



NEAR EAST UNIVERSITY

Faculty of Engineering

Department of Computer Engineering

**DATABANK OF PRODUCER NORTH CYPRUS
PROGRAMMING USING PHP and MYSQL**

**Graduation Project
COM-400**

Student: Mahfuz Daşçı

Supervisor: Mr.Ümit Soyer

Nicosia - 2007

ACKNOWLEDGEMENTS

It is my pleasure to take this opportunity to express my greatest gratitude to man individuals who have given me a lot of supports during my four-year under graduation program in the **Near East University**. Without them, my Graduation Project would not have been successfully compelled on time.

First of all, I would like to express my thanks to my supervisor **Mr. Ümit SOYER** for supervising my project. Under the guidance of him successfully overcome many difficulties and I learned a lot about web designing. He always helped me a lot and I felt remarkable progress during his supervisor. Also I thank for giving his time during my study and my advising.

I also want to thank all my friend and specially **Gökhan TURGAY, Emre Selim ŞAHİN, Engin GEM** and who supported and helped me all the time.

Finally, special thanks for my family, especially my brothers **Mehmet Emin DAŞCI, Mehmet Eta DAŞCI, Hikmet DAŞCI** and **Hifzi DAŞCI** for being patient full during my undergraduate degree study. I could never have completed my study without their encouragement and endless support.

ABSTRACT

The aim of this Project to help the every kind of Product, for example products are; Vegetables, Fruits and Animals by this system the association can register the farm and the farmer.

Using this system Producer can control their production, and association can send information about the result of the production. Also Producer can get information easily and quickly about the announcement and the price of the production.

The producers can easily pursuit the amount of production that Association accepted or not, and also he can see every producer what have a production. The association can add, search and delete all information about the Producer, and create the password for the farmer or for the other Administrator.

TABLE OF CONTENTS

ACKNOWLEDGEMENT	i
ABSTRACT	ii
TABLE OF CONTENTS	iii
INTRODUCTION	1
CHAPTER ONE: ABOUT MY PROJECT	2
1.1 Data Bank of Producers System Programming	2
1.1.1 User Interface Area	3
1.1.2 Admin Interface Area	8
1.1.3 User Interface Area	16
CHAPTER TWO: WHAT IS THE HTML	18
2.1 What is the Html	18
2.1.1 HTML Tags	19
2.1.2 What is IIS?	20
2.1.3 World-Wide Web	21
2.1.4. SLIP and PPP	21
2.1.5 Advanced Conferencing Software	22
2.1.6 Proprietary Software vs. Open Standards	22
2.2 HTTP Authentication with PHP	22
CHAPTER THREE: WHAT IS MYSQL	25
3.1 Understanding MySQL	25
3.1.1 What Is the Enterprise?	27
3.1.2. How hard is it to change?	28
3.1.3. What Is a Relational Database?	29

3.2 Designing Databases	29
3.2.1 Connecting to Database	30
3.2.2 Encrypted Storage Model	30
3.2.3 The Client/Server Paradigm	31
3.2.4 History of Mysql	32
3.2.5 How Large MySQL Tables Can Be	33
3.2.6 Features of MySQL	33
CHAPTER FOUR: INTALLATION APACHE SERVER	34
4.1 Setup a Directory Structure	34
4.2 Install PHP 5	37
4.2.1 Configuring Apache	40
4.2.2 Installing MySQL Server	42
CHAPTER FIVE: WHAT IS APACHE SERVER	45
5.1 What is Apache	45
5.1.1 The Apache Serve	46
5.1.2 Apache's architecture	46
5.1.3 More recent history	47
5.1.4 The Future of Apache	47
5.1.5 Obtaining and Installing Apache	48
5.1.6. Hardware / Software Requirements	48
5.1.7. Configuring	48
5.1.8. Configuration Files	48
CHAPTER SIX: WHAT IS PHP	50
6.1 What is PHP?	50

6.2 What can PHP do?	50
6.3 General Installation Considerations	52
6.4 Installation on Windows Systems	53
6.4.1. Windows InstallShield	54
6.4.2 Building from Source	54
6.4.3 Putting It All Together	54
6.4.4 Compiling	55
6.5 Installation of Windows Extensions	56
6.5.1 Windows and PWS/IIS 3	55
6.5.2 Windows and PWS 4 or newer	58
6.6 The Configuration File	58
6.7 Security	59
6.8 General Considerations	60
6.8.1 Installed as an Apache module	61
6.8.2 Installed as CGI binary	61
CHAPTER SEVEN: APPLICATION CODES	62
7.1 User Interface Area Codes	62
7.1.1 Login_user.php	62
7.1.2 Show producers List.php	64
7.1.3 Edit Producers Information.php	67
7.1.4 Show Location.php	71
7.1.5 Record Person.php	76
CONCLUSION	81
REFERENCES	82

INTRODUCTION

In the early days of the World Wide Web, information was stored in simple files with only HTML markup. Today, the web server often processes scripts within web files that e.g. call other files, style sheets, etc. Furthermore, the scripts may also acquire information from a relational database management system. Such a modern system gets data from database tables and includes them in the web page that is delivered to the user.

To make database-backed system of web pages like Evolutionary Economics. We need a system of tools. Thus we leave the user's viewpoint that there is just a web server that is capable of sending web pages.

In the First Chapter I explained Data Bank of Producers Internet Programming and determined some word that it was mentioned in the system. I have given some example from the program code and also some page from the project

In the Second Chapter I explained about the HTML. What is html and how we use it that code.

In the Third Chapter I write about the Database, I have used MySQL for my project so I gave some information about the MySQL database,

In the Fourth Chapter I mentioned about the Installation Apache server, how is work and setup a directory structure.

In the Fifth Chapter I mentioned about the Apache server, how is work, the apache architecture, the history of apache.

In the Sixth Chapter I write about my fundamental program that it is known PHP (Personal Home Page) and again I write about php program and give some example about this language

In the Last Chapter I write the Project code in tree parts, two of them are Guest, Admin Interface Area, and the other one is User Interface Area.

Given that an adequate software system is related to the web server, the web site developer has a radically increased set of possibilities. A first glimpse of these possibilities may be obtained by inspecting one of the files that are used to produce a web page on such a system. The server-side file will probably have some ordinary text and HTML markup, but it will also have smaller or larger amounts of commands written in a programming ("scripting") language.

CHAPTER ONE ABOUT MY PROJECT

1.1 Data Bank of Producers System Programming

In this chapter; I will explain the contents of project and also the goals of project by the some word that used in the Project.

The main goal of the Project is that, to registers the producer and producer production in the Internet environment.

Also main of the project is in North Cyprus and what get product and where to selling which Company.

What can do admin?

In the system every control is the admin. Admin can do every thing. He can change producer's information and he can change producers' production and he can entry, delete and update for every producer and production and also admin can entry new production for user and he can changes this production.

What can do users (Producers)?

In this system user can do changes own information and own production. Of Course every user can change own information and own production for delete, entry and update. If user is bring up new production he can not add itself only admin do that. He is getting the permission from admin and admin can add that production and permitted to that user.

1.1.1 Guest Interface Area

When we enter the system we will see a page like follow figure 1.1

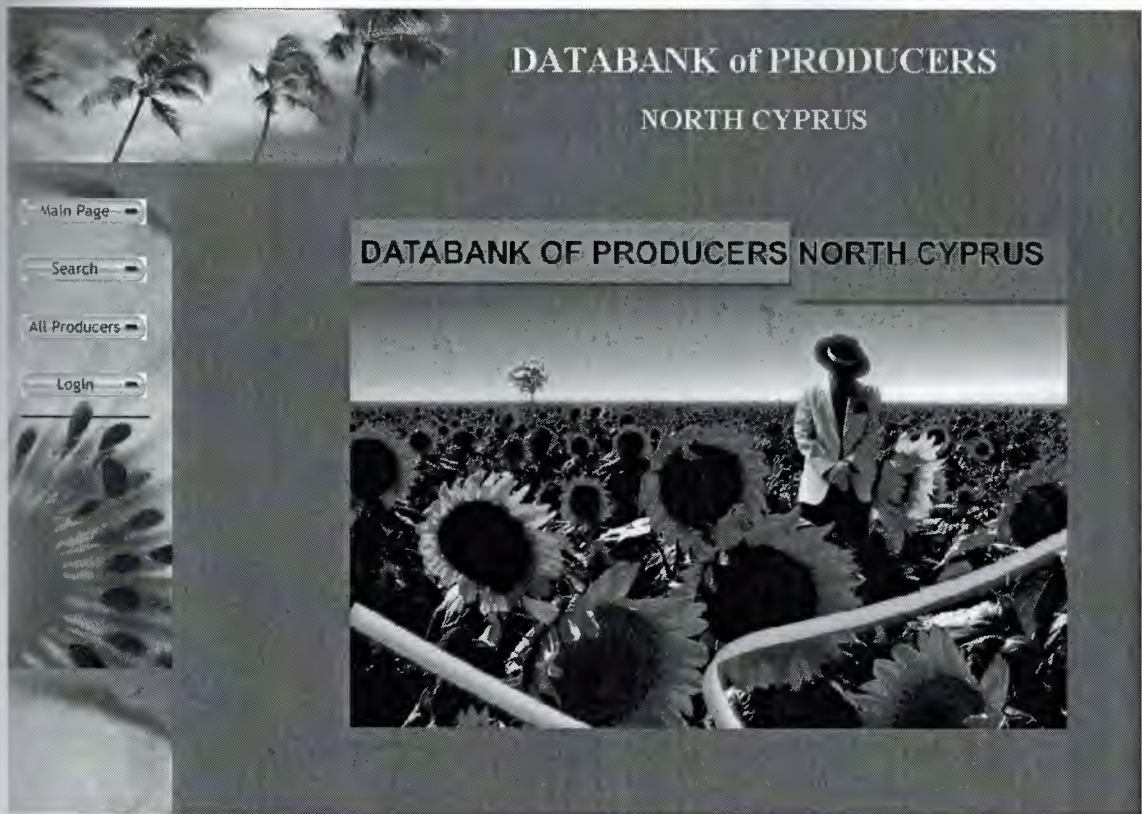


Figure 1.1

We will see 4 buttons in left of the page. In this page for guest that is can see main page, search, and all producers but can not be login in system. For example; when he click the **Search** like the figure 1.2

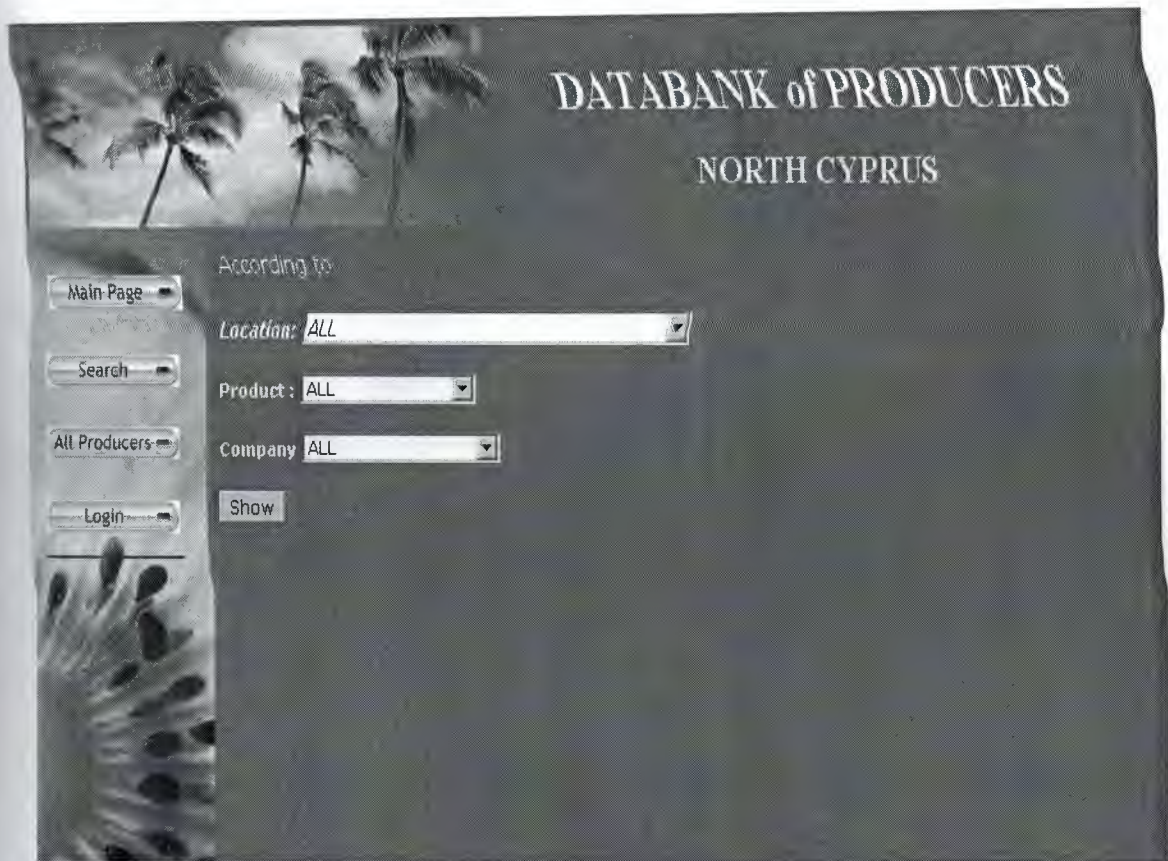


Figure 1.2

In this page (Figure 1.2) guest can see what producer information is and what is producer of product. When guest selected any location and click show it will see how many producers sitting in that location. For example you are selected Yenisehir in the (Figure 1.3).

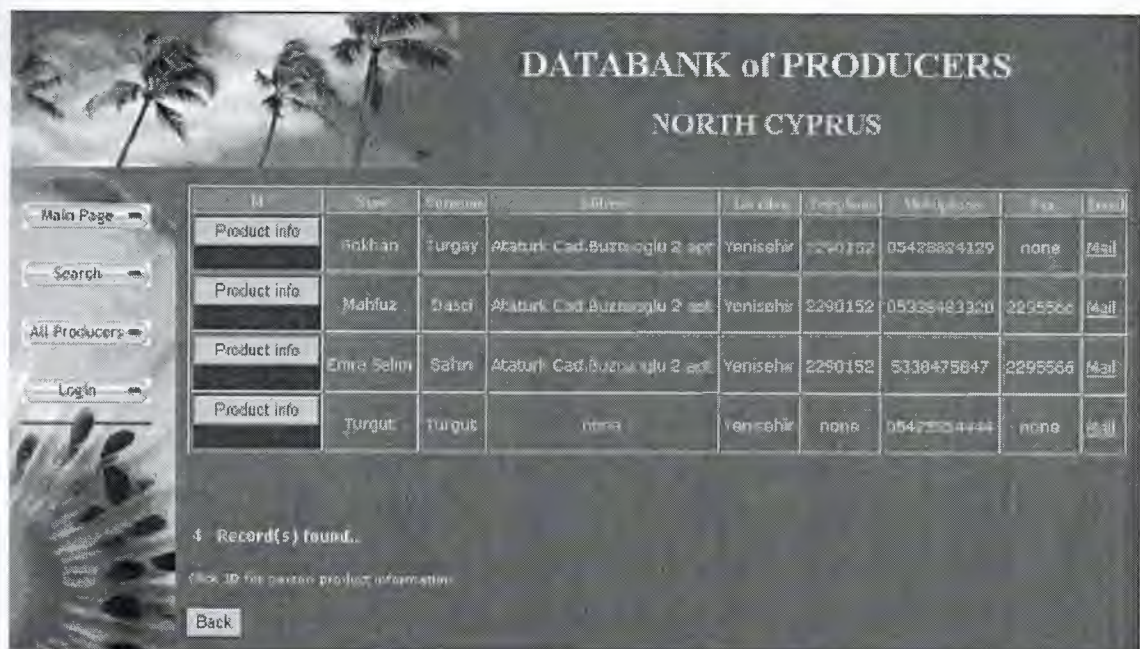


Figure 1.3

In this page (Figure 1.3) we are see who are sitting in that location which is Yenisehir and we can see dependent of information in that producer. In this pages when we click the **Product Info button** we will see this producer what are bring up to product. For example we can see the following Figure 1.4

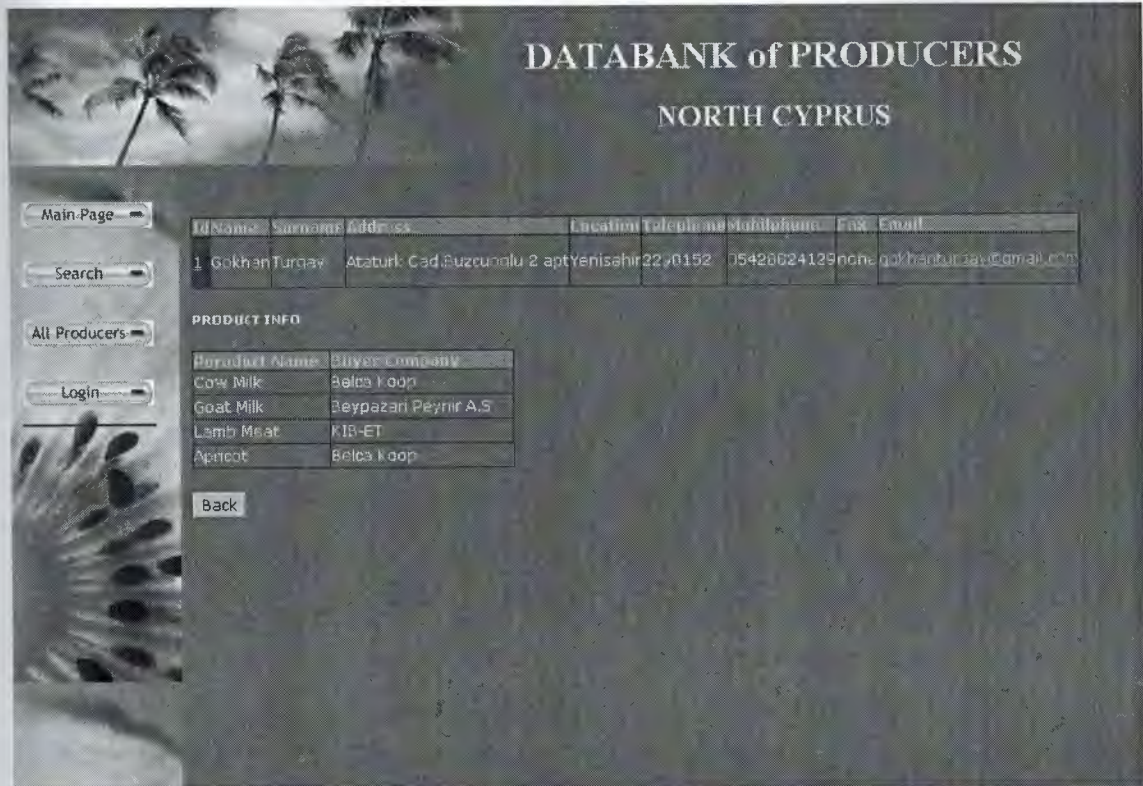


Figure 1.4

As you see in this pages producer name is Gökhan Turgay and he is products cow milk, goat milk, apricot and etc. Of course we can see like that in other producers. When we click the All Producers we will see all producers and information together. Like the figure 1.5

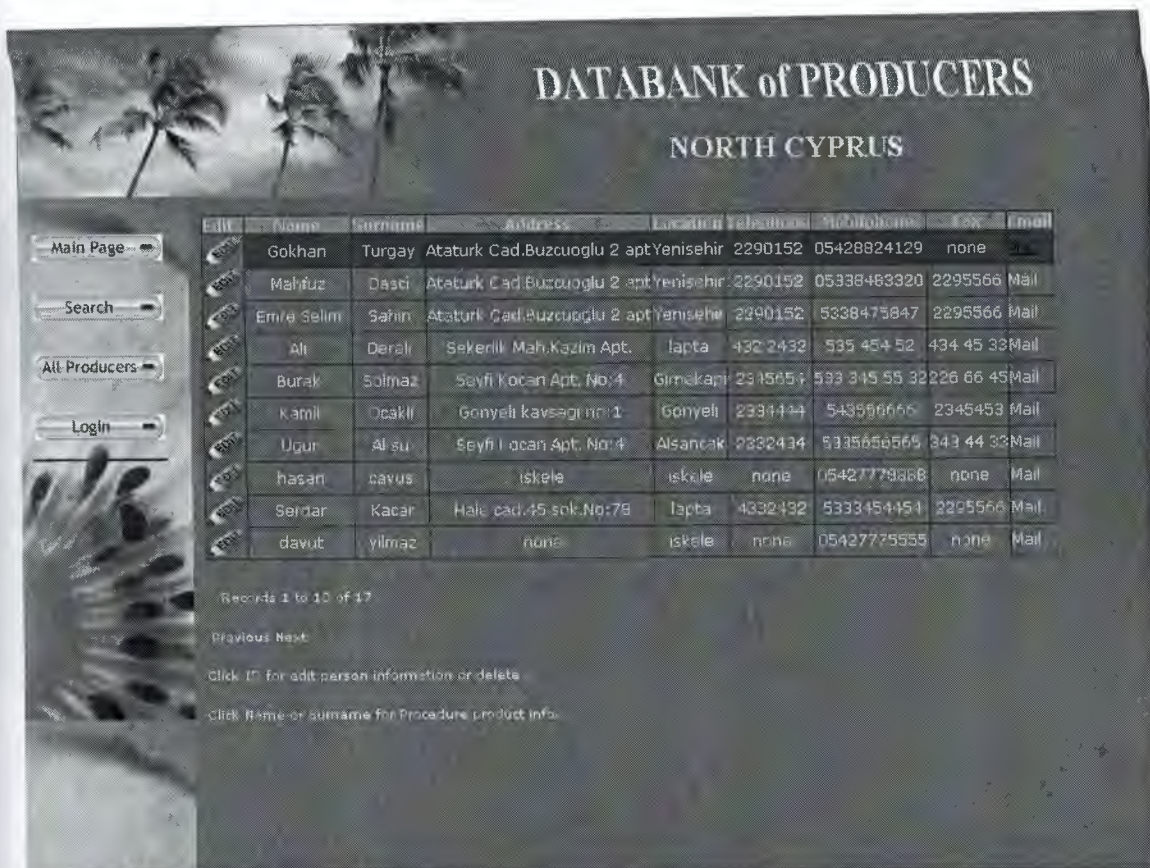


Figure 1.5

If you are guess you can not change any thing and can not edit product or producer and etc. when you pres edit you will see as like the following Figure 1.6

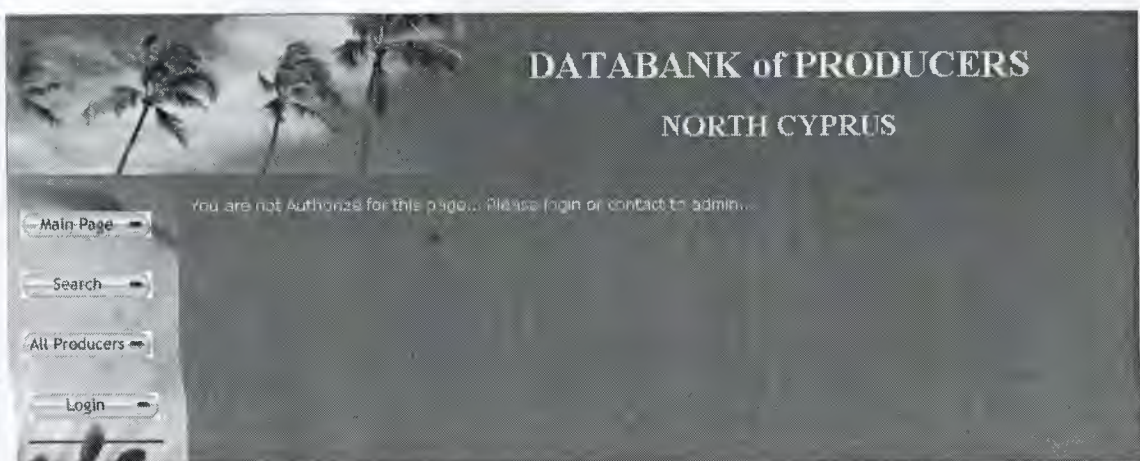


Figure 1.6

As like that you are seen message like that; you are not authorization for this page. Please contact to admin for permit.

Ok now we are login with **admin**. For that we are click the **Login button**. See following Figure 1.7

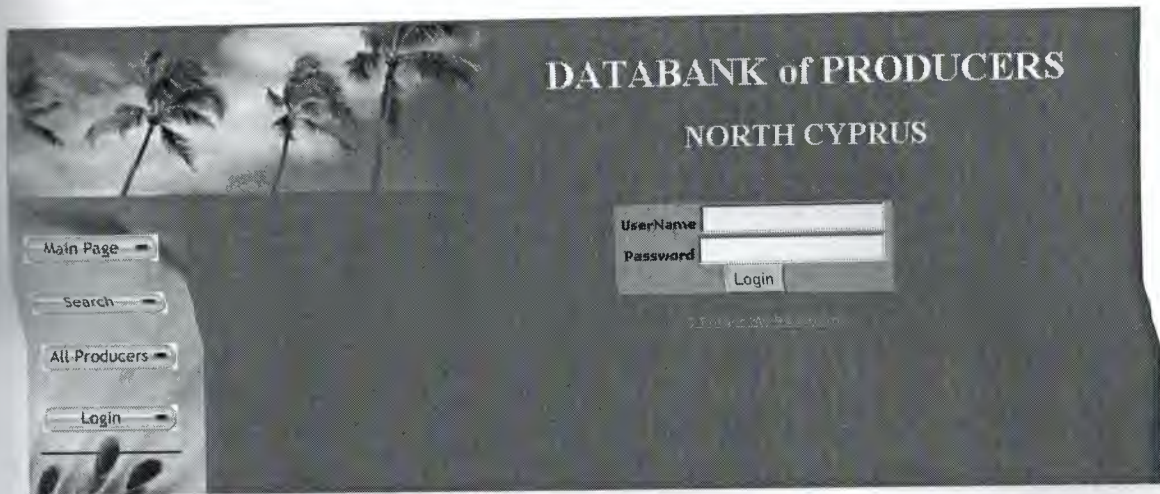


Figure 1.7

In this page we are login by admin. As following Figure 1.8

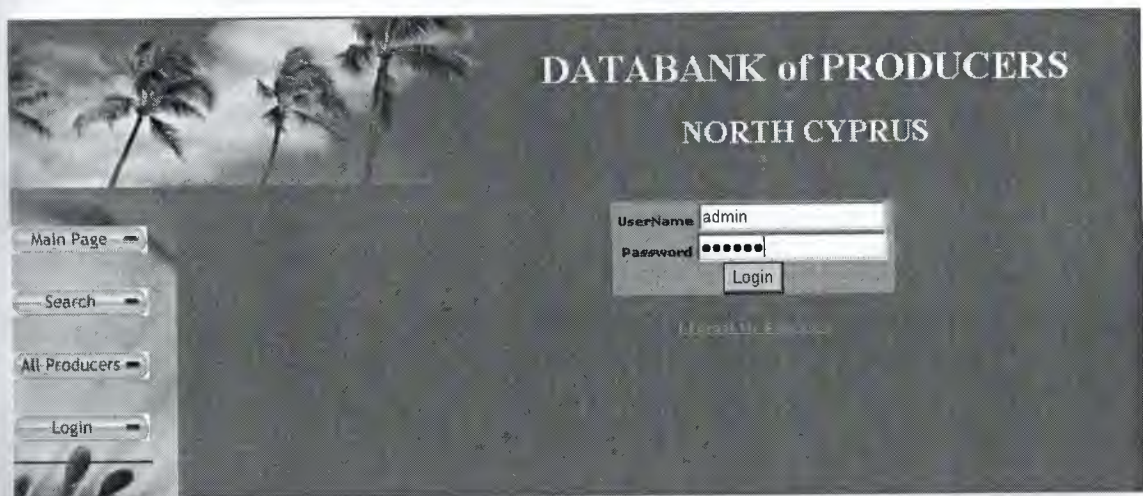


Figure 1.8

When we enter the system by admin we see many buttons in that page.

Figure1.9

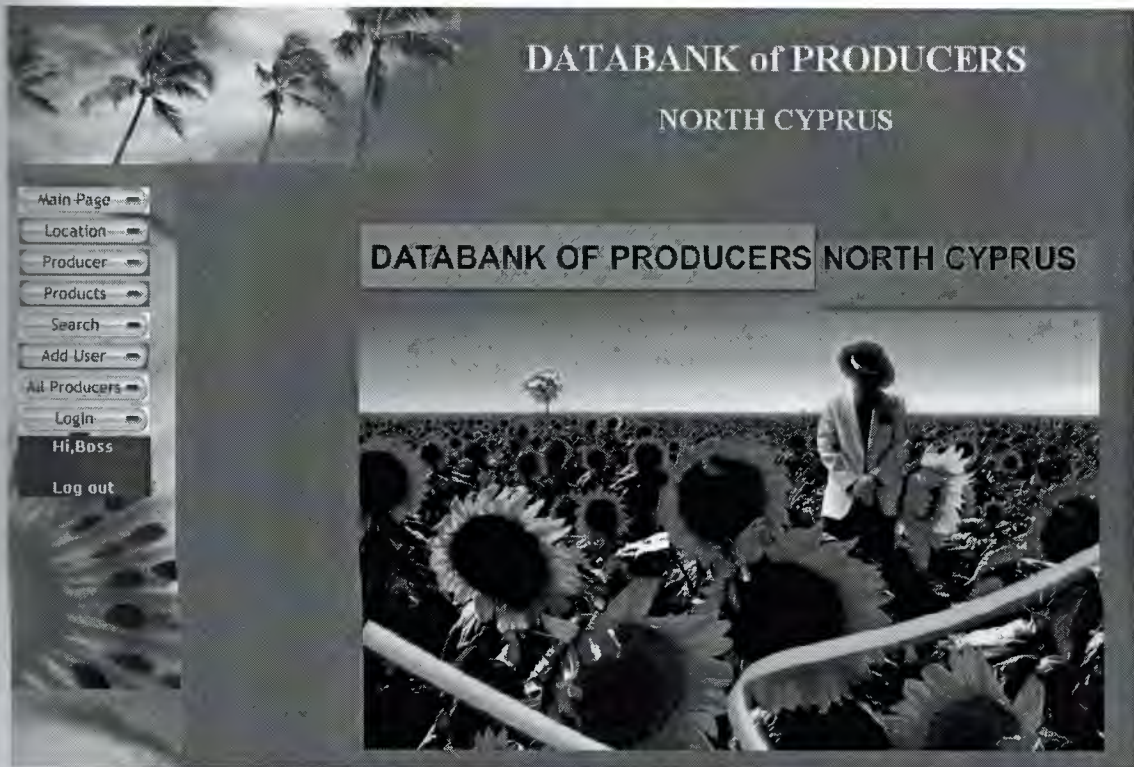


Figure 1.9

1.1.2 Admin Interface Area

In this page every thing under control of admin. Admin can changes any thin. He is can registration of producer information and can registration location. As following Figure 1.10

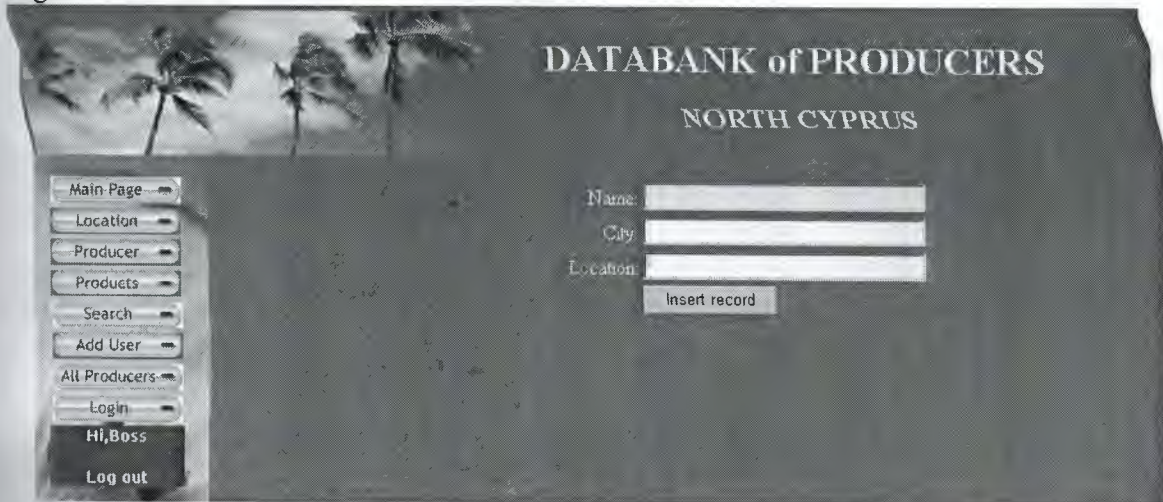


Figure 1.10

In this page **admin** can registration producer of location. When we click the **Producers button** we will see as the following figure 1.11

DATABANK of PRODUCERS
NORTH CYPRUS

Main Page
Location
Producer
Products
Search
Add User
All Producers
Login
Hi, Boss
Log out

Id:
Name:
Surname:
Address:
Location: iskele
Telephone:
Mobilephone:
Fax:
Email:

Insert record ☐ Do you want to be normal user for this person?

Figure 1.11

In this page admin do registration producers information. When we click the **Products** button we will see as like that Figure 1.12

DATABANK of PRODUCERS
NORTH CYPRUS

Main Page
Location
Producer
Products
Search
Add User
All Producers
Login
Hi, Boss
Log out

Add Product
Add Product to Person
Add Buyer Company

Figure 1.12

In this page admin can control **Add Product**, **Add Product to Person** and **Add Buyer Company**. When we click **Add Product** we will see following Figure 1.13

The screenshot shows a web application titled "DATABANK of PRODUCERS NORTH CYPRUS". The background features a tropical scene with palm trees. On the left, there is a vertical sidebar with the following buttons: "Main Page", "Location", "Producer", "Products", "Search", "Add User", "All Producers", "Login", "Hi,Boss" (highlighted in black), and "Log out". The main content area on the right contains three stacked forms: "ADD ANIMAL", "ADD FRUIT", and "ADD VEGETABLE". Each form has input fields for the product name and price, followed by an "Insert record" button. Below these forms is a link labeled "List Products".

DATABANK of PRODUCERS
NORTH CYPRUS

Main Page
Location
Producer
Products
Search
Add User
All Producers
Login
Hi,Boss
Log out

ADD ANIMAL
InputAnimal
Price
Insert record

ADD FRUIT
Input Fruit
Price
Insert record

ADD VEGETABLE
Vegetables
Price
Insert record

[List Products](#)

Figure 1.13

In this page we have three main subjects; Animal, Fruit, Vegetable. And admin can do enter product to producer. And after this process we can see List Product in following Figure 1.14

DATABANK of PRODUCERS									
NORTH CYPRUS									
<div> Main Page Location Producer Products Search Add User All Producers Login Hi,Boss Log out </div>									
id	Animal List	Price (YTL)	id	Fruit List	Price (YTL)	id	Vegetables	Price (YTL)	
Edit	Sheep Meat	8	Edit	Apple	2	Edit	Carrot	1.9	
Edit	Cow Milk	0.7	Edit	Orange	1.23	Edit	Tomato	1.9	
Edit	Hen Meat	1.5	Edit	Banana	1.7	Edit	Potato	1.8	
Edit	Goat Meat	5	Edit	Water Melon	3	Edit	Aubergine	2	
Edit	Cow Meat	12	Edit	Melon	3	Edit	Pepper	3	
Edit	Egg	0.35	Edit	Tangerine	1.3	Edit	Spinach	4	
Edit	Goat Milk	0.5	Edit	Lemon	1.5	Edit	Cabbage	1.55	
Edit	Lamb Meat	13	Edit	Pear	2.5	Edit	Onion	2.1	
			Edit	Bitter Orange	2	Edit	Broccoli	3.3	
			Edit	Grape	1	Edit	Cauliflower	3	
Records 1 to 8 of 8			Records 1 to 10 of 11			Records 1 to 10 of 10			

Figure 1.14

We are seeing all products (Animal List, Fruit List, Vegetables) and with prices. If you click the **Edit button** you will see like that Figure 1.15

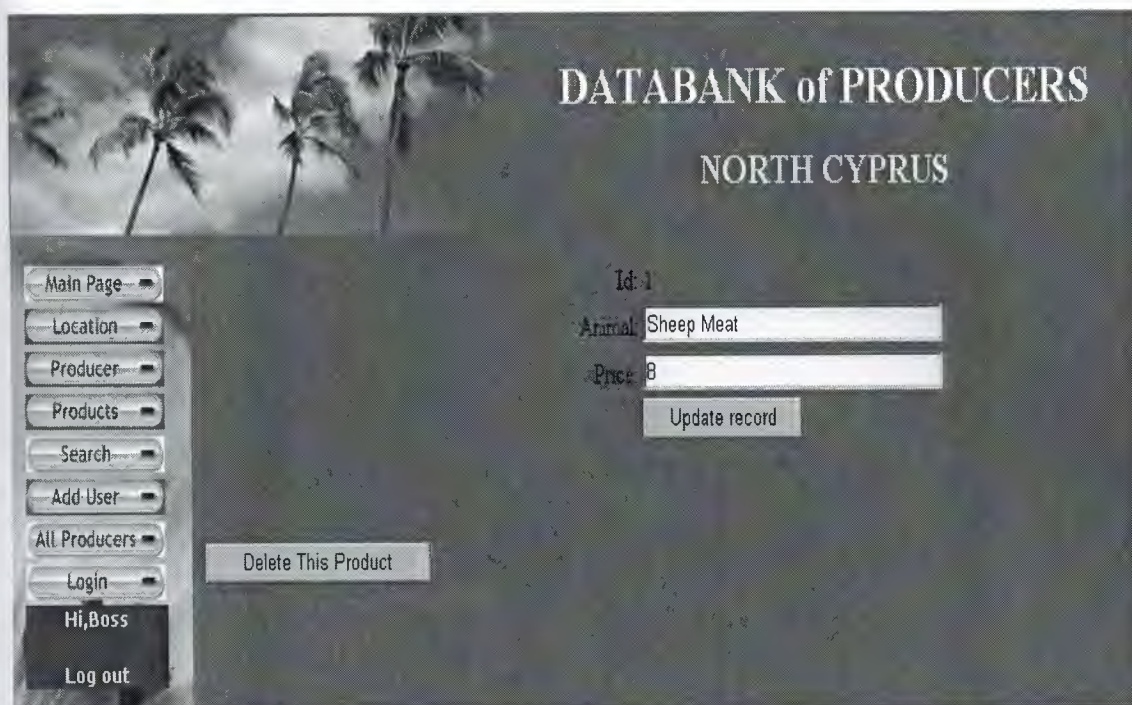


Figure 1.15

In this page **admin** can changes prices but user can not change prices. We will see user interfaces after that process. When we click **Add Product to Person** in Figure 1.12, now we are See Figure 1.16

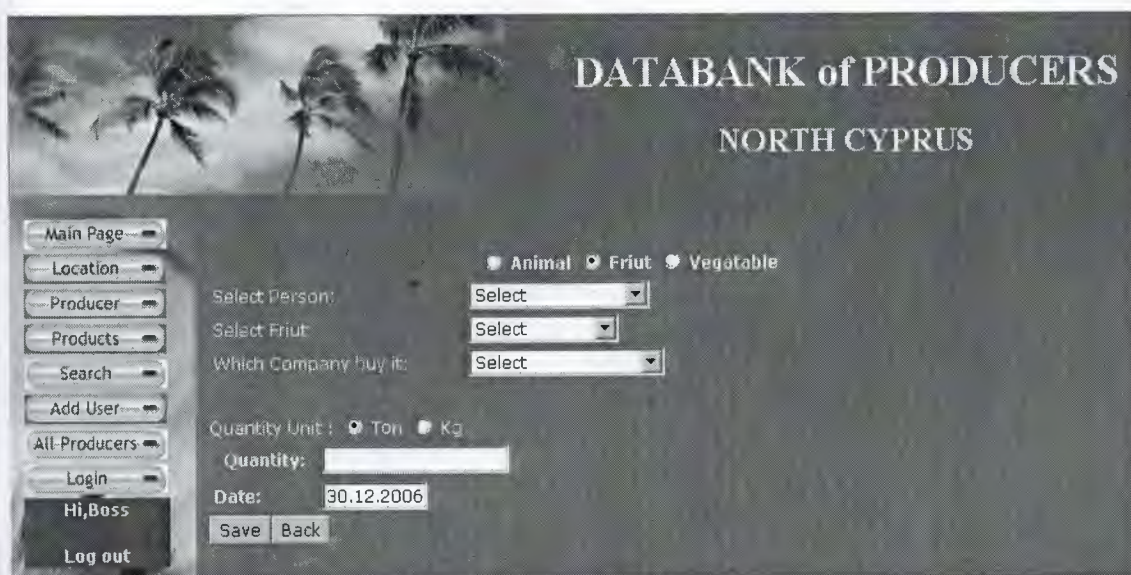


Figure 1.16

In this page admin add product to producer for example if we select person (**Ali Dereli**) and we select Fruit is **Orange** and select buyer company is **Belca Koop** and

keep the quantity unite **Ton** and product quantity **2** ton and press save by date. It is okay. See following figure 1.17

The screenshot shows the 'DATABANK of PRODUCERS NORTH CYPRUS' web application. On the left is a sidebar menu with buttons: Main Page, Location, Producer, Products, Search, Add User, All Producers, Login, Hi,Boss, and Log out. The main content area has a header with the application name and a background image of palm trees. Below the header, there are radio buttons for 'Animal', 'Fruit', and 'Vegetable'. The 'Fruit' option is selected. Below these are three dropdown menus: 'Select Person:' with 'Ali Dereli', 'Select Fruit:' with 'Orange', and 'Which Company buy it:' with 'Belca Koop'. Further down, there are radio buttons for 'Quantity Unit:' with 'Ton' selected and 'Kg'. Below that is a 'Quantity:' input field with the value '2', and a 'Date:' input field with the value '30.12.2006'. At the bottom of the form are 'Save' and 'Back' buttons.

Figure 1.17

After save that process it is registration to account Producer (Ali Dereli) and will see later. We continue **Figure 1.12** for to **Add Buyer Company**. See Figure 1.18

The screenshot shows the 'DATABANK of PRODUCERS NORTH CYPRUS' web application. On the left is a sidebar menu with buttons: Main Page, Location, Producer, Products, Search, Add User, All Producers, Login, Hi,Boss, and Log out. The main content area has a header with the application name and a background image of palm trees. Below the header, there are input fields for 'Id:' (111022), 'Name:' (Reis Market), and 'E-Mail:' (Reismarket@gmail.com). Below these is a dropdown menu for 'Address:' with the value 'yenisehir'. Further down, there are three checkboxes for 'Category:': 'Animal' (checked), 'Fruit' (checked), and 'Vegetable' (checked). Below the category checkboxes are two buttons: 'Insert record' and 'Add Buyer User' (checked). At the bottom of the form is a 'Microsoft Internet Explorer' window with a message: 'Password will be create and send via e-mail, Username will be company e-mail address' and a 'Tamam' button.

Figure 1.18

In this page admin can **Added Buyer Company** by Id, Name, E-Mail, Address and Category, which is Animal, Fruit, vegetable.

When we click **All Producers** we will see following **Figure 1.19**



The screenshot shows a web application titled "DATABANK of PRODUCERS NORTH CYPRUS". On the left is a sidebar with navigation buttons: Main Page, Location, Producer, Products, Search, Add User, All Producers (highlighted), Login, Hi,Boss, and Log out. The main area displays a table of producers with columns: ID, Name, Surname, Address, Location, Telephone, Mobilephone, Fax, and Email. Below the table, it says "Records: 1 to 10 of 18", "Previous Next", and "Click ID for edit person information or delete".

ID	Name	Surname	Address	Location	Telephone	Mobilephone	Fax	Email
1	Golhan	Turgay	Ataturk Cad.Buzcuoglu 2 apt	Yenisehir	2290152	054288324129	none	Mail
2	Mahfuz	Dasci	Ataturk Cad.Buzcuoglu 2 apt	Yenisehir	2290152	05338483320	2295566	Mail
3	Emre Salim	Sahin	Ataturk Cad.Buzcuoglu 2 apt	Yenisehir	2290152	5398475347	2295566	Mail
4	Ali	Direli	Sekerlik Mah.Kazim Apt.	Iapta	432 2432	535 454 52	434 45 33	Mail
5	Burak	Salmaz	Seyfi Kocan Apt. No:4	Ginekapı	2345654	533 345 55	32225 66 45	Mail
6	Kamil	Ocakli	Gonyeli Kaysagi no:1	Gonyeli	2334444	543556666	2345453	Mail
7	Ugur	Aksu	Seyfi Kocan Apt. No:1	Alsancak	2332434	5335555555	343 44 33	Mail
8	Hasan	Cavus	Iskale	Iskale	none	05427776388	none	Mail
9	Serdar	Kacar	Halle cad.45 sok No:78	Iapta	4332432	5333454454	2295566	Mail
10	devut	Yilmaz	none	Iskale	none	05427775555	none	Mail

Figure 1.19

In this page when we click a **Producer Name** (for example **Hasan Cavus**) we can see following in **Figure 1.20**

DATABANK of PRODUCERS

NORTH CYPRUS

[Main Page](#)

[Location](#)

[Producer](#)

[Products](#)

[Search](#)

[Add User](#)

[All Producers](#)

[Login](#)

[Hi,Boss](#)

[Log out](#)

ID	Name	Surname	Address	Location	Telephone	Mobilephone	Fax	Email
2	Mahfuz	Dasci	Ataturk Cad.Buzcuoglu 2 apt	Yenisehir	2290152	05338483320	2295566	mailto:mahfuzdasci@hotmail.com

PRODUCT INFO

	Product Name	Quantity	Unit	Price	Date	Buyer Company
1	Water Melon	2	Ton	6000 YTL	20.12.2006	Arma Meyve Suyu
2	Goat Milk	400	Kg	200 YTL	20.12.2006	Mera Sut-Urunleri
3	Orange	2	Ton	2460 YTL	01.01.2007	Belca Kapp

Total Unit type of Kg : 4400.00 Kg

Total Unit type of Ton : 4.40 Ton

Total Amount : 8660.00 YTL

Figure 1.20

In this page admin can change the any product and producers info and when we click the product (**Orange**) we will see following **Figure 1.21**



The screenshot displays a web application titled "DATABANK of PRODUCERS NORTH CYPRUS". On the left is a vertical navigation menu with buttons: Main Page, Location, Producer, Products, Search, Add User, All Producers, Login, Hi,Boss, and Log out. The main content area shows a form for editing a product record. The form fields are: No: 79, Person: Mahfuz Daso, Product: Orange (selected from a dropdown), Quantity: 2, Unit: Ton (selected from a dropdown), Price: 2460, Dates: 01.01.2007, and Company: Belca Koop (selected from a dropdown). At the bottom right of the form are two buttons: "Update record" and "Delete".

Figure 1.21

In this page admin can changes the products. When we login by user name we will see the following **Figure 1.22**

1.1.2 User Interface Area

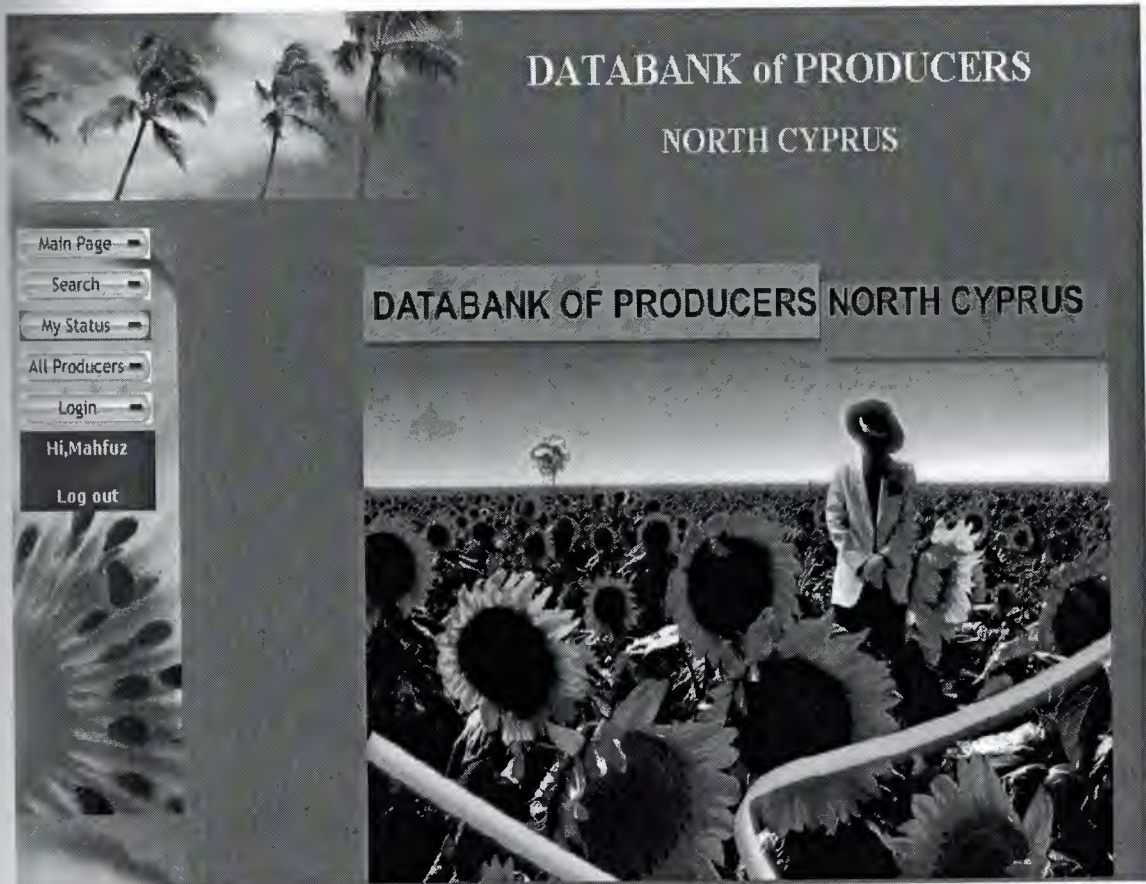


Figure 1.22

This figure is the User interface area. This page has a **User Control** button. In this page every **Users (Producers)** can search, can see all producer and every user of course can change it self products. For example click and see **My Status Button** and see what we have in following **Figure 1.23**

DATABANK of PRODUCERS
NORTH CYPRUS

Main Page
[Search](#)
[My Status](#)
[All Producers](#)
[Login](#)
 Hi, Mahfuz
[Log out](#)

Name Mahfuz
Surname Dasci
Location Yenisehir
E-mail mahfuzdasci@hotmail.com
Amount 0

[Edit My Profile](#) - [Add Product](#) - [Change Password](#)

Product name	Quantity	Unit	Price	Dates	Company
Water Melon	2	Ton	5000 YTL	20.12.2006	Arma Meyve Suyu
Goat Milk	400	Kg	200 YTL	20.12.2006	Mara Sut Unulen
Orange	2	Ton	2450 YTL	01.01.2007	Balca Koop
Lemon	1	Ton	1500 YTL	02.01.2007	Balca Koop
Pear	500	Kg	1250 YTL	02.01.2007	Pais Market

Total Unit type of Kg 5900.00 Kg
 Total Unit type of Ton 5.90 Ton
 Total Amount 11410.00 YTL

Figure 1.23

In this page we have a user an account. User can change same information some product but one information thing if user have a new bring up product he can not add own list. He is get permission admin. Admin can do that. So every user (Producers) has system like that.

CHAPTER TWO: WHAT IS THE HTML

2.1 What Is the Html

The basic language of the Internet is Hyper Text Markup Language (HTML). Unlike a true programming language, HTML doesn't work like an operating system and run your computer. Instead, it allows the author to "mark up" the contents of a document in order to change its visual appearance in a web browser. A browser takes the content as written in the HTML file and represents it on the screen of your computer.

HTML is designed to specify the logical organization of a document, with important hypertext extensions. It is not designed to be the language of a WYSIWYG word processor such as Word or WordPerfect. This choice was made because the same HTML document may be viewed by many different "browsers", of very different abilities. Thus, for example, HTML allows you to mark selections of text as titles or paragraphs, and then leaves the interpretation of these marked elements up to the browser. For example one browser may indent the beginning of a paragraph, while another may only leave a blank line.

HTML instructions divide the text of a document into blocks called elements. These can be divided into two broad categories -- those that define how the BODY of the document is to be displayed by the browser and those that define information 'about' the document, such as the title or relationships to other documents. The detailed rules for HTML (the names of the tags/elements, how they can be used) are defined using another language known as the standard generalized markup language, or SGML. SGML is wickedly difficult, and was designed for massive document collections, such as repair manuals for F-16 fighters, or maintenance plans for nuclear submarines. Fortunately, HTML is much simpler!

However, SGML has useful features that HTML lacks. For this reason, markup language and software experts have developed a new language, called XML (the extensible markup language) which has most of the most useful features of HTML and SGML.

2.1.1 HTML Tags

- HTML Tag `<html> </html>` : This tells the browser that the document is in HTML format.
- Head Tag `<head> </head>` : This section contains information about your site but is not visible in a browser window. This information is helpful when you want to use search engines to promote your site. The one tag you should always include is the `<title>` tag.
- Title Tag `<title> </title>` : This will display the title of your site in the top bar of the browser. You want to include this or your page will appear as an untitled document in the browser and in lists of bookmarks.
- Body tag `<body> </body>` : The visible content of the page must be placed between the `<body>` and `</body>` tags. In our example in above, an attribute of `<bgcolor="#ffffff">` was added to this tag to ensure that the page would have a white background.
- Paragraph tag `<p> </p>` : This tag tells the browser to skip a line and begin a new paragraph. This tag often appears without a corresponding closing tag. Although browsers will still represent your paragraphs correctly, including that closing tag will make possible transitions to future technologies easier.
- Linebreak tag `
` : Content following the `
` tag will appear on the next line.
- Heading tag `<h1> </h1>` : This tag indicates to the browser that you are using a headline. The range of values from `<h1>` to `<h6>` (largest to smallest) sets the size of the headline. Good practice indicates that you should use an `<h1>` tag only once on a web page, just as a newspaper uses the largest headline size only once on the front page. Text formatted with this tag will also be bold. However, do not use this tag if you simply want bold text. Use the `` tag instead.
- Bold tag ` ` : This will make the content between the tags appear bold.
- Italic tag `<i> </i>` : This italicizes the content between the tags.
- Image tag `` : This tag allows you to place graphics onto a web page. You will have to indicate the location of the graphic you want to use. This is done through the attribute `<src="filename">`. You will also want to indicate the

height and width of the graphic. This will allow the browser to continue loading other content on your page as the graphic is downloaded. Finally, use the `<alt>` attribute to describe the graphic. This will be helpful for those surfing with graphics turned off or for persons with visual disabilities using a text reader.

- Anchor tag `<a> ` : The anchor tag creates links to additional content elsewhere on the Web or elsewhere in your site.
- Font tag ` ` : This tag will allow you to change the size, color, and font of the text on your page. You can modify any of these properties by altering these attributes. If you wish to modify the size of the font to be one size larger than normal, you would use ``. While it is possible to set an absolute size, it is better to set a relative size instead such as `"-1"` or `"+2"`. This way you can make your page increasable for someone with a visual problem to read your content.
- Tables: Tables allows for better organization of larger amounts of information into a more cohesive structure.
- Table tag : The `<table>` and `</table>` tags go at the beginning and very end of the table.

2.1.2 What is IIS?

IIS (Microsoft Internet Information Services or Server) is a set of Internet based services for Windows machines. Originally supplied as part of the Option Pack for Windows NT, they were subsequently integrated with Windows 2000 and Windows Server 2003. The current (Windows 2003) version is IIS 6.0 and includes servers for FTP, SMTP, NNTP and HTTP/HTTPS. Earlier versions also included a Gopher server.

The web server itself can not directly perform server side processing but can delegate the task to ISAPI applications on the server. Microsoft provides a number of these, including ones for Active Server Pages and ASP.NET. Third parties have provided support for PHP and Perl languages in the same way.

Internet Information Services is designed to run on Windows server operating systems. A restricted version that supports one web site and a limited number of connections is also supplied with Windows XP Professional.

IIS has been attributed with a number of security exploits, most of which were in fact issues within the lesser used ISAPI handlers. With Windows Server 2003 Microsoft finally elected to turn off all ISAPI handlers by default thereby giving the web servers a much more secure "out of the box" configuration.

Microsoft has also changed the server account that IIS runs on. In versions of IIS before 6.0, all the features were run on the System account, allowing exploits to run wild on the system. Under 6.0 many of the processes have been brought under a Network Services account which has fewer privileges. In particular this means that if there is an exploit on that feature, it wouldn't necessarily compromise the entire system.

Apache is the dominant software in the web server market and IIS's main competitor. Solaris Operating Environment/J2EE also competes in the enterprise web services arena.

2.1.3 World-Wide Web

The Web incorporated the capabilities of most earlier tools, and added the ability to handle various media types in a much more usable graphical hypertext environment. Interfaces for the Web exist on all computer platforms, and are easy to use so much so that Web usage has exploded in the past year, and the amount of academic and commercial information available is increasing at an almost exponential rate. The amount of information available from a desktop computer provides incredible support for research on a variety of topics, including distance education.

2.1.4 SLIP and PPP

These acronyms refer to two standard protocols for utilizing a graphical interface to the Internet through a regular telephone line and modem. Previously, users had to travel to a central location if they wanted to use tools which relied on graphics technologies much like the similar need required to utilize satellite conferencing technology.

2.1.5 Advanced Conferencing Software

Conferencing software is still very much in the early stages on the Internet. Net Phone from Electric Magic permits real-time transfer of audio across the Internet, much like a telephone call. Some users in Hong Kong are now experimenting to see what other components they would need in order to listen to the U.S. National Public Radio broadcasts through the Internet, as they are not available locally. CU-SeeMee is an experimental one-many and/or many-many video repeater. It is an experimental tool using standard network protocols to provide video feeds to anyone who wants one. Videoconferencing from the desktop will be feasible once more network lines have been upgraded to higher signal capacity. Where fiber-optic cable has been installed, the capacity is already far beyond that provided by satellite. Another drawback to computer solutions is the limited image resolution currently available. Now the hardware and software need to take the next step into utility and view ability.

2.1.6 Proprietary Software vs. Open Standards

Many information service companies are offering purpose-built proprietary software for specialized applications on the Internet. A common example is Ameritech's Advanced Video Service, which provides full-motion videoconferencing over standard dial-up lines. Those which fail to incorporate the ability to utilize standard network communication protocols will achieve very limited market penetration. Tremendous amounts of information on every subject imaginable are being provided on servers throughout the Internet. They are following standard protocols, which are constantly under creation or revision. Software which is not able to speak to or query these resources is crippled when it comes to actual utility.

2.2 HTTP Authentication with PHP

The HTTP Authentication hooks in PHP are only available when it is running as an Apache module and is hence not available in the CGI version. In an Apache module PHP script, it is possible to use the `header()` function to send an "Authentication Required" message to the client browser causing it to pop up a Username/Password input window. Once the user has filled in a username and a password, the URL containing the PHP script will be called again with the variables, `$PHP_AUTH_USER`,

`$PHP_AUTH_PW` and `$PHP_AUTH_TYPE` set to the user name, password and authentication type respectively. Only "Basic" authentication is supported at this point. See the `header()` function for more information. An example script fragment which would force client authentication on a page would be the following:

Example 2.2.1. HTTP Authentication example

```
<? php
if (!isset($PHP_AUTH_USER)) {
    header("WWW-Authenticate: Basic realm=\"My Realm\"");
    header("HTTP/1.0 401 Unauthorized");
    echo "Text to send if user hits Cancel button\n";
    exit;
} else {
    echo "<p>Hello $PHP_AUTH_USER.</p>";
    echo "<p>You entered $PHP_AUTH_PW as your password.</p>";
}
```

Instead of simply printing out the `$PHP_AUTH_USER` and `$PHP_AUTH_PW`, you would probably want to check the username and password for validity. Perhaps by sending a query to a database, or by looking up the user in a dbm file.

In order to prevent someone from writing a script which reveals the password for a page that was authenticated through a traditional external mechanism, the `PHP_AUTH` variables will not be set if external authentication is enabled for that particular page. In this case, the `$REMOTE_USER` variable can be used to identify the externally-authenticated user.

Configuration Note: PHP uses the presence of an `AuthType` directive to determine whether external authentication is in effect. Remember to avoid this directive for the context where you want to use PHP authentication (otherwise each authentication attempt will fail).

Note, however, that the above does not prevent someone who controls a non-authenticated URL from stealing passwords from authenticated URLs on the same server.

Both Netscape Navigator and Internet Explorer will clear the local browser window's authentication cache for the realm upon receiving a server response of 401. This can

effectively "log out" a user, forcing them to re-enter their username and password.

Some

people use this to "time out" logins, or provide a "log-out" button.

Example 2.8.2 HTTP Authentication example forcing a new name/password

```
<?php
function authenticate() {
    header( "WWW-Authenticate: Basic realm=\"Test Authentication System\"");
    header( "HTTP/1.0 401 Unauthorized");
    echo "You must enter a valid login ID and password to access this
resource\n";
    exit;
}
if (!isset($PHP_AUTH_USER) || ($SeenBefore == 1 && !strcmp($OldAuth,
$PHP_AUTH_USER))) {
    authenticate();
}
else {
    echo "<p>Welcome: $PHP_AUTH_USER<br>";
    echo "Old: $OldAuth";
    echo "<form action='$PHP_SELF' METHOD='POST'>\n";
    echo "<input type='hidden' name='SeenBefore' value='1'>\n";
    echo "<input type='hidden' name='OldAuth' value='$PHP_AUTH_USER'>\n";
    echo "<input type='submit' value='Re Authenticate'>\n";
    echo "</form></p>\n";
}
?>
```

This behavior is not required by the HTTP Basic authentication standard, so you should never depend on this. Testing with Lynx has shown that Lynx does not clear the authentication credentials with a 401 server response, so pressing back and then forward again will open the resource as long as the credential requirements haven't changed. The user can press the '_' key to clear their authentication information, however.

Also note that this does not work using Microsoft's IIS server and the CGI version of PHP due to a limitation of IIS.

CHAPTER THREE: WHAT IS MYSQL

3.1 Understanding MySQL

MySQL, pronounced "my Ess Que El," is an open source, Enterprise-level, multi-threaded, relational database management system. That sounds like a lot of sales or marketing hype, but it truly defines MySQL. You may not be familiar with some of these terms but, by the end of today, you will be.

MySQL was developed by a consulting firm in Sweden called TcX. They were in need of a database system that was extremely fast and flexible. Unfortunately (or fortunately, depending on your point of view), they could not find anything on the market that could do what they wanted. So, they created MySQL, which is loosely based on another database management system called mSQL. The product they created is fast, reliable, and extremely flexible. It is used in many places throughout the world.

Universities, Internet service providers and nonprofit organizations are the main users of MySQL, mainly because of its price (it is mostly free). Lately, however, it has begun to permeate the business world as a reliable and fast database system. Some examples of commercial use are available on the CD-ROM that accompanies this book.

The reason for the growth of MySQL's popularity is the advent of the Open Source Movement in the computer industry. The Open Source Movement, in case you haven't heard about it, is the result of several computer software vendors providing not only a product but the source code as well. This allows consumers to see how their program operates and modify it where they see fit. This, and the popularity of Linux, has given rise the use of open source products in the business world. Because of Linux's skyrocketing popularity, users are looking for products that will run on this platform. MySQL is one of those products.

MySQL is often confused with SQL, the structured query language developed by IBM. It is not a form of this language but a database system that uses SQL to manipulate, create, and show data. MySQL is a program that manages databases, much like Microsoft's Excel manages spreadsheets. SQL is a programming language that is used by MySQL to accomplish tasks within a database, just as Excel uses VBA (Visual Basic for Applications) to handle tasks with spreadsheets and workbooks. Other programs that manage databases include Microsoft's SQL Server, Sybase Adaptive Server, and DB2.

Now that you know where MySQL came from, look at what it is. To begin with, start with the term database. What is a database? You have probably used one in your lifetime. If you've ever bought anything over the Internet or have a driver's license, you can be assured that you have used one.

A database is a series of structured files on a computer that are organized in a highly efficient manner. These files can store tons of information that can be manipulated and called on when needed. A database is organized in the following hierarchical manner, from the top down. You start with a database that contains a number of tables. Each table is made up of a series of columns.

Data is stored in rows, and the place where each row intersects a column is known as a field. **Figure 3.1** depicts this breakdown. For example, at your favorite online book store there is a database. This database is made up of many tables. Each table contains specific, common data. You would probably see an Authors table or a Books table. These tables are made up of named columns that tell what data is contained in them. When a record is inserted into a table, a row of data has been created. Where a row and a column intersect, a field is created. This is how databases are broken down.

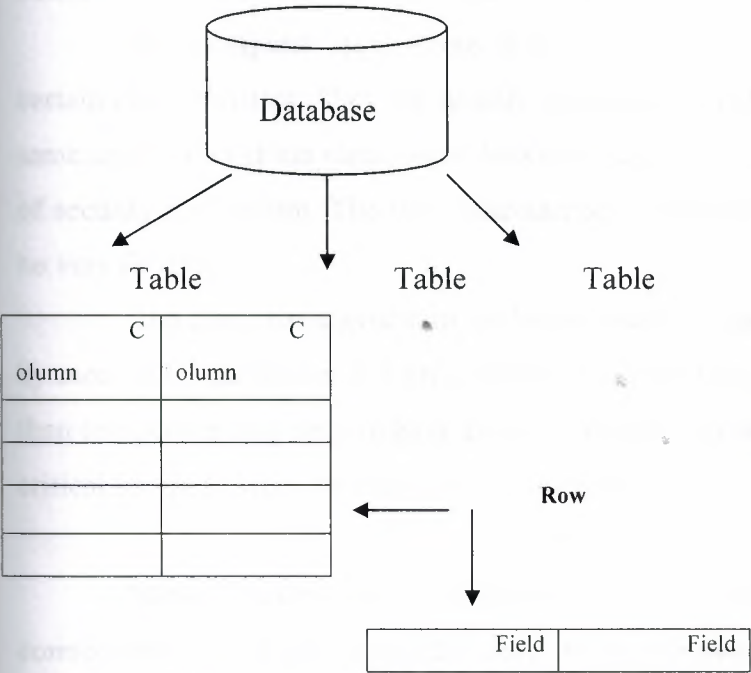


Figure 3.1 The anatomy of a database.

MySQL is more than just a database. It is a system that manages databases. It controls who can use them and how they are manipulated. It logs actions and runs

continuously in the background. This is different from what you may be used to. Most people think about Microsoft Access or Lotus Approach when they think about databases. These are databases, but they are not management systems.

A DBMS can contain many databases. Users connect to the database server and issue requests. The database server queries its databases and returns the requests to the issuers. Databases, such as Approach and Access, are a step down from this type of system. They share their files with multiple users, but there is no interface controlling the connections or answering requests.

There are many uses for a DBMS such as MySQL. Uses can range from help desk systems to Web site applications. The important thing to remember is that MySQL is large enough and quick enough to function in almost any situation. Where it finds itself most comfortable is the Enterprise.

3.1.1 What Is the Enterprise?

The Enterprise is not a starship or a space shuttle. The *Enterprise* is the area in the business world where many large systems interact with one another to accomplish a common goal. Some applications that are at this level of business include SAP, Microsoft SQL Server, Oracle 8i, and Sybase Adaptive Server.

The computer applications that exist at this level of business tend to have certain characteristics. They are usually multi-user in nature—many people can use the same application at the same time. Another characteristic is that they provide some sort of security mechanism. The final characteristic is that applications at this level have to be very flexible.

The first characteristic of an Enterprise-level application is that it can be used by more than one person at a time. This is a requirement at this level of business. More than one person may need to have access to business information at a given time. This is critical for the business to function successfully.

MySQL meets this requirement. It can have up to 101 simultaneous connections. This doesn't mean that only 101 people can use this application. It means it can have 101 connections going on at the same time—which is a little different. A connection is the time it takes for a user to receive the data that he or she has requested. In the case of MySQL, this is hardly any time at all. Most database systems in the same

class as MySQL allow fewer simultaneous connections. Currently, the only DBMS to offer more connections is Microsoft SQL Server.

The next characteristic that an Enterprise-level application must have is security. When dealing with mission-critical information, only people with the need to know should be allowed to view it. Security keeps malicious people at bay; without it, disasters can happen. MySQL meets this requirement.

The security in MySQL is unparalleled. Access to a MySQL database can be determined from the remote machine that can control which user can view a table. The database can be locked down even further by having the operating system play a role in security as well. Very few databases in the same class as MySQL can compare to the level of security that MySQL provides.

3.1.2. How hard is it to change?

MySQL answers these questions very well. It is extremely flexible and easy to use. MySQL can run on almost any platform. If a new CIO wants to change from Windows NT to Linux, fine—MySQL can adapt. MySQL also comes with the source code. If there are any deep-level changes that you need to make, you can edit the source and make these changes yourself.

If MySQL is missing a feature that you can't live without, just add it yourself. No other database on the market can offer you that kind of flexibility. MySQL also has several application-level interfaces in a variety of languages. If yours is mainly a Microsoft shop, you can use ODBC to interact with MySQL. If your company is a UNIX shop, you can use C, Perl, or JDBC. There is no end to the flexibility that MySQL has to offer.

In addition to the previously discussed characteristics, databases at the Enterprise level must be able to work together. Data warehousing is a technique that combines all the data in a business. Because of the flexibility and speed that MySQL has to offer, it can work well in any situation.

The Internet has also become a piece of the Enterprise pie. No large corporation is without an Internet presence. These corporations need databases to sell and compete at this level of business.

MySQL works well as an Internet-based database server. It has been proven in this arena and is the preferred database of many Internet service providers. Because of its speed and multiple application interfaces, MySQL is an ideal choice.

Enterprise applications are the crucial component to a business's decision-making power. Information must be timely and accurate for a business to perform effectively. To do this, applications must work quickly. An application is much like a car. It can look pretty on the outside, but the engine is what gives it its power. The same applies to an application; If its database engine is weak, so is the application. MySQL is clearly the choice for the Enterprise.

3.1.3. What Is a Relational Database?

A *relational database*, simply defined, is a database that is made up of tables and columns that relate to one another. These relationships are based on a key value that is contained in a column. For example, you could have a table called Orders that contains all the information that is required to process an order, such as the order number, date the item was ordered, and the date the item was shipped. You could also have a table called Customers that contains all the data that pertains to customers, such as a name and address. These two tables could be related to each other.

The relational database model was developed by E.F. Codd back in the early 1970s. He proposed that a database should consist of data stored in columns and tables that could be related to each other. This kind of thinking was very different from the hierarchical file system that was used at the time. His thinking truly revolutionized the way databases are created and used.

A relational database is very intuitive. It mimics the way people think. People tend to group similar objects together and break down complex objects into simpler ones. Relational databases are true to this nature. Because they mimic the way you think, they are easy to use and learn. In later days, you will discover how easy a relational database is to design and learn.

Most modern databases use a relational model to accomplish their tasks. MySQL is no different. It truly conforms to the relational model. This further adds to the ease of use of MySQL.

3.2 Designing Databases

The first step is always to create the database, unless you want to use an existing third party's one. When a database is created, it is assigned to an owner, who executed the creation statement. Usually, only the owner (or a superuser) can do anything with the objects in that database, and in order to allow other users to use it,

privileges must be granted. Applications should never connect to the database as its owner or a super user, because these users can execute any query at will, for example, modifying the schema (e.g. dropping tables) or deleting its entire content.

You may create different database users for every aspect of your application with very limited rights to database objects. The most required privileges should be granted only, and avoid that the same user can interact with the database in different use cases. This means that if intruders gain access to your database using one of these credentials, they can only effect as many changes as your application can.

You are encouraged not to implement all the business logic in the web application (i.e. your script), instead to do it in the database schema using views, triggers or rules. If the system evolves, new ports will be intended to open to the database, and you have to re-implement the logic in each separate database client. Over and above, triggers can be used to transparently and automatically handle a field, which often provides insight when debugging problems with your application or tracing back transactions.

3.2.1 Connecting to Database

You may want to establish the connections over SSL to encrypt client/server communications for increased security, or you can use ssh to encrypt the network connection between clients and the database server. If either of them is done, then monitoring your traffic and gaining information's in this way will be a hard work.

3.2.2 Encrypted Storage Model

SSL/SSH protects data traveling from the client to the server, SSL/SSH does not protect the persistent data stored in a database. SSL is an on-the-wire protocol.

Once an attacker gains access to your database directly (bypassing the web server), the stored sensitive data may be exposed or misused, unless the information is protected by the database itself. Encrypting the data is a good way to mitigate this threat, but very few databases offer this type of data encryption.

The easiest way to work around this problem is to first create your own encryption package, and then use it from within your PHP scripts. PHP can assist you in this case with its several extensions, such as Mcrypt and Mhash, covering a wide variety of encryption algorithms. The script encrypts the data be stored first, and decrypts it when retrieving. See the references for further examples how encryption works.

In case of truly hidden data, if its raw representation is not needed (i.e. not be displayed), hashing may be also taken into consideration. The well-known example for the hashing is storing the MD5 hash of a password in a database, instead of the password itself. See also `crypt()` and `md5()`.

3.2.3 The Client/Server Paradigm

The client/server paradigm or model has been around a lot longer than most people think. If you look back to the early days of programming, you remember or have heard or read about the large mainframe computer with many smaller "dumb" terminals. These terminals were called dumb for a reason. No logic or processing was done at the terminals. They were just receptacles for the output of the mainframe. This was the dawn of the client/server age, but the term client/server wasn't the buzzword it is today.

As the personal computer became more prevalent, giving rise to the local area network (LAN), the client/server model evolved. Now processing could be done at the client. Clients started sharing data. This data was stored in sharable computers called file servers. Now, instead of all the processing being done at the server, it was all being done at the client. The server or centralized computer was just a large storage device. It did little or no processing—a complete reversal of earlier thinking. After a couple of years, desktop applications became more powerful. People needed to share more information more quickly. This gave rise to the more powerful server machines. These machines answered requests from clients and processed them. These servers are what you know today as database servers, Web servers, and file servers. This is when people started calling it client/server computing. It is basically a two-tier design; a client issues requests, and a server answers them. All the business logic is at the application level on the client. Two-tier design is still very prevalent today. This is also known as a *fat client* because all the application processing is done at the client level. After a couple of years, servers became the powerhouses of business organizations because of their duties. They were usually top-of-the-line systems with the best hardware and were tweaked for speed. So, it was just a matter of time before someone came up with the idea of moving the guts of their programs to the server. The client would just be a graphical user interface (GUI) and the main application or business logic would be processed on the server. The server would then make the necessary calls to other servers, such as database servers or *file servers, as needed*. This gave birth to the three-tier or *thin client* design. In this design, all processing of the business logic is done at the server level. This allows the

more powerful machine to handle the logic and the slower machines to display the output. Does this sound familiar? It should—we've come full circle. The heavy processing is again done on the more powerful, centralized machines, while all the client machines do is display the output.

The Internet is a prime example of thin client architecture. A very thin client—the browser—sends requests to a Web server, which sends a response back to the browser. The browser then displays the requested information—completely full circle.

Again, we are on the verge of a new era in computing. Applications are becoming more balanced across the network. Because of a decline in computer prices, very good machines are showing up on the desktop as clients. This allows applications to pick up the slack and perform some processing. Server applications are becoming more advanced as well. You can now run functions remotely and accomplish distributed computing fairly easily. These advancements allow your applications to be more robust in nature and more useful to your business.

3.2.4 History of Mysql

We started out with the intention of using the mSQL database system to connect to our tables using our own fast low-level (ISAM) routines. However, after some testing, we came to the conclusion that mSQL was not fast enough or flexible enough for our needs. This resulted in a new SQL interface to our database but with almost the same API interface as mSQL. This API was designed to allow third-party code that was written for use with mSQL to be ported easily for use with MySQL.

The derivation of the name MySQL is not clear. Our base directory and a large number of our libraries and tools have had the prefix “my” for well over 10 years. However, co-founder Monty Wideness’ daughter is also named My. Which of the two gave its name to MySQL is still a mystery, even for us.

The name of the MySQL Dolphin (our logo) is “Sakila,” which was chosen by the founders of MySQL AB from a huge list of names suggested by users in our “Name the Dolphin” contest. The winning name was submitted by Ambrose Twebaze, an Open Source software developer from Swaziland, Africa. According to Ambrose, the feminine name Sakila has its roots in SiSwati, the local language of Swaziland. Sakila is

also the name of a town in Arusha, Tanzania, near Ambrose's country of origin, Uganda.

3.2.5 How Large MySQL Tables Can Be

MySQL 3.22 had a 4GB (4 gigabyte) limit on table size. With the MyISAM storage engine in MySQL 3.23, the maximum table size was increased to 65536 terabytes (2567 – 1 bytes). With this larger allowed table size, the maximum effective table size for MySQL databases is usually determined by operating system constraints on file sizes, not by MySQL internal limits.

The InnoDB storage engine maintains InnoDB tables within a table space that can be created from several files. This allows a table to exceed the maximum individual file size. The table space can include raw disk partitions, which allows extremely large tables. The maximum table space size is 64TB.

3.2.6 Features of MySQL

MySQL is a full-featured relational database management system. It is very stable and has proven itself over time. MySQL has been in production for over 10 years. MySQL is a multithreaded server. *Multithreaded* means that every time someone establishes a connection with the server, the server program creates a thread or process to handle that client's requests. This makes for an extremely fast server. In effect, every client who connects to a MySQL server gets his or her own thread.

MySQL is also fully ANSI SQL92-compliant. It adheres to all the standards set forth by the American National Standards Institute. The developers at TcX take these standards seriously and have carefully adhered to them.

Note: ANSI SQL92 is a set of standards for the Structured Query Language that was agreed on in 1992 by the American National Standards Institute. Another valuable feature of MySQL is its online help system. All commands for MySQL are given at a command prompt. To see which arguments the commands take or what the utility or command does, all you have to do is type the command and include the -help or -? switch. This will display a slew of information about the command.

Yet another feature of MySQL is its portability—it has been ported to almost every platform. This means that you don't have to change your main platform to take advantage of MySQL. And if you do want to switch, there is probably a MySQL port for your new platform.

MySQL also has many different application programming interfaces (APIs). They include APIs for Perl, TCL, Python, C/C++, Java (JDBC), and ODBC. So no matter what your company's expertise is, MySQL has a way for you to access it.

MySQL is also very cheap. For an unlicensed, full version of MySQL, the cost is nothing. To license your copy will currently cost you \$200. This is an incredible deal, considering what you are getting for your money. Database systems that provide half the features that MySQL has can cost tens of thousands of dollars. MySQL can do what they do better and for less.

CHAPTER FOUR: INSTALLATION APACHE SERVER

4.1 Setup a Directory Structure

Firstly, I make a directory where you can put PHP and Apache to avoid any potential problems, it's best to create this directory on a path that does not have any spaces. I put mine on a second partition, in the Web Server directory of the Web folder where I keep all my web development data:

F: WebWebServer

Now it's ready to install. Double-click on the installer package apache.exe. Read the agreements and information, clicking next to move through each screen.

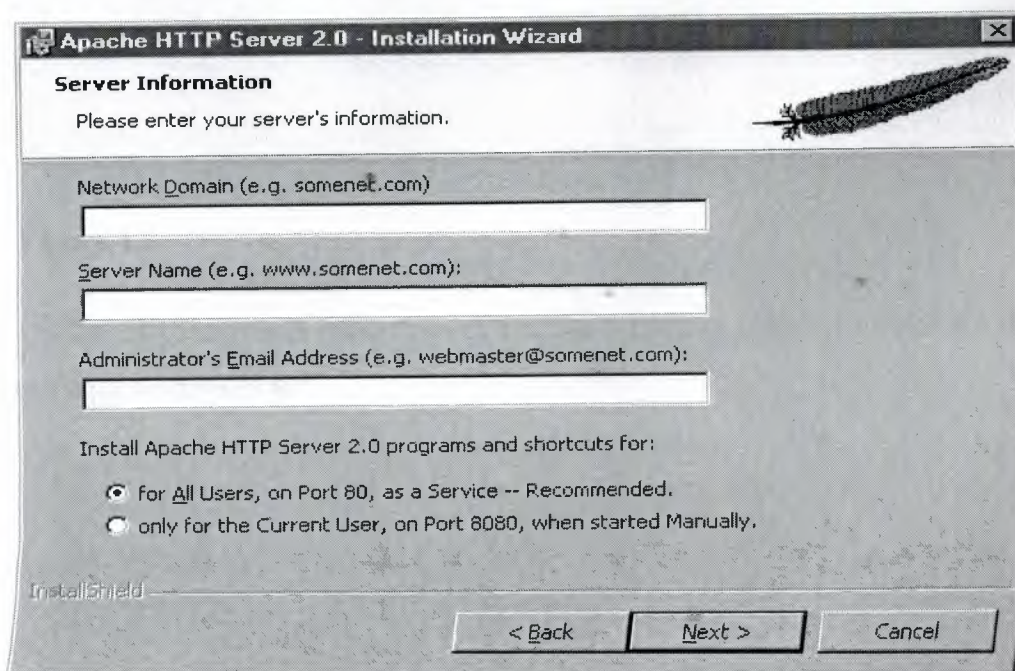


Figure 4.1 Apache Server Installation

If you are going to be using Apache only for testing scripts on your own machine, you can enter localhost for the first two values and anything you'd like for the email address (admin@localhost). If you're only going to be using Apache sometimes and only when you are logged on, it does not make sense to run it as a service, because it will eat up system resources and potentially pose a security risk. Select "only for the Current User, on Port 8080, when started manually". Press "Next" .we are writing local host, localhost and admin@localhost and selecting port 80

The next screen you see will look like this:

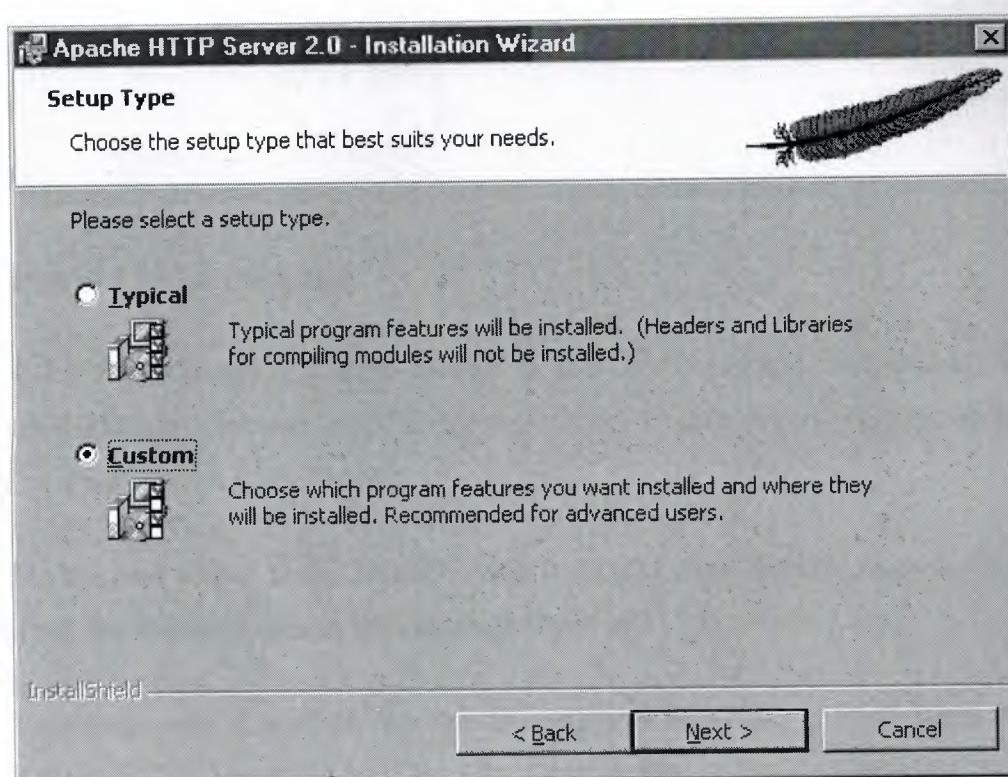


Figure 4.2

Choose "Custom" and press "Next". On the following screen you will see the default path Apache installs to:

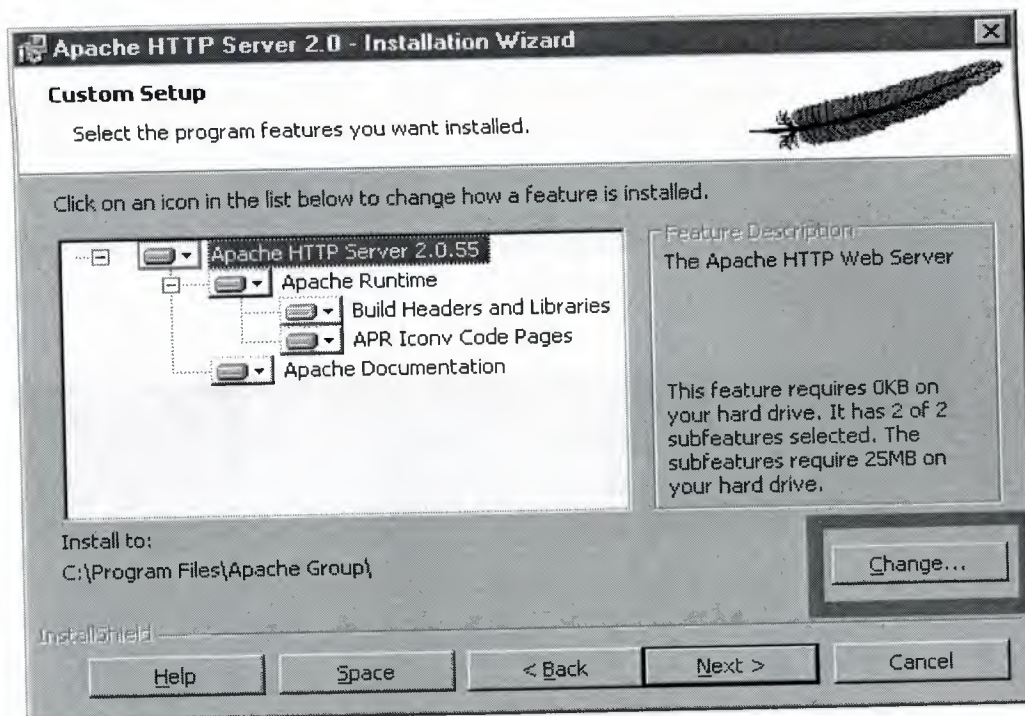


Figure 4.3 Custom Setup


Click the "Change..." button (boxed in red in the picture), and choose the WebServer folder (or whatever other name you chose) I made when you created the directory structure above. Click "Next".

On the next screen click "Install". After it installs click "Finish". Apache. It is located in my WebServer folder in the Apache2 directory.

Let's make sure it worked. Go to your programs folder in the Start menu and find the Apache HTTP Server folder. Go to the Control Apache Server folder, and start Apache



Figure 4.4 Choose Apache Server

If you installed Apache as a service, you can also start it by double-clicking on the Apache Service Monitor tray icon (which looks like: ) and pressing

the "Start" button in the window that appears. If you did not install it as a service, you will see a blank command prompt window appear while Apache is running. This is normal.

The first time you run Apache, you will probably see a security warning pop up:

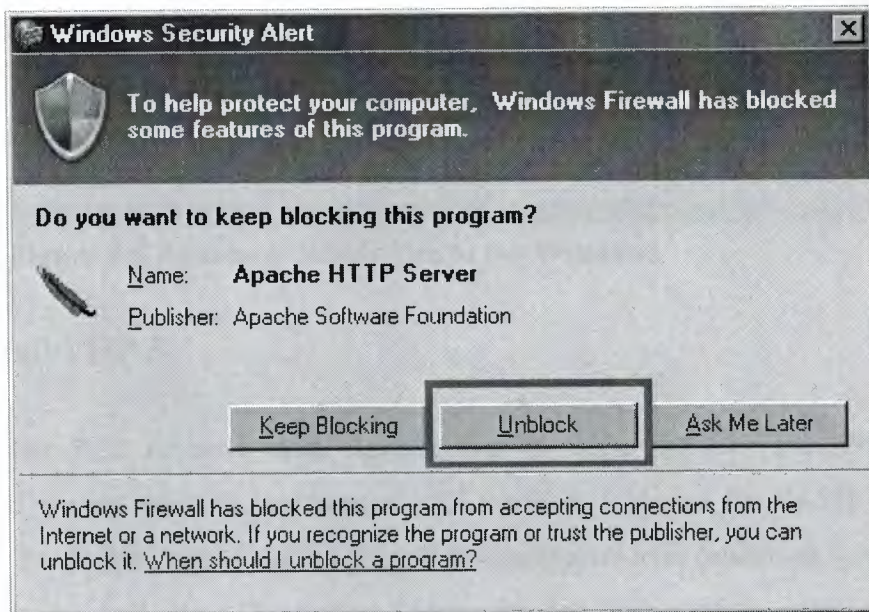


Figure 4.5 Click the Unblock

This is because windows has detected that Apache is opening a port on your computer, and it is checking with you about whether it should let Apache continue. Click the "Unblock" button to allow Apache to function.

Now, let's open up a web browser and make sure Apache is running. Type in the name of my server (local host) in the location bar of the browser.

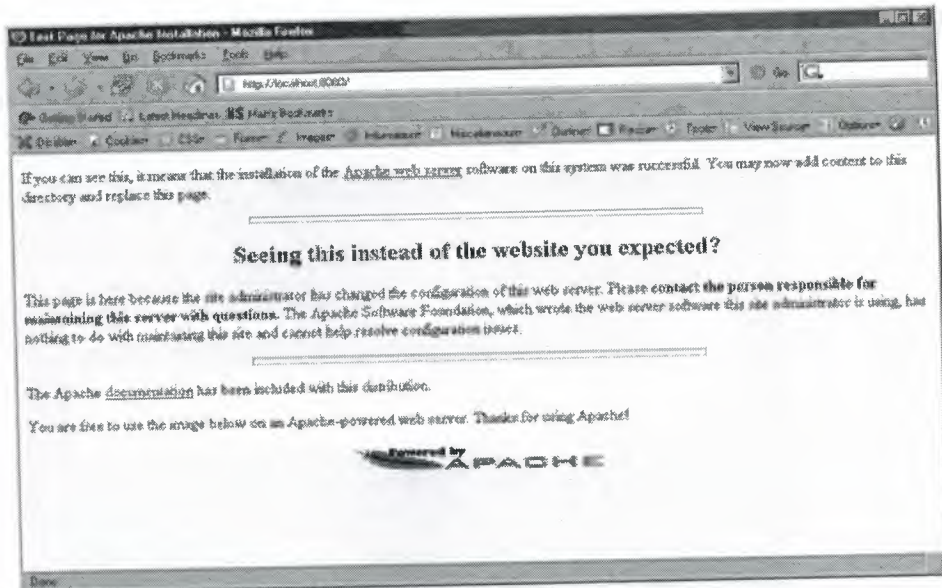


Figure 4.6 Apache is Ready Use to for Windows

4.2 Install PHP 5

For PHP to work with Apache and to make the CLI available from any command prompt window, we need to add the PHP folder to the PATH environment variable. To do this, right click on your "My Computer" icon (either on your desktop or the start menu) and select "Properties." In the window that pops up, select the advanced tab, and click on the "Environment Variables" button.

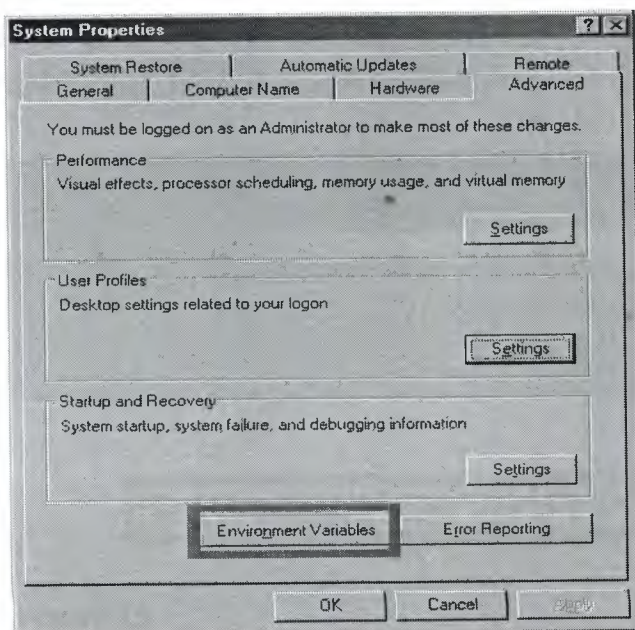


Figure 4.7

In the window that appears, select the "Path" variable line from the "System Variables" menu, and click edit:

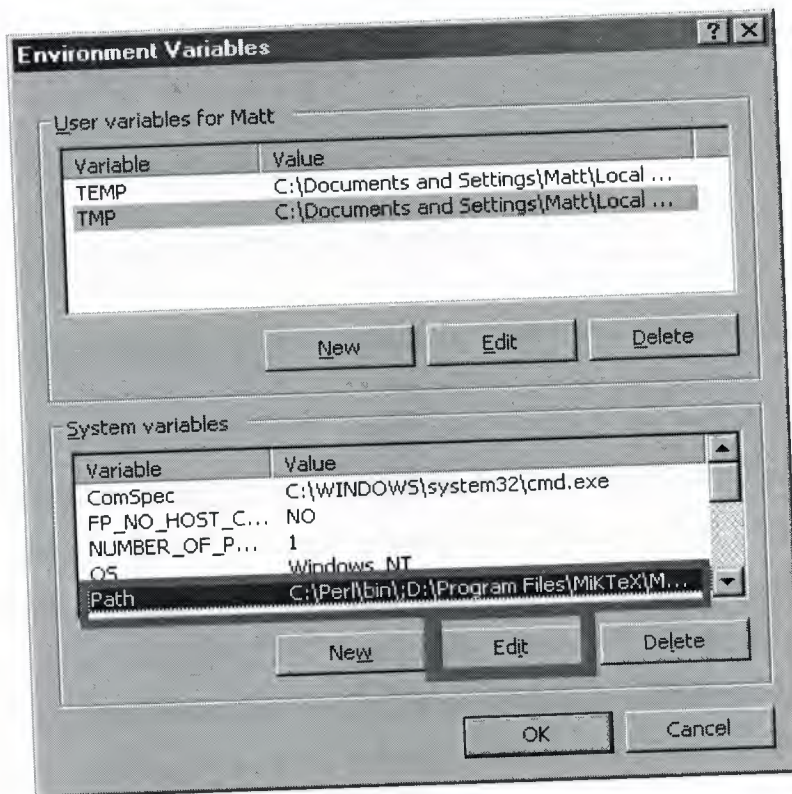


Figure 4.8

At the end of the "Variable Value" field, type a semicolon (;) and the type in the full path to your PHP folder. For me, this means typing the following at the end of the "Variable Value" field; D:WebWebServerPHP.

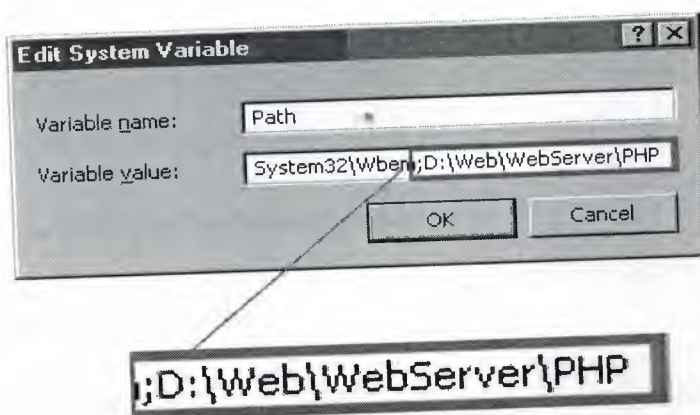


Figure 4.9

Hit OK in each window until they have all closed, and restart your computer. Pick up back here when you've restarted.

OK, you restarted and you're back. Let's make sure PHP is working. Go to Start->Run and type cmd and press OK to bring up a command window. At the command prompt, type php -v and press Enter. If you updated the *PATH* variable correctly, you should see something like:

-PHP 5.1.2 (cli) (built: Jan 11 2006 16:40:00)

-Copyright (c) 1997-2006 The PHP Group

-Zend Engine v2.1.0, Copyright (c) 1998-2006 Zend Technologies

Now open your PHP folder, and change the name of the file php.ini-recommended to php.ini. The settings in this file are all pretty much exactly as they should be. If you are installing Apache and PHP on your own system for testing purposes, it is probably a good idea to open your php.ini file in a text editor, and change the setting on or about line 357 from `display_errors = Off` to `display_errors = On` so that you will see any errors generated by your scripts and will be able to easily debug them. If you are planning on making your server public, it is best to leave this setting Off, because displaying errors can expose information about your file system, databases, and server configuration that should be kept private. Either way, the setting on line 367, `log_errors = On` will ensure that you will have a log of errors that you can review.

Make the following changes to the file:

Line 512: `doc_root = D:\Web\WebServer\Apache2\htdocs`

Line 519: `extension_dir = D:\Web\WebServer\PHP\ext`

Save the php.ini file and close it.

4.2.1 Configuring Apache

Now that we've got PHP and Apache working separately, we need to get them to work together. Go to your Apache2/conf folder, and open up the httpd.conf file in a text editor. You can get Apache to run PHP as either a CGI binary, or an Apache Module. Generally speaking, it is faster and more secure to do the latter, which you do by adding the following to your httpd.conf file:

At the top of the LoadModule section, starting around line 130, add the line `LoadModule php5_module "D:/Web/WebServer/PHP/php5apache2.dll"` (note those are forward slashes, not backslashes, in the path) where you replace `"D:/Web/WebServer/PHP/php5apache2.dll"` with the path to `php5apache2.dll` on your computer. You can actually add settings anywhere you would like, but putting the similar ones together helps keep things organized.

In the AddType section around line 754, add the following line:

```
AddType application/x-httpd-php .php
```

You can add other extensions besides `.php` if you want PHP to parse other file types as well. For example, if you want to be able to include PHP in `.html` documents:

```
AddType application/x-httpd-php .php .html
```

Finally, add the following line, somewhere that seems sensible to you:

```
PHPIniDir "D:/Web/WebServer/PHP/"
```

(again, replace the path I used with the one you chose on your computer).

Save the `httpd.conf` file, and start Apache as discussed above (or restart it if it is running).

And open our Apache2 `htdocs` folder. This is the root folder for all of our web server documents that you will view in your web browser. Start by erasing all the default files currently in there. Next, make a file called `phpinfo.php` that contains the following:

```
<?php phpinfo(); ?>
```

and save it in our `htdocs` folder. Now open a web browser and go to `http://localhost/phpinfo.php`. You should see a page that looks like

Then click "Install" and let it do its thing. When it's done, you can sign up for a MySQL.com account if you want to. Otherwise, choose "Skip Sign-Up" in the bottom left and press Next, and then press Finish on the next screen.

The MySQL Server Instance Configuration Wizard will pop up. Click "Next". Choose "Detailed Configuration" and click "Next". Answer the questions according to what kind of machine your computer is. The instructions are fairly straight forward. On the Windows Options Page, you may want to un-check the "Launch MySQL Server automatically" option, and check the Include Bin Directory in Windows PATH so that you can use the MySQL server from the windows command line. It is easiest to Run MySQL as a Windows service.

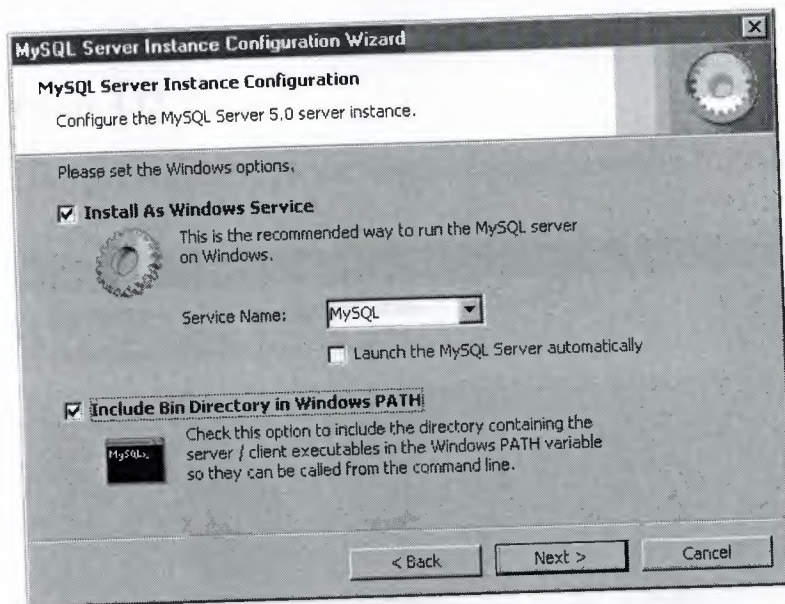


Figure 4.12

Note that you can use the Apache Service monitor to start and stop MySQL (and other services).

On the next screen, pick a root password. This is the main "superuser" password for your server:

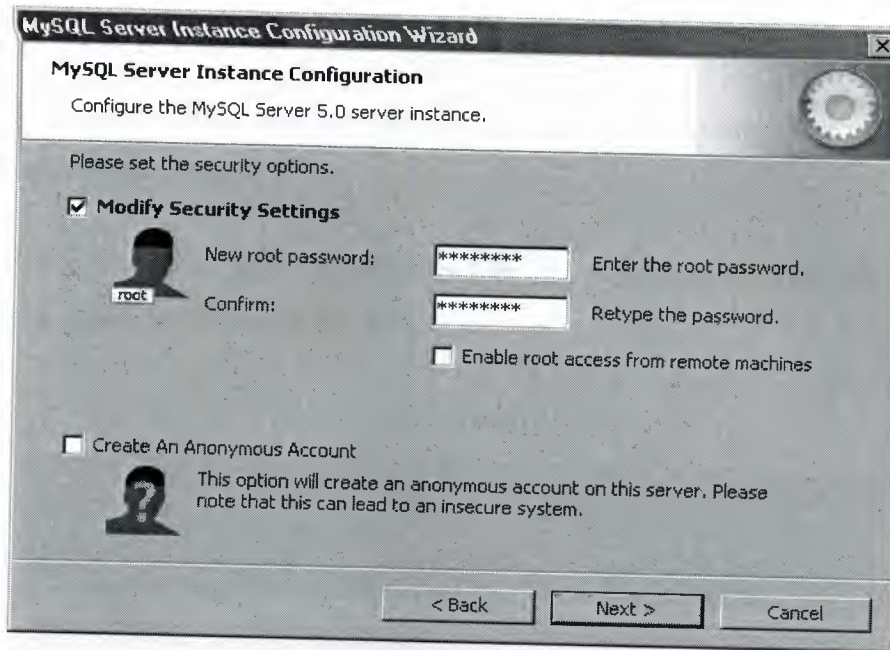


Figure 4.13

Finally, click the "Execute" button to configure your MySQL server

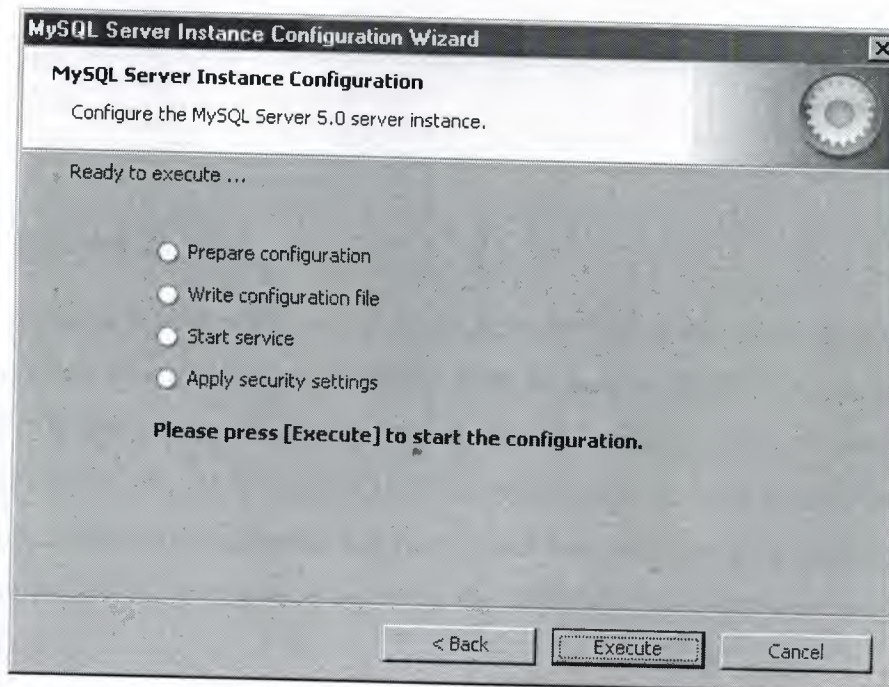


Figure 4.14

Its not a big deal if you do not configure your server right at first, you can always run the config wizard again later. It is in your MySQLbin folder, MySQLInstanceConfig.exe.

Restart (this is so your *PATH* variable gets updated).

Uncomment the line `extension=php_mysql.dll` (around line 650) in your `php.ini` and (re)start Apache.

Have a look at your `phpinfo.php` file. If you successfully set up MySQL, it should have a section that looks like this:

mysql

MySQL Support		enabled
Active Persistent Links		0
Active Links		0
Client API version		4.1.7

Directive	Local Value	Master Value
<code>mysql.allow_persistent</code>	On	On
<code>mysql.connect_timeout</code>	60	60
<code>mysql.default_host</code>	no value	no value
<code>mysql.default_password</code>	no value	no value
<code>mysql.default_port</code>	no value	no value
<code>mysql.default_socket</code>	no value	no value
<code>mysql.default_user</code>	no value	no value
<code>mysql.max_links</code>	Unlimited	Unlimited
<code>mysql.max_persistent</code>	Unlimited	Unlimited
<code>mysql.trace_mode</code>	Off	Off

Figure 4.15

That's all. Remember to use MySQL with PHP, it has to be running, and you need to create users, tables etc. for PHP to work with. Once MySQL is running, you can access it via the command line by typing `mysql -u root -p` and entering your root password (which you set during the MySQL installation) when prompted. You can also access the `mysqladmin` command this way to add other users to your databases (there is also a command line connection option in the MySQL section of your program list in the Start menu).

CHAPTER FIVE: WHAT IS APACHE SERVER?

5.1 What is Apache

The Apache web server project is more than just a piece of software. The Apache web server is the best, and most preferred, HTTP server software in use on the

Internet today, and it was written entirely as a volunteer project, by volunteer programmers, in their spare time. That, in itself, is astonishing. That is, it is to people that are not familiar with the Open Source methodology, and Open Source projects like Linux, Perl, Sendmail, and a variety of others. The interesting thing about these volunteer-written, free software packages is that most of us, and our businesses, rely heavily on them, whether we are aware of it or not.

Before diving directly into talking about what Apache is, it is useful to talk about where Apache came from, and how it came to be.

5.1.1 The Apache Server

When Rob left the project, it left a problem. There were still a lot of people using his code, and actively making patches to the code, but there was no longer anyone collecting those patches.

In 1995, Brian Behlendorf and a small group of other developers started collecting these patches in a central repository. Brian got some space donated on a server, and set up a CVS tree so that developers could check in patches. And in April of 1995, they released the first official release (Version 0.6.2), which was given the name Apache, because it was "a patchy server".

The Apache Group, as they were known at that time, had no formal organizational structure, never met, communicated only over email, and worked entirely in their free time, on a volunteer basis. Early the next year, Apache passed NCSA as the most widely used server on the Internet, and is now used on more than 60% of all web servers on the Internet.

5.1.2 Apache's architecture

Since the 1.0 release of Apache (December 1, 1995) Apache has has a modular design. The core of the server is very light-weight, and all other functions are implemented as modules that plug in to the core. This means that you can keep the size of the executable down by leaving out functionality that you don't need. It also means that if there is some functionality missing that you do need, you can write your own custom module to plug into the core.

5.1.3 More recent history

In the last few years, Open Source has been getting a lot of press, because of Linux, Perl, and Apache. In 1998, IBM decided to abandon development of a web server engine to go into WebSphere - an application server for the web - and use Apache instead. This decision, along with Netscape's decision to release the source code for the Netscape browser, earlier that same year, showed the business world that Open Source was more than just a lot of long-haired anti-establishment types out to bring down the software industry, but that it was actually a good business model. It produces code more quickly, and that code is more reliable, because, in the words of Eric Raymond, with enough eyes, all bugs are shallow.

In June of 1999, The Apache Software Foundation was officially incorporated in the state of Delaware. The ASF has a much broader mission than just the Apache HTTP server, and has several other projects that exist under the larger umbrella of the ASF

The stated goals of the ASF are:

- provide a foundation for open, collaborative software development projects by supplying hardware, communication, and business infrastructure;
- create an independent legal entity to which companies and individuals can donate resources and be assured that those resources will be used for the public benefit;
- provide a means for individual volunteers to be sheltered from legal suits directed at the Foundation's projects; and,
- protect the 'Apache' brand, as applied to its software products, from being abused by other organizations.

Some of the better-known projects under the ASF are the Apache web server, mod_perl, mod_php, and Jakarta.

5.1.4 The Future of Apache

At ApacheCon in Orlando, back in March, Apache 2.0 was released. This is largely a rewrite from earlier versions, and uses a threading model that will increase performance substantially on most platforms. As of this writing, version Alpha 6 of the 2.0 server has been released.

The Apache Group, as mentioned above, has become the Apache Software Foundation, and continues to take on new projects that seem to fit the larger vision that the ASF has for the future. Open Source, and open standards, produce better software. In the end, this makes life better for all of us, and we should support the ASF in all its endeavors, if only for purely selfish reasons.

5.1.5 Obtaining and Installing Apache

Apache is available as source code, and is probably available as a binary installation for your operating system, unless you are running something truly arcane and rare. And, of course, if you are, you can still get the source code, and compile it yourself.

5.1.6. Hardware / Software Requirements

Apache runs on anything. Almost. It will almost certainly run on whatever you have. I've run Apache on a 386 with 4 MB of RAM. And I've run it on a 4-processor machine with 1GB of RAM. It was happy both places. The Apache.org web site does not list any hardware requirements. It will run on any hardware that runs the supported operating systems. Apache will run on any flavor of *nix, and also on Microsoft Windows (95, 98, NT, 2000), Mac, and OS/2.

5.1.7. Configuring

Once you've compiled and installed your server, you need to configure it for your particular environment. Many of the configuration directives got set when you ran configure (or Configure), and so the server should work correctly immediately. However, you will probably want to change some things, since the default installation is very generic, and not precisely suited to your needs.

Apache, unlike most of its competitors in the web server market, lets you configure everything, down to the smallest detail. And if there's really something that you want to configure that you can't, you have the source code, so you can change it if you are so inclined.

5.1.8. Configuration Files

The configuration for your Apache server is located in a file called httpd.conf, which is usually located at /usr/local/apache/conf/httpd.conf.

Note that if you installed Apache with a RPM (don't do that!) then the files will be in bizarre places that have no relation to logic. Uninstall the RPM, and install from source. It's a simple process, and reduces your pain in the long run.

The format of httpd.conf is very simple.

There are comments, which consist of a hash sign (#) at the beginning of a line:

Based upon the NCSA server configuration files originally by Rob McCool.

There are directives, which look like a name, followed by a value:

```
ServerAdmin webmaster@rcbowen.com
```

There are sections, or containers, which look rather like HTML tags:

```
<Directory /usr/local/apache/cgi-bin>
```

```
    Allow Override None
```

```
</Directory>
```

Sections can contain directives, and those directives apply to the resources defined by the container definition. In the above example, the Allow Override directive will apply to files located in the specified directory.

You can edit these configuration files with your favorite text editor. You need to restart the server when you are done editing the configuration files in order for the new configuration to take effect.

```
/usr/local/apache/bin/apachectl configtest.
```


CHAPTER SIX: WHAT IS PHP

6.1 What is PHP?

PHP is a widely-used Open Source general-purpose scripting language that is especially suited for Web development and can be embedded into HTML.

Simple answer, but what does that mean? An example:

Example 6-1

```
<html> <head>
    <title>Example</title>
</head> <body>
    <?php
    echo "Hi, I'm a PHP script!";
    ?>
</body></html>
```

Notice how this is different from a script written in other languages like Perl or C++ instead of writing a program with lots of commands to output HTML, you write an HTML script with some embedded code to do something. The PHP code is enclosed in special start and end tags that allow you to jump into and out of "PHP mode".

What distinguishes PHP from something like client-side JavaScript is that the code is executed on the server. If you were to have a script similar to the above on your server, the client would receive the results of running that script, with no way of determining what the underlying code may be. You can even configure your web server to process all your HTML files with PHP, and then there's really no way that users can tell what you have up your sleeve.

The best things in using PHP are that it is extremely simple for a newcomer, but offers many advanced features for a professional programmer. Don't be afraid reading the long list of PHP's features. You can jump in, in a short time, and start writing simple scripts in a few hours.

6.2 What can PHP do?

Everything. PHP is mainly focused on server-side scripting, so you can do anything any other CGI program can do, such as collect form data, generate dynamic page content, or send and receive cookies. But PHP can do much more.

There are three main fields where PHP scripts are used.

Server-side scripting. This is the most traditional and main target field for PHP. You need three things to make this work. The PHP parser (CGI or server module), a webserver and a web browser. You need to run the webserver, with a connected PHP installation. You can access the PHP program output with a web browser, viewing the PHP page through the server. See the installation instructions section for more information.

Command line scripting. You can make a PHP script to run it without any server or browser. You only need the PHP parser to use it this way. This type of usage is ideal for scripts regularly executed using cron (on *nix or Linux) or Task Scheduler (on Windows). These scripts can also be used for simple text processing tasks. See the section about Command line usage of PHP for more information.

Writing client-side GUI applications. PHP is probably not the very best language to write windowing applications, but if you know PHP very well, and would like to use some advanced PHP features in your client-side applications you can also use PHP-GTK to write such programs. You also have the ability to write cross-platform applications this way. PHP-GTK is an extension to PHP, not available in the main distribution. If you are interested in PHP-GTK, visit its own website.

PHP can be used on all major operating systems, including Linux, many Unix variants (including HP-UX, Solaris and OpenBSD), Microsoft Windows, Mac OS X, RISC OS, and probably others. PHP has also support for most of the web servers today. This includes Apache, Microsoft Internet Information Server, Personal Web Server, Netscape and iPlanet servers, Oreilly Website Pro server, Caudium, Xitami, OmniHTTPd, and many others. For the majority of the servers PHP has a module, for the others supporting the CGI standard, PHP can work as a CGI processor.

So with PHP, you have the freedom of choosing an operating system and a web server. Furthermore, you also have the choice of using procedural programming or object oriented programming, or a mixture of them. Although not every standard OOP feature is realized in the current version of PHP, many code libraries and large applications (including the PEAR library) are written only using OOP code.

With PHP you are not limited to output HTML. PHP's abilities includes outputting images, PDF files and even Flash movies (using libswf and Ming) generated on the fly. You can also output easily any text, such as XHTML and any other XML file. PHP can autogenerate these files, and save them in the file system, instead of printing it out, forming a server-side cache for your dynamic content.

We also have a DBX database abstraction extension allowing you to transparently use any database supported by that extension. Additionally PHP supports ODBC, the Open Database Connection standard, so you can connect to any other database supporting this world standard.

PHP also has support for talking to other services using protocols such as LDAP, IMAP, SNMP, NNTP, POP3, HTTP, COM (on Windows) and countless others. You can also open raw network sockets and interact using any other protocol. PHP has support for the WDDX complex data exchange between virtually all Web programming languages. Talking about interconnection, PHP has support for instantiation of Java objects and using them transparently as PHP objects. You can also use our CORBA extension to access remote objects.

PHP has extremely useful text processing features, from the POSIX Extended or Perl regular expressions to parsing XML documents. For parsing and accessing XML documents, we support the SAX and DOM standards. You can use our XSLT extension to transform XML documents.

While using PHP in the ecommerce field, you'll find the Cybercash payment, CyberMUT, VeriSign Payflow Pro and C CVS functions useful for your online payment programs.

At last but not least, we have many other interesting extensions, the mnoGoSearch search engine functions, the IRC Gateway functions, many compression utilities (gzip, bz2), calendar conversion, translation...

As you can see this page is not enough to list all the features and benefits PHP can offer. Read on in the sections about installing PHP, and see the function reference part for explanation of the extensions mentioned here.

6.3 General Installation Considerations

Before installing first, you need to know what do you want to use PHP for. There are three main fields you can use PHP, as described in the What can PHP do? section:

- Server-side scripting

- Command line scripting

- Client-side GUI applications

For the first and most common form, you need three things: PHP itself, a web server and a web browser. You probably already have a web browser, and depending on

our operating system setup, you may also have a web server (eg. Apache on Linux or IIS on Windows). You may also rent webspace at a company. This way, you don't need to set up anything on your own, only write your PHP scripts, upload it to the server you rent, and see the results in your browser.

While setting up the server and PHP on your own, you have two choices for the method of connecting PHP to the server. For many servers PHP has a direct module interface (also called SAPI). These servers include Apache, Microsoft Internet Information Server, Netscape and iPlanet servers. Many other servers have support for ISAPI, the Microsoft module interface (OmniHTTPd for example). If PHP has no module support for your web server, you can always use it as a CGI processor. This means you set up your server to use the command line executable of PHP (php.exe on Windows) to process all PHP file requests on the server.

If you are also interested to use PHP for command line scripting (eg. write scripts autogenerating some images for you offline, or processing text files depending on some arguments you pass to them), you always need the command line executable. For more information, read the section about writing command line PHP applications. In this case, you need no server and no browser.

With PHP you can also write client side GUI applications using the PHP-GTK extension. This is a completely different approach than writing web pages, as you do not output any HTML, but manage windows and objects within them.

From now on, this section deals with setting up PHP for web servers on Unix and Windows with server module interfaces and CGI executables.

6.4 Installation on Windows Systems

This section applies to Windows 95/98/Me and Windows NT/2000/XP. Do not expect PHP to work on 16 bit platforms such as Windows 3.1. Sometimes we refer to the supported Windows platforms as Win32.

There are two main ways to install PHP for Windows: either manually or by using the InstallShield installer. If you have Microsoft Visual Studio, you can also build PHP from the original source code. Once you have PHP installed on your Windows system, you may also want to load various extensions for added functionality.

6.4.1. Windows InstallShield

The Windows PHP installer available from the downloads page at <http://www.php.net/>, this installs the CGI version of PHP and, for IIS, PWS, and Xitami, configures the web server as well. Also note, that while the InstallShield installer is an easy way to make PHP work, it is restricted in many aspects, as automatic setup of extensions for example is not supported.

Install your selected HTTP server on your system and make sure that it works.

Run the executable installer and follow the instructions provided by the installation wizard. Two types of installation are supported - standard, which provides sensible defaults for all the settings it can, and advanced, which asks questions as it goes along.

The installation wizard gathers enough information to set up the php.ini file and configure the web server to use PHP. For IIS and also PWS on NT Workstation, a list of all the nodes on the server with script map settings is displayed, and you can choose those nodes to which you wish to add the PHP script mappings.

Once the installation has completed the installer will inform you if you need to restart your system, restart the server, or just start using PHP.

6.4.2 Building From Source

Before getting started, it is worthwhile answering the question: "Why is building on Windows so hard?" Two reasons come to mind:

1. Windows does not (yet) enjoy a large community of developers who are willing to freely share their source. As a direct result, the necessary investment in infrastructure required to support such development hasn't been made.
2. Pretty much all of the instructions that follow are of the "set and forget" the.

6.4.3 Putting It All Together

Follow the instructions for installing the unzip utility of your choosing.

Execute setup.exe and follow the installation instructions. If you choose to install to a path other than c:\cygnus, let the build process know by setting the Cygwin environment variable. On Windows 95/98 setting an environment variable can be done by placing a line in your autoexec.bat. On Windows NT, go to My Computer => Control Panel => System and select the environment tab.

Make a directory and unzip win32build.zip into it.

Launch Microsoft Visual C++, and from the menu select Tools => Options. In the dialog, select the directories tab. Sequentially change the dropdown to Executables, Includes, and Library files, and ensure that cygwin\bin, win32build\include, and win32build\lib are in each list, respectively. (To add an entry, select a blank line at the end of the list and begin typing). Typical entries will look like this:

c:\cygnus\bin

c:\php-win32build\include

c:\php-win32build\lib

Press OK, and exit out of Visual C++.

Make another directory and unzip bindlib_w32.zip into it. Decide whether you want to have debug symbols available (bindlib - Win32 Debug) or not (bindlib - Win32 Release). Build the appropriate configuration:

For GUI users, launch VC++, and then select File => Open Workspace and select bindlib. Then select Build=>Set Active Configuration and select the desired configuration. Finally select Build=>Rebuild All.

For command line users, make sure that you either have the C++ environment variables registered, or have run vcvars.bat, and then execute one of the following:

msdev bindlib.dsp /MAKE "bindlib - Win32 Debug"

msdev bindlib.dsp /MAKE "bindlib - Win32 Release"

At this point, you should have a usable resolv.lib in either your Debug or Release subdirectories. Copy this file into your win32build\lib directory over the file by the same name found in there.

5.4.4 Compiling

The best way to get started is to build the standalone/CGI version.

For GUI users, launch VC++, and then select File => Open Workspace and select php4ts. Then select Build=>Set Active Configuration and select the desired configuration. Finally select Build=>Rebuild All.

For command line users, make sure that you either have the C++ environment variables registered, or have run vcvars.bat, and then execute one of the following:

msdev php4ts.dsp /MAKE "php4ts - Win32 Debug_TS"

msdev php4ts.dsp /MAKE "php4ts - Win32 Release_TS"

At this point, you should have a usable php.exe in either your Debug_TS or Release_TS subdirectories.

Repeat the above steps with `php4isapi.dsp` (which can be found in `sapi\isapi`) in order to build the code necessary for integrating PHP with Microsoft IIS.

6.5 Installation of Windows Extensions

After installing PHP and a webserver on Windows, you will probably want to install some extensions for added functionality. The following table describes some of the extensions available. You can choose which extensions you would like to load when PHP starts by uncommenting the: `'extension=php_*.dll'` lines in `php.ini`. You can also load a module dynamically in your script using `dl()`.

The DLLs for PHP extensions are prefixed with `'php_'` in PHP 4 (and `'php3_'` in PHP 3). This prevents confusion between PHP extensions and their supporting libraries.

Note: In PHP 4.0.6 BCMath, Calendar, COM, FTP, MySQL, ODBC, PCRE, Session, WDDX and XML support is built in. You don't need to load any additional extensions in order to use these functions. See your distributions `README.txt` or `install.txt` for a list of built in modules.

Note: Some of these extensions need extra DLLs to work. Couple of them can be found in the distribution package, in the `'dlls'` folder but some, for example Oracle (`php_oci8.dll`) require DLLs which are not bundled with the distribution package.

Copy the bundled DLLs from `'DLLs'` folder to your Windows PATH, safe places are:

`c:\windows\system` for Windows 9x/Me

`c:\winnt\system32` for Windows NT/2000

`c:\windows\system32` for Windows XP

If you have them already installed on your system, overwrite them only if something doesn't work correctly (Before overwriting them, it is a good idea to make a backup of them, or move them to another folder - just in case something goes wrong).

6.5.1 Windows and PWS/IIS 3

The recommended method for configuring these servers is to use the `REG` file included with the distribution (`pws-php4cgi.reg`). You may want to edit this file and make sure the extensions and PHP install directories match your configuration. Or you can follow the steps below to do it manually.

Run Regedit.

Navigate to: HKEY_LOCAL_MACHINE /System /CurrentControlSet
/W3Svc /Parameters /ScriptMap.

On the edit menu select: New->String Value.

Type in the extension you wish to use for your php scripts. For example .php

Double click on the new string value and enter the path to php.exe in the value
field. ex: c:\php\php.exe.

Repeat these steps for each extension you wish to associate with PHP scripts.

The following steps do not affect the web server installation and only apply if
you want your php scripts to be executed when they are run from the command line (ex.
c:\myscripts\test.php) or by double clicking on them in a directory viewer window.
You may wish to skip these steps as you might prefer the PHP files to load into a text
editor when you double click on them.

Navigate to: HKEY_CLASSES_ROOT

On the edit menu select: New->Key.

Name the key to the extension you setup in the previous section. ex: .php

Highlight the new key and in the right side pane, double click the "default
value" and enter phpfile.

Repeat the last step for each extension you set up in the previous section.

Now create another New->Key under HKEY_CLASSES_ROOT and name it
phpfile.

Highlight the new key phpfile and in the right side pane, double click the
"default value" and enter PHP Script.

Right click on the phpfile key and select New->Key, name it Shell.

Right click on the Shell key and select New->Key, name it open.

Right click on the open key and select New->Key, name it command.

Highlight the new key command and in the right side pane, double click the
"default value" and enter the path to php.exe. ex: c:\php\php.exe -q %1. (don't forget the

Exit Regedit.

If using PWS on Windows, reboot to reload the registry.

PWS and IIS 3 users now have a fully operational system. IIS 3 users can use a
tool from Steven Genusa to configure their script maps.

6.5.2 Windows and PWS 4 or newer

When installing PHP on Windows with PWS 4 or newer version, you have two options. One to set up the PHP CGI binary, the other is to use the ISAPI module DLL.

If you choose the CGI binary, do the following:

Edit the enclosed pws-php4cgi.reg file (look into the SAPI dir) to reflect the location of your php.exe. Forward slashes should be escaped, for example:
[HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\w3svc\parameters\Script Map] ".php"="c:\\php\\php.exe"

In the PWS Manager, right click on a given directory you want to add PHP support to, and select Properties. Check the 'Execute' checkbox, and confirm.

If you choose the ISAPI module, do the following:

Edit the enclosed pws-php4isapi.reg file (look into the SAPI dir) to reflect the location of your php4isapi.dll. Forward slashes should be escaped, for example:
[HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\w3svc\parameters\Script Map] ".php"="c:\\php\\sapi\\php4isapi.dll"

In the PWS Manager, right click on a given directory you want to add PHP support to, and select Properties. Check the 'Execute' checkbox, and confirm.

6.6 The Configuration File

The configuration file (called php3.ini in PHP 3.0, and simply php.ini as of PHP 4.0) is read when PHP starts up. For the server module versions of PHP, this happens only once when the web server is started. For the CGI version, it happens on every invocation.

```
include_path = ".;c:\php\lib"
```

When using PHP as an Apache module, you can also change the configuration settings using directives in Apache configuration files and .htaccess files (You will need "AllowOverride Options" or "AllowOverride All" privileges)

With PHP 3.0, there are Apache directives that correspond to each configuration setting in the php3.ini name, except the name is prefixed by "php3_".

With PHP 4.0, there are several Apache directives that allow you to change the PHP configuration from within the Apache configuration file itself.

```
php_value name value
```

This sets the value of the specified variable.

```
php_flag name on|off
```

This is used to set a Boolean configuration option.

`php_admin_value name value`

This sets the value of the specified variable. "Admin" configuration settings can only be set from within the main Apache configuration files, and not from .htaccess files.

`php_admin_flag name on/off`

This is used to set a Boolean configuration option.

Example 5-2. Apache configuration example

```
<IfModule mod_php4.c>
    php_value include_path ".:usr/local/lib/php"
    php_flag safe_mode on
</IfModule>

<IfModule mod_php3.c>
    php3_include_path ".:usr/local/lib/php"
    php3_safe_mode on </IfModule>
```

6.7 Security

PHP is a powerful language and the interpreter, whether included in a web server as a module or executed as a separate CGI binary, is able to access files, execute commands and open network connections on the server. These properties make anything run on a web server insecure by default. PHP is designed specifically to be a more secure language for writing CGI programs than Perl or C, and with correct selection of compile-time and runtime configuration options, and proper coding practices, it can give you exactly the combination of freedom and security you need.

As there are many different ways of utilizing PHP, there are many configuration options controlling its behaviour. A large selection of options guarantees you can use PHP for a lot of purposes, but it also means there are combinations of these options and server configurations that result in an insecure setup.

The configuration flexibility of PHP is equally rivalled by the code flexibility. PHP can be used to build complete server applications, with all the power of a shell user, or it can be used for simple server-side includes with little risk in a tightly controlled environment. How you build that environment, and how secure it is, is largely up to the PHP developer.

This chapter starts with some general security advice, explains the different configuration option combinations and the situations they can be safely used, and describes different considerations in coding for different levels of security.

5.8 General Considerations

A completely secure system is a virtual impossibility, so an approach often used in the security profession is one of balancing risk and usability. If every variable is submitted by a user required two forms of biometric validation (such as a retinal scan and a fingerprint), you would have an extremely high level of accountability. It would also take half an hour to fill out a fairly complex form, which would tend to encourage users to find ways of bypassing the security.

The best security is often in obtrusive enough to suit the requirements without the user being prevented from accomplishing their work, or over-burdening the code author with excessive complexity. Indeed, some security attacks are merely exploits of this kind of overly built security, which tends to erode over time.

A phrase worth remembering: A system is only as good as the weakest link in a chain. If all transactions are heavily logged based on time, location, transaction type, etc. but the user is only verified based on a single cookie, the validity of tying the users to the transaction log is severely weakened.

When testing, keep in mind that you will not be able to test all possibilities for even the simplest of pages. The input you may expect will be completely unrelated to the input given by a disgruntled employee, a cracker with months of time on their hands, or a housecat walking across the keyboard. This is why it's best to look at the code from a logical perspective, to discern where unexpected data can be introduced, and then follow how it is modified, reduced, or amplified.

The Internet is filled with people trying to make a name for themselves by breaking your code, crashing your site, posting inappropriate content, and otherwise making your day interesting. It doesn't matter if you have a small or large site, you are a target by simply being online, by having a server that can be connected to. Many cracking programs do not discern by size, they simply trawl massive IP blocks looking for victims. Try not to become one.

6.8.1 Installed as an Apache module

When PHP is used as an Apache module it inherits Apache's user permissions (typically those of the "nobody" user). This has several impacts on security and authorization. For example, if you are using PHP to access a database, unless that database has built-in access control, you will have to make the database accessible to the "nobody" user. This means a malicious script could access and modify the database, even without a username and password. It's entirely possible that a web spider could stumble across a database administrator's web page, and drop all of your databases. You can protect against this with Apache authorization, or you can design your own access model using LDAP, .htaccess files, etc. and include that code as part of your PHP scripts.

Often, once security is established to the point where the PHP user (in this case, the apache user) has very little risk attached to it, it is discovered that PHP is now prevented from writing any files to user directories. Or perhaps it has been prevented from accessing or changing databases. It has equally been secured from writing good and bad files, or entering good and bad database transactions.

A frequent security mistake made at this point is to allow apache root permissions, or to escalate apache's abilities in some other way.

Escalating the Apache user's permissions to root is extremely dangerous and may compromise the entire system, so sudo'ing, chroot'ing, or otherwise running as root should not be considered by those who are not security professionals.

There are some simpler solutions. By using `open_basedir` you can control and restrict what directories are allowed to be used for PHP. You can also set up apache-only areas, to restrict all web based activity to non-user, or non-system, files.

6.8.2 Installed as CGI binary

Using PHP as a CGI binary is an option for setups that for some reason do not wish to integrate PHP as a module into server software (like Apache), or will use PHP with different kinds of CGI wrappers to create safe chroot and setuid environments for scripts. This setup usually involves installing executable PHP binary to the web server `cgi-bin` directory. CERT advisory CA-96.11 recommends against placing any interpreters into `cgi-bin`. The query information in a url after the question mark (?) is passed as command line arguments to the interpreter by the CGI interface. Usually interpreters open and execute the file specified as the first argument on the command

When invoked as a CGI binary, PHP refuses to interpret the command line arguments.

Accessing any web document on server: `http://my.host/cgi-bin/php/secret/doc.html`. The path information part of the url after the PHP binary name, `/secret/doc.html` is conventionally used to specify the name of the file to be opened and interpreted by the CGI program. Usually some web server configuration directives (Apache: Action) With this setup, the web server first checks the access permissions to the directory `/secret`, and after that creates the redirected request `http://my.host/cgi-bin/php/secret/script.php`. Unfortunately, if the request is originally given in this form, no access checks are made by web server for file `/secret/script.php`, but only for the `/cgi-bin/php` file. This way any user able to access `/cgi-bin/php` is able to access any protected document on the web server.

CHAPTER SEVEN: APPLICATION CODES

7.1 User Interface Area Codes

7.1.1 Login_user.php

```
?php require_once('Connections/mahfuz.php'); ?><?php
*** Validate request to login to this site.
if (!isset($_SESSION)) {
    session_start();
    $loginFormAction = $_SERVER['PHP_SELF'];
    if (isset($_GET['accesscheck'])) {
        $_SESSION['PrevUrl'] = $_GET['accesscheck'];
    }
    if (isset($_POST['textfield'])) {
        $loginUsername=$_POST['textfield'];
        $password=$_POST['textfield2'];
        $MM_fldUserAuthorization = "group";
        $MM_redirectLoginSuccess = "1.php";
        $MM_redirectLoginFailed = "login.php?error1=1";
        $MM_redirecttoReferrer = true;
        mysql_select_db($database_mahfuz, $mahfuz);

        $LoginRS__query=sprintf("SELECT username, password, `group` FROM user
        WHERE username='%s' AND password='%s'",
        addslashes($_GET['accesscheck']) ? $loginUsername : addslashes($loginUsername),
        addslashes($_GET['accesscheck']) ? $password : addslashes($password));

        $LoginRS = mysql_query($LoginRS__query, $mahfuz) or die(mysql_error());
        $loginFoundUser = mysql_num_rows($LoginRS);
```

```

if ($loginFoundUser) {

    $loginStrGroup = mysql_result($LoginRS,0,'group');

    //declare two session variables and assign them
    $_SESSION['MM_Username'] = $loginUsername;
    $_SESSION['MM_UserGroup'] = $loginStrGroup;

    if (isset($_SESSION['PrevUrl']) && true) {
        $MM_redirectLoginSuccess = $_SESSION['PrevUrl'];
    }
    echo "<script language='javascript'>
parent.frames[1].location='3.php';
parent.frames[2].location='1.php';
</script>
";
    header("Location: " . $MM_redirectLoginSuccess );
}
else {
    header("Location: " . $MM_redirectLoginFailed );
}
}
?><!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head><script type="text/javascript">
function uyari()
{
    alert("Your Password or User Name Wrong! Please Try Again...")
}
</script>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />
<title>Untitled Document</title>
<style type="text/css">
<!--
body {
    background-color: #3366CC;
}
.style1 {
    font-family: Verdana, Arial, Helvetica, sans-serif;
    font-size: 10px;
}
a:link {
    color: #CC9900;
}
a:visited {
    color: #FF6600;
}
.style2 {color: #CCCCCC}
-->

```



```

le></head>

y <?PHP if ($error1=="1") {echo "onload=uyari()";} ?>>
n id="form1" name="form1" method="POST" action="<?php echo
nFormAction; ?>">
le width="216" border="0" align="center" cellpadding="0" cellspacing="0"
="border-collapse: collapse" bordercolor="#111111">
r>
th width="210" bgcolor="#999999" scope="col"><label for="textfield"></label>
<div align="right">
<label for="label2"><span class="style1">UserName </span></label>
<input name="textfield" type="text" id="textfield" size="18" maxlength="50" />
</div></th>
tr>
r>
th width="210" nowrap="nowrap" bgcolor="#999999"><label
'label"></label>
<div align="right"> <span class="style1">
<label for="label2"><strong>Password</strong></label>
</span>
<input name="textfield2" type="password" id="label" size="20" maxlength="20"

</div></th>
tr>
r>
td bgcolor="#999999"><label for="Submit"></label>
align="center">

<input type="submit" name="Submit" value="Login" id="Submit" />
</p>
</div></td>
tr>
ble>
m>
align="center" class="style1 style2"><a
"lostpassword.php?idvalue=65ey4hlsbsj7hsfjs87hs7ej" target="mainFrame">I
ot My Password</a></div>
dy>
nl>

```

Show producers List.php

```

p require_once('Connections/mahfuz.php'); ?>
p
ql_select_db($database_mahfuz, $mahfuz);
ry_Recordset1 = "SELECT * FROM person WHERE person.location='$location'";
ordset1 = mysql_query($query_Recordset1, $mahfuz) or die(mysql_error());
v_Recordset1 = mysql_fetch_assoc($Recordset1);
alRows_Recordset1 = mysql_num_rows($Recordset1);

```

```

mysql_select_db($database_mahfuz, $mahfuz);
$query_Recordset1 = "SELECT *
FROM person
WHERE person.location='$select'";
$Recordset1 = mysql_query($query_Recordset1, $mahfuz) or die(mysql_error());
$row_Recordset1 = mysql_fetch_assoc($Recordset1);
$totalRows_Recordset1 = mysql_num_rows($Recordset1);
?><!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />
<title>Untitled Document</title>
<style type="text/css">
<!--
body {
    background-color: #3366CC;
}
.style1 {color: #CCCCCC}
.style2 {color: #993300}
-->
</style>
<style type="text/css" media="all">
<!--
@import url("Eski Klasör/mm_entertainment.css");
.style3 {font-size: 12px}
a:link {
    color: #FFFFFF;
}
.style6 {font-size: 12px; font-family: Verdana, Arial, Helvetica, sans-serif; }
a:visited {
    color: #FF0000;
}
.style7 {font-family: Verdana, Arial, Helvetica, sans-serif}
.style8 {font-size: 12px; font-family: Verdana, Arial, Helvetica, sans-serif; color:
#FFFFFF; }
.style9 {
    color: #FFFFFF;
    font-family: Verdana, Arial, Helvetica, sans-serif;
    font-size: 10px;
}
-->
</style>
</head>

<body>
<table border="1" class="style2">
<tr>
<td bgcolor="#808080"><div align="center"><span class="style3
style1"><strong>Id</strong></span></div></td>

```



```
<td bgcolor="#808080"><div align="center"><span class="style3  
yle1"><strong>Name</strong></span></div></td>  
<td bgcolor="#808080"><div align="center"><span class="style3  
yle1"><strong>Surname</strong></span></div></td>  
<td bgcolor="#808080"><div align="center"><span class="style3  
yle1"><strong>Address</strong></span></div></td>  
<td bgcolor="#808080"><div align="center"><span class="style3  
yle1"><strong>Location</strong></span></div></td>  
<td bgcolor="#808080"><div align="center"><span class="style3  
yle1"><strong>Telephone</strong></span></div></td>  
<td bgcolor="#808080"><div align="center"><span class="style3  
yle1"><strong>Mobilphone</strong></span></div></td>  
<td bgcolor="#808080"><div align="center"><span class="style3  
yle1"><strong>Fax</strong></span></div></td>  
<td bgcolor="#808080"><span class="style3  
yle1"><strong>Email</strong></span></td>  
</tr>  
<?php do { ?>  
<tr>  
<td height="19" bgcolor="#0000FF"><div align="center">  
<form action="searchproduct.php" name="form1 l"> <input name="id"  
ype="submit" value="Product info" />  
<input name="id" type="hidden" value="<?php echo $row_Recordset1['id']; ?>"  
></form>  
</div></td>  
<td><div align="center"><span class="style8"><?php echo  
$row_Recordset1['name']; ?></span>&nbsp;</div></td>  
<td><div align="center"><span class="style8"><?php echo  
$row_Recordset1['surname']; ?></span>&nbsp;</div></td>  
<td><div align="center"><span class="style8"><?php echo  
$row_Recordset1['address']; ?></span>&nbsp;</div></td>  
<td><div align="center"><span class="style8"><?php echo  
$row_Recordset1['location']; ?></span>&nbsp;</div></td>  
<td><div align="center"><span class="style8"><?php echo  
$row_Recordset1['telephone']; ?></span>&nbsp;</div></td>  
<td><div align="center"><span class="style8"><?php echo  
$row_Recordset1['mobilphone']; ?></span>&nbsp;</div></td>  
<td><div align="center"><span class="style8"><?php echo $row_Recordset1['fax'];  
<td><span class="style8"><a href="sendemail.php?email=<?php echo  
$row_Recordset1['email']; ?>">Mail</a></span>&nbsp;</td>  
</tr>  
<?php } while ($row_Recordset1 = mysql_fetch_assoc($Recordset1)); ?>  
</table>  
<p>&nbsp;</p>  
<table width="387" border="0">  
<tr>  
<th width="362" scope="col"><div align="left"><span class="style8"><?php echo  
$totalRows_Recordset1 ?> </span><span class="style8">&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&~Record(s)  
found.</span></div></th>
```

```

    <th width="71" class="style8" scope="col">&nbsp;</th>
</tr>
</table>
<p class="style9">Click ID for person product information </p>
<p class="style9"><span class="style8">
    <input type="button" onclick="history.back(-1)"name="Button" value="Back" />
</span></p>
</body>
</html>
<?php
mysql_free_result($Recordset1);
?>

```

7.1.3 Edit Producers Information.php

```

<?php require_once('Connections/mahfuz.php'); ?>
<?php
if (!isset($_SESSION)) {
    session_start();
}
$MM_authoredUsers = "1";
$MM_donotCheckaccess = "false";

// *** Restrict Access To Page: Grant or deny access to this page
function isAuthorized($strUsers, $strGroups, $UserName, $UserGroup) {
    // For security, start by assuming the visitor is NOT authorized.
    $isValid = False;

    // When a visitor has logged into this site, the Session variable MM_Username set
    equal to their username.
    // Therefore, we know that a user is NOT logged in if that Session variable is blank.
    if (!empty($UserName)) {
        // Besides being logged in, you may restrict access to only certain users based on an
        ID established when they login.
        // Parse the strings into arrays.
        $arrUsers = Explode(",", $strUsers);
        $arrGroups = Explode(",", $strGroups);
        if (in_array($UserName, $arrUsers)) {
            $isValid = true;
        }
        // Or, you may restrict access to only certain users based on their username.
        if (in_array($UserGroup, $arrGroups)) {
            $isValid = true;
        }
        if (($strUsers == "") && false) {
            $isValid = true;
        }
    }
    return $isValid;
}

```



```

MM_restrictGoTo = "access.php";
if (!((isset($_SESSION['MM_Username'])) &&
sAuthorized("", $MM_authorizedUsers, $_SESSION['MM_Username'],
$_SESSION['MM_UserGroup']))) {
$MM_qsChar = "?";
$MM_referrer = $_SERVER['PHP_SELF'];
if (strpos($MM_restrictGoTo, "?") $MM_qsChar = "&";
if (isset($_QUERY_STRING) && strlen($_QUERY_STRING) > 0)
$MM_referrer .= "?" . $_QUERY_STRING;
$MM_restrictGoTo = $MM_restrictGoTo . $MM_qsChar . "accesscheck=" .
urlencode($MM_referrer);
header("Location: " . $MM_restrictGoTo);
exit;
}
}
?php
function GetSQLValueString($theValue, $theType, $theDefinedValue = "",
$theNotDefinedValue = "")
{
$theValue = (!get_magic_quotes_gpc()) ? addslashes($theValue) : $theValue;

switch ($theType) {
case "text":
$theValue = ($theValue != "") ? "'" . $theValue . "'" : "NULL";
break;
case "long":
case "int":
$theValue = ($theValue != "") ? intval($theValue) : "NULL";
break;
case "double":
$theValue = ($theValue != "") ? "'" . doubleval($theValue) . "'" : "NULL";
break;
case "date":
$theValue = ($theValue != "") ? "'" . $theValue . "'" : "NULL";
break;
case "defined":
$theValue = ($theValue != "") ? $theDefinedValue : $theNotDefinedValue;
break;
}
return $theValue;
}

$editFormAction = $_SERVER['PHP_SELF'];
if (isset($_SERVER['QUERY_STRING'])) {
$editFormAction .= "?" . htmlentities($_SERVER['QUERY_STRING']);
}

if ((isset($_POST["MM_update"])) && ($_POST["MM_update"] == "form1")) {

```

```

$updateSQL = sprintf("UPDATE person SET name=%s, surname=%s, address=%s,
location=%s, telephone=%s, mobilphone=%s, fax=%s, email=%s WHERE id=%s",
    GetSQLValueString($_POST['name'], "text"),
    GetSQLValueString($_POST['surname'], "text"),
    GetSQLValueString($_POST['address'], "text"),
    GetSQLValueString($_POST['location'], "text"),
    GetSQLValueString($_POST['telephone'], "text"),
    GetSQLValueString($_POST['mobilphone'], "text"),
    GetSQLValueString($_POST['fax'], "text"),
    GetSQLValueString($_POST['email'], "text"),
    GetSQLValueString($_POST['id'], "int"));

```

```

mysql_select_db($database_mahfuz, $mahfuz);
$result1 = mysql_query($updateSQL, $mahfuz) or die(mysql_error());

```

```

mysql_select_db($database_mahfuz, $mahfuz);
$query_Recordset1 = "SELECT * FROM person WHERE person.id='$id'";
$result1 = mysql_query($query_Recordset1, $mahfuz) or die(mysql_error());
$row_Recordset1 = mysql_fetch_assoc($result1);
$totalRows_Recordset1 = mysql_num_rows($result1);

```

```

mysql_select_db($database_mahfuz, $mahfuz);
$query_Recordset2 = "SELECT location.name FROM location";
$result2 = mysql_query($query_Recordset2, $mahfuz) or die(mysql_error());
$row_Recordset2 = mysql_fetch_assoc($result2);
$totalRows_Recordset2 = mysql_num_rows($result2);

```

```

<style type="text/css">

```

```

<!--
style3 {color: #FFFFFF; font-weight: bold; }
body {

```

```

    background-color: #3366CC;

```

```

style4 {color: #FFFFFF}
-->

```

```

</style>

```

```

<body class="menu">

```

```

<link href="CSS/Level2_Verdana_Forms.css" rel="stylesheet" type="text/css" />

```

```

<form method="post" name="form1" action="<?php echo $editFormAction; ?>">

```

```

    <table align="center">

```

```

        <tr valign="baseline">

```

```

            <td nowrap align="right"><span class="style3">Id:</span></td>

```

```

            <td><span class="style4"><?php echo $row_Recordset1['id']; ?></span></td>

```

```

        </tr>

```

```

        <tr valign="baseline">

```

```

            <td nowrap align="right"><span class="style3">Name:</span></td>

```

```

            <td><input type="text" name="name" value="<?php echo

```

```

$row_Recordset1['name']; ?>" size="32"></td>

```



```

</tr>
<tr valign="baseline">
  <td nowrap align="right"><span class="style3">Surname:</span></td>
  <td><input type="text" name="surname" value="<?php echo
$row_Recordset1['surname']; ?>" size="32"></td>
</tr>
<tr valign="baseline">
  <td nowrap align="right"><span class="style3">Address:</span></td>
  <td><input type="text" name="address" value="<?php echo
$row_Recordset1['address']; ?>" size="32"></td>
</tr>
<tr valign="baseline">
  <td nowrap align="right"><span class="style3">Location:</span></td>
  <td><select name="location">
    <?php
do {
  ?>
    <option value="<?php echo $row_Recordset2['name']?>" <?php if
    (!(strcmp($row_Recordset2['name'], $row_Recordset1['location']))) {echo
    "SELECTED";} ?>><?php echo $row_Recordset2['name']?></option>
    <?php
} while ($row_Recordset2 = mysql_fetch_assoc($Recordset2));
?>
    </select>      </td>
</tr>
<tr valign="baseline">
  <td nowrap align="right"><span class="style3">Telephone:</span></td>
  <td><input type="text" name="telephone" value="<?php echo
$row_Recordset1['telephone']; ?>" size="32"></td>
</tr>
<tr valign="baseline">
  <td nowrap align="right"><span class="style3">Mobilphone:</span></td>
  <td><input type="text" name="mobilphone" value="<?php echo
$row_Recordset1['mobilphone']; ?>" size="32"></td>
</tr>
<tr valign="baseline">
  <td nowrap align="right"><span class="style3">Fax:</span></td>
  <td><input type="text" name="fax" value="<?php echo $row_Recordset1['fax'];
?>" size="32"></td>
</tr>
<tr valign="baseline">
  <td nowrap align="right"><span class="style3">Email:</span></td>
  <td><input type="text" name="email" value="<?php echo
$row_Recordset1['email']; ?>" size="32"></td>
</tr>
<tr valign="baseline">
  <td nowrap align="right">&nbsp;</td>
  <td><input type="submit" value="Update record"></td>
</tr>
</table>

```

```


</form>
<script language="javascript" type="text/javascript">
function alertss()
{
if (confirm("Record will be delete. Are you Sure ?"))
{
parent.document.frames[2].location="deleteperson.php?id=<?php echo
$row_Recordset1['id'];?>";
}
}
}
</script>
<form onclick=alertss() method="post" name="form2" target="_self" id="form2">
<label for="Submit"></label>
<input type="button" name="Submit" value="Delete This Person" id="Submit" />
</form>

<p>&nbsp;</p>
<p>&nbsp;</p>
<?php
mysql_free_result($Recordset1);

mysql_free_result($Recordset2);
?>

```

7.1.4 Show Location.php

```

<?php require_once('Connections/mahfuz.php'); ?><?php
//initialize the session
if (!isset($_SESSION)) {
    session_start();
}

// ** Logout the current user. **
$logouthAction = $_SERVER['PHP_SELF']."?doLogout=true";
if ((isset($_SERVER['QUERY_STRING'])) && ($_SERVER['QUERY_STRING'] !=
"")){
    $logouthAction.="&". htmlentities($_SERVER['QUERY_STRING']);
}

if ((isset($_GET['doLogout'])) && ($_GET['doLogout']=="true")){
    //to fully log out a visitor we need to clear the session variables
    $_SESSION['MM_Username'] = NULL;
    $_SESSION['MM_UserGroup'] = NULL;
    $_SESSION['PrevUrl'] = NULL;
}

```



```

isset($_SESSION['MM_Username']);
isset($_SESSION['MM_UserGroup']);
isset($_SESSION['PrevUrl']);

logoutGoTo = "index.php";
if ($logoutGoTo) {
    echo "<script language='javascript'>
    parent.location='index.php';
    </script>";
    exit;
}

?php
mysql_select_db($database_mahfuz, $mahfuz);
$query_Recordset1 = "SELECT * FROM `user`";
Recordset1 = mysql_query($query_Recordset1, $mahfuz) or die(mysql_error());
$row_Recordset1 = mysql_fetch_assoc($Recordset1);
$totalRows_Recordset1 = mysql_num_rows($Recordset1);
>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />
<title>Untitled Document</title>
<style type="text/css">
<!--
body {
    background-color: #3366CC;
    background-image: url(kivi.jpg);

style1 {color: #CCCCCC}
-->
</style>
<style type="text/css" media="all">
<!--
@import url("Eski Klasör/mm_entertainment.css");
a:link {
    color: #FFFFFF;
    text-decoration: none;

a:visited {
    color: #FFFFFF;
    text-decoration: none;

body, td, th {
    color: #FFFFFF;
    font-family: Verdana, Arial, Helvetica, sans-serif;

```

```

}
a:hover {
    text-decoration: underline;
    color: #000000;
}
a:active {
    text-decoration: none;
    color: #CC9900;
}
}
.style7 {font-size: 12px; color: #FFFFFF; font-weight: bold; }
-->
</style></head>

<body>
<table width="112" height="199" border="0">
<tr>
<th scope="col"><div align="left">
<object classid="clsid:D27CDB6E-AE6D-11cf-96B8-444553540000"
codebase="http://download.macromedia.com/pub/shockwave/cabs/flash/swflash.cab#ve
rsion=5,0,0,0" width="105" height="21">
<param name="movie" value="button7.swf" />
<param name="quality" value="high" />
<param name="bgcolor" value="" />
<embed src="button7.swf" quality="high"
pluginspage="http://www.macromedia.com/shockwave/download/index.cgi?P1_Prod_
Version=ShockwaveFlash" type="application/x-shockwave-flash" width="105"
height="21" bgcolor=""></embed>
</object>
</div></th>
</tr>
<?PHP if ($_SESSION['MM_UserGroup']=="1") { echo " <tr>
<th scope='col'><div align='left'>
<object classid='clsid:D27CDB6E-AE6D-11cf-96B8-444553540000'
codebase='http://download.macromedia.com/pub/shockwave/cabs/flash/swflash.cab#ver
sion=5,0,0,0' width='105' height='21'>
<param name='movie' value='button1.swf' />
<param name='quality' value='high' />
<param name='bgcolor' value='#3366CC' />
<embed src='button1.swf' quality='high'
pluginspage='http://www.macromedia.com/shockwave/download/index.cgi?P1_Prod_V
ersion=ShockwaveFlash' type='application/x-shockwave-flash' width='105' height='21'
bgcolor='#3366CC'></embed>
</object>
</div></th>
</tr>
<tr>
<td><object classid='clsid:D27CDB6E-AE6D-11cf-96B8-444553540000'
codebase='http://download.macromedia.com/pub/shockwave/cabs/flash/swflash.cab#ver
sion=5,0,0,0' width='105' height='21'>
<param name='movie' value='button2.swf' />

```



```

<param name='quality' value='high' />
<param name='bgcolor' value='#3366CC' />
<embed src='button2.swf' quality='high'
pluginspage='http://www.macromedia.com/shockwave/download/index.cgi?P1_Prod_V
ersion=ShockwaveFlash' type='application/x-shockwave-flash' width='105' height='21'
bgcolor='#3366CC'></embed>
</object></td>
</tr>
<tr>
<td><object classid='clsid:D27CDB6E-AE6D-11cf-96B8-444553540000'
codebase='http://download.macromedia.com/pub/shockwave/cabs/flash/swflash.cab#ver
sion=5,0,0,0' width='105' height='21'>
<param name='movie' value='button3.swf' />
<param name='quality' value='high' />
<param name='bgcolor' value='#3366CC' />
<embed src='button3.swf' quality='high'
pluginspage='http://www.macromedia.com/shockwave/download/index.cgi?P1_Prod_V
ersion=ShockwaveFlash' type='application/x-shockwave-flash' width='105' height='21'
bgcolor='#3366CC'></embed>
</object></td>
</tr>";}>
<tr>
<td><object classid="clsid:D27CDB6E-AE6D-11cf-96B8-444553540000"
codebase="http://download.macromedia.com/pub/shockwave/cabs/flash/swflash.cab#ve
rsion=5,0,0,0" width="105" height="21">
<param name="movie" value="button6.swf" />
<param name="quality" value="high" />

<embed src="button6.swf" quality="high"
pluginspage="http://www.macromedia.com/shockwave/download/index.cgi?P1_Prod_
Version=ShockwaveFlash" type="application/x-shockwave-flash" width="105"
height="21"></embed>
</object></td>
</tr>
<?PHP if ($_SESSION['MM_UserGroup']=="2")
{
do {
if($_SESSION['MM_Username']==$row_Recordset1['username'])
{
if ($row_Recordset1['name']==$row_Recordset1['surname'])
{
echo " <tr><td><object classid='clsid:D27CDB6E-AE6D-11cf-96B8-
444553540000'
codebase='http://download.macromedia.com/pub/shockwave/cabs/flash/swflash.cab#ver
sion=5,0,0,0' width='105' height='21'>
<param name='movie' value='button10.swf' />
<param name='quality' value='high' />
<param name='bgcolor' value='#3366CC' />
<embed src='button10.swf' quality='high'
pluginspage='http://www.macromedia.com/shockwave/download/index.cgi?P1_Prod_V

```

```

ersion=ShockwaveFlash' type='application/x-shockwave-flash' width='105' height='21'
bgcolor='#3366CC'></embed>
</object></td>
</tr>";
    } else if ($_SESSION['MM_UserGroup']=="2") { echo "<tr>
<td><div align='left'>
<object classid='clsid:D27CDB6E-AE6D-11cf-96B8-444553540000'
codebase='http://download.macromedia.com/pub/shockwave/cabs/flash/swflash.cab#ver
sion=5,0,0,0' width='105' height='21'>
<param name='movie' value='button9.swf' />
<param name='quality' value='high' />
<param name='bgcolor' value='#3366CC' />
<embed src='button9.swf' quality='high'
pluginspage='http://www.macromedia.com/shockwave/download/index.cgi?P1_Prod_V
ersion=ShockwaveFlash' type='application/x-shockwave-flash' width='105' height='21'
bgcolor='#3366CC'></embed>
</object>
</div></td>
</tr>";}
    }
    } while ($row_Recordset1 = mysql_fetch_assoc($Recordset1));
    $rows = mysql_num_rows($Recordset1);
    if($rows > 0) {
        mysql_data_seek($Recordset1, 0);
        $row_Recordset1 = mysql_fetch_assoc($Recordset1);
    }
    }
    ?>

<?PHP if ($_SESSION['MM_UserGroup']=="1") { echo "<tr>
<td><object classid='clsid:D27CDB6E-AE6D-11cf-96B8-444553540000'
codebase='http://download.macromedia.com/pub/shockwave/cabs/flash/swflash.cab#ver
sion=5,0,0,0' width='105' height='21'>
<param name='movie' value='button8.swf' />
<param name='quality' value='high' />
<param name='bgcolor' value='#3366CC' />
<embed src='button8.swf' quality='high'
pluginspage='http://www.macromedia.com/shockwave/download/index.cgi?P1_Prod_V
ersion=ShockwaveFlash' type='application/x-shockwave-flash' width='105' height='21'
bgcolor='#3366CC'></embed>
</object></td>
</tr>";}}?>
<tr>
<td><object classid="clsid:D27CDB6E-AE6D-11cf-96B8-444553540000"
codebase="http://download.macromedia.com/pub/shockwave/cabs/flash/swflash.cab#ve
rsion=5,0,0,0" width="105" height="21">
<param name="movie" value="button11.swf" />
<param name="quality" value="high" />
<param name="bgcolor" value="" />

```



```

    <embed src="button11.swf" quality="high"
pluginspage="http://www.macromedia.com/shockwave/download/index.cgi?P1_Prod_
Version=ShockwaveFlash" type="application/x-shockwave-flash" width="105"
height="21" bgcolor=""></embed>
</object></td>
</tr>
<tr>
    <td><object classid="clsid:D27CDB6E-AE6D-11cf-96B8-444553540000"
codebase="http://download.macromedia.com/pub/shockwave/cabs/flash/swflash.cab#ve
rsion=5,0,0,0" width="105" height="21">
    <param name="movie" value="button4.swf" />
    <param name="quality" value="high" />

    <embed src="button4.swf" quality="high"
pluginspage="http://www.macromedia.com/shockwave/download/index.cgi?P1_Prod_
Version=ShockwaveFlash" type="application/x-shockwave-flash" width="105"
height="21"></embed>
    </object></td>
</tr>

<tr bgcolor="#990000">
    <td><div align="center">

        <p>

            <span class="style7">
                <?php do {if($_SESSION['MM_Username']==$row_Recordset1['username'])
                    {echo "Hi,".$row_Recordset1['name'];
                    }}while ($row_Recordset1 = mysql_fetch_assoc($Recordset1));?>
            </span></p><p class="style7"><strong><a href="<?php echo $logoutAction ?>"
target="mainFrame" class="style1">
                <?php if ($_SESSION['MM_Username']!="") {echo "Log out";} ?>
            </a></strong></p>
        </div></td>
</tr>
</table>
<p>&nbsp;</p>
<p>&nbsp;</p>
<p>&nbsp;</p>
<p><br />
</p>
</body>
</html>
<?php
mysql_free_result($Recordset1);
?>

```

7.1.5 Record Person.php

```
<?php require_once('Connections/mahfuz.php'); ?>
<?php
if (!isset($_SESSION)) {
    session_start();
}
$MM_authorizedUsers = "";
$MM_donotCheckaccess = "true";

// *** Restrict Access To Page: Grant or deny access to this page
function isAuthorized($strUsers, $strGroups, $UserName, $UserGroup) {
    // For security, start by assuming the visitor is NOT authorized.
    $isValid = False;

    // When a visitor has logged into this site, the Session variable MM_Username set
    equal to their username.
    // Therefore, we know that a user is NOT logged in if that Session variable is blank.
    if (!empty($UserName)) {
        // Besides being logged in, you may restrict access to only certain users based on an
        ID established when they login.
        // Parse the strings into arrays.
        $arrUsers = Explode(",", $strUsers);
        $arrGroups = Explode(",", $strGroups);
        if (in_array($UserName, $arrUsers)) {
            $isValid = true;
        }
        // Or, you may restrict access to only certain users based on their username.
        if (in_array($UserGroup, $arrGroups)) {
            $isValid = true;
        }
        if (($strUsers == "") && true) {
            $isValid = true;
        }
    }
    return $isValid;
}

$MM_restrictGoTo = "access.php";
if (!((isset($_SESSION['MM_Username'])) &&
(isAuthorized("", $MM_authorizedUsers, $_SESSION['MM_Username'],
$_SESSION['MM_UserGroup'])))) {
    $MM_qsChar = "?";
    $MM_referrer = $_SERVER['PHP_SELF'];
    if (strpos($MM_restrictGoTo, "?") $MM_qsChar = "&";
    if (isset($QUERY_STRING) && strlen($QUERY_STRING) > 0)
        $MM_referrer .= "?" . $QUERY_STRING;
    $MM_restrictGoTo = $MM_restrictGoTo. $MM_qsChar . "accesscheck=" .
urlencode($MM_referrer);
    header("Location: ". $MM_restrictGoTo);
```



```

    exit;
}
?>
<?php
function GetSQLValueString($theValue, $theType, $theDefinedValue = "",
$theNotDefinedValue = "")
{
    $theValue = (!get_magic_quotes_gpc()) ? addslashes($theValue) : $theValue;

    switch ($theType) {
        case "text":
            $theValue = ($theValue != "") ? "'" . $theValue . "'" : "NULL";
            break;
        case "long":
        case "int":
            $theValue = ($theValue != "") ? intval($theValue) : "NULL";
            break;
        case "double":
            $theValue = ($theValue != "") ? "'" . doubleval($theValue) . "'" : "NULL";
            break;
        case "date":
            $theValue = ($theValue != "") ? "'" . $theValue . "'" : "NULL";
            break;
        case "defined":
            $theValue = ($theValue != "") ? $theDefinedValue : $theNotDefinedValue;
            break;
    }
    return $theValue;
}

```

```

$editFormAction = $_SERVER['PHP_SELF'];
if (isset($_SERVER['QUERY_STRING'])) {
    $editFormAction .= "?" . htmlentities($_SERVER['QUERY_STRING']);
}

```

```

if ((isset($_POST["MM_insert"])) && ($_POST["MM_insert"] == "form1")) {
    $insertSQL = sprintf("INSERT INTO person (id, name, surname, address, location,
telephone, mobilphone, fax, email) VALUES (%s, %s, %s, %s, %s, %s, %s, %s, %s)",
        GetSQLValueString($_POST['id'], "double"),
        GetSQLValueString($_POST['name'], "text"),
        GetSQLValueString($_POST['surname'], "text"),
        GetSQLValueString($_POST['address'], "text"),
        GetSQLValueString($_POST['location'], "text"),
        GetSQLValueString($_POST['telephone'], "text"),
        GetSQLValueString($_POST['mobilphone'], "text"),
        GetSQLValueString($_POST['fax'], "text"),
        GetSQLValueString($_POST['email'], "text"));
}

```

```

mysql_select_db($database_mahfuz, $mahfuz);
$result1 = mysql_query($insertSQL, $mahfuz) or die(mysql_error());

```

```

}

mysql_select_db($database_mahfuz, $mahfuz);
$query_Recordset1 = "SELECT location.name FROM location";
$Recordset1 = mysql_query($query_Recordset1, $mahfuz) or die(mysql_error());
$row_Recordset1 = mysql_fetch_assoc($Recordset1);
$totalRows_Recordset1 = mysql_num_rows($Recordset1);
?><style type="text/css">
<!--
body {
    background-color: #3366CC;
}
.style1 {color: #FFFFFF}
-->
</style>
<form method="post" name="form1" action="<?php echo $editFormAction; ?>">
<table align="center">
<tr valign="baseline">
<td nowrap align="right"><span class="style1">Id:</span></td>
<td><input type="text" name="id" value="" size="32"></td>
</tr>
<tr valign="baseline">
<td nowrap align="right"><span class="style1">Name:</span></td>
<td><input type="text" name="name" value="" size="32"></td>
</tr>
<tr valign="baseline">
<td nowrap align="right"><span class="style1">Surname:</span></td>
<td><input type="text" name="surname" value="" size="32"></td>
</tr>
<tr valign="baseline">
<td nowrap align="right"><span class="style1">Address:</span></td>
<td><input type="text" name="address" value="" size="32"></td>
</tr>
<tr valign="baseline">
<td nowrap align="right"><span class="style1">Location:</span></td>
<td><label for="select"></label>
<select name="location" id="location">
<?php
do {
?>
<option value="<?php echo $row_Recordset1['name']; ?>"><?php if
(!strcmp($row_Recordset1['name'], $row_Recordset1['name'])) {echo
"selected=\"selected\"";} ?><?php echo $row_Recordset1['name']; ?></option>
<?php
} while ($row_Recordset1 = mysql_fetch_assoc($Recordset1));
$rows = mysql_num_rows($Recordset1);
if($rows > 0) {
    mysql_data_seek($Recordset1, 0);
    $row_Recordset1 = mysql_fetch_assoc($Recordset1);
}

```



```

?>
    </select></td>
</tr>

<tr valign="baseline">
    <td nowrap align="right"><span class="style1"></span></td>
    <td><input type="submit" value="Insert record"></td>
</tr>
</table>
<input type="hidden" name="MM_insert" value="form1">
</form>
<p>&nbsp;</p>
<?php
mysql_free_result($Recordset1);
?>

```

CONCLUSION

Tried and learned web design with PHP Language, MySql Database and Apache Server, these tools are very useful and simply.

This project is good change for me, because it is developed me to learn and design a web page. It is not real Association for Company which web page is prepared for. When design web site we can use some effects and programs. For example Macromedia Dream weaver, Photoshop and Flash. These programs are very useful for designing a web site.

For all of the pages I have worked and research so far. To collect the information about the animal, vegetable and fruit the production.

I have written some useful knowledge that has hyperlink, so that clicking the hyperlink load a Help web page, and a viewer can return to main page using the browser back buttons, or by the Back Button That I have created under the Link menu.

REFERENCES

- 1 – <http://www.php.net>
 - 2 – <http://www.phpnuke.org>
 - 3 – <http://www.apache.org>
 - 4 – <http://www.mysql.com>
 - 5 – MySQL in 21 Days Mark Maslakowski
-