Programski jezik PHP

Seminarski rad u okviru kursa
Metodologija stručnog i naučnog rada
Matematički fakultet

Đorđe Vučković, Tamara Ivanović,
Petar Simić, Stefan Stević
djordjevuckovic996@gmail.com, tamara.d.ivanovic@gmail.com,
petar_ 95@yahoo.com, stefanstevovic@yahoo.com

27. april 2019

Sažetak

U ovom radu su ukratko predstavljene osnovne osobine i specifičnosti programskog jezika PHP. Ovde se možete informisati o razvojnom putu samog jezika, njegovim mogućnostima i primenama.Možete saznati koja su najpoznatija okruženja za programski jezik PHP kao i kako obaviti proces instalacije na Linux i Windows operativnim sistemima. Sam izgled PHP koda je prikazan kroz primere koji prate ovaj rad.

Sadržaj

1 Uvod **2**

2 Razvoj programskog jezika PHP **2**

3 Mogućnosti programskog jezika **3**

4 Osnovne osobine, podržane paradigme i koncepti **4**

5 Najpoznatija okruženja **6**

6 Instalacija **6**

7 Primer kôda **7**

8 Specifičnosti **8**

9 Zaključak **10**

Literatura **10**

1 Uvod

PHP (Hypertext preprocessor) je popularan skriptni jezik opšte namene. Nastao je iz potrebe njegovog stvaraoca za praćenjem poseta svog vebsajta i tokom godina sve je popularniji kod veb programera[**?**]. Kako semantika ovog jezika nije striktna, lak je za učenje, što ga čini pristupačnim programerima sa različitim nivoom znanja [**?**].

O popularnosti jezika govore podaci analize GitHub repozitorijuma i projekata. Tokom godina ostaje u prvih 6 jezika po upotrebi. Statistika za period 2012-2014. godine je prikazana u tabeli 1. Ispred PHP-a 2014. godine su bili jezici JavaScript, Java, Python i CSS[**?**].

Ovaj rad je namenjen svima koji žele da saznaju osnovne karakteristike PHP-a, šta je ono što ga izdvaja od ostalih, kao i kako da ga pokrenu na svojim računarima. Veliki broj radnih okruženja (eng. framework)[**?**] je podržan, ali ovde je fokus stavljen na najpoznatija. Ovaj rad se ne fokusira na sintaksu i nema za cilj da čitaoca nauči kako da programira u PHP-u, ali za to preporučujemo [**?**, **?**, **?**].

Tabela 1: Učešće PHP projekata na GitHub-u i pozicija PHP-a na listi zastupljenosti, u periodu 2012. - 2014. god.

period	PHP repozitorijumi	% zastupljenosti	pozicija
2012, II kvartal	62336	11.3	4
2012, IV kvartal	62336	8.1	4
2013, II kvartal	83169	7.6	5
2013, IV kvartal	91879	7.1	6
2014, II kvartal	118272	5.6	6
2014, IV kvartal	138771	6.3	5

2 Razvoj programskog jezika PHP

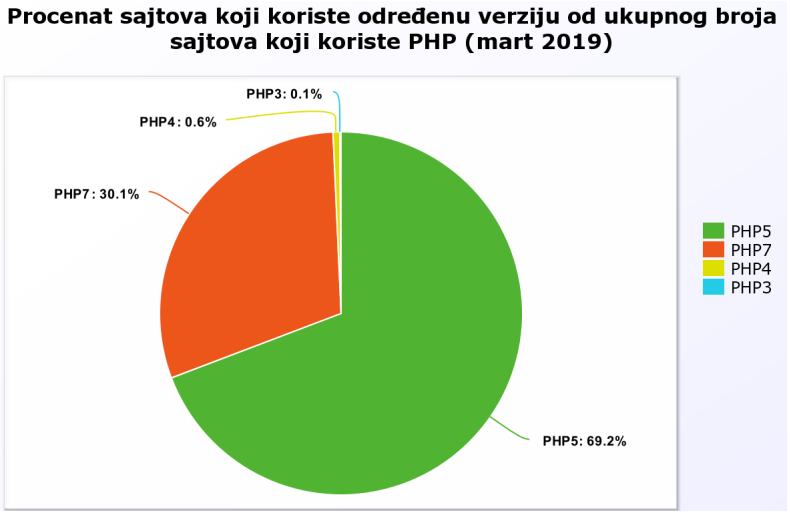
Rasmus Lerdorf je 1994. godine napravio programski jezik PHP[**?**], međutim danas se on značajno razlikuje od početne verzije. Za potrebe napretka svog privatnog vebsajta i praćenja broja poseta, napisao je niz skriptova u Perl-u, a zatim i dodao mogućnost pristupa bazama podataka i kreiranje jednostavnih veb aplikacija razvojem C skriptova što je činilo prvu verziju. Sledeće godine, Rasmus objavljuje PHP 2.0 koji je kombinacija PHP-a i form interpretera koji je on napisao. Kako je ova verzija bila javno dostupna (eng. open-source), mnogi programeri su mogli da unapređuju PHP.**Zeev Suraski** i **Andi Gutmans** su 1997. godine redizajnirali PHP jezgro i objavili **PHP/FI** [**?**].

Lerdorf, Suraski i Gutmans 1997. godine počinju da sarađuju na pisanju novog programskog jezika i PHP/FI 2.0 je preimenovan u PHP 3.0. U PHP 3.0 je uključena podrška objektno-orijentisanom programiranju, sintaksa jezika je postala konzistentna, a krajnjim korisnicima je omogućen interfejs za API-je, protokole i višestruke baze podataka. U svom najvećem uspehu PHP 3.0 je bio instaliran na približno 10% veb servera na Internetu[**?**].

Andri Gutmans i Zeev Suraski su kreirali "**Zend Engine**" [**?**] čija je uloga bila da popravi performanse kompleksnih aplikacija, kao i modulardnost osnove PHP kôda. Zend Engine je zadužen za leksičku analizu,

sintaksno parsiranje i prevođenje koda u bajtkod. Takođe upravlja memorijom i sadrži sakupljač otpadaka. Manipuliše greškama i izuzecima i zadužen je za upravljanje tipovima i promenljivima. Spoj Zend Engine-a i velikog broja dodatnih karakteristika čine PHP 4.0. Njegove glavne karakteristike su podržavanje većeg broja HTTP sesija i veb servera i izlazno baferovanje Takođe, dodati su i novi jezički konstrukti.

Verzija 5.0 izlazi 2004. godine, a godinu dana kasnije verzija 7.0. Ove dve verzije su najzastupljenije, dok prethodne verzije veoma mali broj sajtova koristi. Ovo se može videti na slici 2 koja prikazuje podatke iz 2019. godine prema podacima [**?**].



Slika 1: Zastupljenost verzija PHP-a

Ključnu ulogu u razvoju PHP-a ima PHP/FI, koji se smatra pretečom današnjeg PHP-a. Takođe, bez C i Perl skriptova razvoj ne bi bio moguć. Detaljno razvojno stablo je prikazano na slici 2.

razvojno_stablo.png

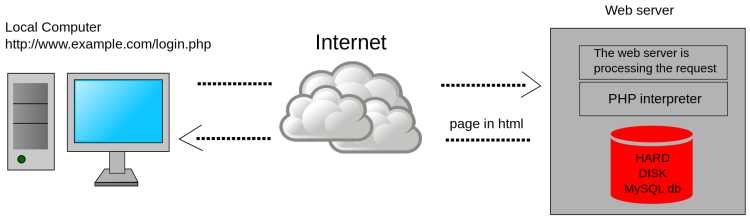
Slika 2: Razvojno stablo PHP-a

3 Mogućnosti programskog jezika

PHP je jezik koji je posebno pogodan za veb razvoj na strani servera, u kom slučaju PHP generalno radi na veb servisu. Svaki PHP kôd u traženoj datoteci izvršava PHP runtime[**?**], obično da bi kreirao dinamičan sadržaj veb stranica ili dinamičke slike koje se koriste na veb stranicama ili drugde. Takođe se može koristiti za skriptovanje komandne linije i aplikacije grafičkog korisničkog interfejsa na strani klijenta (GUI). PHP se može koristiti na većini veb servera, mnogim operativnim sistemima

i platformama, a može se koristiti sa mnogim sistemima za upravljanje relacionim bazama podataka (RDBMS). Većina veb hosting provajdera podržava PHP za upotrebu od strane svojih klijenata. Dostupan je besplatno, a PHP grupa obezbeđuje kompletan izvorni kôd za korisnike koji će graditi, prilagoditi i proširiti za sopstvenu upotrebu.

Izvorno je dizajniran za korišćenje dinamičkih veb stranica, ali PHP se sada fokusira uglavnom na skriptiranje na strani servera i sličan je drugim skriptnim jezicima na strani servera koji pružaju dinamički sadržaj sa veb servera klijentu. PHP je takođe privukao razvoj mnogih softverskih okvira koji pružaju građevinske blokove i strukturu dizajna za promovisanje brzog razvoja aplikacija (RAD). Neki od njih uključuju PRADO, CakePHP, Simfoni, CodeIgniter, Laravel, Iii Framework, Phalcon i Zend Framework, nudeći mogućnosti slične drugim veb okvirima[**?**]. Kako dinamičko skriptiranje funkcioniše vidi se na slici 3.



Slika 3: Dinamička veb stranica: primer serverskog skriptiranja

Za specifične i naprednije scenarije korišćenja, PHP nudi dobro definisan i dokumentovan način za pisanje prilagođenih ekstenzija u C ili C++. Osim proširivanja samog jezika u obliku dodatnih biblioteka, proširenja pružaju način za poboljšanje brzine izvršavanja tamo gde je kritična i postoji prostor za poboljšanja korišćenjem pravom kompiliranog jezika. PHP takođe nudi dobro definisane načine za ugrađivanje u druge softverske projekte. Na taj način se PHP može lako koristiti kao interni skriptni jezik za drugi projekat, a takođe obezbeđuje usko povezivanje sa specifičnim internim strukturama podataka projekta.

4 Osnovne osobine, podržane paradigme i koncepti

PHP dozvoljava programerima da napišu ekstenzije u C-u kao i da dodaju funkcionalnost samom jeziku. PHP ekstenzije se mogu kompajlirati statički u PHP-u ili dinamički učitati tokom izvršavanja. Brojne ekstenzije su napisane kako bi dodale podršku za Windows API, upravljanje procesima na Unix zasnovanim operativnim sistemima, multibajt strune (Unicode), cURL i nekoliko popularnih formata kompresije.

Razne besplatne i open-source biblioteke su uključene u izvornoj distribuciji se koriste i u rezultujućim binarnim build-ovima PHP-a. PHP je u osnovi sistem koji je svestan interneta sa ugrađenim modulima za pristup FTP serverima, Microsoft SKL Server i SKLite (koji je ugrađena baza podataka), LDAP servere i druge. Brojne funkcije poznate C programerima, kao što su one u stdio zaglavlju, dostupne su u standardnim

PHP build-ovima. Ostale PHP funkcije koje su dostupne putem proširenja uključuju integraciju sa IRC-om, dinamičku generaciju slika i Adobe Flash sadržaja, PHP Data Objects(PDO) kao sloj apstrakcije koji se koristi za pristup bazama podataka, pa čak i sinteza govora. Neke od osnovnih funkcija jezika, kao što su one koje se bave nizovima, takođe su implementirane kao proširenja[**?**].

Drugi projekti, kao što je **Zephir**[**?**], pružaju mogućnost da PHP ekstenzije budu kreirane na jeziku visokog nivoa i kompajlirane u izvorne PHP ekstenzije. Takav pristup, umesto pisanja PHP ekstenzija direktno u C, pojednostavljuje razvoj ekstenzija i skraćuje vreme potrebno za programiranje i testiranje.

Origialna, potpuna i najraširenija implementacija PHP-a pokreće Zend Engine i poznata je kao PHP. Da bi se razdvojila od drugih implementacija, ponekad se nezvanično zove „Zend PHP". PHP-ov model „*jedan-zahtev-za-skriptom-izvršenje*"i činjenica da je Zend Engine interpreter dovodi do neefikasnosti. Kao rezultat toga, razvijeni su razni proizvodi koji pomažu poboljšanju performansi PHP-a. Kako bi se ubrzalo vreme izvršavanja i da se ne mora kompajlirati PHP izvorni kôd svaki put kada se pristupi veb stranici, PHP skripte se mogu primeniti u unutrašnjem formatu PHP motora korišćenjem opcode keša. Opcode keš funkcioniše tako što kešira kompajlirani oblik PHP skript (opcodes) u zajedničkoj memoriji kada se skripta pokrene. Jedan od opcode keševa, Zend Opcache[**?**], je ugradenu PHP od verzije 5.5. Drugi primer široko korišćenog opcode cashe-a je alternativni PHP keš (APC)[**?**], koji je dostupan kao PECL ekstenzija.

Dok je Zend PHP još uvek najpopularnija implementacija, razvijeno je nekoliko drugih implementacija. Neki od njih su kompajleri koji podržavaju JIT[**?**] kompilaciju i stoga nude prednosti u odnosu na Zend PHP na račun nedostatka potpune PHP kompatibilnosti. Alternativne implementacije uključuju sledeće:

- **HHVM(HipHop Virtual Machine)**[**?**] - razvijen na Facebook-u i dostupan kao open source, pretvara PHP kôd u bajt-kôd visokog nivoa, koji je obično poznat kao posredni jezik, koji se potom pretvara u k86-64 mašinski kôd. U vremenu izvršavanja kompajler je pravedan u vremenu (JIT), što rezultira poboljšanjem performansi do 6 puta.
- **Parrot**[**?**] - virtuelna mašina dizajnirana da efikasno pokreće dinamičke jezike. Pipp transformiše PHP izvorni kôd u Parrot posrednu reprezentaciju, koja se zatim prevodi u Parrot-ov bajt kôd i virtuelna mašina ga izvršava.
- **Phalanger** - sastavlja PHP u bajt kôd Common Intermediate Language (CIL)
- **Kuercus** - kompajlira PHP u java bajt kôd.

Već je rečeno da je PHP skriptni jezik opšte namene. Pored toga on podržava još mnoge koncepte. Od vezije 5.0 ima jaku podršku za **objektno-orijentisano programiranje**. To uključuje podršku za nasleđivanje, konstruktore, izuzetke, interfejse, klase i apstraktne klase. Takođe, podržava i funkcije prve klase(eng. first-class functions) što znači da funkcija može biti dodeljena promenljivoj. Korisnički definisane funkcije, kao i ugrađene funkcije mogu biti referisane od strane promenljive i pozivaju se dinamički. Podržava rekurziju, odnosno da funkcija poziva samu sebe, mada kod PHP-a je fokus na iteraciji. Funkcije se mogu prosleđivati kao argumenti

tj. funkcije višeg reda i funkcija može vratiti druge funkcije. Od verzije 5.3 uvedena je i podrška za anonimne funkcije sa podrškom za zatvaranje. Verzija 5.4 je dodala mogućnost povezivanja zatvaranja sa opsegom objekta i poboljšanu podršku za callables tako da se mogu koristiti naizmenično sa anonimnim funkcijama u gotovo svim slučajevima. Ove osobine opravdavaju pripadanje **funkcionalnoj paradigmi** [**?**].

5 Najpoznatija okruženja

Na samom početku ovog poglavlja treba da se upoznamo sa terminom **razvojno okruženje** (eng. framework). Razvojno okruženje predstavlja skup već gotovih komponenti koje čine skelet startne platforme. Ta platforma omogućava da se bude brži i efikasniji pri pisanju programa. Najbitniji faktor je to što se povećava brzina programiranja jer se ne počinje svaki put od nule već su na raspolaganju raznorazni gotovi moduli koji se mogu jednostavno koristiti. Razvojno okruženje čini rad na zahtevnim projektima jednostavnijim, omogućava pisanje čistog i ponovo upotrebljivog kôda i sugeriše na moguće greške.

Postoje razna razvojna okruženja za svaki programski jezik pa tako i za PHP. Koje razvojno okruženje izabrati zavisi najviše od samog projekta. Za jezik PHP najpoznatija razvojna okruženja su Laravel[**?**], CodeIgniter[**?**] i Symfony[**?**].

Laravel programeri najviše koriste zbog njegove brzine, fleksibilnog rutiranja i mogućnosti upravljanja preko komandne linije. Odlikuje ga jednostavno i intuitivno korišćenje kao i čist i pregledan kôd. U potpunosti je baziran na MVC arhitekturi eng(Model Wiew Controller).

CodeIgniter je razvojno okruženje koga odlikuje bezbednost, poseduje zaštitu od XSS i CSRF napada. Uopšte nije memorijski zahtevan jer zauzima svega par megabajta. Omogućava MVC arhitekturu, ali nije striktan po tom pitanju. Poseduje sveobuhvatnu dokumentaciju koja se dobija uz samo razvojno okruženje.

Pored sjajnih osobina poput brzine, fleksibilnosti, ponovno upotrebljivih komponenti, razvojno okruženje **Symfony** karakteriše izuzetna podrška. Osim podrške koju pruža sama firma Sensio, tvorac ovog razvojnog okruženja, veliki broj profesionalaca i firmi koristi upravo Symfony, pa je samim tim moguće dobiti odgovore na sva pitanja. Ovo razvojno okruženje podržava trenutne standarde PHP-a i pored toga omogućava korisnicima da koriste delove vlastitog softvera, bez da koriste celo razvojno okruženje.

6 Instalacija

Za kreiranje interaktivnih veb stranica i izvršavanje PHP programa neophodan je server koji podržava PHP. Prilikom samog razvoja bilo kog PHP projekta komunikacija sa tim serverom je veoma česta, stoga je izuzetno nepraktično stalno komunicirati sa nekim udaljenim serverom. Iz tog razloga se preporučuje instalacija lokalnog servera. Osim servera potrebna je i baza podataka. Naravno osim servera i baze, potrebno je instalirati i sam jezik PHP. Na novijim operativnim sistemima PHP uglavnom dolazi uz sam sistem, ali će ovde instalacija biti objašnjena od samog početka, kao da ništa od potrebnog nije instalirano.

Dakle, potrebni su sam PHP programski jezik, server i baza podataka. Postoji mogućnost da se zasebno svaka od navedenih stavki instalira i

podešava, ali postoji i mnogo lakši način, a to su LAMP(linux apache mySql PHP) [?] i WAMP(windows apache mySql PHP) [?] serveri.

Na operativnom sistemu **Windows** sama instalacija se sastoji od preuzimanja WAMP servera sa stranice:http://www.wampserver.com/en/ , zatim sledi pokretanje preuzetog fajla koje automatski pokreće instalaciju celog paketa. Sam proces nadalje je automatizovan i WAMP server paket stiže sa najnovijim verzijama Apache-a, MySQL-a i PHP-a. Nakon procesa instalacije biće kreirana prečica ka WampServer-u kao i folder www, obično na putanji c:\\wamp\\www. Nakon toga u folderu www potrebno je kreirati poddirektorijum koji će služiti za smeštanje PHP fajlova. Na primer, kreiran je fajl 1.php. Moguće je otvoriti ga u veb pregledaču, tako što se u meniju WampServer-a klikne na link "localhost" ili se u pregledaču ukuca http://localhost/1.php.

Kod operativnog sistema **Linux** instalacija se vrši preko terminala i zahteva instaliranje svake komponente pojedinačno, što je prikazano u 1.

```
sudo apt-get update #update sistema
sudo apt-get install apache2 #instalacija Apache-a
sudo apt-get install mysql-server libapache2-mod-auth-mysql php5-
    mysql #instalacija MySQL-a
sudo mysql_install_db #instalacija MySQL servera
sudo /usr/bin/mysql_secure_installation #pokretanje skripte za
    podesavanje MySQL-a
sudo apt-get install php5 libapache2-mod-php5 php5-mcrypt #
    instalacija PHP-a
```

Listing 1: Pokretanje instalacije na Linux operativnom sistemu

Po završetku instalacije MySQL-a može biti traženo da se postavi root šifra što je i potrebno uraditi.

PHP takođe sadrži veliki broj korisnih biblioteka i modula koji se mogu dodati na server. Za izlistavanje potrebno je pokrenuti komandu "apt-cache search php5-" i biće prikazani svi moduli. Za instalaciju nekog od tih modula služi komanda "sudo apt-get install <ime_ modula>". Nakon što je instaliran LAMP može se pokrenuti prethodno napravljeni program, npr. 1.php. Potrebno je smestiti ga u /var/www direktorijum i može se pokrenuti preko pregledača ukoliko se ukuca http://localhost/1.php.

7 Primer kôda

Primer 7.1 *Primer kako se pomoću PHP-a može konektovati na bazu podataka, izvršiti upit i ispitati dobijene rezultate. Cilj u ovom primeru 2 je pronaći sve studente sa zadatim imenom i prezimenom i ispitati njihov datum rođenja.*

```
<?php
  //uspostavljanje konekcije sa bazom podataka
  if(mysqli_connect_errno()){
    die("Problem sa povezivanjem: ".mysqli_connect_error());
  }
  //deklarisanje promenljivih i ucitavanje podataka
  $ime = $_GET['ime'];
  $prezime = $_GET['prezime'];
  //formulisanje i izvrsavanje upita
  $upit = " select datum_rodjenja from dosije where ime = '$ime'
     and prezime = '$prezime' ";
  //rezultat upita se smesta u promenljivu rezultat
  $rezultat = mysqli_query($veza, $upit) or die("Problem prilikom
     izvrsavanja upita: ".mysqli_error($veza));
```

```
14    //racunanje broja redova rezultata (broj studenata sa zadatim
          imenom i prezimenom)
      $broj_pojavljivanja = mysqli_num_rows($rezultat);
16    if($broj_pojavljivanja == 0){
        echo "Nema studenata koji se zovu $ime $prezime.";
18    }
      else{
20      //ukoliko ima studenata ispisujemo ih
        echo "<ul>";
22      for($i=0; $i<$broj_pojavljivanja; $i++){
          $red = mysqli_fetch_assoc($rezultat);
24        echo "<li>";
          echo $red['ime']['prezime']['datum_rodjenja'];
26        echo "</li>";
        }
28      echo "</ul>";
      }
30    //raskidamo konekciju sa serverom
      mysqli_close($veza);
32  ?>
```

Listing 2: Primer PHP kôda

Sav PHP kôd se piše između "<?php" i "?>", slično kao tagovi pri pisanju html kôda. Promenljive deklarišemo sa prefiksom "$" a naredbom "echo" vršimo ispis. $_ GET metod smo koristili kako bismo preneli vrednosti promenljivih kroz veb stranicu do našeg kôda. Vrednosti tih promenljivih se dodaju u URL stranicu. Na primer ako bismo želeli da izlistamo sve studente koji se zovu Marko i prezivaju Marković, to bismo uradili tako što bismo u URL adresu dodali ?ime=Marko& prezime=Markovic.

8 Specifičnosti

Ono što je veoma specifično za ovaj jezik jeste to što nema striktnu sintaksu koja prati veliki broj jezika. Na primer, sledeći fragment kôda možemo zapisati na dva razlicita načina. Kao što je prikazano u 4 standardni pristup koristi vitičaste zagrade, dok je u alternativnom pristupu fokus na elseif grananju.

```
<?php
2 if($a > 0){
    echo "a ima pozitivnu
      vrednost";
4 } elseif($a == 0){
    echo "a je nula";
6 } else{
    echo "a ima negativnu
      vrednost";
8 }
  ?>
```

```
<?php
2 if($a > 0):
    echo "a ima pozitivnu
      vrednost";
4 elseif($a == 0):
    echo "a je nula";
6 else:
    echo "a ima negativnu
      vrednost";
8 endif;
  ?>
```

Slika 4: Primer dva fragmenta koda standardni pristup (levo) i alternativni pristup (desno).

Imenski prostor (eng. namespace) [**?**] je uveden od verzije 5.3 i prevashodno je zamišljen radi rešavanja konflikata oko istoimenih metoda i funkcija, kako bi mogle da se koriste u okviru istog projekta. Ovo je posebno pogodno u slučaju da se radi o kompleksnim projektima, koji su raspoređeni po različitim datotekama i kada u razvoju aplikacije učestvuje grupa programera. Za deklarisanje imenskog prostora koristi se ključna

reč **namespace**. Dozvoljeno je koristiti više imenskih prostora u okviru istog programa, čak i iste datoteke.

Još jedan veoma bitan koncept u radu sa PHP programima jeste rad sa formama. Forma je zapravo deo HTML strukture za grupisanje različitih elemenata čija je funkcija prikupljanje podataka od korisnika. U zavisnosti od akcije korisnika, dobijena informacija se može proveriti kod korisnika. Takođe se može proslediti kao zahtev serveru, eventualno na ponovnu proveru i kasniju obradu. Po potrebi na kraju se može poslati neka vrsta povratne informacije korisniku. Forme se koriste, na primer, kod poručivanja hrane u piceriji preko interneta. Popunjavamo formular gde biramo vrstu pice, dodatke, adresu gde će se pica dostaviti itd. Nakon obrade i čitanja podatka iz forme, šalje se odgovor da je porudžbina primljena i da će pica ubrzo stići.

Jezik PHP se odlikuje ogromnim brojem različitih klasa koje u mnogome olakšavaju rad programerima. Jedna veoma bitna klasa je "PDO (PHP Data Object)"[**?**, **?**] koja se koristi za povezivanje sa SQL bazama podataka. Ta klasa obezbeđuje sloj apstrakcije za skup upravljačkih programa za baze podataka kao što su MySQL, PostgreSQL i MSSQL. To znači da ma koju bazu podataka koristili, ako je PDO podržava, možemo koristiti iste funkcije za izvršavanje istih operacija nad bazom podataka. To čini kôd i samu aplikaciju portabilnom u smislu da se može koristiti u različitim bazama podataka bez nekakve modifikacije samog kôda.

Još jedna veoma korisna i zamiljiva klasa je DateTime klasa [**?**, **?**] koja služi za reprezentaciju vremena i datuma. Pored jednostavnog definisanja i ispisa trenutnog vremena koje je predstavljeno u 3, ova klasa sadrži ogroman broj različitih funkcija za rad sa datumima poput izdvajanja dana, meseca, godine iz datuma i postavljanje vremenskih zona.

```
$d = new DateTime('2019-01-01T15:03:01.012345z')
echo $d -> format('Y-m-d\TH:i:s.u') //2019-01-01T15:03:01.012345
```

Listing 3: Primer upotrebe klase DateTime

Ono što se u jeziku PHP dosta koristi jesu **kolačići** (eng. cookies)[**?**, **?**] koji se često koriste za identifikaciju korisnika. Kolačić je mala datoteka koju server ugrađuje na računar korisnika. Svaki put kada isti računar zahteva stranicu, ona će takođe poslati i kolačić. Pomoću PHP-a možemo kreirati i dohvatiti vrednosti kolačića. Funkcija koja postavlja vrednost kolačića je data u primeru 4.

```
setcookie(name, value, expire); //postavljanje kolacica
setcookie("user", "", time() - 3600); //brisanje kolacicaica
```

Listing 4: Funkcija za postavljanje kolacica

Pristup disku je spor dok je pristup mreži još sporiji. Baze podataka obično koriste oba. Korišćenje lokalnog keša izbegava opterećenje mreže i pristupa disku. Kombinujući ove pristupe dobijamo sistem za memorisanje objekata distribuirane memorije (eng. memcached) [**?**]. Ako naša aplikacija nije distribuirana na više servera, verovatno nam neće trebati memcached. Jednostavniji pristupi keširanju su na primer serijalizovanje podataka i čuvanje u privremenoj datoteci, na primer - mogu eliminisati mnogo redundantnih podataka na svakom pojedinačnom zahtevu[**?**]. Primer keširanja je dat primerom 5.

```
<?php
$feed = apc_fetch('news');
if ($feed === FALSE) {
```

```
4     $feed = file_get_contents('http://example.org/news.xml');
      // Cuvanje podataka u deljenoj memoriji 5 minuta (5*60 sekundi
      )
6     apc_store('news', $feed, 300);
}
8 // Uradi nesto sa $feed.
?>
```

Listing 5: Primer keširanja

9 Zaključak

Programski jezik PHP značajno olakšava veb programiranje i zbog toga je među jednim od najpopularnijih programskih jezika današnjice. Pored toga, rad sa bazama podataka je dosta jednostavan, pa to predstavlja jednu od najbitnijih olakšica u veb programiranju, korišćenjem PHP-a. Mogućnosti ovog jezika su velike, međutim kroz ovaj rad su predstavljene samo one koje ga ističu u odnosu na druge. Takođe, ovim radom smo istakli sve ono što bi programere koji se do sad nisu susreli sa PHP-om zainteresovalo da počnu da ga koriste.