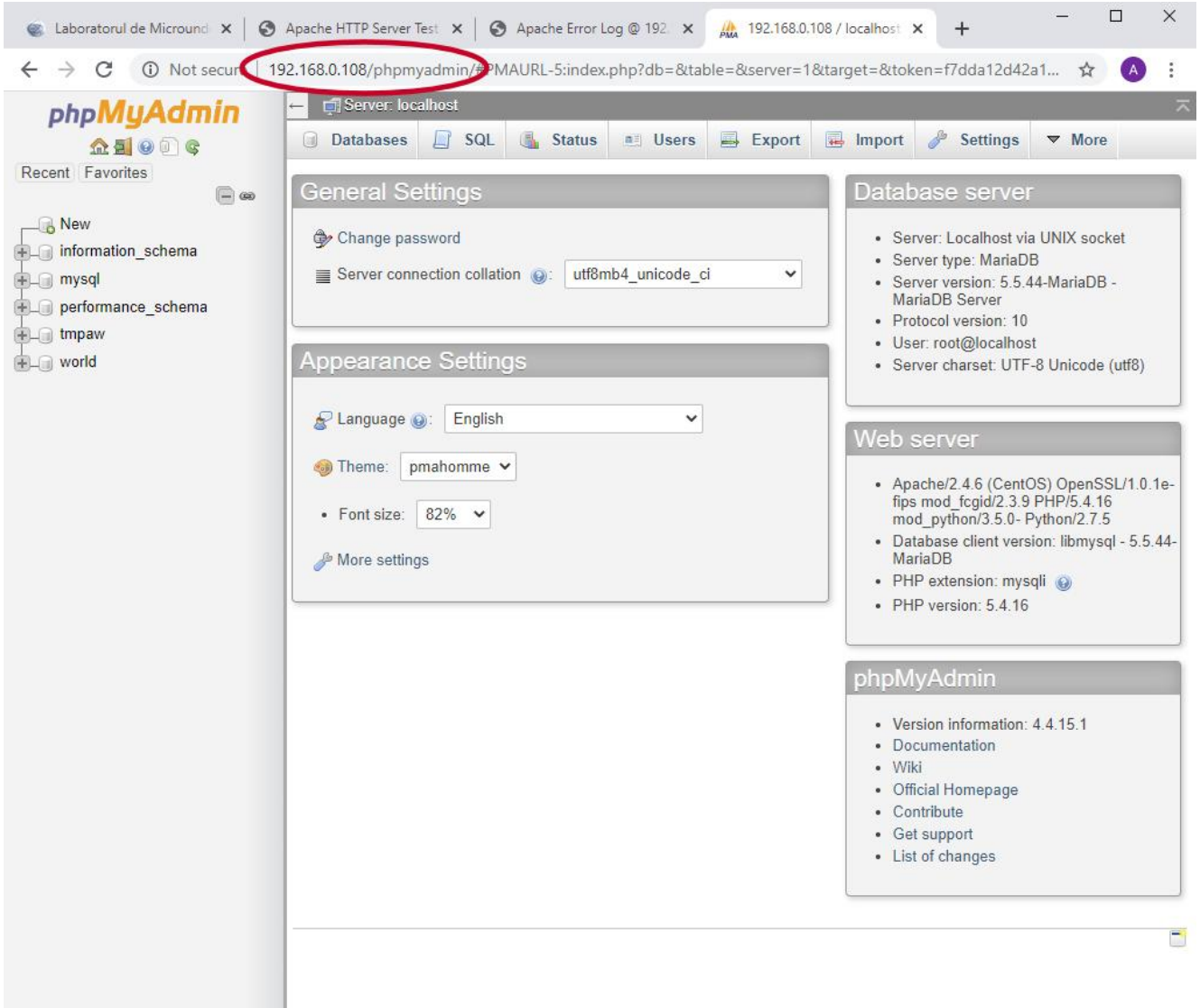




Browser


Laboratorul de Microunde
Apache HTTP Server Test
Apache Error Log @ 192.
192.168.0.108 / localhost
Not secure
192.168.0.108
Testing 123..
This page is used to test the proper operation of the Apache HTTP server after it has been installed. If you can read this page it means that this site is working properly. This server is powered by CentOS.
Just visiting?
The website you just visited is either experiencing problems or is undergoing routine maintenance.
If you would like to let the administrators of this website know that you've seen this page instead of the page you expected, you should send them e-mail. In general, mail sent to the name "webmaster" and directed to the website's domain should reach the appropriate person.
For example, if you experienced problems while visiting www.example.com, you should send e-mail to "webmaster@example.com".
Are you the Administrator?
You should add your website content to the directory /var/www/html/.
To prevent this page from ever being used, follow the instructions in the file /etc/httpd/conf.d/welcome.conf.
Promoting Apache and CentOS
You are free to use the images below on Apache and CentOS Linux powered HTTP servers. Thanks for using Apache and CentOS!
Powered by APACHE
CentOS powered

Server MySQL/MariaDB



Contact

- Laboratorul de microunde si optoelectronica
- https://rf-opto.etti.tuiasi.ro
- rdamian@etti.tuiasi.ro