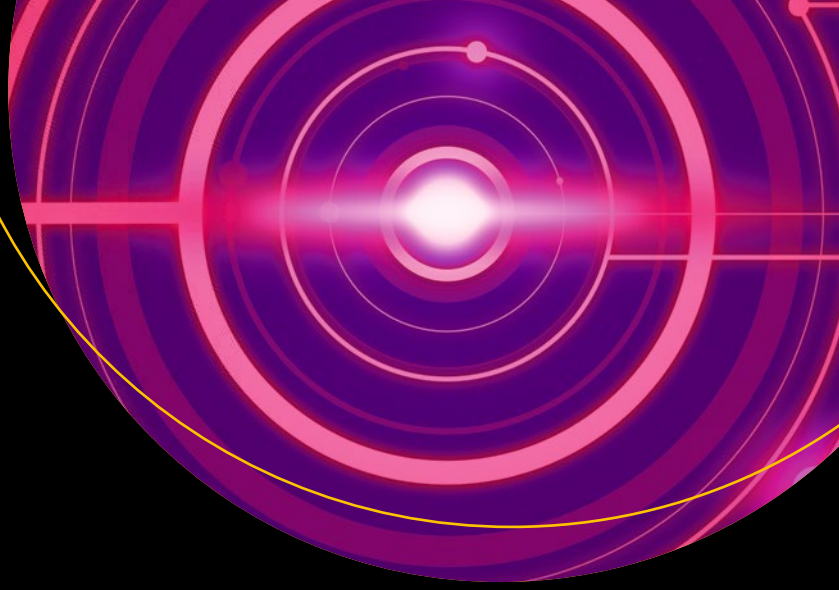

Practical PHP 7, MySQL 8, and MariaDB Website Databases

A Simplified Approach to Developing Database-Driven Websites

—

Second Edition

—

Adrian W. West
Steve Prettyman

Apress®

Practical PHP 7, MySQL 8, and MariaDB Website Databases

A Simplified Approach to Developing Database-Driven Websites

Second Edition

Adrian W. West
Steve Prettyman

Apress®

Practical PHP 7, MySQL 8, and MariaDB Website Databases: A Simplified Approach to Developing Database-Driven Websites

Adrian W. West
Colyton, Devon, United Kingdom

Steve Prettyman
Key West, Florida, USA

ISBN-13 (pbk): 978-1-4842-3842-4
ISBN-13 (electronic): 978-1-4842-3843-1
https://doi.org/10.1007/978-1-4842-3843-1

Library of Congress Control Number: 2018957652

Managing Director, Apress Media LLC: Welmoed Spahr
Acquisitions Editor: Steve Anglin
Development Editor: Matthew Moodie
Coordinating Editor: Mark Powers

Cover designed by eStudioCalamar

Cover image designed by Freepik (www.freepik.com)

Distributed to the book trade worldwide by Springer Science+Business Media New York, 233 Spring Street, 6th Floor, New York, NY 10013. Phone 1-800-SPRINGER, fax (201) 348-4505, e-mail orders-ny@springer-sbm.com, or visit www.springeronline.com. Apress Media, LLC is a California LLC and the sole member (owner) is Springer Science + Business Media Finance Inc (SSBM Finance Inc). SSBM Finance Inc is a **Delaware** corporation.

For information on translations, please e-mail editorial@apress.com; for reprint, paperback, or audio rights, please email bookpermissions@springernature.com.

Apress titles may be purchased in bulk for academic, corporate, or promotional use. eBook versions and licenses are also available for most titles. For more information, reference our Print and eBook Bulk Sales web page at www.apress.com/bulk-sales.

Any source code or other supplementary material referenced by the author in this book is available to readers on GitHub via the book's product page, located at www.apress.com/9781484238424. For more detailed information, please visit www.apress.com/source-code.

Printed on acid-free paper

Contents

About the Authors xiii

Acknowledgments xv

Introduction xvii

Chapter 1: Create and Test a Database and Table 1

Defining Developer, Administrator, and User 2

Defining Interactive Websites 2

Summary 31

Chapter 2: Create Web Pages That Interact with Users 33

Creating the Folder for Holding the Database Pages 34

Creating the Temporary Template 34

Introducing the PHP include() Function 36

The Included Header File 37

The Included Menu File 39

The Included Information Column 40

The Included Footer File 41

The Interactive Version of the Template 42

Connecting to the Database 44

The Registration Page 47

The PHP Keyword echo 58

The “Thank You” Page 58

Displaying Error Messages That Are Collected in an Array 60

Hashing the Password 61
Viewing Members' Records 61
The View Users Page 61
The Change Password Page 65
Confirming a Successful Password Change 72
Testing the Tutorial's Pages 72
More About Arrays 73
Summary 74

Chapter 3: Create Login/Logout Functionality for Members and an Administrator 75

Creating the logindb Database and users Table 76
Removing or Replacing Redundant Menu Buttons in the Headers 77
Adding a Login Button to the Home Page Header 78
Removing Redundant Buttons from the Registration and New Password Headers 79
The Revised Registration Page 79
The New Header for the New Password Page 80
A New Header Menu for the Members page 82
Amend the Header for the "Thank You" Page 82
The Registration Page and Undesirable Characters 84
Registering Some Members 87
Differentiating Between Two Types of Membership 88
Creating User Levels to Limit Access to Private Pages 88
Log In 91
The Header for the Login Page 91
The Login Page 92
Sessions 99
A Members-Only Page 101
Planning the Administrator's Role 104
A New Header for the Administration Page 105

Testing the Login/Logout Function 109
Amending and Deleting Individual Records 109
Summary 110
Chapter 4: Create an Administration Interface 111
The Administration Database 111
The Users Table 113
Revising the View Users Page to Include Editing and Deleting 115
Displaying Pages of Records (Pagination) 121
Planning the Search Criteria 128
A Temporary Search Page for Displaying Specified Members 128
The Search Form 132
The Final Form Handler for Receiving Search Form Input 135
Editing Records 139
Deleting Records 145
Summary 150
Chapter 5: Expand and Enrich Your Website 151
Creating a New Database, a Table with 15 Columns, and a Price Table 151
Creating the File for Connecting to the Database 152
Creating the Tables 153
Using ENUM 154
The Importance of Documentation 154
Extending the Registration Form and Adding a Pull-Down Menu 156
Always Announce Prices and Fee Payments Up-Front 157
Adding PayPal Debit/Credit Card Images 168
Including PayPal on the “Thank You” Page 168
Registering Some Members 173
A Small Amendment to the Login Page 175
Amending the Administrator’s Header 175
Adding Class and Paid to the admin_view_users Table 177

Searching and Editing Records 181
Modifying the Form for Editing Records 184
Summary 190
Chapter 6: Add the Finishing Touches: Security and Validation 191
Creating the Database 191
Creating the File for Connecting to the Database 192
Creating the users Table by Importing a SQL Dump File 193
Registering Some Members Manually 195
Adding a Title Column to the users Table 197
Creating the Prices Table by Importing a SQL Dump File 199
Tidying Up the Folders and Filing System 200
Degrees of Security 200
Adding a Layer of Security 200
An Increased Layer of Protection 202
Validation and Sanitization 202
The filter_var() Function 202
Validation 202
Sanitization 204
Validating Telephone Numbers 205
A Safer Registration Page 205
Searching for a Title, an Address, or a Telephone Number 221
Viewing the Retrieved Title, Address, and Phone Number 225
Editing the Title, Addresses, and Telephone Numbers 228
Summary 236
Chapter 7: Migrate to a Host and Back Up Your Website Database 237
Making Last-Minute Changes 237
Creating a New Database 238
Allowing Members to Update Their Own Records 240
A Secure Feedback Form 254

The Feedback Form 255
The "Thank You" Page and the Error Messages 266
A Common Header 269
Logging Exceptions and Error 272
Migrating the Database and Tables to a Remote Host 274
A Puzzling Error Message 275
Creating and Exporting the SQL File 275
Using the GUIs on a Remote Host's Server 281
Connecting to the Database on the Remote Host 282
Securely Uploading the *mysqli_connection.php* File 283
Uploading the Interactive Pages to the Host 285
Backing Up Your Database 286
Summary 286
Chapter 8: Create a Product Catalog 287
Preparing the Database and Administration Plan 287
Creating a New Database 288
Creating the File for Connecting to the Database 289
Security 290
Creating a Home Page with Search Capability 292
The Header for the Majority of the Pages 293
The Home Page Code 294
Displaying the Catalog 296
The Header for the Page of Search Results 303
Creating the House Details Page 304
Creating the Admin/Adding a House Page 309
The Header for the Administrator's Page 318
Administrator's View of the Entire Stock of Houses for Sale 320

The Administrator's Search Page 325
The Result of a Search 327
The Contact Us Page 328
Summary 334
Chapter 9: Join Multiple Tables and Other Enhancements 335
Introduction to Multiple Tables 335
Normalization 336
Creating the Database and Tables 337
Viewing the Connection File 338
Preparing the Tables for Joining 339
Populating the Two Tables 340
Joining Data from the Two Tables 342
Creating the Third Table 345
Creating Pages to Display the Data from Your Joined Tables 347
The Home Page 347
The Main Menu for the Pages 349
The Header for All the Pages 350
The Page for Viewing the Birds 351
The Page for Viewing the Locations and Habitats of the Reserves 355
Payments by Check 366
A Choice of Payment Method 367
Printing Online Forms 375
Summary 376
Chapter 10: Create a Message Board 377
The Plan 377
Creating the Database 378
Creating the Tables 378
Creating the Second Table 379

Creating the Registration Form 385
The "Thank You" Page 393
Populating the Members Table 394
The Login Page 395
Logging Out 400
Creating a Gateway to a Choice of Quotes 400
The Form for Posting Quotations 402
Posting Some Quotations 408
The Comical Quotes Page 409
The Wise Quotes Page 413
Adding Search Facilities 417
The Header for ViewPosts.php 420
Searching for Specific Words or Phrases 421
The Full Text Search Form 422
Displaying the Search Results 424
The Header for the quotes_found Page 427
Enhancing the Message Board 428
Converting the Message Board to a Forum 428
Summary 429
Chapter 11: E-commerce: A Brief Introduction 431
Security Warning 433
The Plan 433
Creating the PayPal Cart Site 434
Creating the PayPal Cart Database and Tables 434
Viewing the Connection File 435
Populating the PayPal Cart Tables 435
The PayPal Cart Home Page 438
Using the PayPal Cart Home Page to Search for Paintings 441
Integrating with the PayPal Shopping Cart Buttons 444

Creating a Custom Cart 450
The Custom Cart Home Page 452
Create the Custom Cart Database and Tables 452
Retrieving a Forgotten Password 460
The Custom Cart Search Page 464
Adding Paintings to a Table for a Custom Shopping Cart 465
The Checkout Page 477
The Additional Administrative Tasks 477
Summary 478
Chapter 12: Take a Brief Look at Oracle MySQL 8 479
Advantages of Upgrading 479
Installing MySQL 8 Community Server 481
Exploring the Features of MySQL Workbench 488
Connecting PHP 7 to the MySQL 8 Community Server 495
Migrating to MySQL 8 Community Server 498
Using Our PHP Files with MySQL 8 Community Server 502
Summary 505
Appendix A: Troubleshooting 507
Browser Quirks 508
Tables Not Displaying 508
A Style Change Has No Effect 509
Included Items Missing from the Display 509
A Page Fails to Validate 509
A PayPal Pull-Down Menu Does Not Work 509
Access Denied 509
PHP Error Levels 510
Call to an Undefined Function 510
Cannot Redeclare Function 510

Undefined Index or Undefined Variable 510
Empty Variable Value 511
Headers Already Sent 511
Blank Screen 511
Unexpected End of File in Line xxx 511
Parse Error and Unexpected Characters 512
Unexpected T_STRING 512
Unexpected T_ELSE 513
Wrong Equal Sign 513
Failed to Open Stream 513
Syntax Errors 513
Warning: Division by Zero 514
Display Is Not What Was Expected 514
Reference to a Primary Key Could Not Be Created 514
Element <style> Not Allowed as Child of Element 514
Problem with Prepared Statement 514
Logical Errors 515
The Internet Is Your Friend 515
Summary 515

Appendix B: Resources 517

PHP Quick Reference 517
Arrays 517
Bootstrap Template 533
MySQL and phpMyAdmin Quick Reference 534
date_time 535
INSERT 536
SELECT 536
UPDATE 536

What Next? 537
Resources 537
Books on PHP and MySQL for Databases 537
PHP and MySQL Internet Resources 538
E-commerce Resources 538
Summary 539

Index 541

About the Authors

Adrian W. West resigned as a chartered design engineer to become the UK director of a correspondence school. He has been teaching in one form or another since 1982. He introduced computers into his workplace in 1987 and taught the staff how to use them. For four years, he taught undergraduates computer skills at a college in Cheshire in the United Kingdom.

Adrian lives in Colyton, a town in Devon, England, and for the last 18 years, he has designed and produced websites for UK businesses and charities.

Adrian is the author of three books published by Apress: *Practical HTML5 Projects*, a book of tips and tricks (now rather out of date); *Practical PHP and MySQL Website Databases* (the first edition of this book); and *Practical Web Design for Absolute Beginners* (published in 2016).

Steve Prettyman earned his Bachelor of Arts degree in secondary education from Oglethorpe University in 1979. He quickly began his teaching career as a high school mathematics instructor while continuing his education by earning a master's degree in business information systems from Georgia State University (1985). Since then, Steve has spent more than 30 years in the IT industry. The last almost 20 of those he has been an instructor and professor at Chattahoochee Technical College, Kennesaw State University, and Southern Polytechnic State University. He is currently the Computer Science Department chairperson for Florida Keys Community College in Key West, Florida. His primary teaching responsibilities include programming, web design, and web application development.

Acknowledgments

I thank my wife, Janice, for her love, support, and encouragement, and for taking over my share of the chores so that I could concentrate on this edition.

My thanks go to the team at Apress and to all the people in Internet forums who helped me and replied to my queries.

My special thanks go to my co-author, Steve Prettyman, who converted the first edition code and instructions to the latest versions of PHP 7 and MySQL/MariaDB for this second edition.

—Adrian W. West

I thank my partner and wife for her love and support for almost 25 years. Her reminders that I needed to get my head out of the book and take a break to enjoy life were vital in this process. My children also stared at me when I ignored them for too long to let me know that I should stop writing and coding to take them outside to swim in the pool. Can we say that Pixee and Buster, our four-legged children, are a little spoiled?

I especially want to thank Adrian W. West and Apress for allowing me to update his successful first edition to provide a modern-day approach to his logical pattern to teaching interactive database programming using PHP.

—Steve Prettyman

Introduction

What's New in This Edition?

The code and instructions in the first edition of this book (written in 2012) were made obsolete by new and very different versions of XAMPP, EasyPHP, phpMyAdmin, PHP, and MySQL/MariaDB. This second edition contains new code and instructions to match the latest versions of the software.

With the massive increase in cybercrime and other cyber threats, this new edition has been fortified with a much stronger emphasis on security. This book takes the approach of sanitizing any data that has been accepted from any outside source and, additionally, sanitizing any data before it is displayed on a web page. Most of the examples use prepared statements that ensure that any externally accepted data cannot be executed and therefore cannot cause security vulnerabilities.

We have chosen Bootstrap to provide responsive web design (RWD) for each of the book's example websites. While Bootstrap provides the CSS and JavaScript to format the examples in this book for any size device, you can easily reformat these examples with your own CSS code if you desire.

This new edition also takes a brief look at Oracle's MySQL 8. A comparison is provided of the tools available in each version. Step-by-step procedures provide you with the ability to upgrade to MySQL 8.

The Teaching Method

This book uses a different way of teaching website database design compared with the majority of manuals. The usual layout starts with several lessons on PHP followed by snippets of code and may eventually conclude with a project or two. This book abandons that approach. The primary focus is on fully worked, practical MySQL/MariaDB database projects built into real-world web pages.

Instead of presenting PHP, SQL, and MySQL/MariaDB as completely separate topics, they are explained in the context of each project. However, you will find a useful quick reference of PHP syntax in Appendix B.

In this book, practical databases and interactive web pages are presented as early as possible; in fact, you will create a database and a table in the first chapter. In the second chapter, you will embed a database into an interactive web page and test it. Each subsequent chapter will introduce you to increasingly sophisticated and useful database-driven website pages.

We assume that you have little or no knowledge of PHP and databases. This book will demonstrate:

- How to create a free environment for testing database-driven web pages.
- How to embed PHP and interactive databases into real-world web pages. This is the primary theme throughout the book.
- How PHP, HTML, and MySQL/MariaDB work together for creating and maintaining a database and its data.
- How to create a user-friendly interface so that an administrator with minimum computer skills can monitor the database.

Because databases need to be viewed and tested on a server, the first part of Chapter 1 has instructions for using a free server that can be downloaded and installed on your computer. This ensures you will have a safe development platform for learning and testing as you explore the book's practical projects.

Starting with a separate study of PHP theory and syntax can deter learners and prolong the time until they get their hands on a practical application. Learners are enthused when they achieve something. This book jumps into the database driving seat right from the beginning. Essential PHP and MySQL/MariaDB techniques are presented in context within each tutorial where they are most relevant.

Who Is This Book For?

The book assumes you are thoroughly familiar with HTML5 and CSS3. However, we assume you have no knowledge of MySQL/MariaDB, PHP, and phpMyAdmin. As the chapters unfold, you will progress from intermediate level to advanced level.

You do not need to acquire an extensive knowledge of PHP to create interactive databases. We introduce all the PHP you will need in the appropriate place within each project. Each piece of PHP code is explained fully in plain English. The step-by-step, fully worked examples will show you what MySQL/MariaDB and PHP can do and how to do it. This book is for web designers who want to begin developing database-driven websites.

With this in mind, this book uses a highly motivational step-by-step approach. We recognize fully that a sense of achievement encourages readers to look forward eagerly to the next step. The book will teach enough PHP and MySQL/MariaDB to complete all the projects in the book. Web developers who have not kept up-to-date with MySQL/MariaDB and PHP will also benefit from this approach. College and university programming instructors will find that this book provides an excellent text, and the projects can form a basis for students to adapt for their course work.

The "Quick and Easy to Learn" Myth

Books frequently state that PHP and MySQL/MariaDB databases are easily and quickly learned, but this discourages beginners, because when they are confronted with the inevitable difficulties (and error messages), they begin to think that they will never grasp even the basic principles.

Beginners should not be discouraged if they remember the following fact: authors claiming that PHP and MySQL/MariaDB are easily and quickly learned have probably been using PHP and MySQL for more than a decade, and they have forgotten the difficulties they encountered when they first began.

If you accept that some time and effort are required to learn PHP and MySQL/MariaDB, then as you work through the book, it will become increasingly apparent that you are learning something very worthwhile. So, have patience and persevere, and you will then begin to enjoy mastering this valuable discipline.

The Origin of This Book

Most of the PHP/MySQL/MariaDB books tend to demonstrate the author's deep and extensive knowledge of PHP and MySQL/MariaDB instead of teaching how to embed MySQL/MariaDB databases into web pages. In contrast, this book uses fully worked examples to demonstrate how to integrate databases into a website.

The boatload of PHP/MySQL/MariaDB database books that this book's authors own (or borrowed) were unnecessarily complicated. The authors of these books had become used to using neat tricks and shortcuts that were second nature, but these cluttered the code and made it difficult for beginners to discern the essential structure.

This book avoids this mistake; a few useful tricks are introduced gradually and are fully explained in plain English. This book is based on a quote from the composer Brahms.

It is easy to compose but wonderfully hard to let the superfluous notes fall under the table.

Almost all the PHP/MySQL/MariaDB books were written backward; they grind away for chapter after chapter with PHP functions and statements (yawn), and then they add the MySQL/MariaDB bit. *Practical PHP and MySQL Website Databases* explains the necessary PHP and MySQL/MariaDB topics in context within each database tutorial.

MySQL/MariaDB books are nearly always written assuming that the web designer will administer the databases. However, small e-commerce websites, clubs, and societies cannot afford to do this and would prefer that their membership secretary be able to administer the database using a user-friendly interface. The majority of the databases created in this book can be administered by both an unskilled membership secretary and the web designer.

Eventually, Adrian W. Wood concluded that he must write his own manual based on what he could learn by concatenating snippets of information from multiple resources. He also based the manual on his own trial-and-error approach as a raw beginner. This automatically ensured that the manual's content was presented in simple, logical, and progressive steps without suddenly introducing unexplained items.

The homegrown manual was so useful that he decided that it should be shared with other website designers; the first edition of this book was the result of that decision.

Computer software and database techniques are constantly improving and updating. Because of this, we have researched the latest versions of the scripts, tools, and the available software. This ensures that, in this second edition, the content and illustrations will remain relevant for as long as possible.

Following the tutorials in this book requires an absolute minimum of software. Some manuals ask readers to download and learn a new piece of software before they can proceed to each new chapter. In fact, Adrian came across one book that required readers to download MySQL, Apache, PHP, phpMyAdmin, Prototype 1.5, Scriptaculous, Zend Framework, Smarty Template Engine, FCK editor, jQuery, and Ajax. In this book, besides a code editor, the software required is limited, as described next.

What Equipment Is Required

The book assumes that, as a web designer, you will already have an HTML editor such as Blue Griffon (free) or Notepad++ (free).

You will need:

- A notebook (real or electronic) for recording the passwords and file names for your databases and table entries. *Don't rely on memory. Write everything down.*

You will need to download:

- The sample code from the book's page, available at `www.apress.com`
- XAMPP or EasyPHP, which are free, all-in-one packages for testing your work
- The latest browsers (all free): Microsoft Edge, Mozilla Firefox, Safari, Chrome, and Opera

The Conventions Used in This Book

Care has been taken to relate every listing to its screenshot. For instance, Figure 3-6 will be described by Listing 3-6. If two listings are needed, such as the HTML code and the PHP code, both will relate to the screenshot by using *Listing 3-6a* and *Listing 3-6b*. If Figure 4-6 does not need a listing, the next screenshot and listing will use Figure 4-7 and Listing 4-7.

Special tips, notes, and warnings are shown in the following format:

Note Security is important when dealing with databases, especially if they contain personal data. The technique for making your work is secure is woven into each step of the instructions.

All code listings use HTML5 and PHP 7; some meta description and meta keywords have been omitted from each <head></head> section to save space.

Code listings are shown as follows:

```
<div id='container'>
<?php include('header.php'); ?><!--include the new header file-->
```

Code lines are sometimes numbered to help with the explanations as follows:

```
if (empty($errors)) { // If no problems occurred, register the user in the database    #1
```

The line numbers are for explanation only and do not need to be included in your own code.

Interactive vs. Dynamic

Most manuals use the term *dynamic* web pages when referring to interactive pages. The words *dynamic* and *interactive* both describe pages that provide a live link between a user and a database. For instance, a user can register for membership and view account details. A membership secretary can view a table of members, but the table is hidden from ordinary members. Because the word *dynamic* can have so many connotations and meanings, we have chosen to use the more precise term *interactive* in this book.

Source Code

You can access the example code used in this book by navigating to www.apress.com/9781484238424 and clicking the Download Source Code button.

CHAPTER 1

Create and Test a Database and Table

This chapter introduces the concept of a database and a practical way of testing it. Using the examples, you will create a MariaDB or MySQL database and a table. As you work through the examples, you will become familiar with the database administration interface.

After completing this chapter, the student will be able to

- Define and design a database and table
- Install and use a WAMP package
- Use phpMyAdmin to create a database and table
- Secure phpMyAdmin and databases with a user ID and password
- Delete a database and/or table

Databases can be used to store products, details of customers, records of members of a society or a club, and much more. They can store names, passwords, addresses, e-mail addresses, registration dates, blog entries, and telephone numbers. Databases can be regarded as folders containing tables of data. The table of data has *columns* and *rows*; the rows in database tables are called *records*. Table 1-1 shows a typical database table.

Table 1-1. *A Typical Database*

user_id	first_name	last_name	email	password	phone
1	Kevin	Kettle	kev@kettle.com	K3ttl3fur	305 111 1111
2	Susan	Saucepan	sue@kitchen.org.uk	N@sus5	01111 222 1111
3	Oliver	Oven	oliver@cokker.co.uk	H0tst0v3	03333 111 4444

A. W. West and S. Prettyman, *Practical PHP 7, MySQL 8, and MariaDB Website Databases*,
https://doi.org/10.1007/978-1-4842-3843-1_1

Defining Developer, Administrator, and User

In this book, the term *developer* (aka *webmaster*) means the person who designs and produces the database; they will integrate the database into a website. Sometimes the term *webmaster* or *web designer* may be used. When it is used, it usually means the same thing as *developer*. The words *administrator* and *membership secretary* have the same meaning in some of the book's tutorials, which are based on building a database for a club. The word *administrator* means the person responsible for monitoring and maintaining the content of the database tables. Clearly, one person can be both a developer and an administrator. However, most developers will maintain the structure of a database but will not want the hassle of amending and deleting records; that should be the role of an administrator (such as a club or society's membership secretary).

The *user* is any member of the general public viewing and possibly interacting with a website database. For security reasons, users have extremely limited access to the database; however, they will be allowed to register for membership, log in to a special section, or change their password.

■ **Caution** The organization commissioning a database must conform to the rules and laws of the country in which the database is developed and resides. In the United Kingdom, the Data Protection Act for the territory in which the database was developed must be followed. This is especially important if that data is going to be used for profit. This may require obtaining a license. In addition, the developer and administrator must normally sign a document confirming that they will never disclose the details of persons recorded in the database. The UK Information Commissioner's Office (ICO) requires an annual license fee based on the revenues of the organization that owns the database. There is no equivalent law in the United States, but privacy laws can differ between states. It is essential that you understand and obey the data-protection laws for your client's territory.

Databases that are accessible to multiple countries will need to meet the requirements for each country. Sometimes this requires the creation of different versions of the database and/or website, when otherwise it would not be required. Many laws and regulations lag behind the overall need to provide the most secure database environment possible. The developer should always err on the side of the "most secure" when designing and using databases. Databases used for educational purposes only (such as the databases shown in this book) do not have to meet the requirements and regulations (such as licensing).

Defining Interactive Websites

Interactive websites are often called *dynamic* websites; however, this book uses the word *interactive* because *dynamic* can signify so many things. For instance, it can mean moving, powerful, eye-catching, flashy, exciting. To a beginner, none of those meanings defines a web page that interacts with a user.

Dynamic is often used to mean exciting, but there is little excitement to be seen in an interactive registration form. *Dynamic* is also a musical term meaning changes or variations in loudness or speed. If *dynamic* can refer to change, why were dynamic templates designed to provide consistency from one web page to another? The term *interactive* has one clear meaning and will be used from now on in this book.

MariaDB and MySQL (with PHP) allow users and administrators to interact with a database using website pages. For instance, users can register as members of an organization via a registration page on a website. Users will be able to supply their personal data for the membership tables. The database management system then enters the users' input into the administrator's tables automatically; this lightens the workload of the administrator. The website's registration page can be programmed to filter users' data input and verify it. From an interactive page, users may even be allowed to update their own records in a database.

Interactivity means that the administrator's workload is greatly reduced, but not completely. For instance, if the database is for a bookshop, the administrator will still have to enter any new titles and prices. On the other hand, an interactive database can be programmed to alert the administrator when the stock of a certain book needs replenishing.

In Chapter 2, you will learn to develop a simple interactive website.

Using MariaDB or MySQL Only for Interactive Database Tables

A noninteractive data table means that only the administrator can enter or amend the table's information. A noninteractive data table would be more easily created and administered using a spreadsheet, such as Microsoft Excel. However, website users cannot interact with such a data table. Employing a MariaDB or MySQL database management system (DBMS) to create a noninteractive (static) version of the data table would be like using a sledgehammer to crack a nut. Website users would have no input and cannot search or update data.

Using MariaDB or MySQL DBMS for a noninteractive version would not reduce the workload of an administrator; they would have to enter all the members' data and verify that the data is genuine.

■ **Note** A few interactive web pages do not need a database to function. For instance, a Contact Us form can be regarded as interactive because it takes a user's input and transmits it to the website's owner via a PHP form handler via an e-mail; this can be achieved easily without a database. In this book, the term *interactive* always means the user can interact with a database.

Methods for Developing and Maintaining Databases

The four methods for managing databases are as follows (with the easiest method first and hardest last):

- phpMyAdmin (or other administrative tool)
- PHP
- SQL scripts
- Command line

In this book, we will be mainly using the first two methods (phpMyAdmin and PHP). We will use SQL scripts and the command line to briefly demonstrate how to create, update, and distribute your database. For interactive databases, you will need some PHP files. You do not need an extensive knowledge of PHP before you can create interactive databases. We will introduce the PHP syntax you require in the appropriate place in each project—that is, in context. The step-by-step, fully worked examples will show you what MariaDB and MySQL can do and how to do it.

Because of their popularity, graphical user interfaces (GUIs) have been developed to facilitate the task of developing and maintaining databases. These administration tools are part of development packages that include web servers (Apache), databases (MariaDB or MySQL), and programming languages (PHP). In this book, we will introduce two of the most popular packages: XAMPP and EasyPHP.

A Brief Look Inside Web Server Communication

Databases need a server, a DBMS, and a PHP processor, as shown in Figure 1-1. These can be downloaded as an all-in-one, already configured package. The testing and development of the projects in this book are based on the free XAMPP and EasyPHP packages.

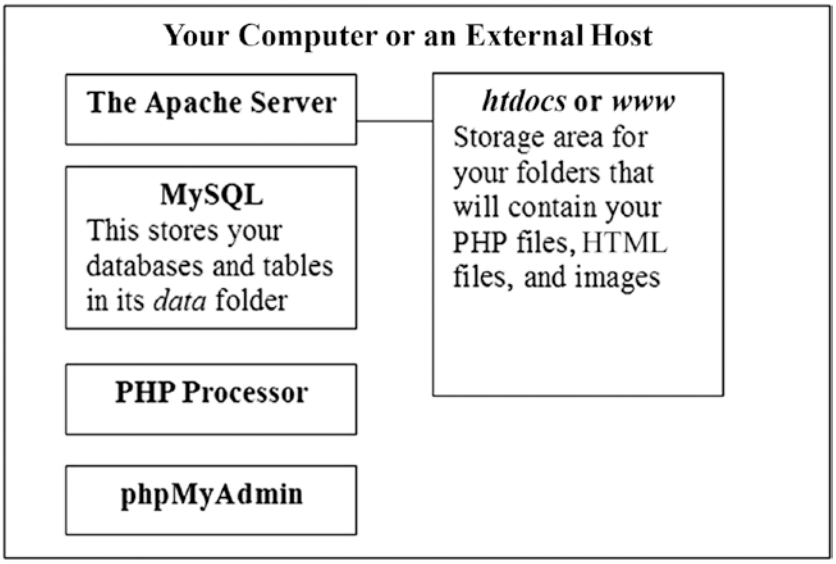

Figure 1-1. *Web server communication*

Figure 1-1 shows the main components built into the XAMPP and EasyPHP development platforms. They are as follows:

- Apache is the web page server used by the great majority of hosts and for web development on local hosts (user computers). Web servers determine whether web pages contain PHP code by looking at the file ending (*.php*). If PHP code exists, the code is passed to the PHP interpreter for execution. If the PHP interpreter determines database-related code or SQL exists within the PHP program, this information is passed to the MySQL or MariaDB database management system. The database management system executes the SQL statements and returns the results to the PHP interpreter. The PHP interpreter completes executing the PHP code and returns the results to the Apache server. The Apache server gathers the results of the PHP code, along with any HTML, CSS, and/or JavaScript, and sends this information to the web browser on the user's machine. The web browser then executes the HTML, CSS, and/or JavaScript code.
- MySQL and MariaDB are not just databases; they are also database management systems. These packages include tools to create, manage, and secure databases.
- The PHP processor interprets and executes any PHP code. It will throw errors or exceptions if there are syntax (coding errors), logical errors, or system errors (out of memory).
- phpMyAdmin is a GUI-based administration tool for creating and maintaining databases and their tables.

A single all-in-one package such as XAMPP or EasyPHP contains the four programs mentioned in Figure 1-1 and is referred to as a WAMP (Windows, Apache, MySQL/MariaDB, and PHP). In WAMPs, the main components are preconfigured so that they can talk to each other. The equivalent on an Apple OS is MAMP, and on a Linux computer it is LAMP.

The folder *htdocs* (or *eds-www*) is the default storage and executable area for your web pages. Apache, by default, looks in *htdocs* (or *eds-www*) for your web pages. These pages may be designed to allow users to interact with the database. Other pages will operate unseen by the user to transmit information back and forth between the browser and database. The pages are usually HTML and PHP files or a combination of both.

■ **Caution** Everything inside Figure 1-1 may be already installed on a remote host, but you should never use a remote host to create a database while you are learning. For security reasons, do not use a remote host until you have become proficient. We recommend you learn and develop a database using an all-in-one package on your own computer. Note that an all-in-one package installed on your own computer is purely a development tool. The database, when developed and thoroughly tested, can eventually be uploaded to a host to make it available to users.

A Free Development Platform for Testing

You will not be able to test your work in the normal way—that is, by using a browser to view a database and the PHP code located on your hard drive. When a web page containing PHP is requested by the browser, the PHP code is executed in the web server. The results of the executing of the code will display in the browser. If you view the browser code (right-click the web page and select View Source), you will not see the PHP code. You will see the results produced by the execution of the code. This confuses beginning developers who are used to seeing any HTML, CSS, or standard JavaScript code they have created in the browser. Remember, HTML, CSS, and standard JavaScript are executed in the browser. PHP code is executed in the web server, even when that server is on your own PC.

Using an all-in-one package on your computer will allow you to see all the code you create and the results of the execution of this code on your PC. This book assumes that you will use a package on your own computer while you are learning and for developing future database-driven websites.

■ **Note** In the current world of hackers who attempt to corrupt or gather personal, corporate, and government data, all web pages on Internet-hosted websites *must* be secured. The earliest projects in this book are necessarily simple and have some secure features. However, it is not recommended that you upload these pages to an Internet host. When you have gained experience and confidence, and you are sure that you understand how to secure websites and databases, you can adapt the book's later projects for use in your own websites and then upload them to a remote host.

Using XAMPP on Your Own Computer

The XAMPP package is free and is preconfigured so that the components will talk to each other. This eliminates the hassle of the usual practice of downloading several individual components and then configuring them to work together. XAMPP includes packages for Windows (WAMP), Linux (LAMP), and the Apple (Mac) OS (MAMP). The examples in this book are developed using a WAMP environment. However, these examples should also work in a LAMP or MAMP environment.

At the time of writing, the most recent version of XAMPP is version 7.2.4. This version is used throughout the book. It has component versions as follows: Apache 2.4.33, MariaDB 10.1.25, PHP 7.2.4, and phpMyAdmin 4.8.0, along with other tools. The examples in this book are not totally compatible with PHP versions before 7.0.

■ **Note** If you are installing a version that is newer than demonstrated in this book, the installation steps may be slightly different. You should refer to the XAMPP site (www.apachefriends.org) or search for installation instructions on the Internet (try youtube.com) for any version not demonstrated in this book. If the newer version includes a major upgrade (for example, PHP 8), some code changes may be required. Check the Apress website for code changes related to major version releases.

Before we give you the instructions for downloading XAMPP, we need to settle a question that bothers every beginner concerning the transferring of a developed database from XAMPP or EasyPHP to the remote host. If you use one of these packages on your own computer, a question will arise, as stated in the title of the next section:

Will I Be Able to Transfer the Database from XAMPP or EasyPHP to a Remote Host?

The main thought that haunts a beginner is this: "If I develop a database on a local WAMP, will I be able to move it easily to a remote host?" Beginners have every reason to be worried because most manuals rarely give even a hint on this topic. However, the answer is this: "Yes, you will be able to move the database." You will find full instructions later in this book.

Now we will provide the information for downloading and installing XAMPP.

■ **Caution** Should you want to install multiple free WAMPs, it is possible to install both EASYPHP and XAMPP on the same computer. However, make sure one of them is shut down before opening the other; otherwise, they will fight for the same ports and cause annoying problems.

Downloading and Installing XAMPP

XAMPP needs minimal configuring. To download the package, go to www.apachefriends.org/.

Click Download from the menu at the top, as shown in Figure 1-2, of the main page. The download page will then display the current versions available. If it's available, select the 7.2.4 version of XAMPP to ensure that all examples you use from this book will be compatible. If there is a minor release of version 7 (such as 7.3.0), it probably will still be compatible. If version 7 is not available, download the latest version. Then check the Apress website to see whether there are any coding changes required to the examples presented in this book. The download page varies from time to time, so you may have to explore the page to find the version shown.

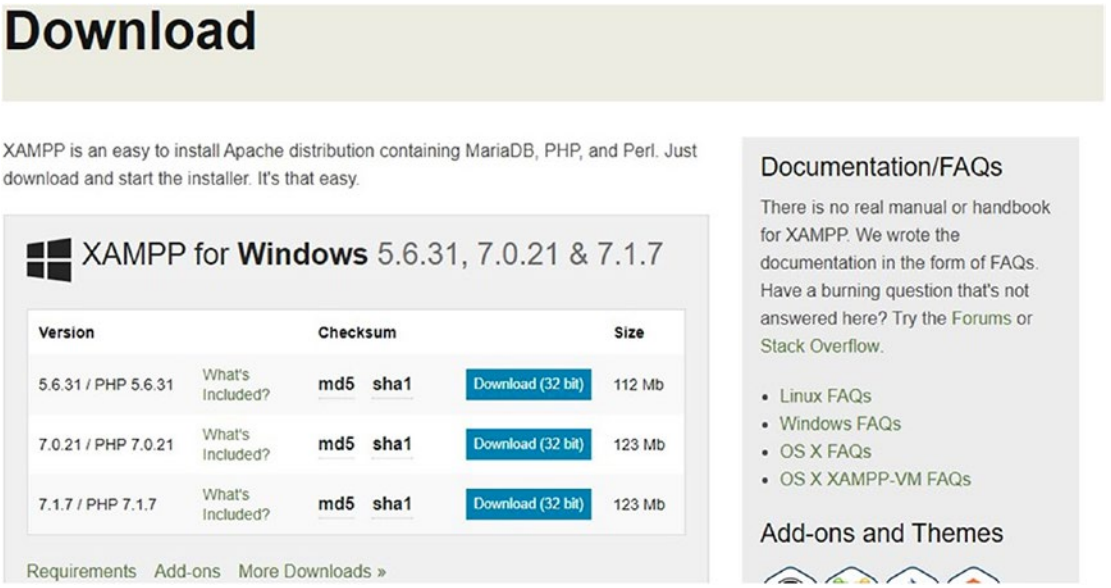

Figure 1-2. *Installing XAMPP*

■ **Note** The download page will state that you must have the C++ runtime libraries installed. XAMPP and its related tools are created using C++. If you are using a current version of Windows, you probably already have the correct version of C++. However, if you receive an error indicating that it is missing, follow the link the error provides to download the correct version.

Click the Download button next to the most current version. This will begin the download process. The file will automatically download to the download folder on your PC. Depending on which browser you are using, you could be asked if you want to run or save the program. Some browsers will automatically save the program and require you to click the file to run it. If you are unaware of how to find downloaded files for the browser you are using, search the Internet for directions. Once the file has downloaded, start the setup process by double-clicking the file (or clicking the Run option if prompted).

The environment will prompt you to ask permission to install the program. Click Yes to continue the process. The setup program will determine the security settings on your machine. It may provide a warning that your security might restrict some of the XAMPP options available. You can choose to either continue the installation or stop the current installation and temporarily turn off your security program. If you turn off your security, make sure your PC is no longer connected to the Internet. The setup program will also look at your User Account Control (UAC) settings. These settings limit access to folders and files based on the user ID in which you are signed into the system. If you are using a computer in which you do not have administrative rights, the setup routine will provide a warning message prompting you to install your program in a different location than the default (program files). Do as the prompt suggests and change the default settings when prompted to a different location, such as *c:/XAMPP*.

A Welcome to XAMPP Setup Wizard screen will appear. Just click Next to continue the setup. The next screen will display all the options available for installation. If you are a beginner, just accept the items already checked by clicking the Next button. Intermediate and advanced users might consider removing items that you are sure you will not use. You can always install them later if you need them. The next screen will show the installation location. Use *c:/XAMPP* to avoid any possible security access issues. Click Next.

The next screen will try to convince you to also look at installing CMS programs (such as WordPress). For now, uncheck the Learn More box. You can install it later if you are interested. Click the Next button. The next screen will indicate the setup is ready. Click Next. The installation will begin. By default, the Start Control Panel check box will be checked. Leave it checked. Click the Finish button. You may be prompted with a language selection prompt before the control panel displays.

If you have errors during installation, copy the error message and paste it into a search engine (like Google). Look at the listing of suggested solutions. Go to a trusted website and follow the directions to correct your error.

The items on the XAMPP control panel labeled Running usually appear automatically, and you will then be able to stop the various modules. If they do not start automatically, click the Start buttons on the XAMPP control panel for Apache and MySQL. If a button says Stop, that module is already running. What next? If you are asked about running the modules as services, choose to run Apache and MySQL as services, and then those modules will automatically start when you double-click the XAMPP desktop icon (Figure 1-3).

Figure 1-3. *The XAMPP icon*

Create a shortcut on your Desktop for XAMPP's *htdocs* folder, and place it alongside the XAMPP icon, as shown in Figure 1-4. Use this shortcut for loading your PHP files into the *C:\xampp\htdocs* folder.

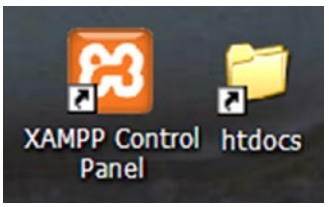

Figure 1-4. *Time-saving shortcuts*

If a desktop icon was not created during the installation, we recommend you go to the *C:\xampp* folder and then create a desktop shortcut for the *xampp-control.exe* file.

For maximum convenience, put the two desktop items side by side, as shown in Figure 1-4. One icon starts and stops XAMPP, and the other allows you to create and modify pages directly in the XAMPP *htdocs* folder.

One common problem is that Skype and other applications also use port 80, the default port for Apache. If another application is using this port, Apache won't start. The log (Figure 1-5) will indicate if Apache could not start because the port was in use. If this occurs, go to the XAMPP control panel and click the Netstat button. This window will list all applications running on your PC and what ports they are using. Determine an unused port (such as 8080) and close the Netstat window. Then click the Config button next to Apache. Select *httpd.conf*. The configuration file will open in Notepad (or your default text editor). Click Edit, click Search, and enter **80**. Click Find Next until you discover the following line:

```
Listen 80
```

Change this line to the following:

```
Listen 8080
```

Or, change it to whatever port you decided to use.
Now go back to search and click Find Next until you find the following line:

```
ServerName: localhost: 80
```

Change this line to the following:

```
ServerName: localhost: 8080
```

Or change it to the same port you decided to use.
Save the file, and close Notepad (or your text editor). Go to the XAMPP control panel and click the Config button at the top-right corner of the page. Click the Services and Port Settings button. Click the tab Apache. Enter the new port for Apache (8080 or whatever port you decided to use). Click Save. Close the Config window. Now click the Start button next to Apache in the control panel (Figure 1-5). After a few seconds, Apache should start. This will be indicated by a green background behind Apache. You will also need to start MySQL because it could not start without Apache running first.

Starting XAMPP

From here onward, to test your pages in XAMPP, double-click the desktop icon and check that Apache and MySQL/MariaDB have started. If they have not started, click the Start button for Apache. Once it starts, then click the Start button for MySQL/MariaDB and then minimize the control panel.
Figure 1-5 shows the XAMPP control panel.

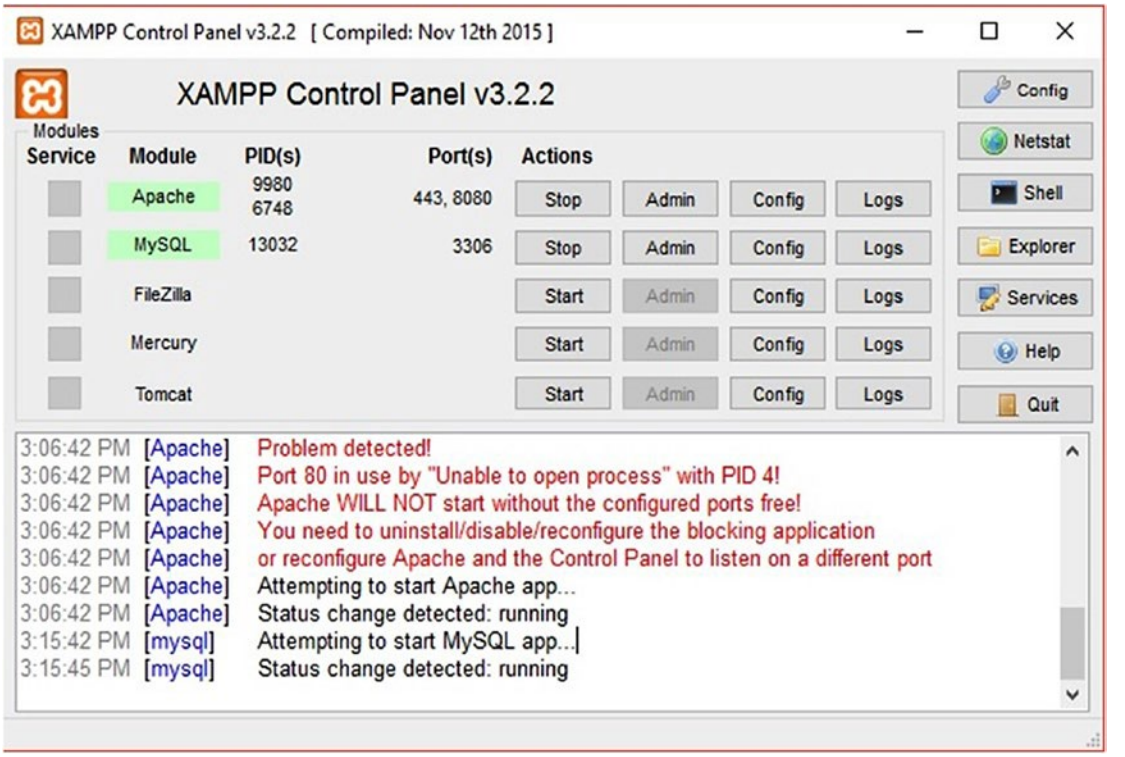

Figure 1-5. *The XAMPP control panel*

We suggest you always minimize the control panel so that you have a clear desktop for starting work on your databases.

After starting Apache and MySQL/MariaDB, you can test your installation and examine all the XAMPP examples and tools; to do this, enter one of the following addresses in your browser:

```
http://localhost/
http://127.0.0.1/
```

If you added a port number, be sure to include the same number in a format like one of the following examples:

```
http://127.0.0.1:8080/
http://localhost:8080/
```

Closing XAMPP

Close XAMPP when you have finished testing your database and PHP files. This will free up memory for tasks other than database development. To close, click the maximize XAMPP control panel on the taskbar and then click the Quit button on the control panel, as shown in Figure 1-6. Alternatively, you can right-click the icon in the Notification area and then click Quit.

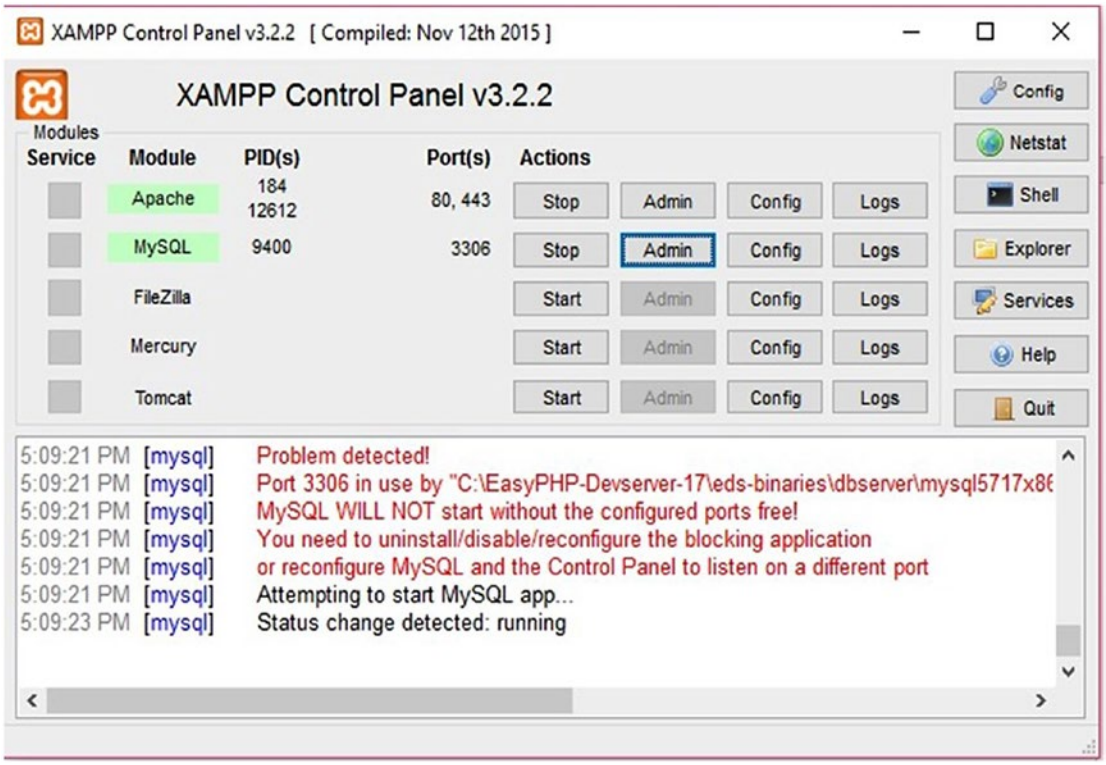

Figure 1-6. *Closing the XAMPP program*

The security of a database and its data is extremely important. XAMPP provides a method for making the database and tables on your computer safe from harmful interference, which is described later in this chapter.

Where Is MariaDB and MySQL 8?

You might be confused at this point about XAMPP and the location of the MariaDB databases. The creators of XAMPP did switch from using MySQL databases to using MariaDB databases; however, everything we just explained seems to indicate that MySQL is still part of XAMPP. However, this is not the case. When the creators decided to switch database systems, they had to make only minor configuration changes to use MariaDB. Since MariaDB does not operate any differently than the free version of MySQL, the creators decided to leave everything with the MySQL title. Since you might be new to this environment, this might be confusing. However, to current users of XAMPP, this allowed an easy transition between database environments. Just remember, wherever you see MySQL in XAMPP, it is MariaDB.

The title of this book includes *MySQL 8*. MySQL 8 is a full, professional version with the latest bells and whistles, including business analytics tools. However, MySQL 8 is not provided with XAMPP or EasyPHP, and the full version is not free. If you are developing sites for corporations or government agencies, you should consider using MySQL 8 (or the latest version). Any version of MySQL can be attached to XAMPP. You do need to read the MySQL documentation to make sure the version you are attaching is compatible with the Apache and PHP versions you have installed. Adding a new version of MySQL is an intermediate skill. We have provided directions in the last chapter to help when you are ready to convert.