

**Stredná odborná škola informačných technológií**

**Hlinícka 1, 831 52 Bratislava**

**Ročníková práca**

**Stredná odborná škola informačných technológií**

**Hlinícka 1, 831 52 Bratislava**

**Študijný odbor:** 2682 K mechanik počítačových sietí

**Téma:**

Využitie jazyka php pri tvorbe webu

## **VYHLÁSENIE**

Vyhlasujem, že som ročníkovú prácu vypracoval samostatne na základe vlastných  
Teoretických a praktických poznatkov s použitím uvedených zdrojov informácií.

Bratislava, 8.4.2022

.....

Nicolas Droppa

# OBSAH

|   |    |
|---|----|
| ÚVOD .....  | 6  |
| 1. HISTÓRIA PROGRAMOVACÍCH JAZYKOV .....                | 7  |
| 2. ROZDELENIE PROGRAMOVACÍCH JAZYKOV .....              | 8  |
| 2.1 PODĽA MIERY ABSTRAKCIE .....                        | 8  |
| 2.2 PODĽA SPÔSOBU PREKLADU .....                        | 8  |
| 2.3 PODĽA PRÍSTUPU K DÁTAM A FUNKCIÁM .....             | 9  |
| 3. NAJČASTEJŠIE POUŽÍVANÉ PROGRAMOVACIE JAZYKY .....    | 9  |
| 4. ČO JE PHP .....                                      | 10 |
| 4.1 HISTÓRIA PHP .....                                  | 10 |
| 4.2 BEZPEČNOSŤ PHP .....                                | 11 |
| 5. POUŽITIE JAZYKA PHP .....                            | 12 |
| 6. VÝZNAMNÉ PROJEKTY .....                              | 13 |
| 7. VÝHODY A NEVÝHODY PHP .....                          | 13 |
| 7.1 VÝHODY PHP .....                                    | 14 |
| 7.2 NEVÝHODY PHP .....                                  | 14 |
| 8. SYNTAX PHP .....                                     | 15 |
| 8.1 DÁTOVÉ TYPY .....                                   | 15 |
| 8.2 OBJEKTY .....                                       | 16 |
| 8.3 VPLYV VEĽKÝCH A MALÝCH PÍSMEN - Kľúčové slová ..... | 17 |
| 8.4 VPLYV VEĽKÝCH A MALÝCH PÍSMEN – premenné .....      | 18 |
| 8.5 OPERÁTORY .....                                     | 18 |
| 9 PRAKTICKÁ ČASŤ .....                                  | 19 |
| 9.1 VYTvorenie prvého súboru .....                      | 19 |
| 9.2 PREPOJENIE DATABÁZY A STRÁNKY .....                 | 20 |
| 9.2.1 VYTvorenie databázy .....                         | 20 |
| 9.2.2 VYTvorenie tabuliek .....                         | 20 |
| 9.2.3 ZAKOMPONOVANIE DATABÁZY .....                     | 21 |
| 9.3 PRÍPRAVA PODSTRÁNOK A ICH FUNKCIA .....             | 22 |
| 9.4 PODSTRÁNKY DOMOV A KONTAKT .....                    | 24 |
| 9.5 REGISTRÁCIA .....                                   | 24 |
| 9.6 PRIHLÁSENIE .....                                   | 26 |
| 9.7 NAVIGAČNÉ MENU PO PRIHLÁSENÍ .....                  | 28 |
| 9.8 ODHLÁSENIE .....                                    | 29 |
| 9.10 ZOBRAZENIE ČLÁNKOV .....                           | 30 |
| 9.11 PRIDANIE ČLÁNKOV .....                             | 30 |
| 9.12 ÚPRAVA ČLÁNKOV .....                               | 31 |

|                                    |           |
|------------------------------------|-----------|
| <b>9.13 VYMAZANIE ČLÁNKU .....</b> | <b>33</b> |
| <b>ZÁVER.....</b>                  | <b>34</b> |

## ÚVOD

Cieľom tejto práce je oboznámiť a naučiť čitateľa o programovacom jazyku Hypertext Preprocessor (skratkovo aj PHP) a o tvorbe webstránok v tomto jazyku. V tejto práci sa zameriavame podrobne na jednotlivé časti tvorby webstránok v tomto skriptovacom jazyku. V tejto práci sa čitateľ dočíta niečo o samotnej histórii jazyka, prejde si cez rozdelenie programovacích jazykov. Čitateľ postupne zistí ako sa dá jazyk použiť a dozvie sa niečo nové aj o používaných architektúrach s týmto jazykom. Čitateľ zistí základnú syntax jazyka, dočíta sa o významných projektoch zostrojených pomocou používania tohto skriptovacieho jazyka, zistí jeho výhody a nevýhody, prejde si poučením o jednotlivých operátoroch, ktoré používame pri práci s databázami a rozdelení operátorov tohto jazyka. V praktickej časti sa čitateľ naučí ako databázu vytvoriť, ako do novo vytvorenej databázy vytvoriť tabuľky, ako databázu prepojiť so stránkou, ako vytvoriť systém pre prihlasovanie, registráciu a odhlasovanie alebo ako sa dá pomocou databázy a php vytvárať, upravovať a mazať články zo stránky a databázy. Po dočítaní tejto práce by mal byť čitateľ schopný samostatne naprogramovať základnú webstránku vo forme blogu s prihlasovacím systémom bez väčších či menších problémov.

## 1. HISTÓRIA PROGRAMOVACÍCH JAZYKOV

Vývoj programovacích jazykov úzko súvisí so samotným vývojom počítačov. Za jeden z prvých pokusov o koncipovanie programátorských jazykov môžeme považovať prácu dcéry básnika Byrona, teda prácu Ady Lovelace. Ada Lovelace bola anglická matematická, no známa je predovšetkým vďaka tomu ako detailne opísala fungovanie Babbageovho mechanického počítača. Neskôr sa aj dostala k spolupráci s Charlesom Babbageom. Vzhľadom na to, že zomrela predčasne sa bohužiaľ zachoval iba program pre výpočet Bernoulliho čísel. Vlastne ona bola tá čo vymyslela prvý algoritmus spracovateľný počítačom a preto je označovaná ako prvá počítačová programátorka.

*Obrázok č. 1 – Ada Lovelace*

V roku 1882 vynášiel americký vynálezca a inžinier Herman Hollerith dierne štítiky. Po krátkom čase si dal tento stroj patentovať. Vlastne princíp toho stroju bol taký, že vyhodnocoval iba „áno“ a „nie“ podľa toho, či mal štítok otvor vydierovaný alebo nie. Toto bol prvý krok vpred k dnešnej technike.



Počas druhej svetovej vojny prichádza nemecký vedec Konrad Zuse s úplne novou radou počítačov Z1-3. No a v Spojených štátoch amerických vynášli MARK 1. Tieto zariadenia zohrali podstatnú rolu pro výpočtoch diferenciálnych rovníc, teda používali sa na automatické riadenie strelby. Tieto zariadenia mohli byť programované manuálne alebo prepojením jednotlivých funkčných blokov.

Za revolučný bod v histórii programovania sa považuje rok 1944 keď maďarský matematik John Von Neumann definuje novú koncepciu počítačov. Celý program a zároveň dáta sa ukladali do pamäte.

Približne v päťdesiatych rokoch začala stúpať náročnosť úloh ktoré museli samotné počítače riešiť, tak začal stúpať aj výkon počítačov. Strojový kód predstavoval pre programátorov zložitý proces, preto sa vytvorili autokódy. Z týchto autokódov sa postupne vytvoril programovací jazyk Assembler.

## 2. ROZDELENIE PROGRAMOVACÍCH JAZYKOV

Je niekoľko kritérií podľa ktorých môžeme programovacie jazyky deliť. Kategórie do ktorých sa delia programovacie jazyky sú:

- Podľa miery abstrakcie
- Podľa spôsobu prekladu
- Podľa prístupu k dátam a funkciám

Obrázok č.2 - JavaScript



### 2.1 PODĽA MIERY ABSTRAKCIE

Podľa miery abstrakcie delíme programovacie jazyky na Vyššie alebo Nižšie.

- **Vyššie programovacie jazyky:** Programovacie jazyky spadajúce pod túto kategóriu sú pre človeka viac zrozumiteľné, znižujú pravdepodobnosť výskytu chýb. Väčšina programov písaná v týchto jazykoch sú krátke a ľahko čitateľné, sem patria napríklad Java, C++, C#, ...

- **Nižšie programovacie jazyky:** Inštrukcie týchto programovacích jazykov zodpovedajú príkazom procesora. Ľahko sa prevádzajú do strojového kódu procesoru. Tieto programovacie jazyky sú zvyčajne ťažšie osvojiteľné. Programy písané v takýchto jazykoch sú rýchle a menej náročné na pamäť, sem patrí napríklad Assembler.

### 2.2 PODĽA SPÔSOBU PREKLADU

Podľa spôsobu prekladu poznáme jazyky:

- **Kompilované:** Vždy sú pred spustením preložené kompilátorom, tieto jazyky sú náročnejšie na správne zapísaný kód, sem patria napríklad: Pascal, C, ...
- **Interpretované:** Pre jeho spustenie je nevyhnutný zdrojový kód a program menom interpret. Ich výhodou je to, že sa dajú ľahko preniesť na inú platformu, sem patria napríklad: JavaScript, BASIC, PHP, ...



## 2.3 PODĽA PRÍSTUPU K DÁTAM A FUNKCIÁM

Podľa prístupu k dátam a funkciám delíme programovacie jazyky na Procedurálne a Neprocedurálne programovacie jazyky.

- **Procedurálne programovacie jazyky = imperatívne**

Toto sú jazyky kde sa realizuje výpočet pomocou príkazov. Procedurálne programovacie jazyky sa ďalej delia na:

- a) **Štruktúrované**

- b) **Objektovo orientované**

A to sú napríklad: ObjectiveC, C++, Python, JavaScript, ...

Obrázok č.3 – C++



- **Neprocedurálne programovacie jazyky = deklaratívne**

V týchto jazykoch sa realizuje programovanie pomocou definícií. Neprocedurálne programovacie jazyky sa ďalej delia na:

- a) **Funkcionálne**

(základom je formálny výpočtový model). Patria sem: Lisp APL/J ML (Standard ML) Sisal, Miranda

- b) **Logické**

(prostriedkom programovania je použitie matematickej logiky)

## 3. NAJČASTEJŠIE POUŽÍVANÉ PROGRAMOVACIE JAZYKY

**JavaScript** – Tento jazyk nemá nič spoločné s jazykom Java ako by si mnohí ľudia mohli myslieť. Tento jazyk sa najnovšie využíva pri tvorení interaktívnych webstránok.

**Python** – Python bol vytvorený v roku 1989 holandským programátorom Guido van Rossumom. Základnou ideológiou tohto jazyku je jednoduchosť a čitateľnosť.

**C** – Jedná sa o imperatívny, kompilovaný jazyk. Vďaka kvalite kompilátora sa v ňom dá vytvárať programy, ktorých rýchlosť ide prirovnať k jazyku Assembler.

**Java** – Jeden z najpopulárnejších jazykov na tvorbu webových aplikácií. Pôvodne bola Java vymyslená pre napredovanie spotrebnej elektroniky no neskôr sa jej cesta obrátila na web.

**Php** – PHP bol vytvorený ako jednoduchý Perl/CGI skript na počítanie počtu návštev a ich zobrazovanie na stránke. Pri rastúcom záujme sa rozhodol PHP rozšíriť aj o ďalšie funkcie a celý ho vyvíjať programovacím jazykom C. Z PHP sa nakoniec stal veľmi výkonný skriptovací jazyk.

## 4. ČO JE PHP

PHP ( Hypertext Preprocessor ) je populárny open source skriptovací jazykom ktorý sa využíva najmä na programovanie aplikácií na strane servera a pre vývoj dynamických webových stránok. PHP bolo inšpirované mnohými jazykmi podporujúce procedurálne programovanie no najviac si prebralo od programovacieho jazyka C a jazyka Perl. Neskôr bol tento jazyk rozšírený o možnosť používať objekty. PHP je možné použiť aj na vývoj aplikácií s užívateľským rozhraním vďaka modulárnemu návrhu. Tento skriptovací jazyk dokáže spolupracovať s relačnými databázami, ako napríklad MySQL, Oracle, IBM, ..., pričom si stále zachováva jednoduchú a priamočiaru syntax.



*Obrázok č. 4 - PHP*

PHP beží na takmer všetkých najrozšírenejších operačných systémoch, vrátane UNIXu, Linuxu, Windows alebo aj na Mac OS X. Zároveň taktiež spolupracuje s najrozšírenejšími webovými servermi.

### 4.1 HISTÓRIA PHP

PHP bolo pôvodne navrhnuté ako niekoľko skriptov v jazyku Perl, neskôr boli tieto skripty prepísané do jazyka C. Autorom bol Rasmus Lerdorf v roku 1994. No však o rok Ramus

Lerdorf zverejnil skripty tohto jazyku pod názvom “Personal Home Page Tools“. Kombináciou s ďalším jeho programom Form Interpreter vzniklo PHP/FI. Neskôr v roku 1997 dvaja izraelskí vývojári menami Zeev Suraski a Andi Gutmans prepísali syntaktický analyzátor na novšiu verziu, ktorá sa stala základom PHP 3. V roku 1999 sa Suraski a Gutmans opäť pustili do prepísania jadra, no tentokrát už založili aj spoločnosť Zend Technologies, ktorá sa odvtedy podieľa aj na ďalšom vývoji PHP. V roku 2000 bolo vydané PHP 4, ktorého jadro tvoril nový Zend Engine 1.0. Ďalej v roku 2004 bola vydaná ďalšia verzia s názvom PHP 5 a s jadrom Zend Engine II.



*Obrázok č. 5 – Ramus Lerdorf*

## **4.2 BEZPEČNOSŤ PHP**

Ak by sme sa pozreli do záznamov z National Vulnerability Database, tak by sme zistili že okolo 30% všetkých zaznamenaných zraniteľností sú spojené s PHP. No však prevažná väčšina týchto zraniteľností sú spôsobené tým, že programátori sa neriadia alebo nedodržia pravidiel praktického programovania. Samotné technické chyby tohto jazyka alebo jeho knižníc nie sú také časté, podľa prieskumu robeného v roku 2008 to bolo okolo jedného percenta zo všetkých chýb. Vzhľadom na to, že programátori robia často chyby pri Programovaní, niektoré jazyky začali obsahovať kontrolu chýb na automatické zisťovanie nedostatočnej validácie vstupu, ktorá vedie k mnohým problémom. Takáto funkcia je vyvíjaná aj pre PHP, no v minulosti bolo jej zahrnutie a vydanie niekoľkokrát zamietnuté.

## 5. POUŽITIE JAZYKA PHP

Tento skriptovací jazyk je špeciálne navrhnutý na tvorbu a prácu s webstránkami bežiacimi na webovom serveri. Celý PHP kód je vykonávaný pomocou PHP runtime, aby mohol dynamicky vytvoriť obsah na webovej stránke. Taktiež môže byť využitý na skriptovanie z príkazového riadku alebo ako už bolo v práci spomenuté, môže byť použitý na klientovo orientované aplikácie s grafickým rozhraním. PHP môže byť nasadené na väčšine Webových serverov, operačných systémov, platformách a môže sa používať v spojení s mnohými relačnými databázami. Veľa webhostingov dokonca ponúka podporu PHP pre svojich klientov. Jedna z výhod PHP je, že je zadarmo dostupné a PHP Group poskytuje zdrojový kód pre používateľov, preto aby ho mohli dotvárať, meniť alebo rozširovať podľa vlastných potrieb a pre ich vlastné použitie. PHP sa primárne správa ako filter, ktorý má na vstupe súbor, alebo prúd dát obsahujúci text alebo PHP inštrukcie a na výstupe prúd dát najčastejšie vo forme HTML. PHP od verzie PHP 4, PHP parser kompiluje vstup preto aby vyprodukoval byte kód pre spracovanie pomocou Zend Engine, ktorý má za úlohu poskytovať vyšší výkon ako jeho interpreti z minulosti. PHP bolo pôvodne vytvorené na tvorbu webových stránok, no v súčasnosti sa zameriava najmä na skriptovanie na strane servera a je podobné iným serverovo orientovaným skriptovacím jazykom, ktoré poskytujú klientom dynamický obsah z webového servera. Patrí sem napríklad ASP .NET od spoločnosti Microsoft, JavaServer Pages od Sun Microsystems alebo mod\_perl. LAMP architektúra sa stala veľmi populárnou vo webovom priemysle ako spôsob vývoja webových aplikácií. Skratka LAMP znamená:

L – Písmeno L odkazuje na operačný systém “Linux“.

A – Písmeno A odkazuje na “Apache“, teda http server.

M – Písmeno M odkazuje na relačnú databázu “MySQL“.

P – Písmeno P odkazuje na “PHP“ alebo aj “Python“ a ešte “Perl“. Samozrejme to môže znamenať aj kombináciu všetkých troch.



Obrázok č. 6 – LAMP

Podobné balíky ako napríklad WAMP a MAMP sú taktiež dostupné pre Windows a OS X, kde prvé písmenka reprezentujú operačný systém. PHP je používaný ako serverovo orientovaný programovací jazyk na 75% zo všetkých web stránok, ktorých je známy ten serverovo orientovaný programovací jazyk a PHP je jeden z najpoužívanejších open source softvérov v spoločnostiach.

## 6. VÝZNAMNÉ PROJEKTY

**Facebook** – rozsiahla sociálna sieť je implementovaná v PHP. (vid' Obrázok č. 7)

**MediaWiki** – software pro tvorbu webových projektov typu wiki, napríklad Wikipédia.

**WordPress** – publikačný systém pre tvorbu blogov a podobných aplikácií.

(vid' Obrázok č. 8)

**Nette Framework** – framework pre tvorbu webových aplikácií v PHP.

**Texy!** – prekladač intuitívnej syntaxe pre formátovanie textu na HTML.

**phpBB** – balík pre prevádzkovanie webového fóra.

**phpMyAdmin** – obľúbená webová aplikácia. (vid' Obrázok č. 9)

**Adminer** – webová aplikácia pre správu databázového systému MySQL.



Obrázok č. 7 – Facebook



Obrázok č. 8 – WordPress



Obrázok č. 9 – phpMyAdmin

## 7. VÝHODY A NEVÝHODY PHP

Skriptovací jazyk PHP má mnoho výhod ale aj pár nevýhod, v podkapitole 8.1 sa pozrieme na výhody tohto skriptovacieho jazyka a v kapitole 8.2 sa pozrieme na pár nevýhod ktoré má skriptovací jazyk PHP. Ak by sme sa pozerali čisto počtovo tak počtom tie výhody určite prevažujú nevýhody.

## 7.1 VÝHODY PHP

- PHP je špecializované na webové stránky.
- Rozsiahly súbor funkcií v základnej knižnici PHP (cez 5,5 tisíc), ďalšie funkcie v PECL.
- Natívna podpora veľa databázových systémov.
- Multiplatformnosť (hlavne Linux a Microsoft Windows).
- Možnosť využitia natívnych funkcií operačného systému (možná nekompatibilita s iným operačným systémom).
- Strmá krivka učení.
- Obrovská podpora na hostingových službách – PHP je fakticky štandardom, ktorý je prakticky všade.
- Obrovské množstvo projektov a kódov, ktoré ide zadarmo využiť (WordPress, phpBB a mnoho ďalších).
- Pomerne slušná dokumentácia.
- Veľmi slobodná licencie, ktorá (v protikladu k napríklad GPL) neobsahuje takzvaný copyleft.

## 7.2 NEVÝHODY PHP

- Nekonzistentné pomenovávanie funkcií, napríklad:  
  
    strpos(), strchr(), ale str\_replace(), str\_pad().  
  
    Nejednotné názvoslovia skupín funkcií, napríklad: mysql\_XXXX, imap\_XXXX, json\_XXXX (s podčiarknutím) versus imageXXXX, bcXXXX, gzXXXX (bez podčiarknutia).
- Nejednotné poradie parametrov, napríklad.: array\_map() vs. array\_filter().
- V štandardnej distribúcii chýba ladiaci nástroj.
- Po spracovaní požiadavky neudržiava kontext aplikácie, vytvára ju vždy znovu (teda oslabuje výkon).

## 8. SYNTAX PHP

PHP skript môže byť umiestnený kdekoľvek v súbore no súbor musí mať ukončenie .php aby skript fungoval. PHP spracuje len kód, ktorý je ohraničený špeciálnymi znakmi, respektíve ešte aj prezývané ako “tagy“, ostatný text odovzdá na výstup bez nejakej zmeny. Najvšeobecnejšie používané tagy sú: `<?php` a `?>`. Použiť je však možné aj `<script language=“php“>` a `</script>`, takisto skrátená forma `<?>`, alebo `<?=` (na výpis reťazca alebo premennej) a `?>`. V takom prípade, že skript obsahuje skrátené formy týchto špeciálnych ohraničovacích znakov, sa skript stáva menej prenosným, vďaka tomu, že ich rozoznávanie je možné v PHP konfigurácii zakázať. Celkovým zmyslom týchto “tagov“, respektíve špeciálnych znakov je rozpoznávať PHP časť kódu a oddeliť ho od zvyšnej časti kódu. Názvy premenných začínajú vždy symbolom \$ (dolár), samotné premenné nemusia mať ani dopredu definovaný typ. Na rozdiel od názov funkcií a tried, v názvoch premenných rozlišujeme veľké a malé písmená. Tabulátor, medzera a nový riadok sú v skriptovacom jazyku PHP interpretované ako biele znaky, teda kdekoľvek v kóde nezáleží na ich počte, ani poradí

(samozrejme výnimkou je vnútro textového reťazca). V PHP existujú tri druhy komentárov.

`/* ... */` - na označenie blokového komentáru

`// ...` - na označenie jednoriadkového komentáru

`# ...` - na označenie jednoriadkového komentáru

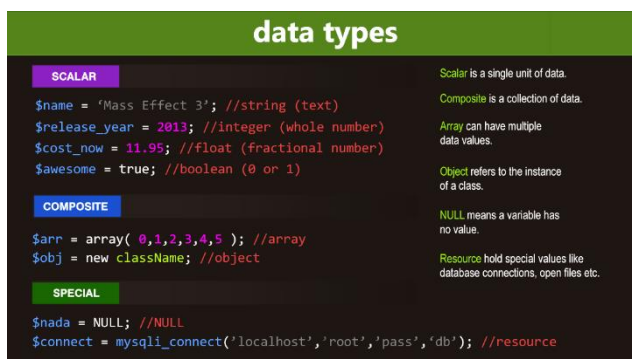
Syntaxou a kľúčovými slovami pripomína PHP jazyk C a jazyky z neho odvodené.

Podmienенý príkaz **if**, cykly **for** a **while** a návratové hodnoty funkcií sú syntakticky veľmi blízke jazykom ako C, C++, Java, alebo Perl.

### 8.1 DÁTOVÉ TYPY

PHP ukladá celé čísla v rozsahu závislom od platformy (teda 64 bitov alebo 32 bitov) ako celé číslo so znamienkom ekvivalentne k typu long v programovacom jazyku C. Celé čísla bez znamienka sú v istých situáciách konvertované na hodnoty so znamienkom, čím sa tento skriptovací jazyk odlišuje od iných programovacích jazykov. Premenné typu celé čísla môžu mať určené používanie desiatkovej, osmičkovej alebo šestnástkovej notácie. Desatinné čísla sú taktiež zoradené podľa rozsahu závislého od platformy. Môžu používať

notáciu desatinných čísel, alebo dve formy vedeckej notácie. PHP má prirodzený booleovský typ, ktorý je podobný prirodzenému booleovskému typu v Jave a C++. Pri použití booleovskej konverzie pravidiel sú nenulové hodnoty interpretované ako pravda, respektíve TRUE a nula je interpretovaná ako nepravda, teda FALSE tak ako aj v programovacích Perl a C++. Prázdný (teda NULL) dátový typ je reprezentovaný ako premenná, ktorá nemá hodnotu. Jej jediná hodnota, ktorú môže nadobudnúť je NULL.



Obrázok č. 10 – Dátové typy

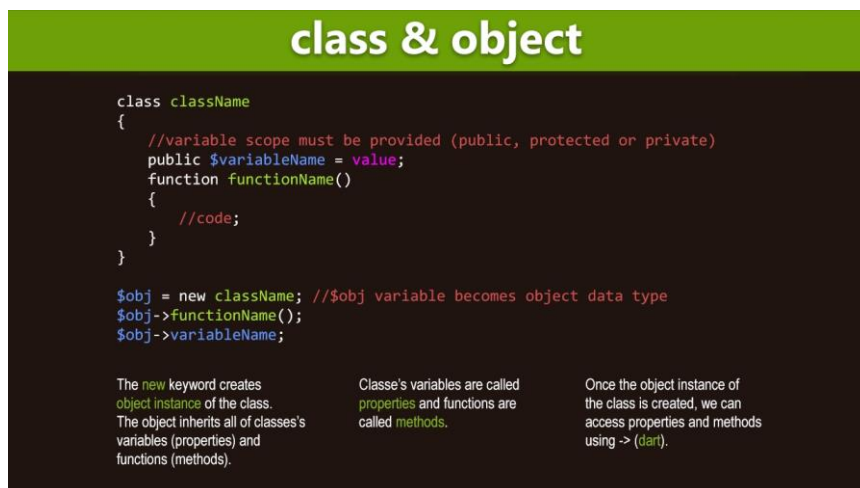
Premenné typu „resource“ reprezentujú odkazy na prostriedky z externých zdrojov. Tieto sa zvyčajne vytvárajú funkciami z konkrétneho doplnku a môžu byť spracované iba funkciami z rovnakého doplnku. Príkladom môžu byť súbory, obrázky alebo databázové zdroje. Polia môžu obsahovať elementy hocakého typu, ktorý skriptovací jazyk PHP dokáže spracovať, vrátane zdrojov, objektov a taktiež aj ďalších polí. Poradie je zachované v listoch hodnôt a v hashoch s hodnotami a kľúčmi, pričom sa môžu navzájom prelínať. PHP taktiež podporuje reťazce, ktoré môžu byť použité s jednoduchými úvodzovkami, dvojitémi úvodzovkami, nowdoc alebo heredoc syntaxou. Štandardná PHP knižnica sa pokúša riešiť bežné problémy a implementuje efektívne rozhrania pre dátové prístupy a triedy.

## 8.2 OBJEKTY

Základná funkcionálna objektovo orientovaného programovania bola pridaná vo verzii PHP 3 a vylepšená vo verzii PHP 4. Zaobchádzanie s objektmi bolo kompletne prepísané vo verzii PHP 5, keď bolo zároveň rozšírené o nové funkcie a vylepšené z hľadiska výkonnosti. V predošlých verziách PHP bolo k objektom pristupované ako k hodnotovým typom. Nedostatkom tejto metódy bolo to, že celý objekt bol skopírovaný, keď bola hodnota priradená, alebo posunutá ako parameter nejakej metóde. V novom prístupe sú objekty odkazované podľa rukoväte a nie podľa hodnoty. PHP 5 predstavilo súkromné a



chránené premenné a metódy objektu, spolu s abstraktnými triedami, konečnými triedami, abstraktnými metódami a konečnými metódami. Taktiež predstavilo štandardný spôsob deklarovania konštruktorov a deštruktorov podobných ako v iných objektovo orientovaných jazykoch ako napríklad C++ a aj štandardný model spracovávanía chyby.



Obrázok č. 11 – objekty v php

Okrem toho PHP 5 pridalo rozhrania a povolilo implementáciu viacerých rozhraní. Existujú špeciálne rozhrania, ktoré objektom dovoľujú interakciu so systémom runtime. Objekty implementujúce prístup k poľu môžu byť použité so syntaxou poľa a objekty implementujúce iteráciu alebo agregáciu iterácie môžu byť použité s konštrukciou foreach. Neexistuje žiadna funkcia virtuálnych tabuliek v jadre, takže statické premenné sú počas kompilácie späté s menom namiesto s referenciou. Ak vývojár vytvorí kópiu objektu pomocou rezervovaného slova clone, Zend engine skontroluje, či bola definovaná metóda \_\_clone() alebo nie. Ak nie je tak sa zavolá predvolená metóda \_\_clone(), ktorá skopíruje vlastnosti objektu. Ak je metóda \_\_clone() definovaná, tak ona bude zodpovedná za nastavenie potrebných vlastností vo vytvorenom objekte. Pre pohodlie, jadro zaobstaráva funkciu, ktorá importuje vlastnosti zdrojového objektu, takže programátor môže začať s úplnou kópiou zdrojového objektu a iba nahradiť vlastnosti, ktoré potrebuje zmeniť.

### 8.3 VPLYV VEĽKÝCH A MALÝCH PÍSMEN - KĽÚČOVÉ SLOVÁ

Kľúčové slová v php ako napríklad (echo, while, if, else) alebo aj funkcie nie sú case sensitive, teda nezáleží či bude príkaz napísaný štýlom, kde v kľúčovom slove budú napríklad prehádzané na striedačku veľké a malé písmená (viď obrázok číslo 12).

```
<?php
ECHO "Hello World!<br>";
echo "Hello World!<br>";
EcHo "Hello World!<br>";
?>
```

Obrázok č. 12 – Vplyv veľkých a malých písmen na kľúčové  
slová

V takomto prípade by program napísal tri krát pod seba Hello World...

```
Hello World!
Hello World!
Hello World!
```

Obrázok č. 13 – Vplyv veľkých a malých písmen na kľúčové slová –  
výsledok príkladu

## 8.4 VPLYV VEĽKÝCH A MALÝCH PÍSMEN – PREMENNÉ

Pri názvoch premenných v programovacom jazyku php si treba dávať pozor na veľké a malé písmená. V prípade priradenia premennej s názvom **\$color** hodnotu typu string “red“ a následnom pokuse o vytlačenie tejto farby do tela stránky (viď Obrázok číslo 14)

```
<?php
$color = "red";
echo "My car is " . $color . "<br>";
echo "My house is " . $COLOR . "<br>";
echo "My boat is " . $coLoR . "<br>";
?>
```

Obrázok č. 14 – Vplyv veľkých a malých  
písmen na názvy premenných

Tak výsledkom by bolo (viď Obrázok číslo 15)

```
My car is red
My house is
My boat is
```

Obrázok č. 15 – Vplyv veľkých a malých písmen na názvy  
premenných, výsledok príkladu

Čiže program by vypísal hodnotu premennej iba tam kde je zavolaná presne jej názvom.

## 8.5 OPERÁTORY

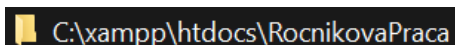
Operátory sú používané na vykonávanie rôznych operácií s premennými a ich hodnotami.

V skriptovacom jazyku PHP poznáme niekoľko typov operátorov. Tieto operátory môžeme poznať aj z iných programovacích jazykov ako napríklad C, Java-Script alebo aj Python.

S pravidla delíme operátory na: Aritmetické, Priradňujúce, Porovnávacie, Increment / Decrement, Logické, String, Array, Podmieneného priradenia

## 9 PRAKTICKÁ ČASŤ

V praktickej časti sa čitateľ dočíta o praktickom vytváraní web stránky v jazyku PHP a prepojení databázy ( pri tvorbe webstránky budeme používať softvér XAMPP ako lokálny web server ktorý nám zároveň aj poskytne databázu MySQL ). Prvým krokom je vytvoriť si zložku do ktorej budeme postupne programovať súbory pre webstránku. Je však veľmi dôležité aby táto zložka bola vytvorená presne kde má v prípade používania XAMPP (viď obrázok číslo 16)

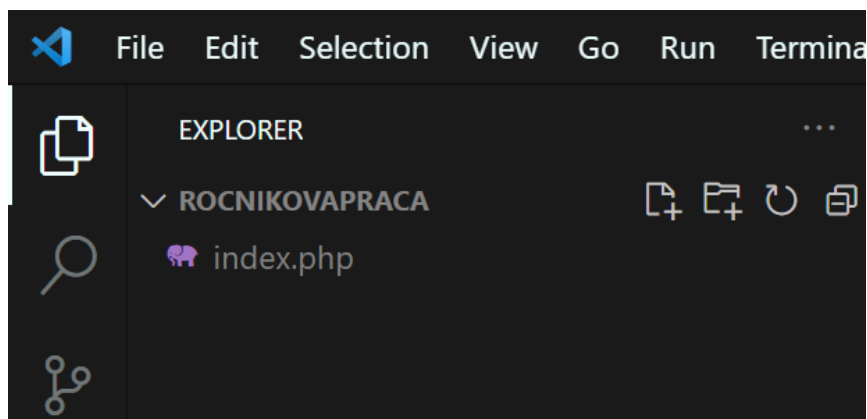
A screenshot of a file explorer window with a dark theme. The address bar at the top shows the path `C:\xampp\htdocs\RocnikovaPraca`. The main area is empty, showing the contents of the selected directory.

Obrázok č. 16 – Cesta uloženej zložky – našej web stránky

### 9.1 VYTVORENIE PRVÉHO SÚBORU

Pri tvorbe web stránky si treba hneď na začiatku dobre rozmyslieť ako chceme tvoriť web stránku. Je niekoľko spôsobov ako na to. Jedným zo spôsobov je vytvoriť si prvý súbor `index.php` v ktorom bude iba navigačné menu a kód ktorý bude používateľa presúvať zo stránky na stránku – teda zo súboru do súboru. Tento spôsob sa robí pomocou podstránok a v jednotlivých podstránkach je už iba kontent tej podstránky, navigačné menu je samozrejme vidieť len kód to navigačné menu už druhý krát obsahovať nemusí. Druhý spôsob je oveľa jednoduchší a ten funguje tak, že každá stránka má v kóde aj menu ( budeme robiť ten prvý spôsob vzhľadom na to, že je síce ťažší ale je trochu viac prehľadný a zakomponováva sa v ňom viac práce v jazyku PHP ).

(viď obrázok číslo 17)



Obrázok č. 17 – Vytvorenie prvého súboru – hlavnej stránky

## 9.2 PREPOJENIE DATABÁZY A STRÁNKY

Pre budúce ukladanie dát a následné načítavanie dát v našom prípade prihlasovacích údajov je potrebné miesto kde sa budú tieto údaje ukladať – teda databázu. Čiže najprv potrebujeme vytvoriť databázu.

### 9.2.1 VYTVORENIE DATABÁZY

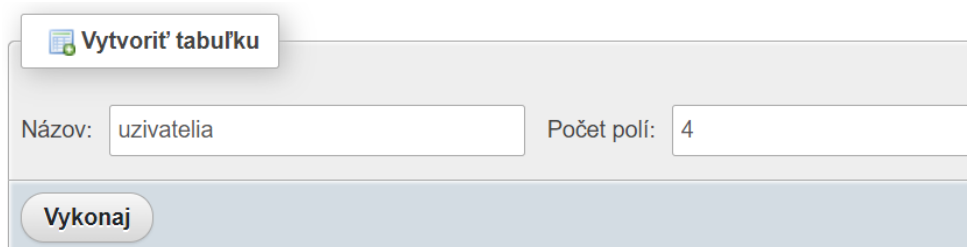
Pri vytváraní databázy treba dbať na meno a na zotriedenie databázy. V našom prípade pomenujeme databázu **RocnikovaPraca** a zotriedenie zvolíme **utf8\_general\_ci**. Pri zadávaní mien či už pre samotnú databázu alebo tabuľky nie je odporúčané dávať -- komplikované názvy alebo používať diakritické znamienka (viď obrázok číslo 18).



Obrázok č. 18 – Vytvorenie databázy

### 9.2.2 VYTVORENIE TABULIEK

Pre fungovanie databázy potrebujeme v databáze vytvoriť príslušný počet tabuliek – v našom prípade vytvoríme dve a to jednu pre prihlasovacie údaje a druhú pre články vytvárané používateľmi. Pri tvorení tabuliek zadávame názov a počet polí – udávame podľa potreby, teda koľko údajov potrebujeme uložiť (viď obrázok číslo 19).



Obrázok č. 19 – Vytvorenie tabuľky

Následne do tabuľky vpíšeme údaje ktoré budeme požadovať pri registrácii na web stránke (viď obrázok číslo 20)

| Názov         | Typ     | Dĺžka/Nastaviť* | Predvolené | Zotriedenie | Atribúty | Nulový                   | Index   | A_                                  |
|---------------|---------|-----------------|------------|-------------|----------|--------------------------|---------|-------------------------------------|
| id_user       | INT     | 11              | Žiadny     |             |          | <input type="checkbox"/> | PRIMARY | <input checked="" type="checkbox"/> |
| name_user     | VARCHAR | 255             | Žiadny     |             |          | <input type="checkbox"/> | ---     | <input type="checkbox"/>            |
| email_user    | VARCHAR | 255             | Žiadny     |             |          | <input type="checkbox"/> | ---     | <input type="checkbox"/>            |
| user_password | VARCHAR | 255             | Žiadny     |             |          | <input type="checkbox"/> | ---     | <input type="checkbox"/>            |

Obrázok č. 20 – Vyplnenie údajov tabuľky pre používateľov

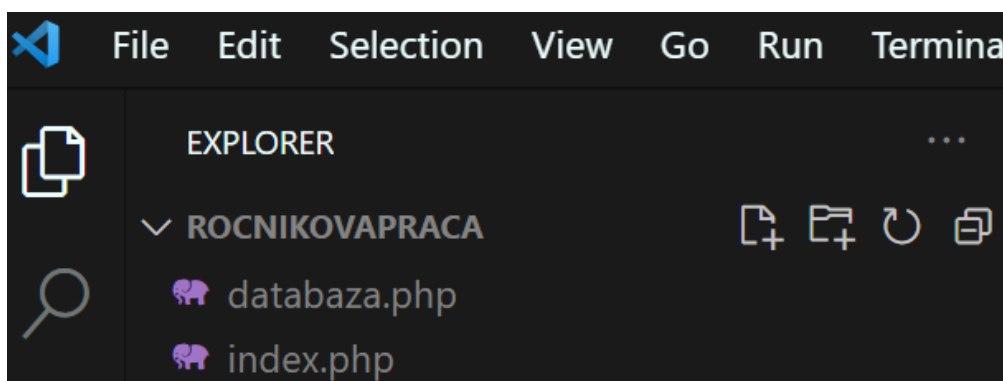
Druhú tabuľku vytvoríme ako už bolo spomenuté pre články a vyplníme údaje potrebné pre ukladanie jednotlivých článkov. (viď obrázok číslo 21)

| # | Názov       | Typ          | Zotriedenie     | Atribúty | Nulový | Predvolené | Komentáre | Extra          | Akcia                 |
|---|-------------|--------------|-----------------|----------|--------|------------|-----------|----------------|-----------------------|
| 1 | id_article  | int(11)      |                 |          | Nie    | Žiadny     |           | AUTO_INCREMENT | Zmeniť Odstrániť Viac |
| 2 | title       | varchar(255) | utf8_general_ci |          | Nie    | Žiadny     |           |                | Zmeniť Odstrániť Viac |
| 3 | content     | text         | utf8_general_ci |          | Nie    | Žiadny     |           |                | Zmeniť Odstrániť Viac |
| 4 | url_article | varchar(255) | utf8_general_ci |          | Nie    | Žiadny     |           |                | Zmeniť Odstrániť Viac |
| 5 | subscribe   | varchar(255) | utf8_general_ci |          | Nie    | Žiadny     |           |                | Zmeniť Odstrániť Viac |
| 6 | id_user     | int(11)      |                 |          | Nie    | Žiadny     |           |                | Zmeniť Odstrániť Viac |

Obrázok č. 21 – Vyplnenie údajov tabuľky pre články

### 9.2.3 ZAKOMPONOVANIE DATABÁZY

Ako konečný krok pre fungovanie databázy je potrebné vytvoriť na našej stránke ďalší súbor (viď obrázok číslo 22) – v našom prípade ho nazveme **databaza.php** a naprogramujeme spojenie databázy so stránkou (viď obrázok číslo 23).



Obrázok č. 22 – Vytvorenie súboru databaza.php

```

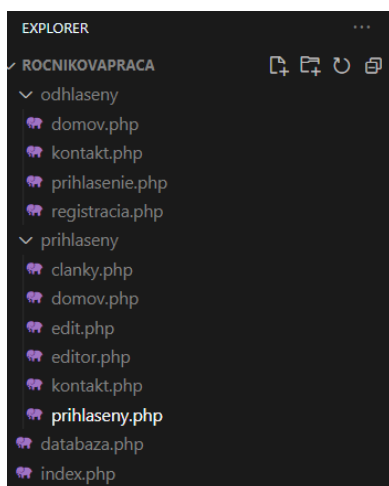
1 <?php
2 $servername = "localhost";           //priradenie do premennej servername hodnotu localhost
3 $username = "root";                 //priradenie do premennej username hodnotu root
4 $password = "";                     //tým že databáza nie je zaheslovaná, tak nmetreba dávať heslo
5 $dbname = "rocnikovapraca";         //priradenie do premennej dbname meno databazy
6
7 $conn = new mysqli($servername, $username, $password, $dbname); //vytvorenie databazy s hodnotami premenných
8 if($conn -> connect_error){          //ak sa nepodari pripojiť alebo nastane chyba,
9     die("Spojenie sa nepodarilo".connect_error); //tak napíše, že sa spojenie nepodarilo
10 }
11 //echo "spojenie sa podarilo";       //napíše nám túto správu ak sa podarí spojenie
12 ?>

```

Obrázok č. 23 – programovanie súboru databaza.php

## 9.3 PRÍPRAVA PODSTRÁNOK A ICH FUNKCIA

Potrebuje si vytvoriť v našej zložke ročníková práca ďalšiu zložku menom odhlásený a do tej zložky povytvárať súbory podľa toho, čo budeme potrebovať – v našom prípade si vytvoríme pre odhlásených používateľov podstránky **domov**, **kontakt**, **registrácia**, **prihlásenie** a pre prihlásených používateľov si vytvoríme podstránky **domov**, **kontakt**, **články**, **prihlásený**, **editor** a **edit** (viď obrázok číslo 24). Najprv sa budeme venovať časti pre odhlásených používateľov. Po dokončení prejdeme na prihlásených používateľov a budeme programovať systém pre nahrávanie, úpravu a mazanie článkov.



Obrázok č. 24 – Tvorba podstránok

Následne potrebujeme urobiť v súbore index.php navigačné menu (viď obrázok číslo 25) a potrebujeme mu pomocou php naprogramovať jednotlivé prekliknutia na naše podstránky. Budeme používať metódu GET – teda super globálnu premennú, ktorá sa používa na zhromažďovanie údajov formulára po odoslaní formulára. Bude to fungovať tak, že keď používateľ klikne na link, tak sa parameter (podstranka) pošle do index.php,

tam sa spracuje a potom presmeruje používateľa na podstránku, ktorú si sám používateľ vybral (viď obrázok číslo 26). Samozrejme treba urobiť hlavičku pomocou html tagov a až potom sa môžeme pustiť na navigačné menu.

```

11 <div class="navigation">
12   <div class="toggle">
13     <ion-icon name="menu-outline" class="open"></ion-icon>
14     <ion-icon name="close-outline" class="close"></ion-icon>
15   </div>
16   <ul>
17     <li class="list">
18       <a href="index.php?stranka=domov">
19         <span class="icon">
20           <ion-icon name="home-outline"></ion-icon>
21         </span>
22         <span class="title">Domov</span>
23       </a>
24     </li>
25     <li class="list">
26       <a href="index.php?stranka=kontakt">
27         <span class="icon">
28           <ion-icon name="call-outline"></ion-icon>
29         </span>
30         <span class="title">Kontakt</span>
31       </a>
32     </li>
33     <li class="list">
34       <a href="index.php?stranka=prihlasenie">
35         <span class="icon">
36           <ion-icon name="log-in-outline"></ion-icon>
37         </span>
38         <span class="title">Prihlasenie</span>
39       </a>
40     </li>
41     <li class="list">
42       <a href="index.php?stranka=registracia">
43         <span class="icon">
44           <ion-icon name="log-in-outline"></ion-icon>
45         </span>
46         <span class="title">Registracia</span>
47       </a>
48     </li>
49   </ul>
50 </div>

```

Obrázok č. 25 – Navigačné menuň

```

54 <?php
55 //presmerovanie v prípade pokusu o prístup na index.php
56 if(!$ _GET['stranka']){
57     header('Location: index.php?stranka=domov');
58 }
59 //priradenie podstránky do premennej adresa po kliknutí na podstránku
60 if(isset($_GET['stranka'])){
61     $adresa = $_GET['stranka'];
62 }
63 else{
64     $adresa = 'domov.php';
65 }
66 //presmerovanie na podstránku
67 if(preg_match('/^[a-z0-9]+$/', $adresa)){
68     $premen = include('odhlaseny/'.$ _GET['stranka'].'.php');
69     if(!$premen){
70         echo "podstranka nenajdena";
71     }
72 }
73 >>

```

Obrázok č. 26 – Kód pre podstránky

Táto časť kódu funguje tak, že ak používateľ klikne v navigačnom menu na podstránku tak pomocou metódy GET sa spracuje podstránka a presmeruje používateľa na danú podstránku. Samozrejme máme aj overenie ak by niekto chcel ísť na index.php tak ho presmerujeme na domovskú stránku.

## 9.4 PODSTRÁNKY DOMOV A KONTAKT

Do podstránok domov a kontakt vložíme iba text pomocou príkazu echo. Rozdelíme si ho pomocou html na dve časti aby sme text mohli ľahšie formátovať v budúcnosti pomocou css (viď obrázok číslo 27).

```
odhlaseny > domov.php
1  <div class = "main">
2    <div class = "title">
3      <?php echo "DOMOV"; ?>
4    </div>
5    <div class = "description">
6      <?php
7        echo "Lorem ipsum dolor sit amet, consectetur adipiscing elit,
8          sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.
9          Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris
10         nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in
11         reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla
12         pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa
13         qui officia deserunt mollit anim id est laborum.Lorem ipsum dolor sit amet,
14         consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et
15         dolore magna aliqua.Ut enim ad minim veniam, quis nostrud exercitation ullamco
16         laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in
17         reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla
18         pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa
19         qui officia deserunt mollit anim id est laborum.";
20      ?>
21    </div>
22  </div>
```

Obrázok č. 27 – Domov a Kontakt

Ako je možné vidieť v kóde tak nie je potreba už opäť otvárať html tagy a znovu písať kód pre navigačné menu.

## 9.5 REGISTRÁCIA

Pri registrácii budeme používať metódu POST a formulár. Pomocou php kódu overíme či boli vyplnené všetky polia a ak tak nebolo urobené tak vypíšeme, že tak treba urobiť. Pre ukladanie údajov ktoré budeme neskôr potrebovať aj v druhých súboroch použijeme session. V prvom rade musíme overiť jednotlivé údaje zadané do formulára používateľom (viď obrázok číslo 28).



```

<?php
//inicializacia session
session_start();
//vyžiadanie databázy zo súboru databaza.php
require_once('databaza.php');
//Toto sa stane po odoslaní formuláru
if($_POST){
    //kontrola políček či boli vyplnené správne
    if($_POST['rok'] != date('Y')) //antispam
        echo '<p>Chybne vyplneny antispam.</p>';
    else if($_POST['heslo'] == '') //kontrola či bolo zadane heslo
        echo '<p>Nevyplnili ste heslo.</p>';
    else if($_POST['heslo'] != $_POST['oheslo']) //kontrola či sa zhoduje heslo
        echo '<p>Hesla sa nezhoduju.</p>';
    else if($_POST['email'] == '') //kontrola či bol email zadany
        echo '<p>Nevyplnili ste email.</p>';
    else if($_POST['meno'] != ''){ //kontrola či už nie je rovnaké meno alebo email použitý
        $meno = $_POST['meno'];
        $email = $_POST['email'];
        $sql = "SELECT name_user FROM uzivatelia WHERE name_user = '$meno' OR email_user = '$email' LIMIT 1;";
        $result = $conn->query($sql);
        //ak sa vyhodnotí, že už niekto má také meno alebo email tak vypíše túto správu
        if($result->num_rows > 0)
            echo '<p>Užívateľ s týmto menom alebo emailom už existuje</p>';
    }
}

```

Obrázok č. 28 – Overenie údajov zadanych do formulára

V nasledujúcej časti zapíšeme údaje ktoré sme získali do databázy – teda vytvoríme účet.

```

else{
    //Priradenie hodnot z formulara do premenných
    $heslo = password_hash($_POST['heslo'], PASSWORD_DEFAULT); //hashovanie hesla
    $meno = $_POST['meno'];
    $email = $_POST['email'];
    //registracia pouzivatela - vlozenie údajov z formularu do databázy
    $sql = "INSERT INTO uzivatelia (name_user, user_password, email_user) VALUES ('$meno', '$heslo', '$email')";
    $conn->query($sql);
    $posledny = $con->insert_id;
    //ulozenie údajov do session
    $_SESSION['uzivatel_id'] = $con->insert_id;
    $_SESSION['uzivatel_meno'] = $_POST['meno'];
    $_SESSION['uzivatel_email'] = $_POST['email'];
    $_SESSION['uzivatel_admin'] = 0;
    //presmerovanie po uspešnej registrácii
    header('Location: prihlaseny/prihlaseny.php?stranka=domov');
    exit();
}

```

Obrázok č. 29 – Zápis údajov do databázy

Následne už iba ostáva vytvoriť formulár pre registrovanie (viď obrázok číslo 30). Vo formulári priradíme do premenných pomocou metódy POST hodnoty ktoré sa potom spracúvajú php kódom. Je dobré myslieť pri registračných formulároch na možnosť spamovania nášho registračného systému a preto je dobrým zvykom vždy pridať nejaký spôsob zamedzenia. V našom prípade budeme používať jednoduchú metódu spočívajúcu vo vpísaní aktuálneho roku do políčka na to vyznačeného. Aktuálny dátum dostaneme php funkciou time().

```

<form method="post">
  <div class="icon-holder">
    <div class="value-holder">
      <ion-icon name="person"></ion-icon>
    </div>
    <?php $vyp_meno = (isset($_POST['meno'])) ? $_POST['meno'] : ''; ?>
    <input type="text" placeholder="Meno" name="meno" value="<?php echo htmlspecialchars($vyp_meno); ?>"> <br>
  </div>
  <div class="icon-holder">
    <div class="value-holder">
      <ion-icon name="key"></ion-icon>
    </div>
    <?php $vyp_heslo = (isset($_POST['heslo'])) ? $_POST['heslo'] : ''; ?>
    <input type="password" placeholder="Heslo" name="heslo" value="<?php echo htmlspecialchars($vyp_heslo); ?>"> <br>
  </div>
  <div class="icon-holder">
    <div class="value-holder">
      <ion-icon name="key"></ion-icon>
    </div>
    <?php $vyp_oheslo = (isset($_POST['oheslo'])) ? $_POST['oheslo'] : ''; ?>
    <input type="password" placeholder="Opakuj Heslo" name="oheslo" value="<?php echo htmlspecialchars($vyp_oheslo); ?>"> <br>
  </div>
  <div class="icon-holder">
    <div class="value-holder">
      <ion-icon name="mail"></ion-icon>
    </div>
    <?php $vyp_email = (isset($_POST['email'])) ? $_POST['email'] : ''; ?>
    <input type="email" placeholder="Email" name="email" value="<?php echo htmlspecialchars($vyp_email); ?>"> <br>
  </div>
  <div class="icon-holder">
    <div class="value-holder">
      <ion-icon name="calendar"></ion-icon>
    </div>
    <?php $vyp_rok = (isset($_POST['rok'])) ? $_POST['rok'] : ''; ?>
    <input type="password" placeholder="Aktuálny Rok" name="rok" value="<?php echo htmlspecialchars($vyp_rok); ?>"> <br>
  </div>
  <input type="submit" name="submit" value="Registrovat sa"> <br>
</form>
</div>

```

Obrázok č. 30 – Formulár pre registráciu

## 9.6 PRIHLÁSENIE

Pri prihlásení budeme používať tak isto ako pri registrácii metódu POST a formulár. Naše prihlasovanie bude fungovať rovnako ako registrácia s malými zmenami v php kóde a pri práci s databázou nebudeme do databázy vpisovať údaje ale budeme ich z databázy požadovať (viď obrázok číslo 31). Taktiež budeme požadovať menej údajov a to konkrétne iba meno alebo email a heslo. Opäť overíme podmienkami či boli zadané všetky údaje a ďalšou podmienkou overíme či bolo zadané heslo správne. Toto overenie bude fungovať tak, že pozrieme sa na zadané meno, prejdeme databázou a ak nájdeme meno tak sa pozrieme pri tomto mene aké bolo nastavené heslo pri registrácii. Ak sa bude zhodovať tak používateľa prihlásime a presmerujeme teda na domovskú stránku. V opačnom prípade vypíšeme, že zadané heslo nie je správne (viď obrázok číslo 32).

```

$sql="SELECT * FROM uzivatelia WHERE name_user = '$meno' OR email_user = '$meno' LIMIT 1";
$result = $conn->query($sql);

```

Obrázok č. 31 – Prihlásenie – získavanie údajov z databázy

```
//prejdenie údajov v databáze na základe zadaného mena
while($row = $result->fetch_row()){
    $id_user = $row[0];
    $name = $row[1];
    $email = $row[2];
    $heslo = $row[3];
    $admin = $row[4];
}
//overenie správnosti hesla
if(!password_verify($_POST['heslo'], $heslo)){
    echo '<p> Nespravne meno alebo heslo</p>';
//priradenie hodnôt do session pre neskorsie použitie
}else{
    $_SESSION['uzivatel_id'] = $id_user;
    $_SESSION['uzivatel_meno'] = $name;
    $_SESSION['uzivatel_admin'] = $admin;
    $_SESSION['uzivatel_email'] = $email;
    //presmerovanie na domovskú stránku po prihlásení
    header('Location: admin_view.php?stranka=administracia');
}
```

Obrázok č. 32 – Prihlásenie – overenie hesla a priradenie hodnôt do session

Ako posledný krok pri prihlásení ostáva dorobiť formulár pre prihlásenie s príslušnými políčkami – teda použijeme dve políčka a to meno a heslo. (viď obrázok číslo 33)

```
<form method="post">
    <div class="icon-holder">
        <div class="value-holder">
            <ion-icon name="person"></ion-icon>
        </div>
        <input type="text" placeholder="Meno" name="meno"> <br>
    </div>
    <div class="icon-holder">
        <div class="value-holder">
            <ion-icon name="key"></ion-icon>
        </div>
        <input type="password" placeholder="Heslo" name="heslo"> <br>
    </div>
    <input type="submit" name="submit" value="Prihlásiť sa"> <br>
</form>
```

Obrázok č. 33 Prihlásenie – Formulár

## 9.7 NAVIGAČNÉ MENU PO PRIHLÁSENÍ

Po prihlásení potrebujeme vedieť písať, upravovať a mazať články, to znamená že potrebujeme urobiť súbor tak ako bol index.php, kde budú menšie zmeny. Po prihlásení chceme aby bolo možné pridávať, upravovať, mazať a zobrazovať články. Preto vytvoríme súbor prihlaseny.php – tento súbor sa nám bude starať o podstránky a bude v sebe držať navigačné menu (vid' obrázok číslo 34). Samozrejme nesmieme zabudnúť na odhlásenie, ktoré pridáme do navigačného menu ako ďalšiu možnosť (vid' obrázok číslo 35). V ďalšej podkapitole si povieme o tom ako funguje toto odhlásenie a prečo vyzerá jeho presmerovanie inak ako pri ostatných podstránkach.

```
<?php
//presmerovanie v prípade pokusu o prístup na index.php
if(!$_GET['stranka']){
    header('Location: prihlaseny.php?stranka=domov');
}
//priradenie podstránky do premennej adresa po kliknutí na podstránku
if(isset($_GET['stranka'])){
    $stranka = $_GET['stranka'];

    if(preg_match('/^[a-z0-9]+$/', $stranka)){
        $vlozene = include($stranka.'.php');
        if(!$vlozene)
            header("Location: index.php");
    }
    else
        echo "Neplatný parameter.";
}
?>
```

Obrázok č. 34 – Presmerovanie na podstránky

```
<li class="list">
    <a href="administracia.php?odhlasit">
        <span class="icon">
            <ion-icon name="log-out-outline"></ion-icon>
        </span>
        <span class="title">Odhlásenie</span>
    </a>
</li>
```

Obrázok č. 35 – Odhlásenie v navigačnom menu

## 9.8 ODHLÁSENIE

Pre odhlásenie nám stačí vytvoriť si nový súbor administracia.php kde budeme pomocou session vypisovať meno a email aktuálne prihláseného používateľa (viď obrázok číslo 36)

```
<div class="main">
    <?php echo $_SESSION['uzivatel_meno']; ?> <br>
    <?php echo $_SESSION['uzivatel_email']; ?>
</div>
```

Obrázok č. 36 – Výpis mena a email-u zo session

V druhej časti súboru administracia.php budeme pomocou php kódu a metódy GET registrovať, či používateľ neklikol v navigačnom menu na možnosť odhlásiť sa. V prípade, že tak učinil tak ho presmerujeme do súboru administracia.php ako je možné vidieť na obrázku číslo 36. V tomto súbore ukončíme session a presmerujeme používateľa do odhlásenej časti na prihlasovaciu podstránku (viď obrázok číslo 37) ,

```
<?php
session_start();
if(isset($_GET['odhlasit'])){
    session_destroy();      // <- ukončenie session
    //presmerovanie
    header('Location: ../index.php?stranka=prihlasenie');
}
```

Obrázok č. 37 – Odhlásenie a zrušenie session

Nesmieme zabudnúť overiť to aby sa používateľ nemohol vrátiť šípkou späť po odhlásení (viď obrázok 38).

```
//ochrana pred vratenim sipkou spat
if(isset($_SESSION['uzivatel_meno']) == ''){
    session_destroy();      // <- ukončenie session
    //presmerovanie
    header('Location: ../index.php?stranka=prihlasenie');
}
```

Obrázok č. 38 – Ochrana pred návratom bez prihlásenia



Súbor `url_create` vytvárame preto, aby sa nám upravovala url adresa nového príspevku a neboli v nej špeciálne znaky (viď obrázok 41).

```
<?php
function createUrlSlug($urlString){
    $slug = preg_replace('/[^A-Za-z0-9-]+/', '-', $urlString);
    return $slug;
}
?>
```

Obrázok č. 41 – Úprava url adresy

Pre samotné pridanie nového príspevku však potrebujeme ešte časť kódu ktorá bude zapisovať do databázy každý nový príspevok. Toto docielime tak, že budeme metódou POST získavať údaje z formuláru, tie potom priradíme do premenných a na základe premenných použijeme príkaz `INSERT INTO` pomocou ktorého zapíšeme údaje do databázy (viď obrázok číslo 42).

```
//privítanie užívateľa na základe jeho mena
echo "<h1>".$SESSION['uzivatel_meno'].", Vitaj v editore článkov</h1>";
require('../databaza.php'); //vyziadanie databázy
if($_POST){ //v prípade stlačenia tlačítka publikovať článok sa udeje všetko v tele tejto podmienky
    $nazov = $_POST['title']; //priradenie názvu do premennej
    $obsah = $_POST['obsah']; //priradenie textu - obsahu do premennej
    $url = createUrlSlug($nazov); //vytvorenie url adresy
    $url = $url.'-'.time(); //pridanie času do url adresy aby sa nemohla nikdy opakovať
    $popis = $_POST['popis']; //priradenie popisu do premennej
    $uziv_id = $SESSION['uzivatel_id']; //priradenie používateľovho id do premennej
    //pridanie článku do databázy
    $sqli = "INSERT INTO clanky (title, content, url_article, subscribe, id_user) VALUES ('$nazov', '$obsah', '$url', '$popis', '$uziv_id')";
    $conn->query($sqli);
    //evidovanie posledného článku
    $posledny = $conn->insert_id;
    $_SESSION['posl_clanok'] = $posledny;
}
```

Obrázok č. 42 – Zápis článkov do databázy

## 9.12 ÚPRAVA ČLÁNKOV

Pre upravovanie článkov budeme používať podobný postup ako pri ich tvorbe len s malou zmenou a to, že budeme z databázy musieť najprv načítať článok (viď obrázok číslo 43), vpísať ho do formuláru (viď obrázok číslo 44) aby používateľ videl to, čo už bolo v článku napísané a mohol pokračovať ďalej. Príkaz `INSERT INTO` nahradíme za `UPDATE` – teda aktualizujeme údaje v databáze aby ostalo id článku a jeho pozícia v grid-e stále rovnaká (viď obrázok číslo 45).

```

<form action="" method="post">
<div class="value-holder">
  <div class="icon-holders">
    <ion-icon name="reader-outline"></ion-icon>
  </div>
  <input type="text" name="titles" value="<?php echo $title ?>"> <br>
</div>
<div class="value-holder">
  <div class="icon-holders">
    <ion-icon name="chatbox-ellipses-outline"></ion-icon>
  </div>
  <input type="text" name="popis" value="<?php echo $popisok ?>"> <br>
</div>
<textarea name="obsah" id="texty" cols="30" rows="10"><?php echo $text ?></textarea><br />
<input type="submit" name="odoslat" value='Upraviť článok'>
</form>

```

Obrázok č. 43 – Formulár pre úpravu článkov

A ako už bolo spomínané vyššie treba článok z databázy načítať. Treba si dávať pozor na hodnoty z databázy a na to ako ich vyžadujeme postupne za sebou. Záleží na poradí v databáze ako sme si ich ukladali (príklad: id, názov, popis, text) – v prípade nedodržania tohto poradia sa hodnoty vpíšu do formulára v inom poradí a po aktualizácii článku by ostal zmätok.

```

//načítanie článku z databázy
$sql = "SELECT * FROM clanky WHERE id_article = $id_article";
$editing = $conn ->query($sql);
if(isset($editing)){
  //priradenie hodnôt do premenných ktoré budeme vypisovať do formuláru
  while($rows = $editing->fetch_row()){
    $id_article = $rows['0'];

    $id_user = $rows[0];
    $title = $rows[1];
    $text = $rows[2];
    $popisok = $rows[4];
  }
}

```

Obrázok č. 44 – Načítanie článku z databázy pre úpravu

Pri samotnej aktualizácii článku je potrebné mať na pamäti presmerovanie používateľa späť na stránku s článkami

```

//aktualizovanie údajov v databáze
$sqlu = "UPDATE clanky SET title = '$nazov', content = '$obsah', subscribe = '$popis' where id_article='$id_article'";
$conn->query($sqlu);
header('Location: prihlaseny.php?stranka=clanky'); //presmerovanie späť na články

```

Obrázok č. 45 – Aktualizácia zmien článku v databáze



## 9.13 VYMAZANIE ČLÁNKU

Pri mazaní článkov budeme používať príkaz DELETE FROM kde ďalej určíme tabuľku databázy, v ktorej sa požadovaný článok na vymazanie nachádza a doplníme iba id článku ktorý bol zvolený pre vymazanie (viď obrázok číslo 46). Druhou časťou programu je, naprogramovať spôsob ktorým budeme zistiť ktorý článok treba vymazať. Pomôže nám metóda GET (viď obrázok číslo 47).

```
if(isset($_GET['delete'])){
    $id_article = $_GET['delete'];
    $conn -> query("DELETE FROM clanky WHERE id_article = $id_article") or die($conn);
    header('location: prihlaseny.php?stranka=clanky'); //resmerovanie späť na články
}
```

Obrázok č. 46 –Vymazanie článku z databázy

Nesmieme zabudnúť, že po vymazaní článku sa nám vďaka GET metóde zmení query string a bude teda po vymazaní treba presmerovať používateľa späť na stránku so všetkými článkami.

```
<a href=clanky.php?delete='.$row['id_article'].'>Vymazať</a>
```

Obrázok č. 47 –Spôsob vymazania článkov

## **ZÁVER**

V tejto práci sa podarilo splniť všetky ciele stanovené na začiatku práce v úvode. Menovite sme splnili ciele ako je tvorba databázy, tvorba príslušných tabuliek, prepojenie databázy s web stránkou, naprogramovanie systému pre registráciu, prihlasovanie, odhlasovanie a v neposlednom rade sa nám podarilo naprogramovať pridávanie článkov, upravovanie článkov a ich samotné mazanie. V teoretickej časti sme sa oboznámili so základnými termínmi, syntaxou, fungovaním jazyka php a ako a kde tento skriptovací jazyk možno použiť. V Praktickej časti sme si všetko prešli pomocou programovania a vysvetľovania za pomoci komentárov v kódach. Naplnili sme teda týmito krokmi celkový cieľ a to naprogramovanie webstránky vo forme blogu v jazyku php. Práca na tejto téme ma obohatila vedomostne v rámci práci s databázou, jej prepájaním, zapisovaním údajov, mazaním údajov z databázy a celkovo som sa naučil niečo nové o programovacích jazykoch a programovaní.

## **POUŽITÁ LITERATÚTA**

1. Peter Šoltýs : Kniha programovanie OOP PHP aplikácii, Pôvodné vydanie, 2019.
2. David Procházka : PHP 6 – Začínáme programovat, Pôvodné vydanie, 2012.



12. [https://sk.wikipedia.org/wiki/PHP\\_\(skriptovac%C3%AD\\_jazyk\)](https://sk.wikipedia.org/wiki/PHP_(skriptovac%C3%AD_jazyk))
13. [https://www.w3schools.com/php/php\\_intro.asp](https://www.w3schools.com/php/php_intro.asp)
14. <https://httpd.apache.org/>
15. <https://www.hostafrica.co.za/blog/wp-content/uploads/2018/03/Linux-LAMP-Stack.png>
16. [https://sk.wikipedia.org/wiki/Rasmus\\_Lerdorf](https://sk.wikipedia.org/wiki/Rasmus_Lerdorf)
17. [https://www.bing.com/images/search?view=detailV2&ccid=m8Bv%2fCUY&id=8E833F3CFA72312929DFD397EAFBCA20250AA4EF&thid=OIP.m8Bv\\_CUYp9lAFn8CxMlhVQHaEK&mediaurl=https%3a%2f%2fclevertie.com%2fimg%2fmain%2fphp-data-types.png&cdnurl=https%3a%2f%2fth.bing.com%2fth%2fid%2fR9bc06ffc2518a7d940167f02c4c96155%3frik%3d76QKJSDK%252b%252bqX0w%26pid%3dImgRaw&exph=608&expw=1080&q=php+data+types&simid=607987285540368754&ck=72135C3780EC1812883140D77744B213&selectedIndex=0&FORM=IRPRST&ajaxhist=0&ajaxserp=0](https://www.bing.com/images/search?view=detailV2&ccid=m8Bv%2fCUY&id=8E833F3CFA72312929DFD397EAFBCA20250AA4EF&thid=OIP.m8Bv_CUYp9lAFn8CxMlhVQHaEK&mediaurl=https%3a%2f%2fclevertie.com%2fimg%2fmain%2fphp-data-types.png&cdnurl=https%3a%2f%2fth.bing.com%2fth%2fid%2fR9bc06ffc2518a7d940167f02c4c96155%3frik%3d76QKJSDK%252b%252bqX0w%26pid%3dImgRaw&exph=608&expw=1080&q=php+data+types&simid=607987285540368754&ck=72135C3780EC1812883140D77744B213&selectedIndex=0&FORM=IRPRST&ajaxhist=0&ajaxserp=0)
18. [https://www.bing.com/images/search?view=detailV2&ccid=eJfFD%2ff4&id=41ABAAA6C5EF35678348D5EA3362BC6778D96CA2&thid=OIP.eJfFD\\_f4klksQWwAFhqCAHaFA&mediaurl=https%3a%2f%2fwww.droid-life.com%2fwp-content%2fuploads%2f2016%2f08%2ffacebook-logo-980x662.jpg&cdnurl=https%3a%2f%2fth.bing.com%2fth%2fid%2fR7897c50ff7f896496ccd05b000586a08%3frik%3domzZeGe8YjPq1Q%26pid%3dImgRaw&exph=662&expw=980&q=facebook+logo&simid=608050485972578298&ck=23AB934FDC29AEA1B56E7E40531BE477&selectedIndex=0&FORM=IRPRST&idpp=overlayview&ajaxhist=0&ajaxserp=0](https://www.bing.com/images/search?view=detailV2&ccid=eJfFD%2ff4&id=41ABAAA6C5EF35678348D5EA3362BC6778D96CA2&thid=OIP.eJfFD_f4klksQWwAFhqCAHaFA&mediaurl=https%3a%2f%2fwww.droid-life.com%2fwp-content%2fuploads%2f2016%2f08%2ffacebook-logo-980x662.jpg&cdnurl=https%3a%2f%2fth.bing.com%2fth%2fid%2fR7897c50ff7f896496ccd05b000586a08%3frik%3domzZeGe8YjPq1Q%26pid%3dImgRaw&exph=662&expw=980&q=facebook+logo&simid=608050485972578298&ck=23AB934FDC29AEA1B56E7E40531BE477&selectedIndex=0&FORM=IRPRST&idpp=overlayview&ajaxhist=0&ajaxserp=0)
19. [https://www.bing.com/images/search?view=detailV2&ccid=WNxzwOyM&id=6DD722E67001DD697222E69021797E7B82D58A32&thid=OIP.WNxzwOyMi9oYWbkXSft0hwAAAA&mediaurl=https%3A%2F%2Fedvservice.bayern%2Fwp-content%2Fuploads%2F2017%2F06%2Fwordpress-logo-300x240.jpg&cdnurl=https%3A%2F%2Fth.bing.com%2Fth%2Fid%2FR58dc73c0ec8c8bda1859b917485b7487%3Frik%3DMorVgnt%252beSGQ5g%26pid%3DImgRaw&exph=240&expw=300&q=wordpress+logo&simid=608002395226651876&ck=5DEFF87F2DA65D30DE44B75B2E43E737&selectedIndex=2&form=IRPRST&ajaxhist=0&ajaxserp=0&pivotparams=insightsToken%3Dccid\\_0BPhKLRK\\*cp\\_BA6FE3956D1D3A7994905FEFC9F434DB\\*mid\\_67E76988A5770F7F61E1B2C93965F4D461062E4C\\*simid\\_608046401460856898\\*thid\\_OIP.0BPhKLRK4Q!\\_I88FyYBt7dAHaEo&vt=0&sim=11&iss=VSI&ajaxhist=0&ajaxserp=0](https://www.bing.com/images/search?view=detailV2&ccid=WNxzwOyM&id=6DD722E67001DD697222E69021797E7B82D58A32&thid=OIP.WNxzwOyMi9oYWbkXSft0hwAAAA&mediaurl=https%3A%2F%2Fedvservice.bayern%2Fwp-content%2Fuploads%2F2017%2F06%2Fwordpress-logo-300x240.jpg&cdnurl=https%3A%2F%2Fth.bing.com%2Fth%2Fid%2FR58dc73c0ec8c8bda1859b917485b7487%3Frik%3DMorVgnt%252beSGQ5g%26pid%3DImgRaw&exph=240&expw=300&q=wordpress+logo&simid=608002395226651876&ck=5DEFF87F2DA65D30DE44B75B2E43E737&selectedIndex=2&form=IRPRST&ajaxhist=0&ajaxserp=0&pivotparams=insightsToken%3Dccid_0BPhKLRK*cp_BA6FE3956D1D3A7994905FEFC9F434DB*mid_67E76988A5770F7F61E1B2C93965F4D461062E4C*simid_608046401460856898*thid_OIP.0BPhKLRK4Q!_I88FyYBt7dAHaEo&vt=0&sim=11&iss=VSI&ajaxhist=0&ajaxserp=0)
20. <https://cs.wikipedia.org/wiki/PHP>
21. [https://www.bing.com/images/search?view=detailV2&ccid=QQ%2BoCwhl&id=3CA80F152F08885F09133E6CD801381015B90F01&thid=OIP.QQ-oCwhl6Tj-e519vKVUwHaEX&mediaurl=https%3A%2F%2Fwww.sklsongkiat.com%2Fhome%2Fimg%2Fportfolio\\_phpmyadmin.jpg&cdnurl=https%3A%2F%2Fth.bing.com%2Fth%2Fid%2FR410fa80b0865a3a4e3f9ee65f6f29553%3Frik%3DAQ%252b5FRA4AdhsPg%26pid%3DImgRaw&exph=949&expw=1612&q=phpmyadmin+logo&simid=607998675784788847&ck=30B998AD16784A03D15A6EF20ED99323&selectedIndex=3&for](https://www.bing.com/images/search?view=detailV2&ccid=QQ%2BoCwhl&id=3CA80F152F08885F09133E6CD801381015B90F01&thid=OIP.QQ-oCwhl6Tj-e519vKVUwHaEX&mediaurl=https%3A%2F%2Fwww.sklsongkiat.com%2Fhome%2Fimg%2Fportfolio_phpmyadmin.jpg&cdnurl=https%3A%2F%2Fth.bing.com%2Fth%2Fid%2FR410fa80b0865a3a4e3f9ee65f6f29553%3Frik%3DAQ%252b5FRA4AdhsPg%26pid%3DImgRaw&exph=949&expw=1612&q=phpmyadmin+logo&simid=607998675784788847&ck=30B998AD16784A03D15A6EF20ED99323&selectedIndex=3&for)

m=IRPRST&ajaxhist=0&ajaxserp=0&pivotparams=insightsToken%3Dccid\_DMIWG2%252FN\*cp\_1DF177  
06B1E5C31E52EA51AD57DDC4C6\*mid\_7329B643A6FBD3C9B75E57BC37B1E4205649EC00\*simid\_6  
08055292042487506\*thid\_OIP.DMIWG2!\_NmNiticR5Z1QOKAHaE8&vt=0&sim=11&iss=VSI&ajaxhist=  
0&ajaxserp=0