

Poznaj zasady wirtualnego handlu i zarabiaj prawdziwe pieniądze

Jak stworzyć doskonałą witrynę sklepu internetowego?
Jak zapewnić maksymalne bezpieczeństwo Twojej strony?
Jak prowadzić sprzedaż i zarządzać stanem magazynowym?

E-commerce

Genialnie proste
tworzenie serwisów

w PHP i MySQL

Larry Ullman

New
Riders



» Idź do

- Spis treści
- Przykładowy rozdział
- Skorowidz

» Katalog książek

- Katalog online
- Zamów drukowany katalog

» Twój koszyk

- Dodaj do koszyka

» Cennik i informacje

- Zamów informacje o nowościach
- Zamów cennik

» Czytelnia

- Fragmenty książek online

» Kontakt

Helion SA
ul. Kościuszki 1c
44-100 Gliwice
tel. 32 230 98 63
e-mail: helion@helion.pl
© Helion 1991–2011

E-commerce. Genialnie proste tworzenie serwisów w PHP i MySQL

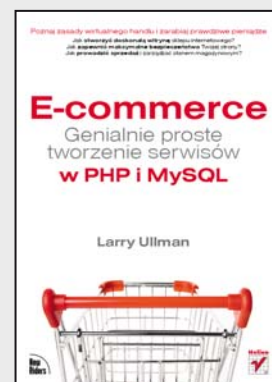
Autor: [Larry Ullman](#)

Tłumaczenie: Aleksander Lamża

ISBN: 978-83-246-3213-8

Tytuł oryginału: [Effortless E-Commerce with PHP and MySQL](#)

Format: 168×237, stron: 400



Poznaj zasady wirtualnego handlu i zarabiaj prawdziwe pieniądze

- Jak stworzyć doskonałą witrynę sklepu internetowego?
- Jak zapewnić maksymalne bezpieczeństwo Twojej strony?
- Jak przeprowadzać sprzedaż i zarządzać stanem magazynowym?

Handel elektroniczny to znakomity sposób prowadzenia działalności zarobkowej. Twoje przedsięwzięcie może odnieść prawdziwy sukces – pod warunkiem, że się do niego dobrze przygotujesz. Oto jedna z nielicznych książek na rynku dostarczających szczegółowych informacji na temat tworzenia serwisów e-commerce z wykorzystaniem PHP i MySQL. Bez względu na to, czy stworzysz dynamiczne strony internetowe od lat, czy dopiero od kilku tygodni, z pewnością znajdziesz tu mnóstwo bezcennych informacji.

Dzięki książce E-commerce. Genialnie proste tworzenie serwisów w PHP i MySQL dowiesz się, jak zaprojektować bazę danych, generować katalog produktów, zarządzać koszykiem zakupów, obsługiwać zamówienia i płatności oraz sprawić, by Twój sklep internetowy nie wymagał od Ciebie pracy ponad siły, a jednak był dochodowy. Podręcznik zawiera również omówienie tak istotnych zagadnień, jak zapewnienie bezpieczeństwa witryny oraz zadbanie o interfejs przyjazny użytkownikom, a także informacje dotyczące modułowego programowania, gotowego do dalszej rozbudowy. Przytoczone tu praktyczne przykłady pozwolą Ci spojrzeć na systemy e-commerce z możliwie jak najszerzej perspektywy.

- Wybór technologii internetowych
- Struktura i projekt witryny
- Zarządzanie zawartością witryny
- Tworzenie kont użytkowników
- Łączenie różnych systemów płatniczych
- Sprzedaż wirtualnych produktów
- Tworzenie bezpiecznego środowiska serwera i baz danych
- Tworzenie paneli administracyjnych
- Zasady składania zamówień

Magia tworzenia profesjonalnych serwisów e-commerce

SPIS TREŚCI

Wprowadzenie	13
Czym jest handel elektroniczny?	13
Podstawowe informacje o książce	14
Wykorzystane technologie	15
Gdzie szukać pomocy?	15
Wymagania	16
Podstawowe umiejętności	16
Serwer internetowy	16
Jeszcze kilka drobiazgów	16
CZĘŚĆ I: PODSTAWY	17
Rozdział 1. Od czego zacząć?	19
Określanie celów biznesowych	20
Analiza kwestii prawnych	21
Prawo państwowe i międzynarodowe	21
Zgodność z PCI	23
Wybór technologii internetowych	23
Wybór hostingu	26
Możliwości hostingu	26
Hosting, który polecam	29
Jak znaleźć dobrego usługodawcę?	30
Korzystanie z systemu płatności	31
Systemy przetwarzania płatności	31
Bramki płatności	32
Który system płatności wybrać?	33

Proces tworzenia witryny	34
Planowanie witryny	35
Projekt HTML	35
Projekt bazy danych	36
Programowanie	38
Testowanie	40
Uruchamianie	42
Utrzymywanie	42
Udoskonalanie	43
Rozdział 2. Podstawy bezpieczeństwa	45
Teoria bezpieczeństwa	45
Żadna witryna nie jest bezpieczna	46
Celem nigdy nie jest zapewnienie maksymalnego bezpieczeństwa	47
Bezpieczeństwo użytkowników	48
Wymagania PCI	50
Bezpieczeństwo serwera	52
Wpływ hostingu na bezpieczeństwo	53
Bezpieczeństwo PHP i serwera WWW	54
Bezpieczeństwo bazy danych	57
Bezpieczne transakcje	59
Typowe słabe punkty	63
Ochrona informacji	63
Ochrona użytkownika	64
Ochrona witryny	65
CZĘŚĆ II: SPRZEDAŻ WIRTUALNYCH PRODUKTÓW	71
Rozdział 3. Pierwsza witryna — struktura i projekt	73
Projekt bazy danych	74
Organizacja plików na serwerze	77
Łączenie się z bazą danych	81
Plik konfiguracyjny	83

Szablon HTML	88
Tworzenie pliku nagłówka	89
Dodawanie dynamicznych funkcjonalności do nagłówka	90
Tworzenie pliku stopki	93
Dodawanie dynamicznych funkcjonalności do stopki	94
Tworzenie strony głównej	96
Rozdział 4. Konta użytkowników	99
Definiowanie funkcji pomocniczych	99
Tworzenie pól formularza	100
Ochrona haseł	104
Przekierowanie przeglądarki	106
Zakładanie kont	108
Tworzenie podstawowej struktury skryptu	108
Tworzenie formularza	110
Przetwarzanie danych z formularza	111
Logowanie	118
Przetwarzanie danych z formularza	118
Tworzenie formularza	120
Wylogowanie	122
Zarządzanie hasłami	123
Odzyskiwanie hasła	123
Zmiana hasła	127
Zwiększenie poziomu bezpieczeństwa	130
Rozdział 5. Zarządzanie zawartością witryny	133
Tworzenie konta administratora	133
Dodawanie stron	134
Tworzenie podstawowego skryptu	134
Dodawanie edytora WYSIWYG	139
Wyświetlanie zawartości strony	143
Przygotowanie skryptu category.php	143
Przygotowanie skryptu page.php	147
Dodawanie plików PDF	149
Konfigurowanie serwera	150
Tworzenie skryptu PHP	152

Wyświetlanie plików PDF	159
Przygotowanie skryptu pdfs.php	159
Przygotowanie skryptu view_pdf.php	160
Rozdział 6. Korzystanie z systemu PayPal	165
Ogólne informacje o systemie PayPal	165
Obsługa płatności w systemie PayPal	167
Przyciski systemu płatności	168
Testowanie systemu PayPal	169
Rejestrowanie w usłudze Sandbox	170
Tworzenie kont testowych	171
Tworzenie przycisku	174
Integracja systemu PayPal z witryną	177
Aktualizacja skryptu rejestrowania	177
Tworzenie skryptu thanks.php	178
Tworzenie skryptu cancel.php	180
Testowanie witryny	181
Korzystanie z mechanizmu IPN	183
Aktywowanie mechanizmu IPN	184
Aktualizacja skryptu rejestrowania	184
Tworzenie skryptu ipn.php	185
Aktualizacja skryptu thanks.php	191
Odnawianie kont	191
Uruchamianie witryny	192

CZĘŚĆ III: SPRZEDAŻ RZECZYWISTYCH PRODUKTÓW ... 195

Rozdział 7. Druga witryna — struktura i projekt	197
Kilka słów o witrynie	197
Co będziemy sprzedawać?	197
Zakupy bez rejestracji	199
Implementowanie architektury MVC	199
Zwiększenie bezpieczeństwa	201

Projekt bazy danych	201
Tabele produktów	202
Tabele klientów	203
Podstawowy kod SQL	205
Konfiguracja serwera	208
Organizacja plików na serwerze	208
Dostosowanie serwera	209
Pliki pomocnicze	216
Łączenie z bazą danych	216
Plik konfiguracyjny	217
Szablon HTML witryny	218
Nowe możliwości bazy MySQL	221
Predefiniowane zapytania	222
Procedury składowane	225
Rozdział 8. Tworzenie katalogu	229
Przygotowanie bazy danych	229
Wypełnianie tabel za pomocą SQL-a	230
Rzut oka na kwerendy procedur składowanych	233
Tworzenie procedur składowanych	239
Dokonywanie zakupów z poziomu kategorii	243
Tworzenie skryptu shop.php	243
Tworzenie plików widoku	245
Wyświetlanie listy produktów	249
Tworzenie skryptu browse.php	249
Tworzenie plików widoku	251
Tworzenie widoku dla braku produktów	255
Informowanie o dostępności	256
Wyświetlanie promocyjnych cen	258
Uaktualnianie procedury składowanej	259
Aktualizowanie skryptu product_functions.inc.php	261
Aktualizowanie pliku list_products.html	262
Aktualizowanie pliku list_coffees.html	263
Wyróżnianie promocji	263
Tworzenie strony głównej	263
Tworzenie stron służących do sprzedaży	266

Rozdział 9. Tworzenie koszyka na zakupy	269
Definiowanie procedur składowanych	269
Dodawanie produktów	270
Usuwanie produktów	271
Aktualizowanie koszyka	271
Pobieranie zawartości koszyka	273
Definiowanie funkcji pomocniczych	274
Budowanie koszyka na zakupy	275
Tworzenie skryptu PHP	275
Tworzenie plików widoku	279
Budowanie przechowalni	283
Tworzenie skryptu PHP	283
Tworzenie plików widoku	285
Obliczanie kosztów wysyłki	287
Rozdział 10. Składanie zamówienia	289
Authorize.net	289
Tworzenie konta próbnego	291
Przygotowanie witryny	293
Nowy szablon HTML	293
Funkcja pomocnicza	295
Tworzenie procedur	298
Pobieranie danych do wysyłki	306
Tworzenie skryptu PHP	307
Tworzenie plików widoku	314
Pobieranie danych posiadacza rachunku	321
Tworzenie podstawowego skryptu PHP	322
Tworzenie pliku widoku	323
Sprawdzanie danych z formularza	327
Obsługa karty kredytowej	332
Tworzenie pliku gateway_setup.php	333
Tworzenie pliku gateway_process.php	334
Analiza odpowiedzi serwera	337
Uaktualnienie pliku billing.php	338

Finalizacja zamówienia	341
Tworzenie skryptu PHP	341
Tworzenie pliku widoku	343
Testowanie strony	344
Uruchomienie strony	345
Rozdział 11. Administrowanie witryną	347
Konfigurowanie serwera	348
Uwierzytelnianie dostępu	348
Tworzenie szablonu	348
Korzystanie z menu Superfish	352
Aktualizowanie funkcji create_form_input()	354
Dodawanie produktów	355
Dodawanie innych produktów	355
Dodawanie kawy	363
Zmiana stanu magazynowego	367
Definiowanie promocji	371
Przeglądanie zamówień	376
Wyświetlanie wszystkich zamówień	377
Przeglądanie jednego zamówienia	379
Dostarczanie zamówień	384
Tworzenie skryptu gateway_setup_admin.php	384
Aktualizowanie skryptu view_order.php	384
Skorowidz	389

9 | TWORZENIE KOSZYKA NA ZAKUPY

W poprzednim rozdziale został opracowany katalog produktów zawierający przyciski służące do dodawania wybranych produktów do koszyka na zakupy. Teraz przyszedł czas na zaimplementowanie koszyka. Dobrze zaprojektowany koszyk pozwala klientom wyświetlić wszystkie wybrane przez nich produkty, w łatwy sposób usunąć wskazane pozycje lub zmienić ilość, a także przejść do etapu realizacji zamówienia. Projektowana witryna ma posiadać również funkcję „przechowalni”, do której klient może dodać produkty, ale nie musi ich od razu zamawiać.

Główna uwaga w tym rozdziale skupi się na zaimplementowaniu koszyka i przechowalni. Musisz jednak zacząć od zdefiniowania nowych procedur składowanych. Na końcu rozdziału są omówione zagadnienia związane z różnymi możliwościami uwzględnienia kosztów wysyłki oraz inne dodatkowe funkcje, które można zaimplementować.

DEFINIOWANIE PROCEDUR SKŁADOWANYCH

Ponieważ w projekcie witryny został zastosowany wzorzec MVC, pracę nad koszykiem rozpoczniemy od zaprojektowania **modelu**, czyli bazy danych. W tym rozdziale zostanie opisanych osiem nowych procedur składowanych: cztery dla koszyka na zakupy i cztery dla przechowalni. Tylko dwie z nich są skomplikowane porównywalnie do tych opisanych w rozdziale 8., „Tworzenie katalogu”, jednak tym razem w procedurach znajdzie się więcej tzw. logiki. Ponieważ tabele `carts` i `wish_lists` mają taką samą strukturę i będą wykorzystywane w ten sam sposób, opiszemy tylko procedury dla koszyka. Aby utworzyć procedury dla przechowalni, wystarczy zmienić nazwę tabeli `cart` na `wish_list`.

Dodawanie produktów

Wskazówka

Sposób tworzenia procedur składowanych w bazie danych jest opisany w rozdziale 8.

Procedura składowana `add_to_cart()` będzie wywoływana po żądaniu zgłoszonym przez klienta wkładającego produkt do koszyka. Procedurze trzeba przekazać cztery argumenty: unikalny identyfikator użytkownika, typ produktu (*kawa* lub *inne*) i jego identyfikator, a także ilość. Dodatkowo, jeśli w koszyku nie ma jeszcze danego produktu, procedura powinna go **dodać**, natomiast jeśli już jest, powinna tylko **zwiększyć jego ilość**. Poniżej znajduje się kod procedury i jego objaśnienie:

```
DELIMITER $$
CREATE PROCEDURE add_to_cart (uid CHAR(32), type VARCHAR(6), pid MEDIUMINT,
↳qty TINYINT)
BEGIN
    DECLARE cid INT;
    SELECT id INTO cid FROM carts WHERE user_session_id=uid AND product_type=type
↳AND product_id=pid;
    IF cid > 0 THEN
        UPDATE carts SET quantity=quantity+qty, date_modified=NOW( ) WHERE id=cid;
    ELSE
        INSERT INTO carts (user_session_id, product_type, product_id, quantity)
↳VALUES (uid, type, pid, qty);
    END IF;
END$$
DELIMITER ;
```

Wskazówka

Pełny kod SQL i pozostałe pliki możesz pobrać ze strony www.DMCInsights.com/ecom/.

Najpierw, tak jak to już robiliśmy, zmieniamy symbol zakończenia definicji na dwa znaki dolara (\$\$). Następnie jest definiowana **sygnatura** procedury, czyli jej nazwa i lista argumentów. Pierwszy argument (`uid`) to identyfikator sesji użytkownika, który będzie się składał z dokładnie 32 znaków. Kolejny (`type`) to typ produktu, który może mieć długość do sześciu znaków (w naszym przypadku nazwy obu typów mają po cztery znaki: *kawa* i *inne*). Dwa ostatnie argumenty to identyfikator produktu (`pid`) oraz ilość (`qty`).

Wskazówka

Symbol zakończenia definicji procedury musi być zmieniony na coś innego niż znak średnika.

Aby w procedurze sprawdzić, czy dany produkt znajduje się już w koszyku, konieczne jest wprowadzenie dodatkowej, wewnętrznej zmiennej. Można ją zadeklarować za pomocą instrukcji `DECLARE`, po której

następuje nazwa zmiennej i typ danych MySQL. W kodzie jest tworzona zmienna `cid` (skrót od *cart ID*, czyli identyfikator koszyka).

Uwaga

Zmienne muszą być deklarowane bezpośrednio pod instrukcją `BEGIN`.

Kwerenda `SELECT` poszukuje w tabeli `carts` określonego produktu. Jeśli go znajdzie, zapisuje jego identyfikator (`id`) do zmiennej `cid`, za co odpowiada konstrukcja `SELECT . . . INTO`.

Po kwerendzie `SELECT` znajduje się instrukcja warunkowa `IF-ELSE`, która — w zależności od wartości zmiennej `cid` — wykonuje zapytanie `UPDATE` lub `INSERT`. Wartość `cid` większa od zera oznacza, że produkt już jest w koszyku, więc trzeba jedynie zaktualizować ilość. W przeciwnym przypadku do tabeli jest wstawiany nowy rekord. Kolumna `date_modified` jest uaktualniana tylko w przypadku wykonania zapytania `UPDATE`.

Usuwanie produktów

Wskazówka

Argumenty przekazywane do procedur składowanych mogą być wykorzystywane w kwerendach, tak samo jak w przypadku predefiniowanych zapytań, co oznacza, że łańcuchy tekstowe nie mogą być umieszczane w cudzysłowach.

Procedura składowana usuwająca produkt z koszyka jest najprostszą z wszystkich czterech opisanych w tym rozdziale. Wymaga przekazania trzech z czterech argumentów zdefiniowanych w procedurze `add_to_cart()` (z oczywistych względów podczas usuwania produktu nie jest wymagane podanie jego ilości). Procedura wykonuje tylko jedną kwerendę `DELETE`:

```
DELIMITER $$
CREATE PROCEDURE remove_from_cart (uid CHAR(32), type VARCHAR(6), pid MEDIUMINT)
BEGIN
    DELETE FROM carts WHERE user_session_id=uid AND product_type=type AND
        ↳product_id=pid;
END$$
DELIMITER ;
```

Aktualizowanie koszyka

Wskazówka

Są dwie możliwości usunięcia produktu z koszyka: kliknięcie linku usuwającego lub wpisanie zera w polu ilości.

Procedura składowana aktualizująca koszyk będzie wywoływana po kliknięciu przez użytkownika przycisku *Aktualizuj koszyk* znajdującego się na stronie koszyka (rysunek 9.1). Procedura przyjmuje te same cztery parametry co `add_to_cart()`, ale nie musi sprawdzać, czy produkt już istnieje w tabeli (ponieważ produkt musi znajdować się w bazie, by był wyświetlony w koszyku). Jeśli jednak użytkownik w polu ilości wpisze zero, procedura powinna usunąć taki produkt. Zamiast umieszczać w tej procedurze kod usuwający produkt, wywołamy procedurę `remove_from_cart()` realizującą to zadanie.

Twój koszyk

Użyj tego formularza do zmiany zawartości koszyka. Możesz zmienić ilość, całkowicie usunąć produkty z koszyka albo przenieść je do przechowalni, by zamówić je później. Koszt dostawy i obsługi zamówienia jest podany nad wartością koszyka. Jeśli chcesz złożyć zamówienie, kliknij przycisk Złóż zamówienie. Zostaniesz przeniesiony do bezpiecznej strony, na której sfinalizujesz zakupy.

Produkt	Ilość	Cena	Wartość	Opcje
Kubki::"Czerwony smok"	<input type="text" value="3"/>	21.00 zł	63.00 zł	Przenieś do przechowalni Usuń z koszyka
Kona::1 kg - bezkof. - ziarnista	<input type="text" value="2"/>	52.50 zł	105.00 zł	Przenieś do przechowalni Usuń z koszyka
Transport i obsługa			28.20 zł	
Suma			196.20 zł	

Aktualizuj koszyk

Złóż zamówienie

Rysunek 9.1

```
DELIMITER $$
CREATE PROCEDURE update_cart (uid CHAR(32), type VARCHAR(6), pid MEDIUMINT,
↪qty TINYINT)
BEGIN
  IF qty > 0 THEN
    UPDATE carts SET quantity=qty, date_modified=NOW( ) WHERE user_session_id=uid
    ↪AND product_type=type AND product_id=pid;
  ELSEIF qty = 0 THEN
    CALL remove_from_cart (uid, type, pid);
  END IF;
END$$
DELIMITER ;
```

Wskazówka

Za każdym razem, gdy w tabelach `carts` lub `wish_lists` zmieni się ilość jakiegoś produktu (ale nie zostanie on usunięty), zaktualizowana zostanie również data modyfikacji (`date_modified`).

Pobieranie zawartości koszyka

Czwarta i ostatnia procedura składowana (dla koszyka) wykonuje kwerendę `SELECT` pobierającą całą zawartość koszyka. Stopień skomplikowania kwerendy zależy od tego, jak dużo informacji chcesz uzyskać, jednak jej podstawowa budowa przypomina kwerendy związane z promocjami, opisane w rozdziale 8. Zapytanie łączy dwie kwerendy `SELECT` za pomocą instrukcji `UNION`: pierwsza to złączenie `JOIN` czterech tabel, a druga — aż pięciu. W poniższym kodzie kwerenda jest przedstawiona w czytelny sposób z podziałem na wiersze. Procedura wymaga podania jedynie jednego argumentu, którym jest identyfikator sesji użytkownika.

```
DELIMITER $$
CREATE PROCEDURE get_shopping_cart_contents (uid CHAR(32))
BEGIN
    SELECT CONCAT("I", ncp.id) AS sku, c.quantity, ncc.category,
    ncp.name, ncp.price, ncp.stock, sales.price AS sale_price
    FROM carts AS c
    INNER JOIN non_coffee_products AS ncp ON c.product_id=ncp.id
    INNER JOIN non_coffee_categories AS ncc ON ncc.id=ncp.non_coffee_category_id
    LEFT OUTER JOIN sales ON
    (sales.product_id=ncp.id AND sales.product_type='inne' AND
    ((NOW( ) BETWEEN sales.start_date AND sales.end_date) OR (NOW( ) > sales.start_date
    ↪AND sales.end_date IS NULL)) )
    WHERE c.product_type="inne" AND c.user_session_id=uid
    UNION
    SELECT CONCAT("K", sc.id), c.quantity, gc.category,
    CONCAT_WS(" - ", s.size, sc.caf_decaf, sc.ground_whole), sc.price, sc.stock,
    ↪sales.price
    FROM carts AS c
    INNER JOIN specific_coffees AS sc ON c.product_id=sc.id
    INNER JOIN sizes AS s ON s.id=sc.size_id
    INNER JOIN general_coffees AS gc ON gc.id=sc.general_coffee_id
    LEFT OUTER JOIN sales ON
    (sales.product_id=sc.id AND sales.product_type='kawa' AND
    ((NOW( ) BETWEEN sales.start_date AND sales.end_date) OR (NOW( ) > sales.start_date
    ↪AND sales.end_date IS NULL)) )
    WHERE c.product_type="kawa" AND c.user_session_id=uid;
END$$
DELIMITER ;
```

Na rysunku 9.2 widać efekt wywołania procedury. Zwróć uwagę, że kwerenda zwraca domyślną cenę (w kolumnie `price`), stan magazynowy (`stock`) i cenę promocyjną (`sale_price`), jeżeli obowiązuje dla danego produktu. Te informacje będą wykorzystane później.

```
mysql> CALL get_shopping_cart_contents('53f316a881cb9e0f47d144e9c46c8733');
+-----+-----+-----+-----+-----+-----+-----+
| sku | quantity | category | name | price | stock | sale_price |
+-----+-----+-----+-----+-----+-----+-----+
| I2 | 3 | kubki | "czerwony smok" | 23.99 | 4 | 21.00 |
+-----+-----+-----+-----+-----+-----+-----+
| K9 | 1 | kona | 1 kg - bezkof. - ziarnista | 52.50 | 15 | NULL |
+-----+-----+-----+-----+-----+-----+-----+
2 rows in set (0,01 sec)

query OK, 0 rows affected (0,01 sec)

mysql>
```

Rysunek 9.2

DEFINIOWANIE FUNKCJI POMOCNICZYCH

Zanim przejdziemy do tworzenia koszyka na zakupy, przygotujemy dwie funkcje pomocnicze, które będą potrzebne w skrypcie koszyka. Pierwsza funkcja przyjmuje dwie ceny — normalną oraz promocyjną — i zwraca tę, która powinna być uwzględniona w koszyku:

```
function get_just_price($regular, $sales) {
    if ((0 < $sales) && ($sales < $regular)) {
        return number_format($sales, 2);
    } else {
        return number_format($regular, 2);
    }
}
```

Podobny kod znajduje się w funkcji `get_price()` zdefiniowanej w rozdziale 8., ale tam jest zwracana wartość liczbowa ceny (bez żadnego kontekstu).

Druga funkcja przetwarza numer SKU na dwie wartości, np. K12 zostanie zamienione na kawa i 12. Ta operacja będzie często potrzebna, ponieważ w bazie danych każdy produkt jest opisany typem i identyfikatorem. Funkcja ta przyjmuje jeden argument — numer SKU — i zwraca dwuelementową tablicę:

```
function parse_sku($sku) {

    // Wydziel pierwszy znak:
    $type_abbr = substr($sku, 0, 1);

    // Wydziel pozostałe znaki:
    $pid = substr($sku, 1);

    // Sprawdź typ:
    if ($type_abbr == 'K') {
        $sp_type = 'kawa';
    } elseif ($type_abbr == 'I') {
        $sp_type = 'inne';
    }
}
```

```

} else {
    $sp_type = NULL;
}

// Sprawdź identyfikator produktu:
$pid = (filter_var($pid, FILTER_VALIDATE_INT, array('min_range' => 1))) ?
↳$pid : NULL;

// Zwróć wartości:
return array($sp_type, $pid);

} // Koniec funkcji parse_sku().

```

Numer SKU jest dzielony na części za pomocą dwóch funkcji `substr()`. Następnie na podstawie pierwszego znaku numeru SKU (który może przyjąć dwie wartości: K dla kawy i I dla innych produktów) jest ustalany typ (zmienna `$sp_type`). Jeżeli nie jest spełniony żaden warunek, zmiennej `$sp_type` jest przypisywana wartość `NULL`.

Druga część numeru SKU musi być liczbą całkowitą co najmniej równą 1. Do sprawdzenia jej poprawności służy funkcja `filter_var()`. Jeżeli liczba jest prawidłowa, jest przypisywana zmiennej `$pid`, a w przeciwnym przypadku przyjmuje wartość `NULL`.

Na koniec obie wartości są zwracane jako tablica, więc w kodzie wywołującym tę funkcję trzeba zastosować instrukcję `list()`:

```
list($type, $id) = parse_sku($sku);
```

Funkcja `list()` przypisuje zmiennym `$type` i `$id` wartości z tablicy zwróconej przez wyrażenie z prawej strony przypisania, czyli funkcję `parse_sku()`.

Najbardziej odpowiednim miejscem dla tych dwóch funkcji jest skrypt `product_functions.inc.php` znajdujący się w katalogu `includes`. Przed przejściem do kolejnego etapu koniecznie umieść tam te funkcje.

BUDOWANIE KOSZYKA NA ZAKUPY

Po zdefiniowaniu procedur składowanych i funkcji pomocniczych musisz się zająć utworzeniem trzech plików:

- `cart.php`, który wykonuje wszystkie niezbędne operacje (jest kontrolerem),
- `cart.html`, który jest plikiem widoku dla koszyka,
- `emptycart.html`, który jest plikiem widoku dla pustego koszyka.

Utworzysz teraz te pliki, rozpoczynając od skryptu PHP.

Tworzenie skryptu PHP

Skrypty PHP opracowane w rozdziale 8., dzięki zastosowaniu procedur składowanych i dołączanych plików HTML, były bardzo krótkie i stosunkowo „czyste”. Cały plik `cart.php` składa się tylko z 50 wierszy kodu,

włączając w to komentarze i puste wiersze! Większość kodu odpowiada za uruchomienie odpowiednich procedur składanych w zależności od sposobu wywołania skryptu.

1. W edytorze lub IDE utwórz nowy skrypt PHP i zapisz go w głównym katalogu serwera WWW pod nazwą `cart.php`.

2. Dołącz plik konfiguracyjny:

```
<?php
require ('./includes/config.inc.php');
```

3. Sprawdź, czy jest już ustawiony identyfikator sesji użytkownika, a jeżeli nie — utwórz go.

```
if (isset($_COOKIE['SESSION'] )) {
    $uid = $_COOKIE['SESSION'];
} else {
    $uid = md5(uniqid('biped',true));
}
```

Ta witryna korzysta tylko z jednego ciasteczka do obsługi zarówno koszyka, jak i przechowania. Ciasteczko ma nazwę `SESSION`. Chociaż nie są tu stosowane prawdziwe sesje PHP, nazwa wskazuje przeznaczenie ciasteczka.

Jeśli ciasteczko istnieje, jego wartość jest przypisywana zmiennej `$uid`. W przeciwnym przypadku musi zostać utworzony nowy identyfikator sesji. Służy do tego kombinacja funkcji `uniqid()` i `md5()`.

Wskazówka

Funkcja `md5()` zwraca łańcuch tekstowy o długości 32 znaków.

4. Wyślij ciasteczko:

```
setcookie('SESSION', $uid, time()+(60*60*24*30));
```

Gdy użytkownik powraca na stronę albo pojawia się na niej pierwszy raz, ciasteczko musi zostać przesłane. Dla nowych użytkowników jest to oczywiste, natomiast w przypadku powracających klientów chodzi o zaktualizowanie ciastka, tak by było dłużej ważne. Przy przedstawionych ustawieniach termin ważności ciasteczka mija po 30 dniach od daty utworzenia.

Uwaga

Ciasteczka muszą zostać przesłane do przeglądarki internetowej jako pierwsze.

5. Dołącz plik nagłówka:

```
$page_title = 'Kawy świata – koszyk na zakupy';
include ('./includes/header.html');
```

6. Dołącz skrypty odpowiedzialne za połączenie z bazą danych oraz funkcje pomocnicze:

```
require (MYSQL);
include ('./includes/product_functions.inc.php');
```

7. Jeśli w adresie URL znajduje się wartość SKU, podziel ją na części:

```
if (isset($_GET['sku'])) {
    list($sp_type, $pid) = parse_sku($_GET['sku']);
}
```

Dzięki wywołaniu funkcji `parse_sku()` numer SKU, który może być przekazany w adresie URL, zostanie podzielony na dwie części: typ produktu oraz jego identyfikator.

8. Sprawdź, czy produkt ma być dodany do koszyka:

```
if (isset($_sp_type, $pid, $_GET['action']) && ($_GET['action'] == 'add')) {
    $r = mysqli_query($dbc, "CALL add_to_cart('$uid', '$sp_type', $pid, 1)");
```

Wskazówka

Za pomocą funkcji `isset()` można za jednym razem sprawdzić istnienie wielu zmiennych.

Cała logika tego skryptu opiera się na długiej instrukcji warunkowej IF-ELSEIF, sprawdzającej różne możliwości wywołania skryptu. Pierwszą możliwością jest kliknięcie przycisku *Do koszyka*, co powoduje utworzenie adresu URL o postaci np. <http://nazwahosta/cart.php?sku=C8&action=add>. W tym przypadku numer SKU zostanie podzielony na wartości `$sp_type` i `$pid`, a zmienna `$_GET['action']` będzie równa `add`.

Jeśli wszystkie warunki są spełnione, jest wywoływana procedura składowana `add_to_cart()` i przekazywany jest jej identyfikator użytkownika (`$uid`), typ (`$sp_type`) i identyfikator produktu (`$pid`) oraz ilość wynosząca 1.

Wskazówka

Jeśli chcesz umożliwić klientom dodawanie do koszyka większych ilości produktu za jednym razem, musiałyby pobrać tę wartość z adresu URL, sprawdzić ją i przekazać procedurze `add_to_cart()`.

9. Sprawdź, czy produkt ma zostać usunięty z koszyka:

```
} elseif (isset($_sp_type, $pid, $_GET['action']) && ($_GET['action']
↳ == 'remove')) {
    $r = mysqli_query($dbc, "CALL remove_from_cart('$uid', '$sp_type', $pid)");
```

Warunek sprawdzany za pomocą funkcji `isset()` jest podobny do poprzedniego (dla dodawania produktów). Jeśli wartość `$_GET['action']` jest równa `remove`, wywoływana jest procedura składowana `remove_from_cart()`. Do takiej sytuacji dojdzie po kliknięciu przez użytkownika linku *Usuń z koszyka* (patrz rysunek 9.1).

10. Sprawdź, czy produkt ma zostać przeniesiony do koszyka:

```
} elseif (isset($_sp_type, $pid, $_GET['action'], $_GET['qty']) &&
↳ ($_GET['action'] == 'move')) {
    $qty = (filter_var($_GET['qty'], FILTER_VALIDATE_INT, array('min_range' => 1)))
↳ ? $_GET['qty'] : 1;
```

```
$r = mysqli_query($dbc, "CALL add_to_cart('$uid', '$sp_type', $pid, $qty)");
$r = mysqli_query($dbc, "CALL remove_from_wish_list('$uid', '$sp_type', $pid)");
```

Klient ma możliwość przenoszenia produktów między koszykiem i przechowalnią. Jeśli coś znajduje się w przechowalni i jest przenoszone do koszyka, wykonywana jest podobna operacja do dodawania do koszyka. Między tymi działaniami są jednak dwie istotne różnice. Po pierwsze, brana jest pod uwagę również liczba produktów, czyli jeśli w przechowalni były np. trzy kubki, do koszyka zostaną przeniesione wszystkie trzy. Po drugie, po przeniesieniu produktu do koszyka, musi on zostać usunięty z przechowalni.

Aby to wszystko zadziało, w instrukcji warunkowej jest sprawdzane istnienie wartości `$_GET['qty']` oraz to, czy `$_GET['action']` jest równe `move`. Następnie jest sprawdzana poprawność wartości ilości (liczba całkowita nie mniejsza niż 1).

Wyniki wszystkich procedur składowanych wywoływanych w skrypcie są przypisywane zmiennej `$r`, mimo że tak naprawdę jest używany wynik tylko jednej z nich (mowa o procedurze zwracającej zawartość koszyka). Mimo to lepiej przypisywać wyniki do zmiennej, ponieważ ułatwia to debugowanie.

Jeśli w czasie pracy nad skryptem pojawią się problemy, po wywołaniu procedury możesz wpisać następujący kod:

```
if (!$r) echo mysqli_error($dbc);
```

11. Sprawdź, czy został przesłany formularz:

```
} elseif (isset($_POST['quantity'])) {
```

Wszystkie poprzednie warunki korzystały z żądań typu GET, natomiast ta strona może być również wywołana za pomocą żądania POST. Do tego typu sytuacji dojdzie, gdy użytkownik zmieni ilość produktów w koszyku i prześle formularz koszyka.

12. Przejdź w pętli przez wszystkie przesłane pozycje koszyka:

```
foreach ($_POST['quantity'] as $sku => $qty) {
    list($sp_type, $pid) = parse_sku($sku);
    if (isset($sp_type, $pid)) {
        $qty = (filter_var($qty, FILTER_VALIDATE_INT, array('min_range' => 0))) ?
            ↪ $qty : 1;
        $r = mysqli_query($dbc, "CALL update_cart('$uid', '$sp_type', $pid, $qty)");
    }
}
```

W `$_POST['quantity']` znajduje się tablica elementów w postaci `SKU => ilość`. Dla każdego elementu w `$_POST['quantity']` musi zostać zaktualizowany odpowiedni produkt w koszyku. W tym celu najpierw jest przetwarzany i sprawdzany numer SKU, a następnie sprawdzana jest ilość. Wartość ta zostanie użyta, jeśli jest to liczba całkowita równa co najmniej 0 (ponieważ klient może wpisać zero, by usunąć produkt z koszyka). Jeśli z jakiegoś powodu klient wprowadzi nieprawidłową wartość, zostanie zastosowana domyślna, wynosząca 1. Na końcu jest wywoływana procedura składowana `update_cart()` z prawidłowymi argumentami.

13. Zakończ główną instrukcję warunkową i pobierz zawartość koszyka:

```
} // Koniec głównej instrukcji warunkowej IF.
$r = mysqli_query($dbc, "CALL get_shopping_cart_contents('$uid')");
```

Bez względu na wykonaną akcję zawsze jest wyświetlana aktualna zawartość koszyka.

14. Dołącz właściwy plik widoku:

```
if (mysqli_num_rows($r) > 0) {
    include ('./views/cart.html');
} else { // Pusty koszyk!
    include ('./views/emptycart.html');
}
```

15. Zakończ kod strony:

```
include ('./includes/footer.html');
```

16. Zapisz plik.

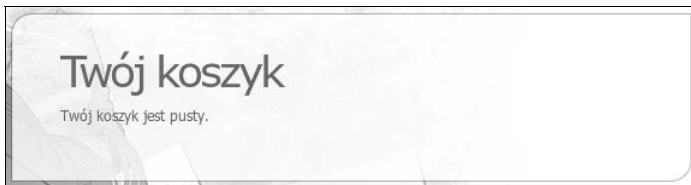
Tworzenie plików widoku

Uwaga

Oba pliki widoków muszą być oczywiście zapisane w katalogu *views*.

Koszyk na zakupy używa dwóch plików widoku: jeden służy do wyświetlania produktów, a drugi informuje o tym, że koszyk jest pusty. Ten ostatni — *emptycart.html* — jest bardzo prosty (rysunek 9.3):

```
<!-- początek bloku -->
<div class="box alt"><div class="left-top-corner"><div class="right-top-corner">
↳<div class="border-top"></div></div></div><div class="border-left">
↳<div class="border-right"><div class="inner">
<h2>Twój koszyk</h2>
<p>Twój koszyk jest pusty.</p>
</div></div></div><div class="left-bot-corner"><div class="right-bot-corner">
↳<div class="border-bot"></div></div></div></div>
<!-- koniec bloku -->
```



Rysunek 9.3

Drugi plik widoku jest trochę bardziej skomplikowany. Musi wyświetlać wszystkie produkty zwrócone przez procedurę składowaną, czyli produkty dodane do koszyka. Jednak lista musi być formularzem HTML, tak by użytkownik mógł zaktualizować ilość. Ponadto każdy produkt powinien być uzupełniony linkami do funkcji usuwania z koszyka i przenoszenia do przechowalni. Poza tym muszą być obliczane i wyświetlane informacje o wartości poszczególnych pozycji koszyka oraz o wartości całego koszyka (na rysunku 9.4 jest przedstawiona początkowa wersja tego widoku, a końcówką można zobaczyć na rysunku 9.1).



Rysunek 9.4

1. W edytorze lub IDE utwórz nowy plik HTML i zapisz go w katalogu *views* pod nazwą *cart.html*.
2. Rozpocznij blok HTML i nagłówek:

```
<div class="box alt"><div class="left-top-corner"><div class="right-top-corner">
<div class="border-top"></div></div></div><div class="border-left">
↳<div class="border-right"><div class="inner">
<h2>Twój koszyk</h2>
<p>Użyj tego formularza do zmiany zawartości koszyka. Możesz zmienić ilości,
↳całkowicie usunąć produkty z koszyka albo przenieść je do przechowalni,
↳by zamówić je później. Koszt dostawy i obsługi zamówienia jest podany nad
↳wartością koszyka. Jeśli chcesz złożyć zamówienie, kliknij przycisk Złóż
↳zamówienie. Zostaniesz przeniesiony do bezpiecznej strony, na której
↳sfinalizujesz zakupy.</p>
```

Przygotowując tekst na stronę koszyka, musisz go odpowiednio wyważyć, by zawierał najważniejsze informacje, ale nie był zbyt długi. Pamiętaj, że podstawową funkcją strony koszyka na zakupy jest skłonienie klienta do złożenia zamówienia!

3. Rozpocznij kod formularza:

```
<form action="/cart.php" method="POST">
```

Formularz odsyła dane z powrotem do skryptu *cart.php* za pomocą metody POST. Wartość atrybutu *action* rozpoczyna się od ukośnika, by wskazać, że skrypt *cart.php* znajduje się w głównym katalogu serwera WWW. Akurat w tym przypadku nie jest on niezbędny, ale w wielu innych tak, więc zastosowałem go w celu zachowania spójności i konsekwencji.

4. Rozpocznij definicję tabeli:

```
<table border="0" cellspacing="8" cellpadding="6">
<tr>
```

```

    <th align="center">Produkt</th>
    <th align="center">Ilość</th>
    <th align="right">Cena</th>
    <th align="right">Wartość</th>
    <th align="center">0pcje</th>
</tr>

```

Do wyświetlenia zawartości koszyka stosujemy stary, sprawdzony sposób z tabelą. W tym przypadku potrzebnych jest pięć kolumn.

5. Rozpocznij blok kodu PHP:

```

<?php
$total = 0;

```

W bloku kodu PHP zmienna `$total` jest inicjowana wartością 0, tak by całkowita wartość koszyka została prawidłowo obliczona.

6. Pobierz każdą pozycję koszyka:

```

while ($row = mysqli_fetch_array($r, MYSQLI_ASSOC)) {
    $price = get_just_price($row['price'], $row['sale_price'] );
    $subtotal = $price * $row['quantity'];
}

```

Wskazówka

Ze względów bezpieczeństwa cena jest zawsze pobierana z bazy danych, tak by klient nie mógł jej w żaden sposób zmienić.

W pętli najpierw jest określana obowiązująca cena, za co odpowiada funkcja `get_just_price()`, której przekazuje się normalną i promocyjną cenę produktu. Następnie jest obliczana wartość danej pozycji koszyka jako iloczyn ceny i ilości.

7. Wyświetl wiersz tabeli:

```

echo '<tr>
<td>' . $row['category'] . ' :: ' . $row['name'] . '</td>
<td align="center"><input type="text" name="quantity[' . $row['sku'] . ']'
↳value="' . $row['quantity'] . '" size="2" /></td>
<td align="right">' . $price . ' zł</td>
<td align="right">' . number_format($subtotal, 2) . ' zł</td>
<td align="right"><a href="/wishlist.php?sku=' . $row['sku'] . '&action=move&qty=' .
↳$row['quantity'] . '">Przenieś do przechowalni</a><br /><a href="/cart.php?sku=' .
↳$row['sku'] . '&action=remove">Usuń z koszyka</a></td>
</tr>
';

```

Dla każdej pozycji koszyka jest tworzony osobny wiersz tabeli. W pierwszej kolumnie jest wyświetlana nazwa, która składa się z nazwy kategorii i produktu, tak samo, jak było na stronach z produktami. W drugiej kolumnie znajduje się ilość, przy czym jest to wartość, którą można edytować, więc jest wyświetlana w polu formularza. Nazwa każdego pola jest tworzona według schematu `quantity[sku]`, więc po wysłaniu formularza będzie dostępna wartość zarówno SKU, jak i nowo ustawionej ilości (rysunek 9.5).

```

<tr><td>Kubki::"Czerwony smok"</td>
<td align="center"><input type="text" name="quantity[I2]" value="1" size="2" class="small" /></td>
<td align="right">21.00 zł</td>
<td align="right">21.00 zł </td>
<td align="right"><a href="/wishlist.php?sku=I2&action=move&qty=1">Przenieś do przechowalni</a><br />
<a href="/cart.php?sku=I2&action=remove">Usuń z koszyka</a></td>
</tr>

<tr><td>Kona::1 kg - bezkof. - ziarnista</td>
<td align="center"><input type="text" name="quantity[K9]" value="1" size="2" class="small" /></td>
<td align="right">52.50 zł</td>
<td align="right">52.50 zł </td>
<td align="right"><a href="/wishlist.php?sku=K9&action=move&qty=1">Przenieś do przechowalni</a><br />
<a href="/cart.php?sku=K9&action=remove">Usuń z koszyka</a></td>
</tr>

```

Rysunek 9.5

W dwóch kolejnych kolumnach jest wyświetlana cena i wartość. W piątej kolumnie są umieszczone dwa linki. Pierwszy służy do przenoszenia pozycji koszyka do przechowalni i przekazuje skryptowi *wishlist.php* wartość SKU, bieżącą ilość i akcję *move*. Drugi link prowadzi do tego samego skryptu (*cart.php*) i powoduje usunięcie pozycji z koszyka.

8. Dodaj komunikat o błędzie, jeśli w magazynie nie ma wystarczającej ilości produktu:

```

if ($row['stock'] < $row['quantity'] ) {
    echo '<tr class="error"><td colspan="5" align="center">W magazynie zostało
    ↳ tylko ' . $row['stock'] . ' sztuk produktu ' . $row['name'] . '. Zmień ilość
    ↳ określoną w zamówieniu, usuń całkowicie tę pozycję lub przenieś ją do
    ↳ przechowalni.</td></tr>';
}

```

Jeśli klient chce zamówić więcej sztuk jakiegoś produktu, niż aktualnie jest w magazynie, musi zostać powiadomiony o problemie (rysunek 9.6). W takiej sytuacji klient poznaje stan magazynowy i jest proszony o samodzielne rozwiązanie problemu. W kolejnym rozdziale zmodyfikujesz kod, tak by produkty, których jest za mało, były automatycznie usuwane z zamówienia.

Produkt	Ilość	Cena	Wartość	Opcje
Kubki::"Czerwony smok"	<input type="text" value="6"/>	21.00 zł	126.00 zł	Przenieś do przechowalni Usuń z koszyka
W magazynie zostało tylko 5 sztuk produktu "Czerwony smok". Zmień ilość określoną w zamówieniu, usuń całkowicie tę pozycję lub przenieś ją do przechowalni.				

Rysunek 9.6

9. Do zmiennej `$total` dodaj wartość pozycji (`$subtotal`) i zakończ pętlę:

```

$total += $subtotal;
} // Koniec pętli WHILE

```

10. W tabeli wyświetl całkowitą wartość koszyka (`$total`) i zakończ blok kodu PHP:

```

echo '<tr>
    <td colspan="3" align="right"><strong>Wartość</strong></td>
    <td align="right">$' . number_format($total, 2) . '</td>
    <td>&nbsp;&nbsp;&nbsp;</td>
</tr>
';
?>

```

11. Zakończ definicję tabeli i utwórz dwa przyciski:

```
</table><br /><p align="center">
<input type="submit" value="Aktualizuj koszyk" class="button" /></form></p><br />
<p align="center"><a href="https://<?php echo BASE_URL; ?>"checkout.php?session=
↳<?php echo $uid; ?>" class="button">Złóż zamówienie</a></p></div>
```

Pierwszy przycisk służy do przestania formularza (w celu zaktualizowania ilości). Kliknięcie drugiego przycisku powoduje rozpoczęcie procesu finalizowania zamówienia. Link ten kieruje do skryptu *checkout.php* poprzez bezpieczne połączenie HTTPS. Aby wygenerować adres, jest wymagana wartość stałej `BASE_URL`. Zwróć uwagę, że w adresie jest przekazywany identyfikator sesji użytkownika, tak więc w skrypcie *checkout.php* będzie można z niego skorzystać. O tym, że jest to niezbędne, przekonasz się w kolejnym rozdziale.

Uwaga

Proces finalizowania zamówienia musi się odbywać na bezpiecznych stronach, tak by klient czuł się bezpiecznie!

12. Zakończ kod strony:

```
</div></div><div class="left-bot-corner"><div class="right-bot-corner">
↳<div class="border-bot"></div></div></div></div>
<!-- koniec bloku -->
```

13. Zapisz plik i przetestuj go w przeglądarce internetowej.

Cała funkcjonalność witryny, z wyjątkiem przechowalni, powinna już działać, więc możesz ją przetestować. W sklepie możesz już dodawać produkty do koszyka, a na stronie koszyka możesz aktualizować ilość produktów i je usuwać.

BUDOWANIE PRZECHOWALNI

Procedury składowane używane w zarządzaniu przechowalnią są w zasadzie identyczne z tymi, które są stosowane w koszyku na zakupy. To samo dotyczy skryptu PHP i plików HTML — będą prawie wiernymi kopiami już napisanych. Przedstawię je tu w całości, więc jeśli chcesz poznać szczegóły, wróć do opisu ich odpowiedników dla koszyka na zakupy.

Tworzenie skryptu PHP

Skrypt *wishlist.php* zapisany w głównym katalogu serwera WWW różni się od *cart.php* tylko czterema elementami:

- tytułem strony,
- nazwą wywoływanej procedury składowanej,
- innymi plikami widoków,
- brakiem instrukcji warunkowej sprawdzającej dodawanie produktów.

Tak naprawdę, poza ostatnią różnicą, skrypt dla przechowalni możesz przygotować metodą „wyszukaj i zamień”, traktując jako bazę kod skryptu *cart.php*. Ostatnia różnica polega na tym, że w tym skrypcie produkty są dodawane do przechowalni przez przeniesienie ich z koszyka na zakupy.

```
<?php // wishlist.php
require ('./includes/config.inc.php');
if (isset($_COOKIE['SESSION'] )) {
    $suid = $_COOKIE['SESSION'];
} else {
    $suid = md5(uniqid('biped',true));
}
setcookie('SESSION', $suid, time( )+(60*60*24*30));
$page_title = 'Kawy świata - przechowalnia';
include ('./includes/header.html');
require (MYSQL);
include ('./includes/product_functions.inc.php');
if (isset($_GET['sku'] )) {
    list($sp_type, $pid) = parse_sku($_GET['sku'] );
}
if (isset ($sp_type, $pid, $_GET['action'] ) && ($_GET['action'] == 'remove')) {
    $r = mysqli_query($dbc, "CALL remove_from_wish_list('$suid', '$sp_type', $pid)");
} elseif (isset ($sp_type, $pid, $_GET['action'], $_GET['qty'] ) && ($_GET['action']
↳ == 'move') ) {
    $qty = (filter_var($_GET['qty'], FILTER_VALIDATE_INT, array('min_range' => 1)))
↳ ? $_GET['qty'] : 1;
    $r = mysqli_query($dbc, "CALL add_to_wish_list('$suid', '$sp_type', $pid, $qty)");
    $r = mysqli_query($dbc, "CALL remove_from_cart('$suid', '$sp_type', $pid)");
} elseif (isset($_POST['quantity'] )) {
    foreach ($_POST['quantity'] as $sku => $qty) {
        list($sp_type, $pid) = parse_sku($sku);
        if (isset($sp_type, $pid)) {
            $qty = (filter_var($qty, FILTER_VALIDATE_INT, array('min_range' => 0))) ?
↳ $qty : 1;
            $r = mysqli_query($dbc, "CALL update_wish_list('$suid', '$sp_type', $pid,
↳ $qty)");
        }
    }
}
}
$r = mysqli_query($dbc, "CALL get_wish_list_contents('$suid')");
if (mysqli_num_rows($r) > 0) {
    include ('./views/wishlist.html');
} else {
    include ('./views/emptylist.html');
}
include ('./includes/footer.html');
?>
```

Tworzenie plików widoku

Uwaga

Oczywiście oba pliki widoku muszą być zapisane w katalogu *views*.

Gdy przechowalnia klienta jest pusta, używany jest plik widoku *emptylist.html* (rysunek 9.7).



Rysunek 9.7

```
<!-- początek bloku -->
<div class="box alt"><div class="left-top-corner"><div class="right-top-corner">
↳<div class="border-top"></div></div></div><div class="border-left">
↳<div class="border-right"><div class="inner">
<h2>Twoja przechowalnia</h2>
<p>Twoja przechowalnia jest pusta.</p>
</div></div></div><div class="left-bot-corner"><div class="right-bot-corner">
↳<div class="border-bot"></div></div></div></div>
<!-- koniec bloku -->
```

Plik widoku *wishlist.html* wyświetla listę produktów w tabeli HTML. Podobnie jak w koszyku na zakupy, tu również zawartość listy można modyfikować na kilka sposobów:

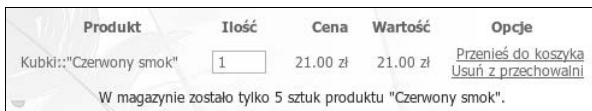
- zmieniając ilość produktów,
- przenosząc do koszyka na zakupy,
- usuwając z przechowalni.

Inaczej niż w skrypcie koszyka, na stronie przechowalni nie są wyświetlane ani informacje o wartości wszystkich produktów, ani link do strony zatwierdzania zamówienia (rysunek 9.8). Z kolei zamiast informowania o niewystarczającym stanie magazynowym na stronie przechowalni pojawia się jedynie komunikat z informacją o kończących się zapasach, co w zamysle ma skłonić klienta do natychmiastowego zamówienia danego produktu (rysunek 9.9).

```
<div class="box alt"><div class="left-top-corner"><div class="right-top-corner">
↳<div class="border-top"></div></div></div><div class="border-left">
↳<div class="border-right"><div class="inner">
<h2>Twoja przechowalnia</h2>
<p>Użyj tego formularza do zmiany zawartości przechowalni. Możesz zmienić ilości,
↳całkowicie usunąć produkty z przechowalni albo przenieść je do koszyka na zakupy.</p>
<form action="/wishlist.php" method="POST">
<table border="0" cellspacing="8" cellpadding="6">
```



Rysunek 9.8



Rysunek 9.9

```

<tr>
  <th align="center">Produkt</th>
  <th align="center">Ilość</th>
  <th align="right">Cena</th>
  <th align="right">Wartość</th>
  <th align="center">Opcje</th>
</tr>
<?php
while ($row = mysqli_fetch_array($r, MYSQLI_ASSOC)) {
  $price = get_just_price($row['price'], $row['sale_price'] );
  $subtotal = $price * $row['quantity'];
  echo '<tr>
    <td>' . $row['category'] . '::~' . $row['name'] . '</td>
    <td align="center"><input type="text" name="quantity[' . $row['sku'] . ']'
      ↳value="' . $row['quantity'] . '" size="2" /></td>
    <td align="right">$' . number_format($price, 2) . '</td>
    <td align="right">$' . number_format($subtotal, 2) . '</td>
    <td align="right"><a href="/cart.php?sku=' . $row['sku'] . '&action=move&qty='
      ↳$row['quantity'] . '">Przenieś do koszyka</a><br /><a href="/wishlist.php?
      ↳sku=' . $row['sku'] . '&action=remove">Usuń z przechowalni</a></td>
</tr>
';
// Sprawdź status:
if ( ($row['stock'] > 0) && ($row['stock'] < 10) ) {
  echo '<tr class="error"><td colspan="5" align="center">W magazynie zostało
    ↳tylko ' . $row['stock'] . ' sztuk produktu ' . $row['name'] . ' .</td></tr>';
}
}

```

```

} // Koniec pętli WHILE.
?> </table><p align="center"><input type="submit" value="Aktualizuj przechowalnię"
↳class="button" /></form></p></div></div></div></div><div class="left-bot-corner">
↳<div class="right-bot-corner"><div class="border-bot"></div></div></div></div>
<!--koniec bloku -->

```

OBLICZANIE KOSZTÓW WYSYŁKI

Do koszyka warto dodać przydatną funkcję podającą koszt wysyłki. Dla wielu klientów opłaty za transport i obsługę zamówienia są istotnym czynnikiem, który może decydować o dokonaniu zakupu. Koszty te mogą być stałe lub można je uzależnić od:

- wagi przesyłanych produktów,
- odległości między nadawcą a odbiorcą,
- wymiarów paczki (przesłanie np. mebli wiąże się z dużymi kosztami),
- wielkości zamówienia.

W witrynie zostanie zastosowane ostatnie kryterium.

Przed wszystkim musisz zdefiniować funkcję obliczającą koszty wysyłki według ustalonej zależności. Załóżmy, że ustalamy pewien bazowy koszt, który trzeba przeznaczyć na wynagrodzenie pracownika kompletującego i pakującego zamówienie. Do tego należy doliczyć koszt częściowo zależny od wielkości zamówienia — co prawda większe zamówienia wiążą się z większą i cięższą przesyłką, jednak również więcej się na nich zarabia. Z tego względu koszty wysyłki zdefiniujemy tak, by były odwrotnie proporcjonalne do wielkości zamówienia (określanego na podstawie całkowitej wartości zamówienia). W związku z tym funkcja może wyglądać następująco:

```

function get_shipping($total = 0) {
    // Ustal bazowe koszty obsługi zamówienia:
    $shipping = 3;
    // Ustal współczynnik zależny od całkowitej wartości zamówienia:
    if ($total < 50) {
        $rate = .25;
    } elseif ($total < 100) {
        $rate = .20;
    } elseif ($total < 200) {
        $rate = .18;
    } elseif ($total < 500) {
        $rate = .16;
    } else {
        $rate = .15;
    }
    // Oblicz całkowity koszt przesyłki:
    $shipping = $shipping + ($total * $rate);
    // Zwróć całkowity koszt przesyłki:
    return number_format($shipping, 2);
} // Koniec funkcji get_shipping().

```

Jak się możesz domyślać, funkcję należy umieścić w pliku *product_functions.inc.php*, tak by była dostępna dla wielu skryptów (np. *cart.php* czy *checkout.php*, który zostanie utworzony w kolejnym rozdziale).

Aby skorzystać z tej funkcji w skrypcie *cart.php* (w przechowalni nie jest wyświetlana całkowita wartość zamówienia), dodaj poniższy kod przed wierszem wyświetlającym całkowitą wartość koszyka (rysunek 9.10).

Produkt	Ilość	Cena	Wartość	Opcje
Kubki::"Czerwony smok"	<input type="text" value="1"/>	21.00 zł	21.00 zł	Przenieś do przechowalni Usuń z koszyka
Kona::500 g - kof. - ziarnista	<input type="text" value="2"/>	25.80 zł	51.60 zł	Przenieś do przechowalni Usuń z koszyka
Transport i obsługa			17.52 zł	
Suma			90.12 zł	

Rysunek 9.10

```
$shipping = get_shipping($total);  
$total += $shipping;  
echo '<tr>  
<td colspan="3" align="right"><strong>Transport i obsługa</strong></td>  
<td align="right">' . $shipping . ' zł</td>  
<td>&nbsp;</td>  
</tr>  
';
```

SKOROWIDZ

?=, 114
127.0.0.1, 57
777, 150

A

Account, 345
Add
 Bank Account, 173
 Credit Card, 173
add_to_cart, 277
addslashes, 83, 309, 328, 339
administracja witryną, 347
administrator, 107
 kontakt, 36
 konto, 133
 uprawnienia, 57
Adobe Dreamweaver, 16
adres, 301, 309, 329
 e-mail, 203, 301, 311, 320
 walidacja, 113, 119
 klienta, 312
 mechanizm przepisywania, 215
 pocztowy, 203
 przyjazny, 212
 URL, 212
 witryny, 84
 weryfikacja, 34
Advanced
 Integration Method, 290, 292
 PHP Debugger, 41
After how many cycles should billing stop?, 175
AIM, 290, 292, 335, 337
aktualizowanie koszyka, 272
aktywacja
 konta, 117
 SSL, 215
algorytm tworzenia skrótów, 105
allow, 211
ALTER, 58
 ROUTINE, 228, 260

Alternative PHP Cache, 41
American Express, 23
 numer karty, 328
anonimowa funkcja, 353
anulowanie płatności, 176
Apache, 15, 16, 54, 348
ApacheBench, 40
APC, 41
APD, 41
Aptana Studio, 16
argument
 nazwa, 300
 procedura składowana, 241
 wchodzący, 300
 wychodzący, 300
asercje sprawdzające następstwo znaków, 114
atak
 brute force, 65
 CSRF, 68
 DoS, 65
 hakerów, , 47, 56
 przez wstrzykiwanie kodu SQL, 82
 przez wymuszanie sesji, 131
 sitowy, 65
 słownikowy, 105
 SQL Injection, 66
 wzrost zagrożenia, 47
 XSS, 64, 137
 zapobieganie, 65
Authorize.net, 32, 289, 330, 332, 337
 konto rzeczywiste, 345
 pola odpowiedzi, 338
 testowania funkcji, 292
 uruchomienie systemu płatności, 345
autoryzowanie płatności, 338

B

back_log, 38
backtrace, 85
basename, 91

baza danych, 57, 269
 bezpieczeństwo, 58
 blok kodu, 225
 dodanie nowego rekordu, 114
 zamówienia, 302
 indeks, 37
 kodowanie, 81
 niepotrzebne zapytania, 95
 normalizacja, 37
 połączenie, 81
 projekt, 36, 74, 201
 replikacja, 38
 silnik, 37
 sprawdzanie zawartości, 233
 struktura tabel, 232
 tworzenie, 76
 użytkownicy, 57
 wstawianie rekordów, 230
 wydajność, 58
 zoptymalizowanie, 38
 wypełnianie, 230, 231
BEGIN, 271
 bezpieczeństwo, 45, 120, 281, 326
 całkowite, 46
 bazy danych, 58
 danych, 38
 dostarczanych przez użytkowników, 63
 luka, 87, 150
 minimum, 52
 naruszenie zasad, 332
 odpowiedni poziom, 47
 ograniczenie
 dostępu do katalogu, 80
 ilości danych, 63
 PHP, 54
 podwyższenie, 52, 130, 133, 151, 154, 179, 208, 221, 225, 306, 348
 serwera, 52, 54
 sposób na zwiększenie, 78
 systemu kont użytkowników, 113
 witryny, 48
 wyjątkowe, 226
 bezpieczna sieć komputerowa, 50
 bezpieczny identyfikator konta sprzedawcy, 175
Billing
 amount each cycle, 175
 cycle, 175
 blok kodu PHP, 91
 błąd
 dołączania pliku, 83
 informacja, 322
 komunikat, 84, 87, 103
 powiadomienie administratora, 331
 raportowane w postaci e-maili, 56

 tworzenie opisu, 85
 zabezpieczenie przed, 100
 bramka płatności, 31, 32, 299, 311, 330, 333
 rejestr zapytań, 299
 brute force, 65

C

całkowita wartość zamówienia, 302, 331, 336, 385
CAPTCHA, 69
 Card Verification Value, 34, 325
 Cart Not Present, 292
 Cascading Style Sheet, 24, 25, 26, 78, 92, 94, 101, 144, 210, 224, 299, 370, 354
 cele biznesowe, 20
 cena, 202, 281, 365
 domyślna, 273
 promocyjna, 258, 259261, 273, 373
 centrum certyfikacji, 60
 Certifying Authority, 60
 certyfikat
 SSL, 60
Checkout, 31
Choose
 a button type, 175
 IPN Settings, 184
 ciasteczko, 107, 204, 276, 307
 ograniczenie czasu życia, 68
 sesji, 122
CKEditor, 139
CNP, 292
 commands out of sync, 317
Completed, 188
Completely Automated Public Turing test to tell Computers and Humans Apart, 69
CONCAT_WS, 235
 concatenation with separator, 235
Cracklib PECL, 113
CREATE, 58
 PROCEDURE, 226
 ROUTINE, 228
Create
 a Sandbox Test Account, 172
 Account, 173
 Button, 174, 176
 manually, 171
 PayPal payment button, 174
 create_form_input, 110, 157, 354
Cross-Site
 Request Forgery, 68
 Scripting, 64, 137
CSRF, 68
CSS, 24, 25, 26, 78, 92, 94, 101, 144, 210, 224, 299, 370, 354

cURL
 odpowiedź, 337
 opcje, 337
 Currency, 175
 CVC, 325
 CVV, 34, 325
 CVVC, 325
 czas
 złożenia zamówienia, 204
 życia ciasteczka, 68
 czek elektroniczny, 290
 czytelność instrukcji, 92

D

dane
 adresowe, 176, 306
 bezpieczeństwo, 38
 binarne, 105
 do wysyłki, 323
 klienta, 67, 201, 301
 osobowe, 22
 ochrona, 36
 posiadacza rachunku, 323
 poufne, 64, 321, 327, 334, 335
 przekazywane przez użytkowników, 64
 szyfrowanie, 58, 59
 walidacja, 111
 właścicieli kart kredytowych, 50
 darowizna
 przyjmowanie, 168
 data
 utworzenia, 202
 ważności konta, 75
 Datatables, 379
 Datepicker, 372, 375
 aktywacja, 376
 debug_backtrace, 86
 debugowanie
 narzędzia, 16
 skryptów JavaScript, 354
 DECLARE, 270
 DELETE, 271, 299
 definiowanie funkcji, 102
 deklarowanie zmiennej, 270
 Denial of Service, 65
 deny, 211
 diagram przepływu, 38
 Discover, 23
 display_errors, 56
 długość cyklu płatności, 175
 Do not receive IPN messages, 184
 Do you need your customer's shipping address?, 176
 dobre zasady tworzenia oprogramowania, 38
 DocumentRoot, 54

dodanie nowego zamówienia do bazy danych, 302
 dokonywanie płatności, 321
 dokumentacja, 50
 dolara znak, 214
 dołączanie pliku
 błąd, 83
 dostęp do stron
 ograniczanie, 106
 DoS, 65
 DROP, 58
 działanie skryptu
 zakończenie, 107
 dziennik błędów, 313

E

E_ALL | E_STRICT, 56
 echo, 157
 e-commerce, 13
 edytor
 do pracy na stronach internetowych, 139
 kodowanie, 81
 problemy, 142
 element
 nazwa, 174
 else, 95
 ELSEIF, 354
 e-mail, 203, 301, 311, 320
 Enter Sandbox Test Site, 174
 ENUM, 114
 error, 103, 157
 error_log, 86
 EV, 63
 exec, 56, 67
 EXECUTE, 228, 244, 246
 exit, 107
 explode, 337
 Extended Validation, 63

F

falszywa rejestracja, 117
 FCKeditor, 139
 Federal Trade Commission, 22
 FILE, 58
 Filter, 113
 filter_var, 113
 Firebug, 354
 Firefox, 16
 firma
 godna zaufania, 60
 hostingowa, 53
 first_name, 185
 Flash, 24

fopen, 67
 formularz, 100

- błędne pole, 101
- definiowanie, 365
- instrukcja wypełniania, 110
- logowania, 118
- ułatwienie obsługi, 99
- zapamiętanie wpisanych wartości, 100

 forum poświęcone książce, 15
 fragment numeru karty kredytowej, 204
 fsockopen, 186, 193
 FTC, 22
 funkcja my_error_handler, 85
 funkcja

- anonimowa, 353
- definiowanie, 102
- haszująca, 104
- manipulująca plikami, 56
- obsługi błędów, 85, 87, 96
- skrót, 104

G

G, 235
 gadżety, 244
 gateway

- payment, 31

 General Security Settings, 345
 GET, 307
 get_order_contents, 301
 get_password_hash, 106, 114, 125
 Git, 39
 GoDaddy, 60
 Google API, 351, 375
 gość, 147
 GRANT, 58
 grupy użytkowników, 150

H

handel elektroniczny, 13, 20
 Hash

- rozszerzenie, 105

 hash_algos, 105
 hash_hmac, 105, 106
 hasło, 216

- bezpieczne, 99, 104
- bieżące, 128
- domyślne, 50
- odzyskanie, 123
- siła, 113
- zarządzanie, 123
- zmiana, 127

haszująca funkcja, 104
 header, 107
 hosting, 26, 53

- dedykowany, 53
- kryteria wyboru, 30
- najtańszy, 30
- plan, 27, 209, 215
- polecany, 29
- sposób na znalezienie dobrego, 30
- współdzielony, 48, 53, 56, 152
- wymagania minimalne, 27
- zmiana, 85

 hoverIntent, 352
 htaccess, 210, 212
 html, 80

- usuwanie kodu, 65

 htmlspecialchars, 100
 HTTPS, 62, 86, 135, 171, 178, 179, 210, 215, 217, 283, 285, 295, 309, 310, 311, 327, 342, 348

I

IDE, 16

- kodowanie, 81

 indeks

- baza danych, 37

 identyfikator

- klienta, 312, 385
- koszyka, 308
- sesji, 308
- transakcji, 385
- zamówienia, 301, 377

 IF-ELSE, 271, 339
 IF-ELSE IF, 240, 277
 IFNULL, 304
 IIS, 16
 imię, 301, 308, 326, 327

- i nazwisko klienta, 377

 inc.php, 80
 include, 67, 96, 97, 108
 indeksowanie tablicy, 373
 index.php, 96
 informacje o

- błędach, 322
- kartach kredytowych, 201
- klientach
 - zarządzanie, 290

 Information Server, 16
 init, 141
 INNER JOIN, 259
 InnoDB, 37, 208
 INSERT, 271, 302, 304
 Instant Payment Notification, 183
 Preferences, 184

instrukcja
 czytelność, 92
 integracja
 z witryną, 177
 zaawansowana, 290, 335
 Integrated Development Environment, 16
 interfejs sprzedawcy, 290
 imię
 walidacja, 112
 IPN, 180, 183, 185
 isset, 277
 Item name, 174

J

JavaScript, 15, 24
 debugowanie skryptów, 354
 usuwanie kodu, 65
 JCB, 23
 jednostka magazynowa, 198
 JOIN, 235, 259, 273, 301, 303, 386
 jQuery, 15, 351
 UI, 372, 375

K

kalkulator
 kosztów dostawy, 166
 podatków, 166
 karta kredytowa
 dane właściciela, 50
 informacja, 201
 numer
 fragment, 204
 niewłaściwy, 340
 testowy, 344
 problemy, 330
 sprzedawca nie akceptuje danego typu karty, 340
 termin ważności, 297, 329
 minął, 340
 karta testów, 51
 katalog
 administracyjny, 348
 produktów, 202
 zabezpieczenie przed nieuprawnionym dostępem, 210
 kategorie, 202
 lista, 243
 key_buffer_size, 38
 klasa znaków, 213
 klient
 dane, 67, 201, 301
 adresowe, 176, 306
 identyfikator, 312, 385
 imię i nazwisko, 377

informowanie po opłaceniu zamówienia, 343
 mysql
 konsola tekstowa, 230
 nazwa, 203
 uprawnienia, 57
 zaufanie, 64
 kłódka zamknięta, 59
 kod
 autoryzacyjny, 329
 odpowiedzi, 337
 pocztowy, 301, 310, 320, 377
 SQL
 wstrzykiwanie, 82, 113
 kodowanie
 bazy danych, 81
 edytor, 81
 znaków, 81
 kolory
 witryny, 35
 kolumna
 nazwa, 300
 komunikacja między PHP i MySQL, 105
 komunikat o
 błędzie, 84, 87, 103
 kończących się zapasach, 285
 koniec
 łańcucha, 214
 wiersza, 86
 konkatenacja z separatorem, 235
 konsola klienta mysql, 230
 kontakt z administratorem witryny, 36
 konto
 administratora, 133
 aktywacja, 117
 aktywne, 161
 bankowe, 173
 bezpieczeństwo, 113
 ważność, 119, 181
 zakładanie, 108
 kontrolowanie zalogowanych użytkowników, 85
 kopia bezpieczeństwa, 42, 64
 koszty, 34, 47
 dostawy, 204
 kalkulator, 166
 obsługi transakcji, 32
 transportu, 302, 316
 wysyłki, 287
 koszyk, 36, 204, 269, 273, 278
 aktualizowanie, 272
 identyfikator, 308
 plik widoku, 275, 279
 pobieranie zawartości, 273
 usuwanie produktu, 271
 zawartość, 314, 322

książka

forum, 15

kwerenda, 271, 364

symbol zakończenia, 240

warunek, 235

wybierająca, 234, 237

wybieranie losowe, 238

wynik wykonania, 234

wyświetlenie, 235

złączenie, 235

złożenie tabel, 237

znaki psujące, 105

kurs waluty

przeliczanie, 166

L

last_name, 185

LFI, 67

libcurl, 291

link

zarabianie na umieszczaniu, 14, 20

lista

kategorii, 243

produktów, 249, 267

Local File Inclusion, 67

localhost, 57

Login Email, 173

logowanie, 118

formularz, 118

lokalne dołączanie pliku, 67

Ł

łańcuch

koniec, 214

tekstowy, 271

M

macierz RAID, 64

Magic Quotes, 82, 101, 102, 309

Malicious File Execution, 67

MAMP, 16, 231

MasterCard, 23

max_connections, 38

MD5, 104, 105, 125

mechanizm przepisywania adresów, 215

menu

kaskadowe, 352

kategorii, 365

Merchant

Interface, 290

Services, 174

miasto, 301, 310, 377

mikropłatności, 34

MIME, 154

mod_rewrite, 212, 215, 265, 348

Model-View-Controller, 39, 199, 200, 209, 243, 269

model-widok-kontroler, 39, 199, 200, 209, 243, 269

move_uploaded_file, 155

MVC, 39, 199, 200, 209, 243, 269

wady, 201

zalety, 200

MyISAM, 37

MySQL, 37, 230, 299, 312, 331

5.0, 15

5.1.44, 15

Improved, 222

procedura składowana, 240

wersja, 221

mysqli_connect, 81

mysqli_real_escape_string, 82, 105, 106, 113

mysqli_set_charset, 81

N

nagłówek, 135

dokumentu HTML, 89

narzędzia do debugowania, 16

natychmiastowe powiadamianie o płatności, 183

nawigacja, 91

witryna, 36

nazwa

argument, 300

domenowa, 85

elementu, 174

generowanie, 358

klienta, 203

kolumna, 300

procedura składowana, 240

użytkownika, 112, 210, 216

zmiennej, 119

nazwisko, 301, 308, 326, 327

Never, 175

New test account, 171

niezrealizowane transakcje, 345

niski stan magazynowy, 205

n2br, 86

No, 176

normalizacja bazy danych, 37

notice, 87

nowa oferta, 205

numer karty kredytowej, 325, 328

format, 328

fragment, 204

niewłaściwy, 340

pole, 296

testowy, 344

usuwanie spacji i łączników, 328

numer
 sesji, 308
 telefonu, 203, 301, 310, 320

O

object-oriented programming, 39
 obrazek, 202
 skalowanie, 362
 obsługa błędów, 56
 funkcja, 85, 87, 96
 ochrona
 danych osobowych, 36
 przed oszustwami, 167, 188
 użytkownika, 64
 odczyt, 150
 odmowa
 reguła, 211
 usługi, 65
 odpowiedzialność za nieprawidłowe działanie sklepu, 21
 odwołanie
 bezwzględne, 85
 wsteczne, 214
 odzyskiwanie hasła, 123
 oferta
 nowa, 205
 ogólne ustawienia zabezpieczeń, 345
 OOP, 39
 opcode caching, 41
 open_basedir, 55
 opłacenie zamówienia, 343
 opłata za każdy cykl, 175
 oprogramowanie
 aktualizacja, 54
 dobre zasady tworzenia, 38
 wspomagające
 prowadzenie firmy, 38
 zarządzanie projektem, 38
 order_contents, 301
 osCommerce, 25, 166
 oszustwo
 próba, 381
 wykrywanie, 290
 z użyciem kart kredytowych, 311
 zapobieganie, 34, 167, 188, 193, 381
 OUTER JOIN, 259, 260

P

pasek
 adresu, 63
 wyświetlanie na zielono, 63
 postępu, 319

Payment
 Card Industry Data Security Standard, 23
 Data Transfer, 183
 gateway, 31
 processor, 31
 payment_status, 188
 PayPal, 31, 64, 75, 84, 165, 289
 gniazdo, 186
 integracja z witryną, 177
 opłaty w systemie, 167
 personalizacja konta, 192
 przelew na konto bankowe, 166
 przycisk, 174
 Sandbox, 169
 testowanie, 169
 PCI, 33, 50, 167, 332
 DSS, 23, 290
 PDF, 159, 160
 PDT, 183
 Pending, 188
 personalizacja sesji użytkownika, 85
 PHP
 5.1.2, 105
 5.2, 15, 113
 5.3.2, 15
 blok kodu, 91
 konfiguracja, 55
 plik dołączany, 78
 procedura składowana, 244
 profilowanie kodu, 41
 rozszerzenie, 80
 usuwanie kodu, 65
 wersja, 221
 php.ini, 55
 PHPDoc, 39
 phpinfo, 54, 64
 phpMyAdmin, 228, 230
 plan hostingowy, 27, 209, 215
 planowanie witryny, 35
 plik
 arkuszy stylów CSS, 77
 danych multimedialnych, 78
 do pobrania, 15
 dołączanie
 błąd, 83
 zdalne, 67
 dołączany PHP, 78
 funkcja manipulująca, 56
 JavaScript, 78
 konfiguracyjny, 83, 106, 217
 lokalne dołączanie, 67
 nazwa, 358
 obrazka poprawność, 357
 obrazów, 77
 PDF, 149

- plik
 - sprawdzanie rozmiaru, 154
 - stron administracyjnych, 77
 - zmiana nazwy, 67
 - plik widoku, 218, 245, 251, 253, 255, 267, 279, 314
 - koszyka, 275, 279
 - пустego, 275
 - strony głównej, 265
 - płatności, 75
 - anulowanie, 176
 - autoryzowane, 338
 - cykliczne, 23
 - długość cyklu, 175
 - dokonanie, 332
 - internetowe, 289
 - natychmiastowe powiadomienie, 183
 - nieautoryzowane, 338
 - okresowe, 169
 - ponawiane, 290
 - sfinalizowanie, 176
 - za pomocą kart kredytowych, 167
 - podatki
 - kalkulator, 166
 - pole, 103
 - ceny, 357
 - kategorii, 360
 - poprawność wyboru, 364
 - numeru karty kredytowej, 296
 - tekstowe, 102
 - wielowierszowe, 103, 354
 - typ, 102
 - wprowadzania haseł, 102
 - wypełnienie nieprawidłowe, 362
 - polskie znaki diakrytyczne, 112
 - połączenie z bazą danych, 216
 - poprawność wysokości opłaty, 188
 - POST, 280, 291, 295, 307
 - powiadomienie, 87
 - prawo
 - do uruchomienia procedury składowanej, 246
 - dostępu, 150
 - Preconfigured, 171
 - predefiniowane zapytania, 66
 - prepared statements, 66
 - procedura składowana, 201, 225, 233, 239, 246, 255, 259, 269, 270, 293, 298, 300
 - aktualizacja, 260
 - argumenty, 241
 - MySQL, 240
 - nazwa, 240
 - prawo do uruchomienia, 246
 - testowanie, 242
 - wywołanie przez PHP, 244
 - zmienna, 305
 - proces tworzenia witryny, 34
 - PROCESS, 58
 - procesem zarządzanie, 295
 - processor payment, 31
 - produkt, 202
 - dodawanie do bazy, 355
 - dostępny, 370, 374
 - lista, 249, 267
 - niedostępny, 315
 - sprzedaż, 20
 - wprowadzony do sprzedaży, 205
 - zarabianie na promocji, 20
 - profilowanie kodu PHP, 41
 - program antywirusowy, 51
 - programowanie
 - sterowane testami, 40
 - zorientowane
 - obiektowo, 39
 - proceduralnie, 39
 - projekt sklepu, 197
 - promocja, 258, 259, 261, 273
 - cena, 373
 - data rozpoczęcia, 373
 - próba
 - oszustwa, 381
 - sabotażu, 53
 - zaatakowania systemu, 190
 - przecena, 263
 - przechowalnia, 204, 278, 283
 - przeglądanie zamówień, 376
 - przenoszenia pozycji z koszyka do przechowalni, 282
 - przenośność aplikacji, 225
 - przesyłanie
 - danych karty kredytowej, 47
 - plików na serwer, kody błędów, 156
 - przetwarzanie
 - płatności, 31, 384
 - sfinalizowanych zamówień, 384
 - zamówień, 376
 - przyciski
 - tworzenie, 174
 - przyjazne adresy URL, 212
 - przyjmowanie czeków elektronicznych, 290
 - przypomnienie o konieczności sfinalizowania zamówienia, 205
- ## R
- RAID
 - macierz, 64
 - rand, 125
 - Receive IPN messages, 184
 - redirect_invalid_user, 128, 135
 - register_globals, 56

reguły
 odmowy, 211
 zezwolenia, 211
 rejestr zapytań do bramki płatności, 299
 rejestracja
 fałszywa, 117
 klientów, 203
 zakup bez rejestracji, 199
 reklama
 zarabianie na umieszczeniu, 14, 20
 rekord
 dodanie do bazy danych, 114
 RELOAD, 58
 Remote File Inclusion, 67
 replikacja bazy danych, 38
 require, 67, 96, 97
 return mysqli_real_escape_string, 82
 rezerwacja pieniędzy dla sprzedawcy, 334
 RFI, 67
 robots.txt, 79
 rozmieszczenie plików i katalogów na serwerze, 208
 rozszerzenie
 html, 80
 php, 80

S

sabotaż, 53
 safe mode, 54
 Save button at PayPal, 175
 Search, 345
 separator konkatencji, 235
 Secure Socket Layer, 47, 58, 59, 105, 201, 231, 291, 294
 SELECT, 233, 237, 238, 239, 241, 271, 273, 303, 304, 317, 381
 SELECT...INTO, 271
 Selling Preferences, 184
 serwer, 26
 baz danych, 306
 przeciążenie, 226
 bezpieczeństwo, 52, 54
 konfigurowanie, 348
 obciążenie, 119
 organizacja plików, 77
 rozmieszczenie plików i katalogów, 208
 WWW, 16
 odciążenie, 226
 wydajność, 27
 sesja
 identyfikator, 131
 numer, 308
 użytkownika
 personalizacja, 85
 wymuszanie, 131
 zakończenie, 122

SESSION, 295
 session fixation attack, 131
 session_destroy, 122
 session_regenerate_id, 131
 session_start, 85
 set_error_handler, 87
 setcookie, 123
 SHA, 105
 SHA1, 105
 SHUTDOWN, 58
 Siege, 40
 silnik bazy danych, 37
 siła hasła, 113
 siłowy atak, 65
 SIM, 290, 336
 Simple Checkout Server Integration Method, 290
 --skip-name-resolve, 57
 --skip-networking, 57
 sklep
 odpowiedzialność za nieprawidłowe działanie, 21
 projekt, 197
 skalowalność
 witryny 200
 obrazka, 362
 składanie zamówienia, 293, 295
 zarządzanie procesem, 295
 skrótu algorytm tworzenia, 105
 skrypt
 nastuchujący, 185
 PHP, 108
 przekierowujący, 135
 rejestracji, 117
 zakończenie działania, 107
 SKU, 369, 373
 słownikowy atak, 105
 sprawdzanie rozmiaru pliku, 154
 sprzedawcy interfejs, 290
 sprzedaż
 produktów rzeczywistych, 197
 przedmiotów
 pojedynczych, 168
 wielu, 168
 subskrypcji, 168
 SQL
 Injection, 66
 polecenia, 231
 wstrzykiwanie kodu, 82, 113
 SSL, 47, 58, 59, 105, 201, 231, 291, 294
 aktywowanie, 215
 etapy, 59
 wymuszanie zastosowania, 215
 stan, 301
 magazynowy, 202, 256, 273, 282, 357, 365, 386
 niski, 205
 zarządzanie, 166, 367

stopka, 93
 str_replace, 310
 strip_tags, 65, 136
 stripslashes, 82, 309
 strona
 administracyjna, 348
 główna, 96, 263
 plik widoku, 265
 HTML kodowanie, 81
 ograniczenie dostępu, 106
 plik widoku, 265
 przekierowanie, 99
 sprzedaży, 266
 szybkość ładowania, 140
 tytuł, 91
 wyświetlanie, 143
 Subscribe, 174
 subskrypcja, 174
 substr, 275
 Subversion, 39
 suckerfish, 349
 suma zamówienia, 204
 switch, 156
 sygnatura procedury, 270
 symbol
 zakończenia kwerendy, 240
 zastępczy, 223
 system, 56
 kontroli wersji, 39
 płatności, 31
 kryteria wyboru, 33
 standardowy, 167
 uruchomienie, 345
 zaawansowany, 167
 włamanie, 161
 zaatakowanie, 190
 szablon
 HTML, 88
 strony
 bezpłatny, 36
 komercyjny, 36
 szybkość ładowania stron, 140
 szyfrowanie danych, 58, 59

S

ścieżki względne, 221
 śledzenie prób dostępu, 54

T

tablica, 102, 103
 indeksowanie, 373
 opisu błędów, 111
 podział, 274

Take customers to this URL when they
 cancel their checkout, 176
 finish checkout, 176
 TDD, 40
 tekstowa konsola klienta mysql, 230
 termin
 dostarczenia towarów, 198
 ważności karty kredytowej, 297, 325, 329
 minął, 340
 Test
 Accounts, 173, 174
 Mode, 345
 Test-Driven Development, 40
 testowanie
 jednostkowe, 40
 procedury składowanej, 242
 witriny, 49
 wydajnościowe, 40
 textarea, 103
 TextMate, 16
 Thawte, 60
 thread_cache_size, 38
 timestamp, 119
 TinyMCE, 139
 dodatki, 141
 Toad, 299
 towaru zwrot, 36
 transakcja, 204
 identyfikator, 385
 koszt obsługi, 32
 międzynarodowa, 290
 niezrealizowana, 345
 transport
 koszt, 302, 316
 transfer danych o płatności, 183
 trigger_error, 85, 115
 tryb
 bezpieczny, 54
 testowy, 345
 Turn Test Off, 345
 tworzenia oprogramowania, 38
 typ
 pliku sprawdzanie, 357
 serwera, 215
 tytuł strony, 91

U

UML, 38
 Unified Modeling Language, 38
 Uniform Resource Locator, 291
 UNION, 237, 238, 273, 303, 375, 381
 uniqid, 125
 Unsettled Transactions, 345
 UPDATE, 271, 386
 update_cart, 278

uprawnienia
 administratora, 57
 klienta, 57
 użytkowników, 57
 URL, 291
 uruchomienie, 150
 szkodliwego pliku, 67
 urząd certyfikacji, 215
 Use my secure merchant account ID, 175
 usługa
 dla sprzedawców, 174
 odmowa wykonania, 65
 zarabianie na sprzedaży, 20
 ustawienia podstawowe, 96
 usuwanie kodu
 HTML, 65
 JavaScript, 65
 PHP, 65
 UTF-8, 77, 81
 utrzymanie witryny, 42
 użytkownicy
 grupy, 150
 nazwa, 112, 210, 216
 ochrona, 64
 personalizacja sesji, 85
 zalogowani, 107, 147

V

Verified Merchant Seal, 290
 VeriSign, 60
 Visa, 23
 numer karty, 328
 VPS, 53

W

w magazynie nie ma wystarczającej ilości produktu, 282
 walidacja
 adresu e-mail, 113, 119
 danych, 111
 imienia, 112
 waluta
 przeliczanie kursu, 166
 wartość zamówienia, 377
 całkowita, 302, 331, 336, 385
 warunek kwerendy, 235
 ważność konta, 75
 Website Payments, 167
 Standard, 31, 167
 weryfikacja adresu zamówienia, 34
 What You See Is What You Get, 73, 134
 WHERE, 235

witryna
 administracja, 347
 adres, 84
 bezpieczeństwo, 48
 użytkownika, 49
 czcionki, 35
 dobry wzorzec, 35
 kolory, 35
 nawigacja, 36
 planowanie, 35
 projekt, 35
 skalowalna, 200
 słabe punkty, 63
 testowanie, 40, 42
 udoskonalanie, 43
 utrzymanie, 42
 wydajność, 38, 40, 48
 wygląd, 35
 po zalogowaniu, 36
 zadania, 35
 włamanie do systemu, 161
 właściciela karty kredytowej dane, 50
 województwo, 310, 377
 wskazówki dla hakera, 65
 wskaźnik postępu, 319, 323
 wstrzykiwanie
 kodu SQL, 82, 113
 wszystkie dostępne produkty, 374
 wydajność, 58
 bazy danych, 38
 testowanie, 40
 witryny, 97
 poprawienie, 40
 wzrost, 57, 223, 225, 306
 wygląd
 witryny, 35
 po zalogowaniu 36
 wykrywanie oszustw bankowych, 290
 wylogowanie, 122
 wymuszanie
 sesji, 131
 zastosowania protokołu SSL, 215
 wynik wykonania kwerendy, 234
 wyprzedaż, 203
 wyrażenia regularne, 213
 WYSIWYG, 73, 134
 wysokość opłaty za każdy cykl, 175
 wysyłka
 koszt, 287
 wyświetlanie paska adresu na zielono, 63
 wzrost zagrożenia atakami hakerów, 47

X

XAMPP, 16, 231
Xdebug, 41
XSS, 64

Z

zabezpieczenie
katalogów przed nieuprawnionym dostępem, 210
ogólne ustawienia, 345
przed
błędem, 100
oszustwami, 381
zadania witryny, 35
zakup bez rejestracji, 199
zamówienie, 204
czas złożenia, 204
częściowe, 315
finalizowanie, 48, 283
identyfikator, 301, 377
istniejące, 333
nie może być ukończone, 322
nowe, 333
opłacenie, 343
składanie, 293, 295
wartość, 377
całkowita, 302, 331, 336, 385
weryfikacja adresu, 34
znaki diakrytyczne polskie, 112
zapasy
komunikat o kończących się, 285
zapis, 150
zapobieganie atakom, 65
zapora sieciowa, 50
zapytanie
niepotrzebne, 95
predefiniowane, 66, 222, 224, 359

zarabianie na, 13
linkach, 14, 20
promocji produktów lub usług, 20
reklamach, 14, 20
sprzedaży produktów lub usług, 20
zawartość koszyka, 314, 322
pobieranie, 273
zarządzanie
informacjami o klientach, 290
zawartością, 133
zaufanie klientów, 64
zdalne dołączanie pliku, 67
zdefiniowania bloków kodu w bazie danych, 225
ZenCart, 25, 166
zero-width positive lookahead assertion, 114
zespół programistów, 39
zestaw znaków, 77
zezwolenie
reguła, 211
zmiana
hasła, 127
hostingu, 85
nazwy
domenowej, 85
przesyłanych plików, 67
zmienna
deklarowanie, 270
czyszczenie, 359
nazwy, 119
procedura składowana, 305
znacznik
czasu, 119
zamykający, 83
znak
dolar, 214
klasa, 213
końca wiersza, 86
zestaw, 77
zwyroty, 36

E-commerce

Genialnie proste tworzenie serwisów

w PHP i MySQL

Handel elektroniczny to znakomity sposób prowadzenia działalności zarobkowej. Twoje przedsięwzięcie może odnieść prawdziwy sukces – pod warunkiem że się do niego dobrze przygotujesz. Oto jedna z nielicznych na rynku książek dostarczających szczegółowych informacji na temat tworzenia serwisów e-commerce z wykorzystaniem PHP i MySQL. Bez względu na to, czy tworzysz dynamiczne strony internetowe od lat, czy dopiero od kilku tygodni, z pewnością znajdziesz tu mnóstwo bezcennych informacji.

Dzięki książce „E-commerce. Genialnie proste tworzenie serwisów w PHP i MySQL” dowiesz się, jak zaprojektować bazę danych, generować katalog produktów, zarządzać koszykiem zakupów, obsługiwać zamówienia i płatności oraz sprawić, by Twój sklep internetowy nie wymagał od Ciebie pracy ponad siły, a jednak był dochodowy. Podręcznik zawiera również omówienie tak istotnych zagadnień, jak zapewnienie bezpieczeństwa witryny oraz zadbanie o interfejs przyjazny użytkownikom, a także informacje dotyczące modułowego programowania, gotowego do dalszej rozbudowy. Przytoczone tu praktyczne przykłady pozwolą Ci spojrzeć na systemy e-commerce z możliwie jak najszerszej perspektywy.

- Wybór technologii internetowych
- Struktura i projekt witryny
- Zarządzanie zawartością witryny
- Tworzenie kont użytkowników
- Łączenie różnych systemów płatniczych
- Sprzedaż wirtualnych produktów
- Tworzenie bezpiecznego środowiska serwera i baz danych
- Tworzenie paneli administracyjnych
- Zasady składania zamówień

Magia tworzenia profesjonalnych serwisów e-commerce

helion.pl
księgarnia internetowa

Nr katalogowy: 6274



Księgarnia internetowa:
<http://helion.pl>



Zamówienia telefoniczne:
0 801 339900



0 601 339900



Helion

Sprawdź najnowsze promocje:

🔴 <http://helion.pl/promocje>

Książki najchętniej czytane:

🔴 <http://helion.pl/bestsellery>

Zamów informacje o nowościach:

🔴 <http://helion.pl/nowości>

Helion SA

ul. Kościuszki 1c, 44-100 Gliwice

tel.: 32 230 98 63

e-mail: helion@helion.pl

<http://helion.pl>

Cena 69,00 zł

ISBN 978-83-246-3213-8



9 788324 632138

Informatyka w najlepszym wydaniu