5 OBTINEREA SI PROCESAREA DATELOR DE LA UTILIZATOR

Studentul poate utiliza prezentul material si informatiile continute in el exclusiv in scopul asimilarii cunostintelor pe care le include, fara a afecta dreptul de proprietate intelectuala detinut de InfoAcademy.

5.1 SURSE EXTERNE DE INFORMATIE PENTRU INTERPRETORUL PHP

5.1.1 Ce informatii pot fi folosite pentru generarea dinamica a paginilor web?

Generarea dinamica a paginilor web reprezinta crearea pe loc a unei pagini pe serverul web, ca urmare a cererii primite de la un client. Prin contrast, in cazul paginilor statice, acestea sunt deja create si stocate in fisiere HTML, ele trebuind doar livrate clientului ca raspuns la cererea sa. In ambele cazuri insa, orice actiune a serverului web este determinata de o cerere a unui client.

Generarea dinamica a unei pagini web se foloseste atunci cand continutul paginii nu poate fi pre-generat si stocat intr-un fisier, ci depinde de parametri care se pot schimba in timp sau in functie de client (ex: data/ora, adresa IP a clientului, portul acestuia, informatii provenite de la client etc). Spre exemplu, un magazin online, unde utilizatorul filtreaza produsele in functie de categorie, caracteristici etc. si unde nu exista pagini pregenerate pentru toate combinatiile posibile de filtre, ci se genereaza pe loc pagina dorita in functie de optiunile utilizatorului.

Atunci cand generarea dinamica a unei pagini web este facuta de catre un script PHP rulat prin intermediul serverului web, lansarea in executie a interpretorului PHP se face ca urmare a cererii unui client, iar scriptul poate lua decizii in functie de informatii provenite din diferite surse (server, sistemul sau de operare, clientul care a facut cererea etc). Parametrii in functie de care scriptul PHP poate lua decizii pot fi incadrati in 3 categorii:

- date produse intern de catre interpretorul PHP – spre exemplu, numere aleatoare sau data/ora. Exemplu: un script care genereaza o pagina ce contine un banner publicitar si care alege in mod aleator bannerul afisat dintr-o lista predefinita
- date externe interpretorului PHP
    - informatii locale – provenite de la server sau de la sistemul de operare. Aceste informatii nu depind de clientul care a facut cererea ce a determinat rularea interpretorului PHP
        - informatii despre server: numele serverului, softul de server web folosit, adresa si portul serverului etc.
        - informatii provenite de la sistemul de operare, prin intermediul variabilelor de mediu
    - informatii provenite de la clientul web
        - caracteristici ale clientului (adresa si port, browser folosit, tipuri de continut pe care acesta la accepta etc)
        - date provenite de la client (de exemplu, informatiile introduse de utilizator intr-un formular HTML)

5.1.2 Cum ajung informatiile externe la interpretorul PHP?

5.1.2.1 Informatiile locale

Informatiile locale devin disponibile interpretorului PHP prin mecanisme interne ale serverului web sau ale sistemului de operare in care acesta ruleaza:

- variabilele de mediu - fiecare proces din sistemul de operare dispune de asa-numitul "environment" – un set de informatii sub forma de perechi nume-valoare, care pot fi mostenite sau setate la pornirea procesului si la care are procesul in cauza are acces
- informatiile despre server – interfata intre modulul PHP si serverul web este de asa natura gandita incat serverul poate transmite interpretorului PHP informatii despre sine

5.1.2.2 Informatiile de la clientul web

Informatiile de la client ajung in posesia interpretorului PHP prin intermediul mesajelor HTTP ce circula intre client si server. Cand clientul trimite serverului un mesaj HTTP ce contine informatie, acesta din urma receptioneaza mesajul si paseaza informatia interpretorului PHP, care o poate apoi folosi pentru a lua decizii in momentul generarii paginii web.

Studentul poate utiliza prezentul material si informatiile continute in el exclusiv in scopul asimilarii cunostintelor pe care le include, fara a afecta dreptul de proprietate intelectuala detinut de InfoAcademy.

In protocolul HTTP, dialogul intre client si server consta din:

- cereri ale clientului, in care acesta solicita serverului o anumita resursa
- raspunsuri ale serverului, ce contin datele corespunzatoare resursei solicitate. Serverul nu trimite date catre client decat ca urmare a unei cereri a acestuia din urma.

Observatie: *serverul HTTP nu initiaza comunicatia cu clientul – el doar reactioneaza la cereri ale acestuia.*

O cerere HTTP contine:

- resursa solicitata de catre client – de obicei, sub forma unui URL sau URI (ex: /products/index.php – calea relativa la document root)
- tipul de cerere – exista mai multe tipuri de cereri HTTP, insa cele prin intermediul carora interpretorul PHP poate primi date de la client sunt doua: GET si POST (ele vor fi detaliate in cadrul acestui material). Tipul de cerere specifica operatia aplicata resursei cerute; in functie de aceasta, cererea poate sa contina sau nu date de la client (iar datele respective pot avea o limita mai mica sau mai mare), cererea poate produce sau nu modificari pe server (creare/stergere de fisiere, adaugare/modificare/stergere de inregistrari dintr-o baza de date) etc
- (optional) informatii de la client - fie caracteristici ale clientului, fie date introduse de catre utilizator

Exemple:

- utilizatorul scrie in browser adresa www.example.com/index.html si apasa Enter. Rezultatul va fi trimiterea catre server a unei cereri HTTP de tip GET, care este inofensiva (scopul ei este simpla obtinere de informatie de la server, ea putand fi efectuata in mod repetat fara a produce modificari pe server)
- utilizatorul completeaza un formular HTML si apasa butonul Submit. Ca urmare, catre server va pleca o cerere HTTP de tip POST ce contine informatiile introduse de catre utilizator in formular

Nota: *este posibil ca clientul sa trimita informatii catre server si prin intermediul unei cereri de tip GET (vezi sectiunea din material dedicata lui GET)*

5.1.3 In ce fel sunt disponibile informatiile externe in PHP?

Interpretorul PHP memoreaza datele externe receptionate pe caile descrise mai sus in variabile predefinite cu domeniu de vizibilitate global (superglobals). Valorile acestor variabile pot fi apoi folosite in cadrul scriptului PHP pentru a genera pagina web, in doua feluri:

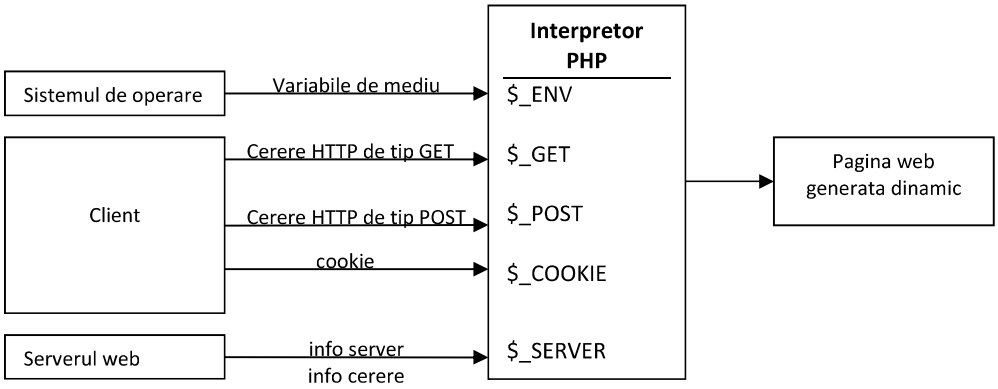
- includerea informatiilor in cauza in pagina web (ex: un script PHP genereaza o pagina web care ii raporteaza clientului adresa sa IP)
- folosirea informatiilor pentru a lua decizii ce determina continutul paginii web (ex: intr-un motor de cautare scris in PHP, cuvintele cheie introduse de utilizator sunt transmise serverului, de aici interpretorului PHP iar acesta genereaza o pagina web ce depinde de cuvintele cheie alese)

Iata cele mai importante variabile de tip superglobal ce memoreaza informatii externe interpretorului PHP:

- $_ENV – tablou cu informatiile extrase din variabilele de mediu (environment)
- $_GET – tablou cu informatiile provenite prin intermediul cererii HTTP de tip GET
- $_POST – tablou cu informatiile provenite prin intermediul cererii HTTP de tip POST
- $_COOKIE – informatii pe care serverul le-a stocat anterior pe client sub forma de cookie si care ii sunt retrimise cu ocazie cererilor urmatoare
- $_SERVER – informatii despre server (adresa, nume, port, soft etc) si despre cererea curenta (tip de cerere, URI, calea in sistemul de fisiere catre resursa solicitata etc)
- $_FILE – tablou cu informatii despre fisierele trimise de utilizator (din formulare care permit upload de fisiere)

Studentul poate utiliza prezentul material si informatiile continute in el exclusiv in scopul asimilarii cunostintelor pe care le include, fara a afecta dreptul de proprietate intelectuala detinut de InfoAcademy.

- $_REQUEST – tablou asociativ cu informatii cumulate (GET+POST+cookie)



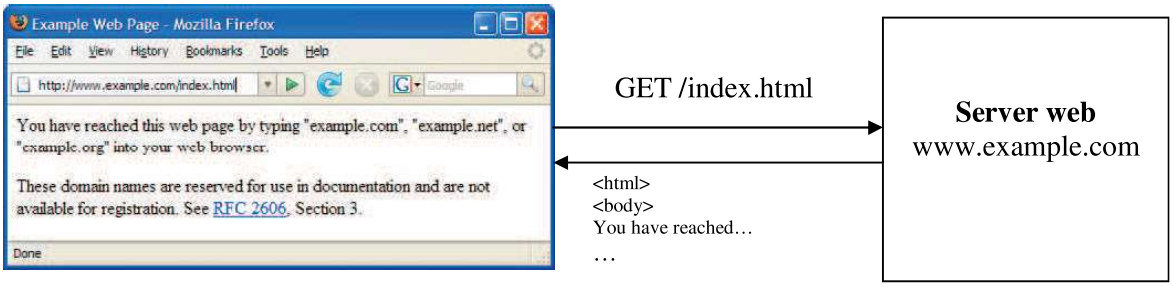
Nota: *In documentatiile PHP se intalnesc acronime precum EGPCS sau GPC, care se refera la sursele de informatii externe interpretorului:* ***E****nvironment,* ***G****ET,* ***P****OST,* ***C****ookie,* ***S****erver.*

Fiecare script PHP rulat prin intermediul serverului web este lansat in executie in urma unei cereri emise de catre un anume client web. Fiecare script rulat astfel va contine in variabilele predefinite enumerate mai sus informatiile legate de acel client particular si de cererea sa. Cu alte cuvinte, daca mai multi clienti efectueaza cereri catre acelasi server web solicitand o aceeasi adresa ce are ca efect executia unui script PHP, fiecare instanta a scriptului in cauza va contine alte valori pentru variabilele predefinite – cele proprii clientului a carui cerere a determinat rularea scriptului.

5.1.4 Cererea HTTP de tip GET

5.1.4.1 Descriere, utilizare si interactiunea cu PHP

Cererea de tip GET este cea trimisa serverului atunci cand utilizatorul scrie in browser o adresa si apasa Enter. Este folosita in cele mai dese cazuri pentru a solicita pagini serverului web. Atunci cand utilizatorul scrie in browser http://www.example.com/index.html, catre server va pleca o cerere *GET /index.html*:



Exista posibilitatea ca clientul sa trimita informatii serverului web prin intermediul unei cereri de tip GET folosind asa-numitul *query string*. Query string-ul este o parte optionala a URL-ului; el urmeaza adresei propriu-zise, incepe cu ? si continua cu perechi nume=valoare separate prin caracterul &. Iata exemplul cautarii cuvantului php folosind google:

```
http://www.google.com/search?hl=en&q=php&btnG=Google+Search
                            query string
```

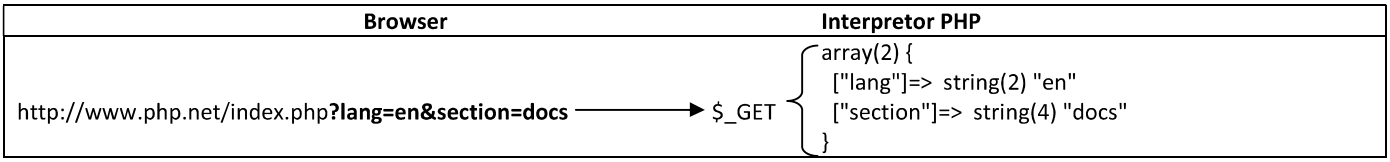
Studentul poate utiliza prezentul material si informatiile continute in el exclusiv in scopul asimilarii cunostintelor pe care le include, fara a afecta dreptul de proprietate intelectuala detinut de InfoAcademy.

Observatie: *atunci cand un URL contine query string, resursa referita de catre acel URL trebuie sa fie un program capabil sa acceseze si sa prelucreze datele primite. Nu ar avea sens, de exemplu, un URL cu query string care pointeaza catre un fisier HTML.*

Browserul web poate ajunge sa ceara serverului un URI ce contine query string din diferite motive – iata cateva:

- utilizatorul scrie de mana URL inclusiv query string
- utilizatorul da click pe un link a carui tinta este un URL ce contine si query string
- utilizatorul completeaza un formular HTML care a fost configurat sa trimita datele catre server prin intermediul unei cereri de tip GET (vezi sectiunea despre formulare HTML)

Daca, corespunzator URL-ului cerut de client, serverul web lanseaza interpretorul PHP, acesta plaseaza automat informatiile primite prin query string in variabila de tip superglobal $_GET, sub forma unui tablou asociativ ale carui perechi cheie-valoare corespund perechilor nume-valoare din query string:



Informatiile provenite din query string pot fi folosite de programul PHP pentru a lua decizii:

```
$languages = array("en","ro","fr");       // lista de limbi disponibile
$lang = "en";                             // limba din oficiu

/* verificam daca s-a cerut o anumita limba prin intermediul query string si,
in caz afirmativ, daca aceasta este disponibila */
if(isset($_GET['lang']) && in_array($_GET['lang'], $languages))
      $lang = $_GET['lang'];

switch($lang){
      case "en": echo "Hello world!"; break;
      case "ro": echo "Soro lume!"; break;
      case "fr": echo "Bonjour monde!"; break;
}
```

Presupunand ca acest program este salvat intr-un fisier numit hello.php, aflat in document root-ul unui server HTTP ce ruleaza pe aceeasi masina cu browserul, iata ce rezultate am obtine pentru diverse URL-uri incercate:

Daca scriem in browser URL-ul...	...obtinem...	Comentarii
http://localhost/hello.php?lang=fr	Bonjour monde!	$_GET['lang'] va avea valoarea 'fr', care este o limba disponibila
http://localhost/hello.php?lang=ro	Soro lume!	Analog cu cazul "fr"
http://localhost/hello.php?lang=en	Hello world!	Chiar daca "en" este default, se procedeaza la fel ca mai sus
http://localhost/hello.php?lang=de	Hello world!	"de" nu se afla in lista de limbi disponibile; se foloseste default-ul "en"
http://localhost/hello.php	Hello world!	$_GET['lang'] nu este definit, asadar $lang ramane cu valoarea default
http://localhost/hello.php?section=docs	Hello world!	$_GET['lang'] nu este definit, asadar se procedeaza ca mai sus. Exista in schimb $_GET['section'] dar care nu este folosit in script

Studentul poate utiliza prezentul material si informatiile continute in el exclusiv in scopul asimilarii cunostintelor pe care le include, fara a afecta dreptul de proprietate intelectuala detinut de InfoAcademy.

5.1.4.2 Avantaje si dezavantaje ale cererii de tip GET

Precum s-a observat, in cererea de tip GET toate datele trimise de utilizator serverului fac parte din URL, ceea ce are urmatoarele implicatii:

- datele sunt la vedere
  - avantaj: in acest fel, utilizatorul poate salva un bookmark catre o pagina generata pe baza input-ului sau, pentru ca URL-ul contine toate informatiile necesare generarii acelei pagini particulare
  - dezavantaj: nu este recomandabila transmiterea informatiilor sensibile in acest fel (ex: informatii de login).
- lungimea datelor pasabile prin intermediul URL-ului este drastic limitata, deoarece lungimea maxima a URL-ului este in general in jurul a 255 de caractere
- valorile folosite in query string trebuie sa treaca printr-un proces de URL encoding, deoarece exista caractere care in URL au semnificatie speciala (ex: /, &, = etc) sau sunt nepermise

Introduction to URL encoding: http://www.permadi.com/tutorial/urlEncoding/

Date fiind cele enumerate mai sus, deducem ca cererea de tip GET nu este in general potrivita pentru a transmite serverului web informatiile introduse de catre utilizator intr-un formular HTML. Daca formularul este unul de login, userul si parola introduse de utilizator s-ar vedea in URL dupa Submit; chiar si in cazul in care formularul are alt scop, oricum lungimea datelor ce pot fi introduse de utilizator ar fi mult limitata.

Cererea de tip GET a fost gandita pentru a solicita date serverului web fara a modifica starea acestuia (in terminologie HTTP, este o cerere de tip *safe*). Chiar si atunci cand prin intemediul ei se trimit date catre server, acestea nu trebuie sa produca modificari permanente (ex: introducerea/stergerea/modificarea informatiilor dintr-o baza de date). Query string-ul este folosit in general pentru a personaliza ceea ce se afiseaza intr-o pagina generata dinamic (ex: limba folosita, categorii de continut dorite, numar de rezultate afisate in urma interogarii unei baze de date etc).

5.1.5 Cererea HTTP de tip POST

Daca cererea de tip GET era in general folosita pentru a solicita resurse serverului (insa cu posibilitatea de a-i transmite acestuia informatii in caz de nevoie, ca parte a URL-ului), cererea de tip POST a fost gandita special pentru ca clientul sa trimita date serverului in vederea procesarii. POST este folosit in general pentru expedierea catre server a datelor introduse de utilizator intr-un formular (de exemplu, in urma apasarii butonului cu rol de Submit).

Datele trimise cu POST nu mai fac parte din URL, ci sunt transmise ca payload (incarcatura) a mesajului HTTP, consecintele fiind:

- datele trimise serverului nu mai sunt vizibile in URL
  - avantaj: putem folosi POST pentru a transmite informatii de autentificare dintr-un formular HTML
  - dezavantaj: utilizatorul nu mai poate face bookmark unei pagini generate pe baza informatiilor din POST, deoarece URL-ul este insuficient de aceasta data pentru a reconstrui acea pagina particulara. Daca utilizatorul completeaza un formular de mai multe ori, cu date diferite, dupa fiecare expediere a datelor catre server URL-ul vizibil in browser va fi acelasi
- limita impusa cantitatii de date trimise serverului este mult mai sus decat in cazul GET, si este configurabila din setarile serverului web si ale interpretorului PHP (ex: in fisierul de configurare php.ini, limita default este de 2MB)

Atentie! *Faptul ca datele din POST nu sunt vizibile in URL nu inseamna ca utilizatorul nu are acces la ele sau nu le poate falsifica! In masura in care avem acces la mesajele HTTP traficate intre client si server, putem folosi un analizor de protocol pentru a interpreta contitnutul lor si a obtine datele trimise serverului. In plus, exista extensii ale diverselor browsere care permit utilizatorului sa vizualizeze si chiar sa modifice datele trimise prin POST inaintea expedierii acestora. Concluzie:*

Studentul poate utiliza prezentul material si informatiile continute in el exclusiv in scopul asimilarii cunostintelor pe care le include, fara a afecta dreptul de proprietate intelectuala detinut de InfoAcademy.

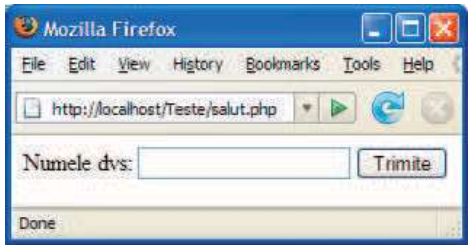
chiar daca impunem ca un formular sa returneze un set limitat de valori (folosind componente de tip drop-down list, checkbox etc) nu trebuie sa ne bazam exclusiv pe aceasta limitare.

In interiorul mesajului HTTP, datele din formular sunt transmise tot sub forma unei succesiuni de perechi nume=valoare, separate prin &. Un interpretor PHP care ruleaza prin intermediul serverului web va receptiona aceste informatii si le va memora in variabila de tip superglobal cu numele $_POST, analog cu cazul lui GET (tablou asociativ).

Exemplu - un formular in care utilizatorul isi introduce numele, acesta fiind ulterior afisat pe ecran:

```
$formular = <<<GATA
<html><body>
<form method="post" action="/Teste/salut.php">
Numele dvs:
<input type=text name=nume>
<input type=submit value=Trimite>
</form>
GATA

// daca a fost introdus un nume, il salutam
if(!empty($_POST['nume'])){
      echo "Salut ".$_POST['nume']."!";
}
else echo $formular;
```



Formularul contine un camp denumit *nume* a carui valoare va fi trimisa serverului in momentul apasarii pe Trimite. Fisierul este plasat in subdirectorul *Teste* din document root si se numeste *salut.php*.

Nota: *discutia detaliata a formularelor HTML este facuta intr-o sectiune separata a materialului, exemplul de mai sus avand doar scopul clarificarii mecanismului cererii de tip POST si a lucrului cu datele provenite din POST in PHP.*

In functie de actiunea utilizatorului, putem vedea urmatoarele manifestari ale formularului nostru:

Actiunea utilizatorului	Ce se afiseaza in browser	Comentarii/explicatii
Scrie in browser http://localhost/Teste/salut.php	formularul	$_POST['nume'] nu este definit, deoarece cererea trimisa serverului este de tip GET
Apasa Trimite fara a scrie nume	formularul	$_POST['nume'] este definit, insa are ca valoare sirul vid, deoarece in campul de nume nu a fost introdus nimic
Scrie "Marius" si apasa Trimite	Salut Marius!	$_POST['nume'] exista si are valoarea "Marius" de tip string

5.2 FORMULARE HTML SI TRATAREA INPUT-ULUI PROVENIT DIN ACESTEA

5.2.1 Generalitati

Un formular HTML este format dintr-un ansamblu de elemente ce permit utilizatorului sa ofere input sub diverse forme:

- alegerea dintr-o lista de valori impuse
- introducerea de mana a unei valori, a unui sir de caractere etc.
- bifarea unei optiuni independente sau care este parte a unui grup

Fiecare element din formular are o valoare din oficiu si o valoare curenta (ex: un camp pentru introducerea varstei are ca valoare initiala sirul vid, urmand ca ea sa se schimbe in momentul in care utilizatorul introduce text in acel camp). Utilizatorul completeaza formularul modificand valorile diferitelor elemente componente. Odata informatiile introduse in formular, trimiterea lor catre server se face de obicei ca urmare a apasarii pe un buton cu rol de Submit (expediere a datelor).

Studentul poate utiliza prezentul material si informatiile continute in el exclusiv in scopul asimilarii cunostintelor pe care le include, fara a afecta dreptul de proprietate intelectuala detinut de InfoAcademy.

Nota: *este posibil ca trimiterea datelor din formular sa se faca si in urma altei actiuni decat apasarea pe butonul de submit, daca se folosesc limbaje de scripting care ruleaza pe client (ex: JavaScript)*

Definirea unui formular HTML se face folosind tag-ul FORM, care este analizat mai jos:

```
<form method="POST" action="/cale/catre/script.php">
…
elemente de formular HTML
…
</form>
```

Tag-ul **form** poate avea diferite atribute, dintre care de interes pentru noi sunt urmatoarele:

- **method** – specifica tipul de cerere HTTP care va fi folosita pentru a trimite datele catre serverul web cand utilizatorul apasa pe butonul de submit. Daca se foloseste GET, toate datele trimise vor fi incluse in URL, sub forma de query string (cu toate implicatiile discutate mai sus, la cererea de tip GET). Daca se foloseste POST, URL-ul afisat de browser ramane neschimbat dupa apasarea butonului de Submit, insa datele vor fi receptionate de catre server ca parte a mesajului HTTP.
- **action** – specifica adresa web a programului care va prelucra datele introduse in formular. Adresa poate indica un alt fisier decat cel in care este definit formularul, sau acelasi – nu este deloc neobisnuit ca un script PHP sa efectueze fie prelucrarea datelor, fie generarea codului HTML al paginii web, fie ambele - in functie de datele provenite de la utilizator.

Nota: *atunci cand acelasi script efectueaza atat generarea codului HTML cat si prelucrarea datelor, valoarea atributului action al tag-ului form poate fi setata ca $_SERVER['PHP_SELF'], care reprezinta calea catre fisierul PHP relativa la document root. Aceasta ne scuteste de a modifica toate formularele atunci cand locatia sau numele scriptului se schimba (asa cum s-ar intampla daca am folosi ca valoare a lui action calea scrisa de mana ("hard-coded")).*

Iata un exemplu de formular simplu, ce contine doua campuri pentru introducere de text, si care trimite datele catre server sub forma unei cereri de tip GET:

```
<form method="GET" action="http://www.exemplu.ro/persoana.php">
      <input type="text" name="nume" />
      <input type="text" name="varsta" />
      <input type="submit" value="Trimitere date"
</form>
```

Formularul contine doua campuri in care utilizatorul poate introduce text, si un buton de submit. Daca utilizatorul introduce numele Andrei si varsta 30, apasand apoi pe butonul Trimitere, catre serverul www.exemplu.ro va pleca urmatoarea cerere:

```
GET /persoana.php?nume=Andrei&varsta=30
```

Informatiile introduse in formular vor fi automat disponbile in interiorul scriptului persoana.php prin intermediul elementelor $_GET['nume'] si $_GET['persoana'] ale tabloului de tip superglobal $_GET (detalii la sectiunea despre cererea de tip GET).

Daca atributul *method* al formularului ar fi avut valoarea "POST", cererea plecata catre serverul web ar fi aratat astfel:

```
POST /persoana.php
```

insa in interiorul mesajului HTTP (in payload) s-ar fi gasit sirul *nume=Andrei&varsta=30*.

Studentul poate utiliza prezentul material si informatiile continute in el exclusiv in scopul asimilarii cunostintelor pe care le include, fara a afecta dreptul de proprietate intelectuala detinut de InfoAcademy.

In continuarea materialului vor fi prezentate principalele elemente ce intervin in formularele HTML, impreuna cu felul in care input-ul furnizat de acestea parvine scriptului PHP si de sugestii/practici uzuale de prelucrare a datelor primite.

5.2.2 Lista de elemente ale formularelor HTML

Iata o lista a principalelor tipuri de elemente ce pot aparea intr-un formular HTML:

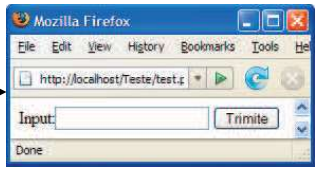
- **butoane**
  - cu rol de submit – apasarea lor determina trimiterea datelor curente ale formularului catre server
  - cu rol de reset – aduc valorile elementelor din formular la cele default
  - fara rol special – li se pot atasa actiuni prin intermediul unor limbaje de scripting ce ruleaza pe client
- **text** – un camp in care utilizatorul poate scrie/edita text, pe o singura linie
- **text area** – un camp de introducere/editare de text pe mai multe linii
- **drop-down list** – o lista de elemente predefinite din care utilizatorul poate alege o singura valoare
- **lista cu selectie multipla** – o lista de elemente predefinite in care utilizatorul poate selecta mai multe valori simultan
- **checkbox** – casuta care poate fi bifata sau debifata independent de altele
- **radio button** – elementele de acest tip sunt folosite pentru a crea un set de optiuni din care numai una poate fi selectata la un moment dat; selectarea altei optiuni o deselecteaza automat pe cea anterioara

5.2.3 Butonul de expediere a formularului (submit)

5.2.3.1 Definire si atribute

Butonul de submit este un element al formularului care are functie speciala: apasarea sa determina trimiterea datelor curente din formular catre server, sub forma unei cereri HTTP de tip GET sau POST (in functie de valoarea atributului *method* al elementul *form*). Se defineste cu ajutorul tag-ului HTML INPUT, folosind atributul *type* cu valoarea "submit". Valoarea atributului *value* al acestui element reprezinta textul afisat pe buton:

```
<form method="post" action="/Teste/test.php">
      Input: <input type="text" name="input">
      <input type="submit" name="expediere" value="Trimite">
</form>
```



5.2.3.2 Accesarea datelor din PHP

Apasarea butonului de submit determina trimiterea catre server a tuturor valorilor curente ale elementelor formularului, sub forma de perechi nume-valoare. Una dintre acestea este si cea formata din valorile atributelor *name* si *value* ale butonului, daca amandoua au fost definite. Daca programul care prelucreaza aceste informatii este un script PHP, informatiile vor fi incluse in tablourile $_GET/$_POST. In cazul exemplului de mai sus, daca utilizatorul nu introduce nimic in campul Input, $_POST va arata astfel:

```
$_POST = array(2) {
           ["input"]=>  string(0) ""
           ["expediere"]=>  string(7) "Trimite"
         }
```

5.2.4 Elemente pentru introducere/editare de text de o singura linie (text input)

5.2.4.1 Definire si atribute

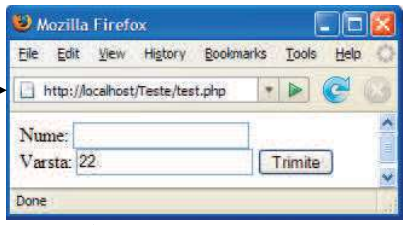
Iata principalele caracteristici ale acestui tip de elemente de formular:

- sunt create folosind tag-ul HTML <input> cu atributul type=text

Studentul poate utiliza prezentul material si informatiile continute in el exclusiv in scopul asimilarii cunostintelor pe care le include, fara a afecta dreptul de proprietate intelectuala detinut de InfoAcademy.

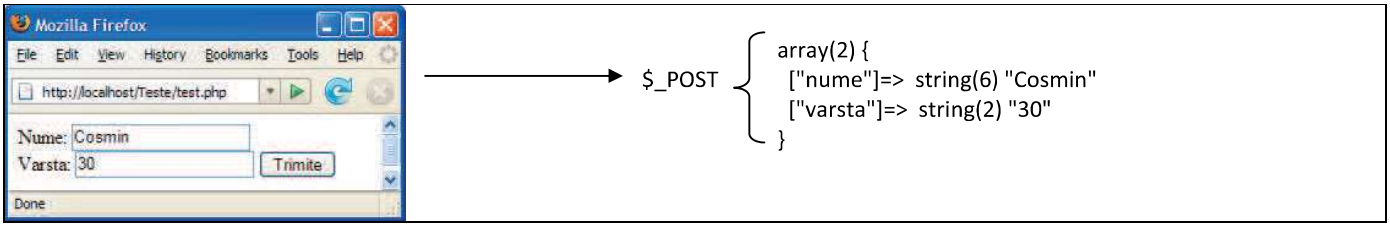
- sunt prezentate utilizatorului sub forma unor campuri de editare de text de o singura linie
- utilizatorul introduce manual valori in aceste campuri
- numele elementului din formular:
  - se defineste cu ajutorul atributului *name* al tag-ului <input>
  - determina cheia elementului corespunzator din tabloul $_POST/$_GET
- valoarea default a elementului (textul deja scris la afisarea campului) se poate specifica cu atributul *value*

```
<form>
Nume:  <input type="text" name="nume"><br />
Varsta: <input type="text" name="varsta" value=22>
<input type="submit" value="Trimite">
</form>
```



5.2.4.2 Accesarea datelor din PHP

Dupa submit, elementul de tip text input determina aparitia unui element in tabloul $_GET sau $_POST a carui cheie este data de valoarea atributului *name* al tag-ului <input> si a carui valoare este cea introdusa de utilizator in campul de editare afisat in browser (sau sirul vid daca acesta nu a introdus nimic).



Atentie! *Chiar daca input-ul de la utilizator este un numar (adica o succesiune de caractere de tip cifra), valoarea corespunzatoare din tabloul $_GET/$_POST are tipul de date string! De aceea functiile is_int(), is_float() vor returna false, solutia eficace fiind functia* ***is_numeric()****.*

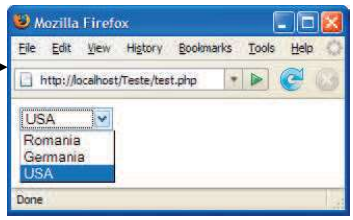
5.2.5 Elemente de tip lista cu selectie unica (drop-down list)

5.2.5.1 Definire si atribute

Un drop-down list se defineste cu ajutorul constructiei select, continutul listei fiind compus dintr-o succesiune de elemente option, ca in exemplul de mai jos:

Studentul poate utiliza prezentul material si informatiile continute in el exclusiv in scopul asimilarii cunostintelor pe care le include, fara a afecta dreptul de proprietate intelectuala detinut de InfoAcademy.

```
<select name="tara">
      <option value="ro">Romania</option>
      <option value="de">Germania</option>
      <option value="us" selected="selected">USA</option>
</select>
```

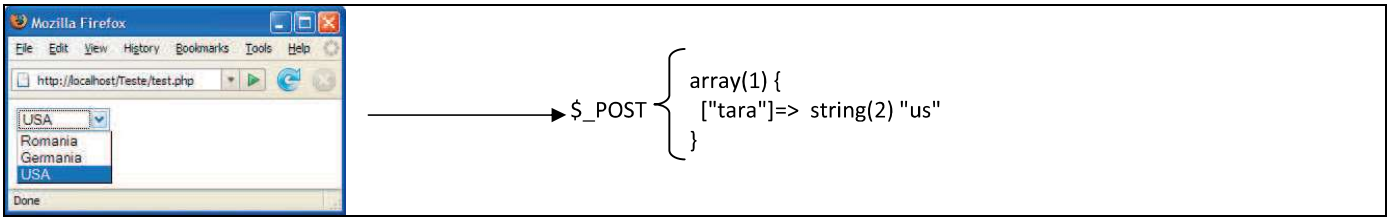


Elementul *option* care are atributul *selected* setat va fi cel selectat din oficiu la afisarea formularului.

5.2.5.2 Accesarea datelor din PHP

Dupa submit, la receptionarea datelor, interpretorul PHP va crea in tabloul $_GET/$_POST un element cu urmatoarele caracteristici:

- cheie – valoarea atributului *name* al tag-ului <select>
- valoare – valoarea atributului *value* al elementului <option> selectat de catre utilizator



Atentie! *Desi felul in care browserul prezinta un element de tip select ne-ar lasa sa credem ca utilizatorul nu poate alege alte valori decat cele specificate in lista, in realitate informatiile transmise serverului pot fi alterate sau generate integral de catre utilizator. De aceea este sanatos ca scriptul PHP care receptioneaza datele sa verifice daca ele fac parte din lista de valori posibile inainte de a le folosi:*

```
$tari = array("ro", "de", "us");
if(!empty($_POST['tara'])){
      if( !in_array($_POST['tara'], $tari) ) echo "Tara invalida!";
      else  // procesarea datelor
}
```

5.2.6 Elemente de tip lista cu selectie multipla

5.2.6.1 Definire si atribute

Un astfel de element de formular permite utilizatorului sa aleaga mai multe valori simultan dintr-o lista predefinita (ex: o lista de hobby-uri, de cursuri etc). Crearea unui astfel de element se face la fel ca in cazul listei cu selectie unica, cu doua exceptii:

- tag-ul <select> trebuie sa aiba atributul *multiple* setat
- atributul *name* al aceluiasi tag trebuie sa se termine cu paranteze drepte, ca in sintaxa folosita pentru crearea unui element intr-un tablou PHP fara specificarea cheii (vezi explicatia in sectiunea urmatoare)

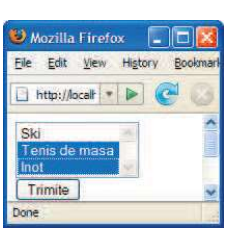
```
<select name="sporturi[]" multiple="multiple">
      <option value="ski" selected="selected">Ski</option>
      <option value="ping-pong">Tenis de masa</option>
      <option value="inot" selected="selected">Inot</option>
</select>
```

Studentul poate utiliza prezentul material si informatiile continute in el exclusiv in scopul asimilarii cunostintelor pe care le include, fara a afecta dreptul de proprietate intelectuala detinut de InfoAcademy.

5.2.6.2 Accesarea datelor din PHP

Daca valoarea atributului *name* al tag-ului <select> este scrisa corect (se termina cu paranteze drepte), efectul in interpretorul PHP va fi crearea unui element in tabloul $_GET/$_POST cu urmatoarele caracteristici:

- cheie – valoarea atributului *name* ale tag-ului <select>
- valoare – un tablou indexat numeric, ale carui valori sunt valorile atributelor *value* ale optiunilor selectate de catre utilizator



$_POST

```
array(1) {
  ["sporturi"]=> array(2) {
                   [0]=> string(9) "ping-pong"
                   [1]=> string(4) "inot"
                 }
}
```

Nota: *daca unul dintre elementele <option> nu are atributul value specificat, atunci valoarea trimisa serverului va fi textul acelei optiuni (sirul de caractere dintre <option…> si </option>)*

5.2.6.3 De ce valoarea atributului name trebuie sa aiba sintaxa de tablou PHP

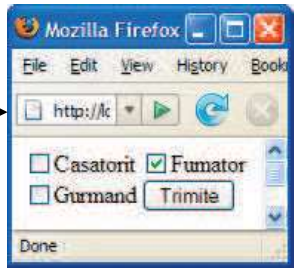
Interpretorul PHP prelucreaza automat datele primite de la server prin cereri GET sau POST; pentru fiecare pereche *nume-valoare* receptionata, el incearca sa creeze un element cu cheia *nume* si valoarea *valoare*. Atunci cand utilizatorul selecteaza mai multe valori intr-o lista cu selectie multipla, in urma submit-ului se obtin mai multe perechi nume-valoare care au acelasi nume – cel din atributul *name* al tag-ului select. Motorul PHP va face pe rand asignarea $_POST['nume'] = valoare, suprascriind de fiecare data valoarea deja existenta cu cea noua pentru perechile cu acelasi nume. Asa se face ca, daca valoarea atributului *name* al tag-ului select nu se termina cu paranteze drepte, chiar daca utilizatorul selecteaza mai multe valori, in $_POST se va crea o singura intrare ce va contine ultima valoare selectata (obtinuta prin suprascrieri succesive).

5.2.7 Elemente de tip checkbox

5.2.7.1 Definire si atribute

Elementele de tip checkbox reprezinta casute cu doua stari - bifat si debifat. Ele sunt folosite in general in formulare pentru a reprezenta optiuni cu doua variante. Se definesc cu ajutorul tag-ului <input>, folosind valoarea "checkbox" pentru atributul *type*. Daca este specificat si atributul "checked", casuta va fi bifata din oficiu:

```
<form method=post action="/Teste/test.php">
<input type="checkbox" name="casatorit" value="da">Casatorit
<input type="checkbox" name="fumator" checked="checked">Fumator
<input type="checkbox" name="gurmand">Gurmand
<input type=submit value="Trimite">
</form>
```



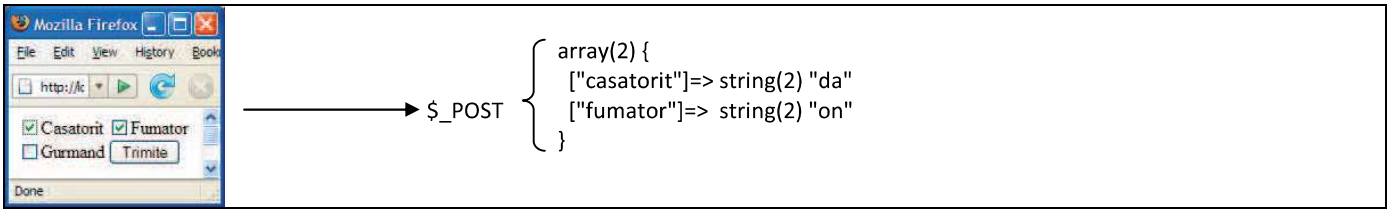
5.2.7.2 Accesarea datelor din PHP

Atunci cand datele unui formular care contine checkbox-uri sunt expediate catre server si sunt prelucrate de un script PHP:

- elementele checkbox bifate de utilizator care au specificate atat atributul *name* cat si *value* vor determina aparitia in tabloul $_GET/$_POST a unui element ce are ca cheie valoarea atributului *name* si ca valoare valoarea atributului *value* al tag-ului <input> corespunzator
- checkbox-urile nebifate, sau care sunt bifate dar nu au atribut name NU vor genera intrari in $_GET/$_POST

Studentul poate utiliza prezentul material si informatiile continute in el exclusiv in scopul asimilarii cunostintelor pe care le include, fara a afecta dreptul de proprietate intelectuala detinut de InfoAcademy.

- checkbox-urile bifate care au atribut name dar nu si value vor trimite catre server valoarea "on"



Se observa mai sus ca checkbox-ul corespunzator optiunii "Casatorit", care are atat atribut name cat si value, a generat $_POST["casatorit"] = "da". Checkbox-ul optiunii "Fumator" a generat valoarea "on", deoarece ii lipsea atributul *value*, iar checkbox-ul "Gurmand" nu a generat nimic deoarece nu a fost bifat.

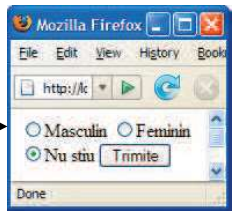
5.2.8 Elemente de tip radio button

5.2.8.1 Definire si atribute

Un radio button este un element cu doua stari (la fel ca checkbox-ul), insa cu urmatoarea particularitate: putem crea grupuri de elemente de acest fel, impunand ca numai unul dintre ele sa fie bifat la un moment dat. Selectarea altui buton decat cel curent selectat duce la deselectarea automata a celui curent.

Definirea se face cu ajutorul tag-ului <input>, folosind atributul *type* cu valoarea "radio". Se poate preciza butonul din grup care sa fie bifat din oficiu, folosind atributul *checked* al tag-ului <input>:

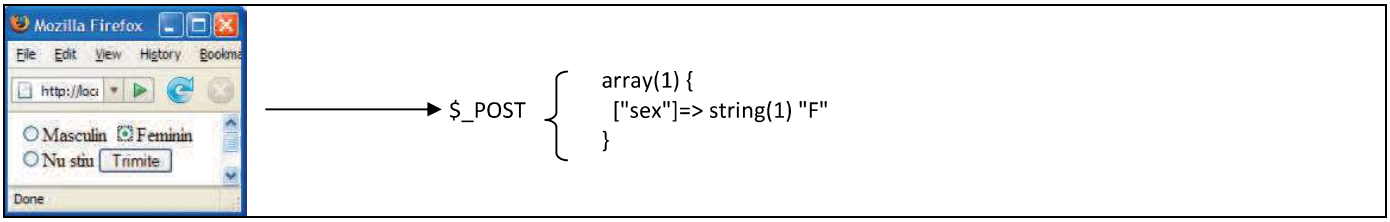
```
<form method=post>
<input type="radio" name="sex" value="M">Masculin
<input type="radio" name="sex" value="F">Feminin
<input type="radio" name="sex" value="N" checked="checked">Nu stiu
<input type=submit value="Trimite">
</form>
```



Atentie! *Pentru ca butoanele sa faca parte din acelasi grup (si sa fie deselectat cel curent la apasarea altuia) toate elementele de tip radio ale grupului trebuie sa aiba aceeasi valoare a atributului name.*

5.2.8.2 Accesarea datelor din PHP

Ca mod de accesare a datelor din scriptul PHP ce primeste datele formularului, radio button-urile se manifesta la fel ca checkbox-urile: sunt create in tabloul $_GET/$_POST elemente cu perechile name-value, iar daca atributul *value* al butonului selectat lipseste, valoarea va fi automat "on".



Nota: *daca nu folosim aceeasi valoare a atributului name pentru butoanele dintr-un astfel de grup:*

- *butoanele vor putea fi selectate independent*

Studentul poate utiliza prezentul material si informatiile continute in el exclusiv in scopul asimilarii cunostintelor pe care le include, fara a afecta dreptul de proprietate intelectuala detinut de InfoAcademy.

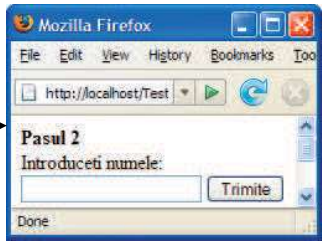
- *$_GET/$_POST va contine perechi diferite nume-valoare pentru fiecare nume distinct corespunzator unui radio button selectat*

5.2.9 Elemente ascunse (hidden field)

5.2.9.1 Definire si atribute

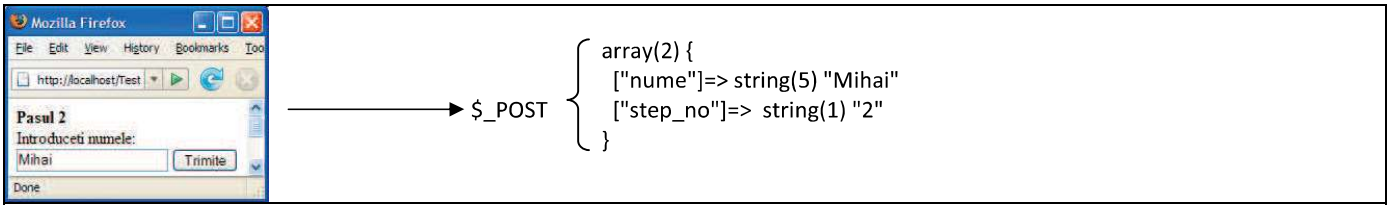
Reprezinta elemente de formular HTML invizibile – ele nu sunt afisate in browser ca parte a formularului, dar determina trimiterea unei perechi nume-valoare catre server la expedierea formularului. Sunt folosite in general pentru a propaga informatie de la o pagina la alta fara a o include in URL, in mod transparent pentru utilizator (de exemplu, in cazul in care utilizatorul completeaza un formular impartit in mai multe pagini sau mai multi pasi). Se definesc cu ajutorul tag-ului <input> folosind valoarea "hidden" pentru atributul *type*:

```
<form method="POST">
<strong>Pasul 2</strong><br/>
Introduceti numele:
<input type="text" name="nume">
<input type="hidden" name="step_no" value="2">
<input type=submit value="Trimite">
</form>
```



5.2.9.2 Accesarea datelor din PHP

Elementul de formular de tip hidden genereaza in $_GET/$_POST propria pereche nume-valoare, constand din valorile atributelor *name* si *value* ale tag-ului <input> corespunzator:

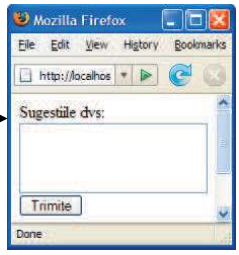


5.2.10 Elemente pentru introducere/editare de text pe mai multe linii (text area)

5.2.10.1 Definire si atribute

Un text area este un element al formularului in care utilizatorul poate introduce si edita text pe mai multe randuri. Este folosit in general atunci cand utilizatorul trebuie sa introduca cantitati mai mari de text (care eventual contin caractere newline), caz in care un element input de tip text nu mai este potrivit. Se defineste folosind tag-ul <textarea>, care dispune de atribute ce specifica dimensiunea elementului (numarul de linii si de coloane afisate):

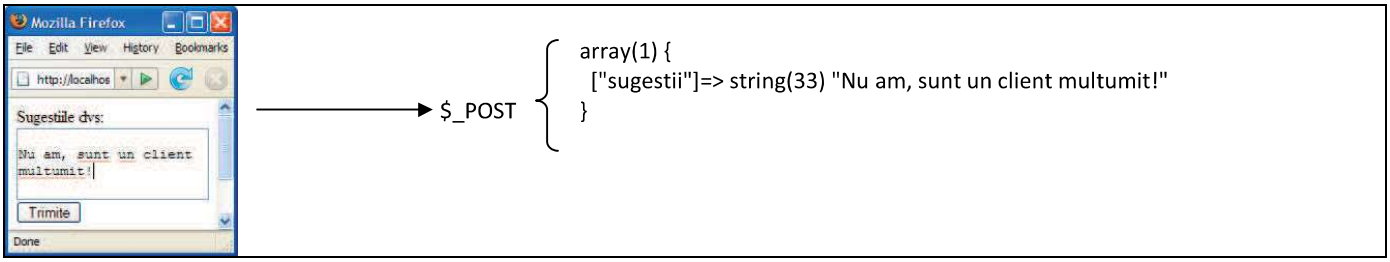
```
<form method="post" action="/Teste/test.php">
      Sugestiile dvs: <br/>
      <textarea name="sugestii" rows="3" cols="20">
      <input type="submit" value="Trimite">
</form>
```



5.2.10.2 Accesarea datelor din PHP

La submit, un textarea are ca efect crearea unui element in tabloul $_GET/$_POST, perechea sa cheie-valoare fiind formata din valoarea atributului *name* al tag-ului <textarea> si informatia introdusa de utilizator:

Studentul poate utiliza prezentul material si informatiile continute in el exclusiv in scopul asimilarii cunostintelor pe care le include, fara a afecta dreptul de proprietate intelectuala detinut de InfoAcademy.



5.2.11 Alte elemente de formular HTML

Mai exista cateva elemente de formular HTML care nu au fost acoperite aici:

- butoanele de reset – au rol de aducere a elementelor formularului la valorile default
- butoanele cu alt rol decat submit si reset
- elementul <input> de tip *file*, folosit pentru a oferi utilizatorului posibilitatea de a uploada fisiere, si care este tratat la lectia despre lucrul cu fisiere

5.3 BIBLIOGRAFIE

- **O'Reilly – Learning PHP and MySQL 2nd Edition – Chapter 10: Working with forms – pag 199-215**
- **PHP5 and MySQL Bible – Chapter 7: Passing Information between Pages – pag 119-136**
- **PHP Manual - Variables from outside PHP: http://www.php.net/variables.external**
- **PHP Manual - PHP and HTML FAQ: http://www.php.net/manual/en/faq.html.php**
- **HTML Forms - http://www.w3.org/TR/html401/interact/forms.html**
- **Specificatia protocolul HTTP: http://tools.ietf.org/html/rfc2616**
- **Validarea input-ului din formulare HTML:** http://hudzilla.org/phpwiki/index.php?title=Validation_reminder_sheet

Studentul poate utiliza prezentul material si informatiile continute in el exclusiv in scopul asimilarii cunostintelor pe care le include, fara a afecta dreptul de proprietate intelectuala detinut de InfoAcademy.