

**THE DESIGN AND IMPLEMENTATION OF AN E-COMMERCE
WEB-SITE FOR ONLINE MERCHANDISE SALES**

A PROJECT REPORT

Submitted by

991764075 PRANAY SEELA

In partial fulfillment for the award of the degree
Of
MASTER'S OF SCIENCE
IN
COMPUTER SCIENCE AND ENGINEERING

INDIANA STATE UNIVERSITY, TERRE HAUTE

INDIANA - 47809

August - 2016

Abstract

The business-to-buyer part of electronic trade (e-trade) is the most noticeable business utilization of the World Wide Web. The essential objective of an e-business website is to offer/sell merchandise online.

This project deals with developing an online website for Merchandise Sale. It provides the user with a menu of different items available. In order to do online purchase a shopping cart is provided to the user. This is implemented using a 3-tier approach, with a backend database, a middle tier of Microsoft IIS(Internet Information Services) and ASP.NET(Active server pages), and a web browser as the front end.

In order to develop an e-commerce website, a number of Technologies must be studied. These include multi-tiered architecture, server and client side scripting languages, implementation technologies such as ASP.NET, programming language (such as C#), Databases (such as MySQL).

This is a project with the objective to develop a basic website where a consumer is provided with a shopping cart application and also to know about the technologies used to develop such an application.

This document will discuss each of the underlying technologies to create and implement an E-Commerce website.

TABLE OF CONTENTS

[Abstract](#)

[TABLE OF CONTENTS](#)

[ACKNOWLEDGMENTS](#)

[Introduction](#)

[Literature Review](#)

[Project Design](#)

[Data Model](#)

[Database Design/Tables](#)

[User Interface Diagram](#)

[Implementation Technologies](#)

[Internet Information Services \(IIS\)](#)

[ASP.NET](#)

[Powerful database-driven functionality:](#)

[Faster web applications:](#)

[Memory leak and crash protection:](#)

[Authentication in ASP.NET](#)

[MySQL Database](#)

[Integrating IIS and ASP.NET](#)

[Integrating the Website and Database](#)

[Web Page Programming Options](#)

[Server-side processing.](#)

[Client-Side Processing.](#)

[Web Based Application Development](#)

[Database Connectivity](#)

[ADO.NET](#)

[DataSet](#)

[Data Provider](#)

[The Connection Object](#)

[The Command Object](#)

[Connecting ASP.NET application to a Database](#)

[The E-Commerce Shopping Cart Application](#)

[Limitations and Future Development](#)

[Conclusion](#)

[Bibliography](#)

ACKNOWLEDGMENTS

In completing this graduate project I have been fortunate to have help, support and encouragement from many people. I would like to acknowledge them for their cooperation.

First, I would like to thank **Jeff Kinne**, my project advisor, for guiding me through each and every step of the process with support. Thank you for your advice, guidance and assistance.

Finally, I would also like to thank **Michael**, my project manager, who showed immense patience and understanding throughout the project and provided suggestions.

1. Introduction

E-commerce/E-trade is quick making progress as an acknowledged and utilized business worldview. More organizations are executing online web-sites giving usefulness to performing business exchanges over the web. It is sensible to say that the way toward shopping on the web is getting to be ordinary.

The goal of this is to build up an E-business store where any product(such as shirts, jeans, a wide range of garments, and stock) can be purchased from the solace of home through the Internet. In any case, for execution purposes, will deal with an online store.

An online store is a virtual store on the Internet where customers can browse the catalog/menu and select products. The selected items may be collected in a online cart. At checkout time, the items in the shopping cart will be presented as an order. At that time, more information will be needed to complete the transaction. Usually, the customer will be asked to fill or select a billing address, a shipping address, a shipping option, and payment information such as credit card number or cash on delivery.

2. Literature Review

Electronic Commerce (e-commerce) applications support the interaction between different parties participating in a commerce transaction via the network, as well as the management of the data involved in the process.

The increasing importance of e-commerce is apparent in the study conducted by researchers at the Gvu (Graphics, Visualization, and Usability) Center at the Georgia Institute of Technology. In their summary of the findings from the eighth survey, the researchers report that “e-commerce is taking off both in terms of the number of users shopping as well as the total amount people are spending via Internet based transactions”. Over three quarters of the 10,000 respondents report having purchased items online. The most cited reason for

using the web for personal shopping was convenience (65%), followed by availability of vendor information (60%), no pressure from sales person (55%) and saving time (53%).

Although the issue of security remains the primary reason why more people do not purchase items online, the GVA survey also indicates that faith in the security of e-commerce is increasing. As more people gain confidence in current encryption technologies, more and more users can be expected to frequently purchase items online.

A good E-Commerce site should present the following factors to the customers for better usability:

- Knowing when an item was saved or not saved in the shopping cart.
- Returning to different parts of the site after adding an item to the shopping cart.
- Easy scanning and selecting items in a list.
- Effective categorical organization of products.
- Simple navigation from home page to information and order links for specific products.
- Obvious shopping links or buttons.
- Minimal and effective security notifications or messages.
- Consistent layout of product information.

Another important factor in the design of an e-commerce site is feedback. The interactive cycle between a user and a web site is not complete until the web site responds to a command entered by the user. According to Norman, "feedback--sending back to the user information about what action has actually been done, what result has been accomplished--is a well-known concept in the science of control and information theory. Imagine trying to talk to someone when you cannot even hear your own voice, or trying to draw a picture with a pencil that leaves no mark: there would be no feedback".

Website feedback often consists of a change in the visual or verbal information presented to the user. Simple examples include highlighting a selection made by the user or filling a field on a form based on a user's selection from a pull down list. Another example is using the sound of a cash register to confirm that a product has been added to an electronic shopping cart.

Completed orders should be acknowledged quickly. This may be done with an acknowledgment or fulfillment page. The amount of time it takes to generate and download this page, however, is a source of irritation for many e-commerce users. Users are quick to attribute meaning to events. A blank page, or what a user perceives to be "a long time" to receive an acknowledgment, may be interpreted as "there must be something wrong with the order." If generating an acknowledgment may take longer than what may be reasonably expected by the

user, then the design should include intermediate feedback to the user indicating the progress being made toward acknowledgment or fulfillment.

Finally, feedback should not distract the user. Actions and reactions made by the web site should be meaningful. Feedback should not draw the user's attention away from the important tasks of gathering information, selecting products, and placing orders.

3. Project Design

In order to design a web site, the relational database must be designed first. Conceptual design can be divided into two parts: The data model and the process model. The data model focuses on what data should be stored in the database while the process model deals with how the data is processed. To put this in the context of the relational database, the data model is used to design the relational tables. The process model is used to design the queries that will access and perform operations on those tables.

3.1. Data Model

A data model is a conceptual representation of the data structures that are required by a database. The first step in designing a database is to develop an Entity-Relation Diagram (ERD). The ERD serves as a blueprint from which a relational database may be deduced. Entity A matches exactly one record in entity B and every record in B matches exactly one record in A. One to many means that every record in A matches zero or more records in B and every record in B matches exactly one record in A. If there is a one to many relationship between two entities, then these entities are represented as Associative Entities. In the Relational Database model, each of the entities will be transformed into a table. The tables are shown below along with the attributes.

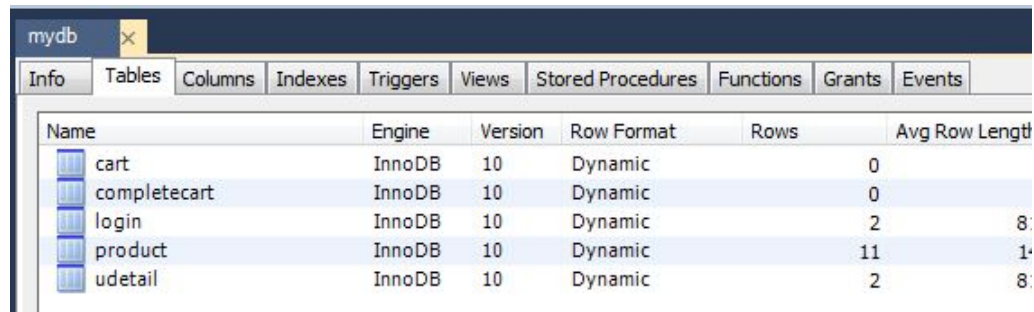
3.2. Database Design/Tables

Technologies used :

1. MySQL Server 5.7
2. MySQL Connector Net 6.9.9
3. MySQL for Visual Studio 1.2.6
4. MySQL Installer - Community
5. MySQL Workbench 6.3 CE

Tables and Schema information :

Tables :



Name	Engine	Version	Row Format	Rows	Avg Row Length
cart	InnoDB	10	Dynamic	0	
completecart	InnoDB	10	Dynamic	0	
login	InnoDB	10	Dynamic	2	8
product	InnoDB	10	Dynamic	11	14
udetail	InnoDB	10	Dynamic	2	8

Columns :

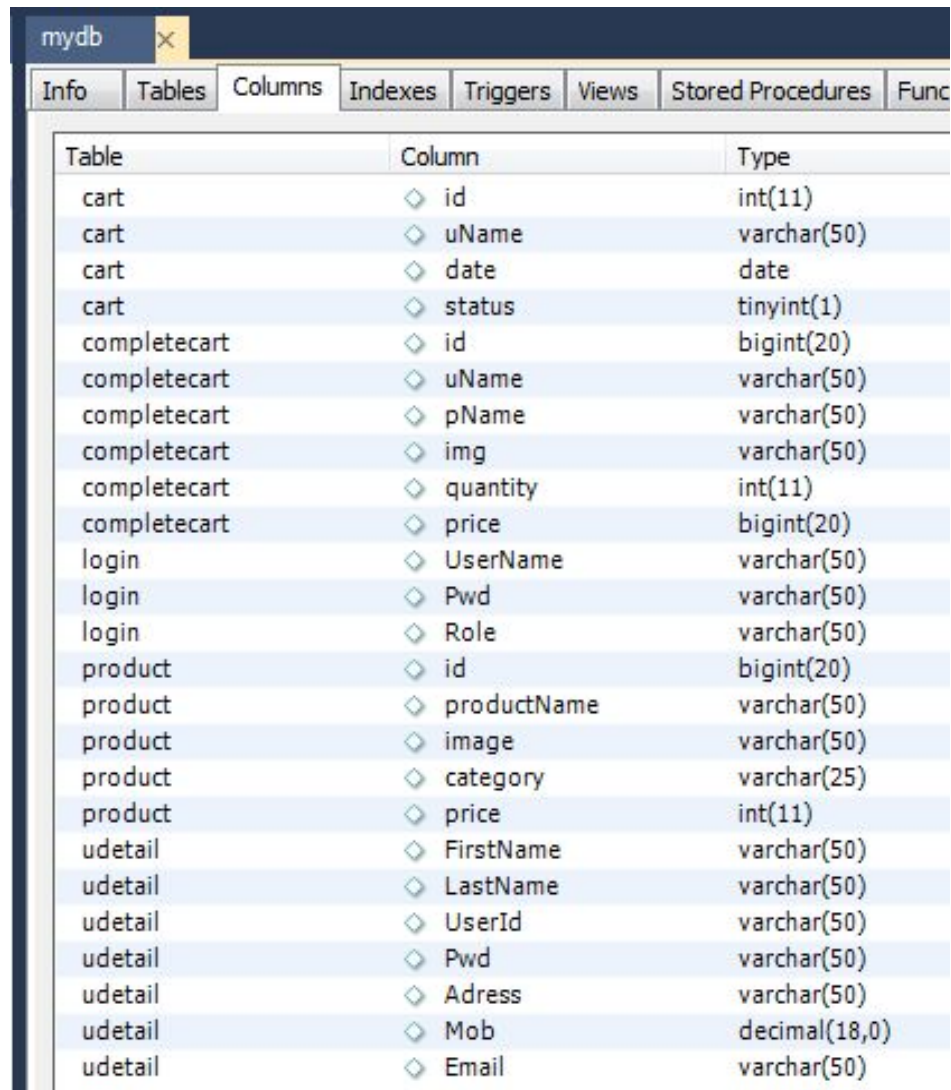


Table	Column	Type
cart	id	int(11)
cart	uName	varchar(50)
cart	date	date
cart	status	tinyint(1)
completecart	id	bigint(20)
completecart	uName	varchar(50)
completecart	pName	varchar(50)
completecart	img	varchar(50)
completecart	quantity	int(11)
completecart	price	bigint(20)
login	UserName	varchar(50)
login	Pwd	varchar(50)
login	Role	varchar(50)
product	id	bigint(20)
product	productName	varchar(50)
product	image	varchar(50)
product	category	varchar(25)
product	price	int(11)
udetail	FirstName	varchar(50)
udetail	LastName	varchar(50)
udetail	UserId	varchar(50)
udetail	Pwd	varchar(50)
udetail	Adress	varchar(50)
udetail	Mob	decimal(18,0)
udetail	Email	varchar(50)

Indexes :

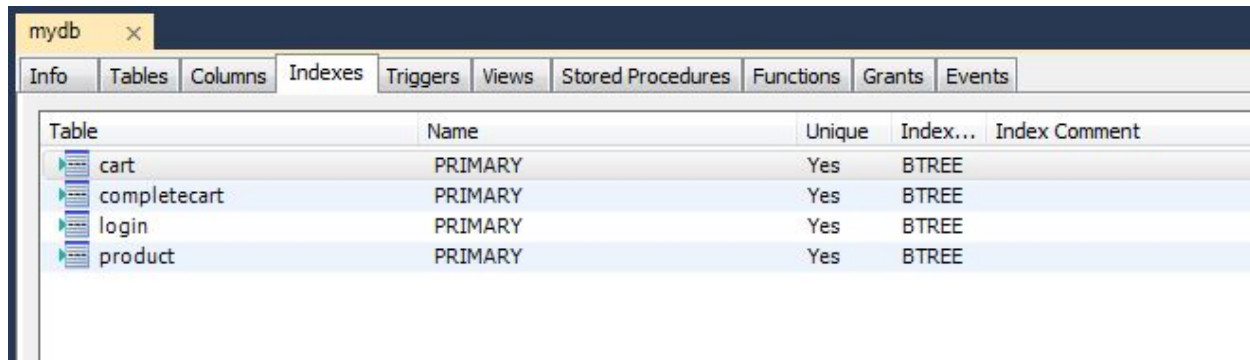


Table	Name	Unique	Index...	Index Comment
cart	PRIMARY	Yes	BTREE	
completecart	PRIMARY	Yes	BTREE	
login	PRIMARY	Yes	BTREE	
product	PRIMARY	Yes	BTREE	

4. User Interface Diagram

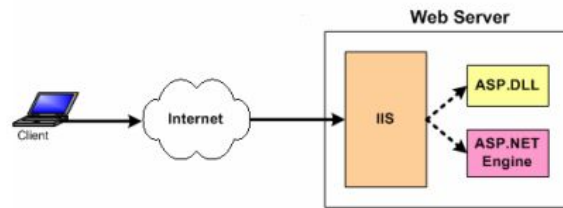
Before implementing the actual design of the project, a few user interface designs were constructed to visualize the user interaction with the system as they browse for items, create a shopping cart and purchase items.

<<User interface diagram goes here >>

<<User interface diagrams of various pages goes here >>

5. Implementation Technologies

The objective of this project is to develop an online e-commerce store. When the user types in the URL of the Store in the address field of the browser, a Web Server is contacted to get the requested information. In the .NET Framework, IIS (Internet Information Service) acts as the Web Server. The task of a Web Server is to accept incoming HTTP requests and to return the requested resource in an HTTP response. The first thing IIS does when a request comes in is to decide how to handle the request. Its decision is based upon the requested file's extension. For example, if the requested file has the .asp extension, IIS will route the request to be handled by asp.dll. If it has the extension of .aspx, .ascx, etc, it will route the request to be handled by ASP.NET Engine.



The ASP.NET Engine then gets the requested file, and if necessary contacts the database through ADO.NET for the required file and then the information is sent back to the Client's browser. Figure shows how a client browser interacts with the Web server and how the Web server handles the request from client.

5.1. Internet Information Services (IIS)

IIS 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 (a software standard for transferring computer files between machines with widely different operating systems), SMTP (Simple Mail Transfer Protocol, is the de facto standard for email transmission across the Internet) and HTTP/HTTPS (is the secure version of HTTP, the communication protocol of the World Wide Web).

Features: The web server itself cannot directly perform server side processing but can delegate the task to ISAPI (Application Programming Interface of IIS) applications on the server. Microsoft provides a number of these including ones for Active Server Page and ASP.NET.

Compatibility: 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.

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 that has fewer privileges. In particular this means that if there were an exploit on that feature, it would not necessarily compromise the entire system.

5.2. ASP.NET

ASP.NET is a programming framework built on the common language runtime that can be used on a server to build powerful Web applications. ASP.NET has many advantages – both for programmers and for the end users

because it is compatible with the .NET Framework. This compatibility allows the users to use the following features through ASP.NET:

5.2.1. Powerful database-driven functionality:

ASP.NET allows programmers to develop web applications that interface with a database. The advantage of ASP.NET is that it is object-oriented and has many programming tools that allow for faster development and more functionality.

5.2.2. Faster web applications:

Two aspects of ASP.NET make it fast -- compiled code and caching. In ASP.NET the code is compiled into "machine language" before a visitor ever comes to the website. Caching is the storage of information in memory for faster access in the future. ASP.NET allows programmers to set up pages or areas of pages that are commonly reused to be cached for a set period of time to improve the performance of web applications. In addition, ASP.NET allows the caching of data from a database so the website is not slowed down by frequent visits to a database when the data does not change very often.

5.2.3. Memory leak and crash protection:

ASP.NET automatically recovers from memory leaks and errors to make sure that the website is always available to the visitors.

ASP.NET also supports code written in more than 25 .NET languages (including VB.NET, C#, and Jscript.Net). This is achieved by the Common Language Runtime (CLR) compiler that supports multiple languages.

5.3. Authentication in ASP.NET

There are two separate authentication layers in an ASP.NET application. All requests flow through IIS before they are handed to ASP.NET, and IIS can decide to deny access before ASP.NET even knows about the request. Here is how the process works :

1. IIS checks to see if an incoming request is coming from an IP address that is allowed access to the domain. If not, the request is denied.
2. IIS performs its own user authentication, if it is configured to do so. By default, IIS allows anonymous access and requests are authenticated automatically.
3. When a request is passed from IIS to ASP.NET with an authenticated user, ASP.NET checks to see whether impersonation is enabled. If so, ASP.NET acts as though it were the authenticated user. If not, ASP.NET acts with its own configured account.

4. Finally, the identity is used to request resources from the operating system. If all the necessary resources can be obtained, the user's request is granted; otherwise the request is denied.

5.4. MySQL Database

In this project, MySQL is used as the backend database. MySQL is an open source database management system. The features of MySQL are given below:

- MySQL is a relational database management system. A relational database stores information in different tables, rather than in one giant table. These tables can be referenced to each other, to access and maintain data easily.
- MySQL is open source database system. The database software can be used and modified by anyone according to their needs.
- It is fast, reliable and easy to use. To improve the performance, MySQL is multithreaded database engine. A multithreaded application performs many tasks at the same time as if multiple instances of that application were running simultaneously.

In being multi threaded MySQL has many advantages. A separate thread handles each incoming connection with an extra thread that is always running to manage the connections. Multiple clients can perform read operations simultaneously, but while writing, only hold up another client that needs access to the data being updated. Even though the threads share the same process space, they execute individually and because of this separation, multiprocessor machines can spread the thread across many CPUs as long as the host operating system supports multiple CPUs. Multithreading is the key feature to support MySQL's performance design goals. It is the core feature around which MySQL is built.

MySQL database is connected to ASP.NET using an ODBC driver. Open Database Connectivity (ODBC) is a widely accepted application-programming interface (API) for database access. The ODBC driver is a library that implements the functions supported by ODBC API. It processes ODBC function calls, submits SQL requests to MySQL server, and returns results back to the application. If necessary, the driver modifies an application's request so that the request conforms to syntax supported by MySQL.

5.5. Integrating IIS and ASP.NET

When a request comes into IIS Web server its extension is examined and, based on this extension, the request is either handled directly by IIS or routed to

an ISAPI extension. An ISAPI extension is a compiled class that is installed on the Web server and whose responsibility is to return the markup for the requested file type. By default, IIS handles the request, and simply returns the contents of the requested file.

This makes sense for static files, like images, HTML pages, CSS files, external JavaScript files, and so on. For example, when a request is made for a .html file, IIS simply returns the contents of the requested HTML file.

For files whose content is dynamically generated, the ISAPI extension configured for the file extension is responsible for generating the content for the requested file. For example, a Web site that serves up classic ASP pages has the .asp extension mapped to the asp.dll ISAPI extension. The asp.dll ISAPI extension executes the requested ASP page and returns its generated HTML markup. If the Web site serves up ASP.NET Web pages, IIS has mapped the .aspx to aspnet_isapi.dll, an ISAPI extension that starts off the process of generating the rendered HTML for the requested ASP.NET Web page.

The aspnet_isapi.dll ISAPI extension is a piece of unmanaged code. That is, it is not code that runs in the .NET Framework. When IIS routes the request to the aspnet_isapi.dll ISAPI extension, the ISAPI extension routes the request onto the ASP.NET engine, which is written in managed code - managed code is code that runs in the .NET Framework.

The ASP.NET engine is strikingly similar to IIS in many ways. Just like IIS has a directory mapping file extensions to ISAPI extensions, the ASP.NET engine maps file extensions to HTTP handlers. An HTTP handler is a piece of managed code that is responsible for generating the markup for a particular file type.

5.6. Integrating the Website and Database

Customers ordering from an e-commerce website need to be able to get information about a vendor's products and services, ask questions, select items they wish to purchase, and submit payment information. Vendors need to be able to track customer inquiries and preferences and process their orders. So a well organized database is essential for the development and maintenance of an e-commerce site.

In a static Web page, content is determined at the time when the page is created. As users access a static page, the page always displays the same information. Example of a static Web page is the page displaying company information. In a dynamic Web page, content varies based on user input and data received from external sources. We use the term "data-based Web pages" to refer to dynamic Web pages deriving some or all of their content from data files or databases.

A data-based Web page is requested when a user clicks a hyperlink or the submit button on a Web page form. If the request comes from clicking a hyperlink, the link specifies either a Web server program or a Web page that calls a Web server program. In some cases, the program performs a static query, such as "Display all items from the Inventory". Although this query requires no user

input, the results vary depending on when the query is made. If the request is generated when the user clicks a form's submit button, instead of a hyperlink, the Web server program typically uses the form inputs to create a query. For example, the user might select five books to be purchased and then submit the input to the Web server program. The Web server program then services the order, generating a dynamic Web page response to confirm the transaction. In either case, the Web server is responsible for formatting the query results by adding HTML tags. The Web server program then sends the program's output back to the client's browser as a Web page.

6. Web Page Programming Options

An e-commerce organization can create data-based Web pages by using serverside and client-side processing technologies or a hybrid of the two. With server-side processing, the Web server receives the dynamic Web page request, performs all processing necessary to create the page, and then sends it to the client for display in the client's browser. Client-side processing is done on the client workstation by having the client browser execute a program that interacts directly with the database.

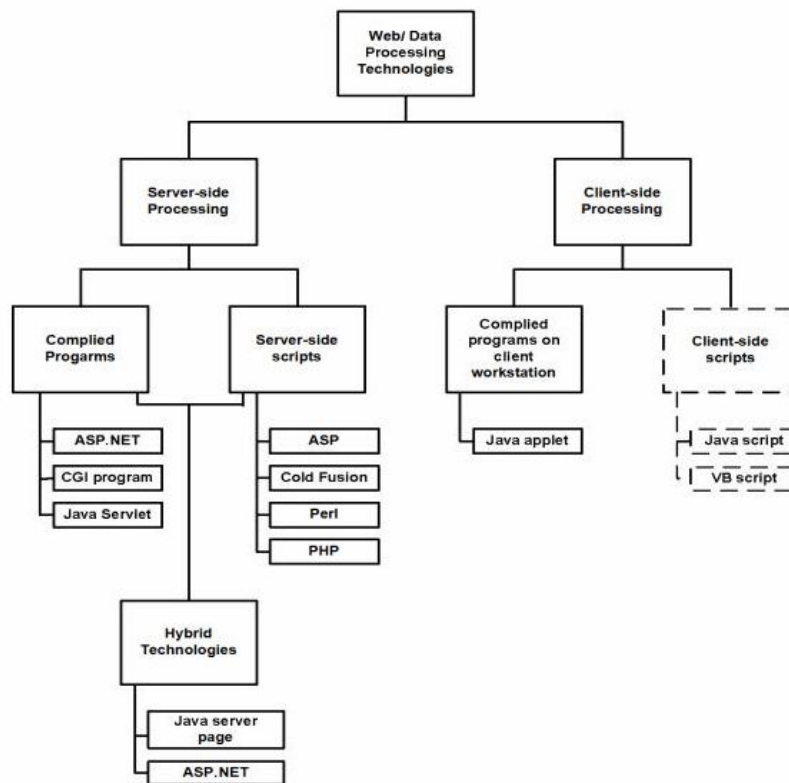


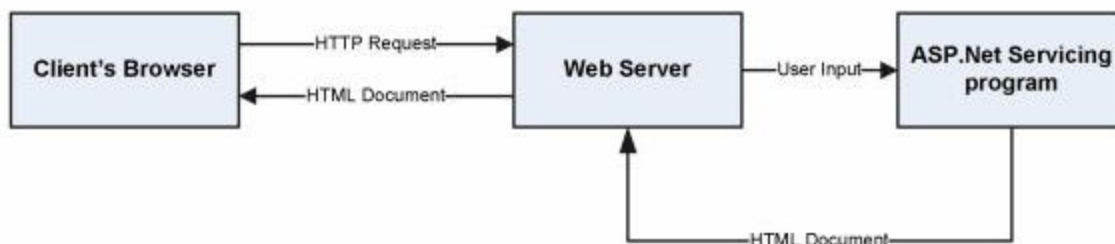
Figure outlines commonly used server-side, client-side, and hybrid Web and data processing technologies; client-side scripts are in dashed lines to indicate they are

unable to interact directly with a database or file but are used to validate user input on the client, then send the validated inputs to the server for further processing.

6.1. Server-side processing.

Generally dynamic or data-driven Web pages use HTML forms to collect user inputs, submitting them to a Web server. A program running on the server processes the form inputs, dynamically composing a Web page reply. This program, which is called, servicing program, can be either a compiled executable program or a script interpreted into machine language each time it is run.

Compiled server programs. When a user submits HTML-form data for processing by a compiled server program, the Web Server invokes the servicing program. The servicing program is not part of the Web server but it is an independent executable program running on the Web server; it processes the user input, determines the action which must be taken, interacts with any external sources (Eg: database) and finally produces an HTML document and terminates. The Web server then sends the HTML document back to the user's browser where it is displayed. Figure shows the flow of HTTP request from the client to the Web server, which is sent to the servicing program. The program creates an HTML document to be sent to the client browser.



Popular languages for creating compiled server programs are Java, Visual Basic, and C++, but almost any language that can create executable programs can be used, provided that it supports commands used by one of the protocols that establish guidelines for communication between Web servers and servicing programs. The first such protocol, introduced in 1993, for use with HTML forms was the Common Gateway Interface (CGI); many servicing programs on Web sites still use CGI programs. However, a disadvantage of using CGI-based servicing programs is that each form submitted to a Web server starts its own copy of the servicing program on the Web server.

A busy Web server is likely to run out of memory when it services many forms simultaneously; thus, as interactive Web sites have gained popularity, Web server vendors have developed new technologies to process form inputs without starting a new copy of the servicing program for each browser input. Examples of

these technologies for communicating with Web servers include Java Servlets and Microsoft's ASP.NET; they allow a single copy of the servicing program to service multiple users without starting multiple instances of the program.

ASP.NET has introduced many new capabilities to server-side Web programming, including a new category of elements called server controls that generate as many as 200 HTML tags and one or more JavaScript functions from a single server control tag. Server controls support the processing of user events, such as clicking a mouse or entering text at either the client browser or the Web server. Server controls also encourage the separation of programming code into different files and/or areas from the HTML tags and text of a Web page, thus allowing HTML designers and programmers to work together more effectively.

6.2. Client-Side Processing.

Client-side Web page processing is achievable through compiled programs downloaded, installed, and executed on the client workstation or by creating scripts with the HTML Web page commands interpreted by the client browser.

Downloading and running compiled programs on client workstations. When a user clicks a hyperlink on a Web page associated with a compiled client-side program, the user's browser must have the ability to run the executable program file; this program interacts with the user, sending and retrieving data from a database server as needed.

Many times, the user is asked to install certain ActiveX components to view some animations or play games. This new component plugs in into the existing system, thus extending the functionality of the system.

7. Web Based Application Development

The Web is built on the HyperText Transfer Protocol. HTTP is a client/server request/reply protocol that is stateless. That is, the protocol does not make any association between one transaction and another; e.g.: time since the last transaction, type or client involved in the last transaction, what data was exchanged between the client and the server. As far as HTTP is concerned, each transaction is a discrete event. But this is not what we want in a shopping cart application because we need to preserve the user's shopping selection as they proceed with their purchase, in addition it is useful to have the access to their past purchase history and personal preferences.

Carrying information from one page to another can be achieved by several ways, such as Cookies, Session variables, Post variables, etc.

8. Database Connectivity

In e-commerce applications it is very typical for the Web server to contact the database to get information as needed. ASP.NET uses a technology called ActiveX Data Objects.NET (ADO.NET) to connect to the database

8.1. ADO.NET

Classic ASP pages used ActiveX Data Objects (ADO) to access and modify databases. ADO is a programming interface used to access data. This method was efficient and fairly easy for developers to learn and implement. However, ADO suffered from a dated model for data access with many limitations, such as the inability to transmit data so it is easily and universally accessible. Coupled with the move from standard SQL databases to more distributed types of data (such as XML), Microsoft introduced ADO.NET.

Although ADO.NET is known as the next evolution of ADO, it is very different from its predecessor. Whereas ADO was connection-based, ADO.NET relies on short, XML message-based interactions with data sources. This makes ADO.NET much more efficient for Internet-based applications.

A fundamental change from ADO to ADO.NET was the adoption of XML for data exchanges. XML is a text-based markup language, similar to HTML that presents an efficient way to represent data. This allows ADO.NET to reach and exchange. It also gives ADO.NET much better performance because XML data is easily converted to and from any type of data

Another major change is the way ADO.NET interacts with databases. ADO requires “locking” of database resources and lengthy connections for its applications, but ADO.NET does not; it uses disconnected data sets, which eliminates lengthy connections and database locks. This makes ADO.NET much more scalable because users are not in contention for database resources.

In ADO.NET there are two core objects that allow us to work with data initially: the `DataReader` and the `DataSet`. In any .NET data access page, before we connect to a database, we first have to import all the necessary namespaces that will allow us to work with the objects required. Namespace in .NET is a set of classes that can be used while creating an application. The .NET Framework has about 3,500 classes which can be accessed through a namespace. The application will be using a technology known as Open DataBase Connectivity (ODBC) to access the database; therefore we must first import necessary namespaces. Below is a sample namespace declaration used by .NET.

After all the necessary namespaces are imported, a connection to the database is made.

The above statement creates a connection to the database with an OdbcConnection object. This object tells ASP.NET where to go to get the data it needs. Since the data is stored in the same computer as the application, the SERVER is given as localhost. Next we open the connection object. Listed below are the common connection object methods we could work with:

- Open - Opens the connection to our database
- Close - Closes the database connection
- Dispose - Releases the resources on the connection object. Used to force garbage collecting, ensuring no resources are being held after our connection is used.
- State - Tells you what type of connection state your object is in, often used to check whether the connection is still using any resources.

Once the connection is made, in order to access the data in a database, ADO.NET relies on two components: DataSet and Data Provider. These components are explained below.

8.1.1. DataSet

The dataset is a disconnected, in-memory representation of data. It can be considered as a local copy of the relevant portions of the database. The DataSet resides in memory and the data in it can be manipulated and updated independent of the database. If necessary, changes made to the dataset can be applied to the central database. The data in DataSet can be loaded from any valid data source such as a text file, an XML database, Microsoft SQL server database, an Oracle database or MySQL database.

8.1.2. Data Provider

The Data Provider is responsible for providing and maintaining the connection to the database. A DataProvider is a set of related components that work together to provide data in an efficient and performance driven manner. Each DataProvider consists of the following component classes:

- The Connection object which provides a connection to the database
- The Command object which is used to execute a command
- The DataReader object which provides a read only, connected recordset
- The DataAdapter object which populates a disconnected DataSet with data and performs the update.

8.1.3. The Connection Object

The Connection object creates the connection to the database. Microsoft Visual Studio .NET provides two types of Connection classes: the SqlConnection object, which is designed specifically to connect to Microsoft SQL Server 7.0 or later, and the OleDbConnection object, which can provide connections to a wide range of database types like Microsoft Access and Oracle. The Connection object contains all of the information required to open a connection to the database.

8.1.4. The Command Object

The Command object is represented by two corresponding classes: SqlCommand and OleDbCommand. Command objects are used to execute commands to a database across a data connection. The Command objects can be used to execute stored procedures on the database, SQL commands, or return complete tables directly. Command objects provide three methods that are used to execute commands on the database:

ExecuteNonQuery:

Executes commands that have no return values such as INSERT, UPDATE or DELETE.

ExecuteScalar:

Returns a single value from a database query

ExecuteReader:

Returns a resultset by way of a DataReader object

The DataAdapter Object

The DataAdapter is the class at the core of ADO .NET's disconnected data access. It is essentially the middleman facilitating all communication between the database and a DataSet. The DataAdapter is used either to fill a DataTable or DataSet with its Fill method. After the memory-resident data has been manipulated, the DataAdapter can commit the changes to the database by calling the Update method. The DataAdapter provides four properties that represent database commands:

SelectCommand

InsertCommand

DeleteCommand

UpdateCommand

When the Update method is called, changes in the DataSet are copied back to the database and the appropriate InsertCommand, DeleteCommand, or UpdateCommand is executed.

8.2. Connecting ASP.NET application to a Database

The steps required to connect our ASP.NET application to the MySQL database and access the data are given below:

1. Import the required namespaces.

```
using System;
```

```
using System.Data;
```

```
using System.Data.Odbc;
```

2. Create a connection object.

```
string myConnectionString;
```

```
myConnectionString = "DRIVER = {MySQL ODBC 3.51 Driver};
```

```
SERVER=localhost;DATABASE = project; UID
```

```
= root; PASSWORD = ""
```

```
OdbcConnection odbcCon = new  
OdbcConnection(myConnectionString)
```

3. Create a SQL query

```
string str;
```

```
str="Select * from Customer where UserID='admin';
```

4. Create a Command object to run the SQL query

```
odbcCmd=new OdbcCommand(str,odbcCon);
```

5. DataReader to read the result

6. Close odbcReader and odbcConnection

```
odbcReader.Close();
```

```
odbcCon.Close();
```

The data can now be used as desired by the application

9. The E-Commerce Shopping Cart Application

Website consists of the following web pages:

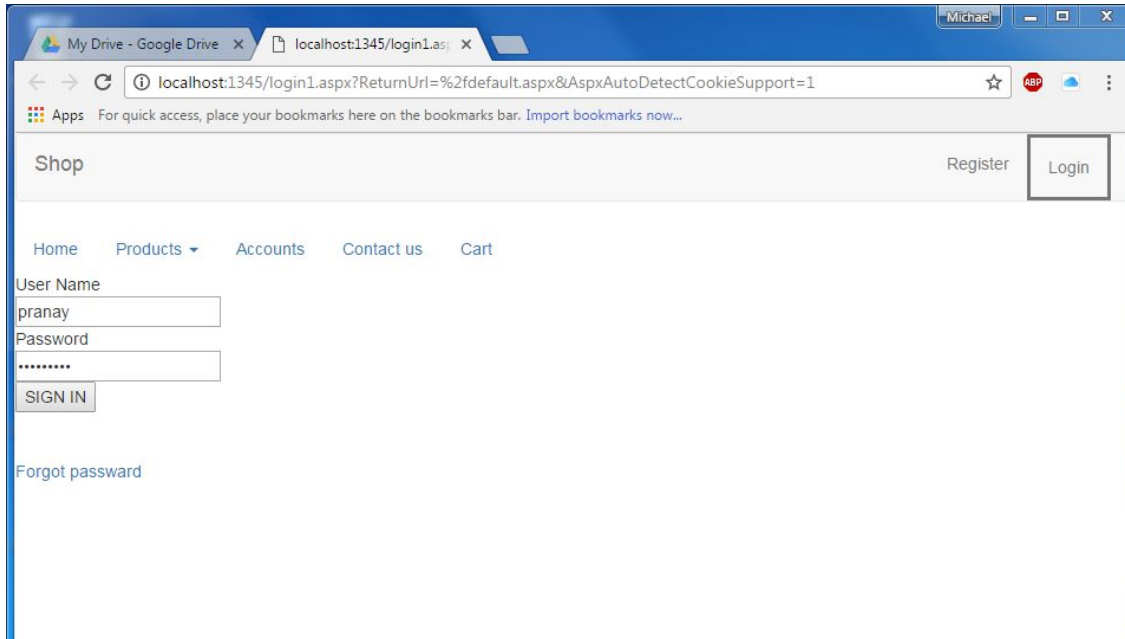
1. Account.aspx
2. AdminDelete.aspx
3. AdminEntry.aspx
4. cCart.aspx
5. confirm.aspx
6. contact.aspx
7. default.aspx
8. error.aspx
9. login1.aspx
10. ManageUser.aspx
11. pant.aspx
12. productDetail.aspx
13. Shorts.aspx
14. site1.master
15. tshirt.aspx
16. Web.config

These are the mail server pages. There are many sub-pages corresponding to these pages.. Such as Account.aspx.cs ..etc

Below figures show some screenshots taken from running the application. All the functionalities are explained accordingly.

10. Screenshots of the application

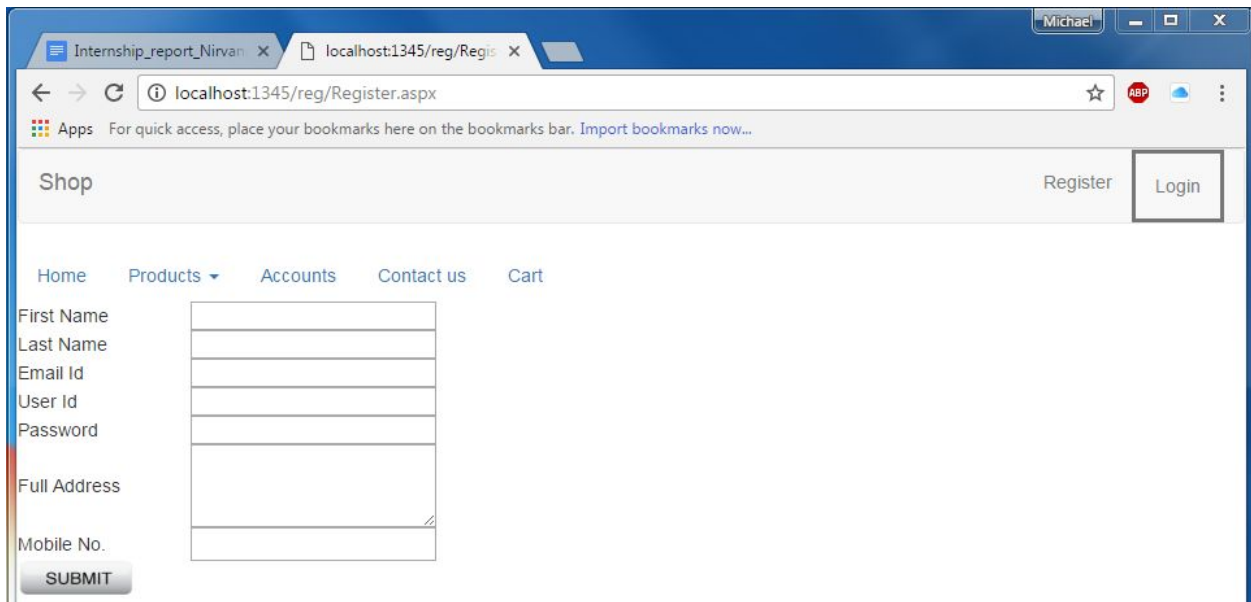
Login Page :



A screenshot of a web browser displaying the login page of an application. The browser's address bar shows the URL `localhost:1345/login1.aspx?ReturnUrl=%2fdefault.aspx&AspxAutoDetectCookieSupport=1`. The page has a header with a "Shop" label on the left and "Register" and "Login" buttons on the right. Below the header is a navigation menu with links for "Home", "Products", "Accounts", "Contact us", and "Cart". The main form area contains labels for "User Name" and "Password", each followed by a text input field. The "User Name" field contains the text "pranay". Below the password field is a "SIGN IN" button. At the bottom of the form is a link that says "Forgot password".

This is the login page. Where an admin/user can login through this page.

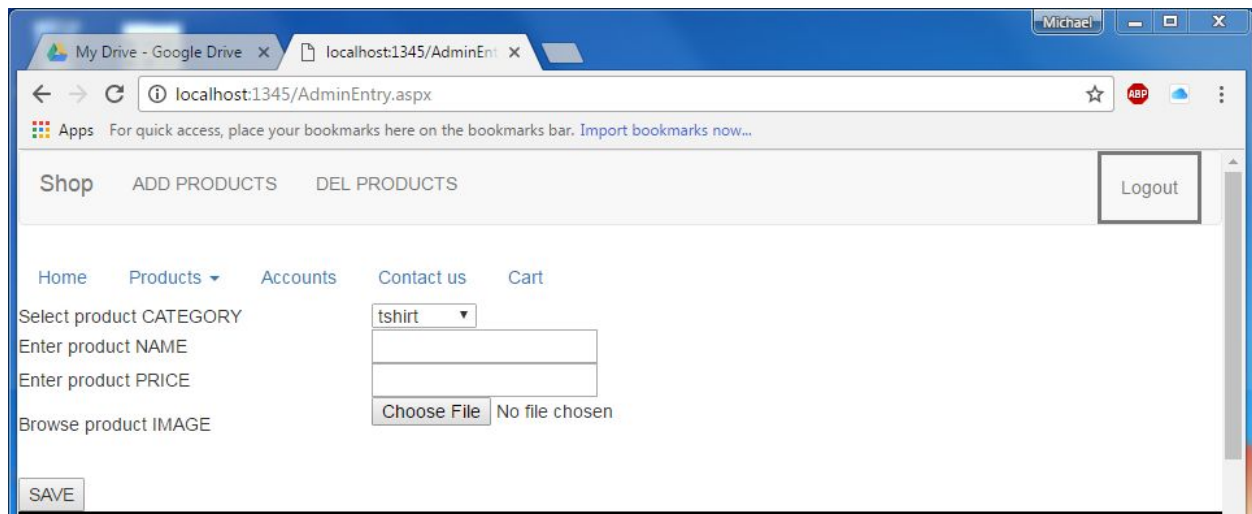
Registration page :



A screenshot of a web browser displaying the registration page. The browser's address bar shows the URL `localhost:1345/reg/Register.aspx`. The page layout is similar to the login page, with a "Shop" header and "Register" and "Login" buttons. The navigation menu is also present. The registration form includes labels for "First Name", "Last Name", "Email Id", "User Id", "Password", "Full Address", and "Mobile No.", each followed by a text input field. A "SUBMIT" button is located at the bottom of the form.

A user can register through this page.

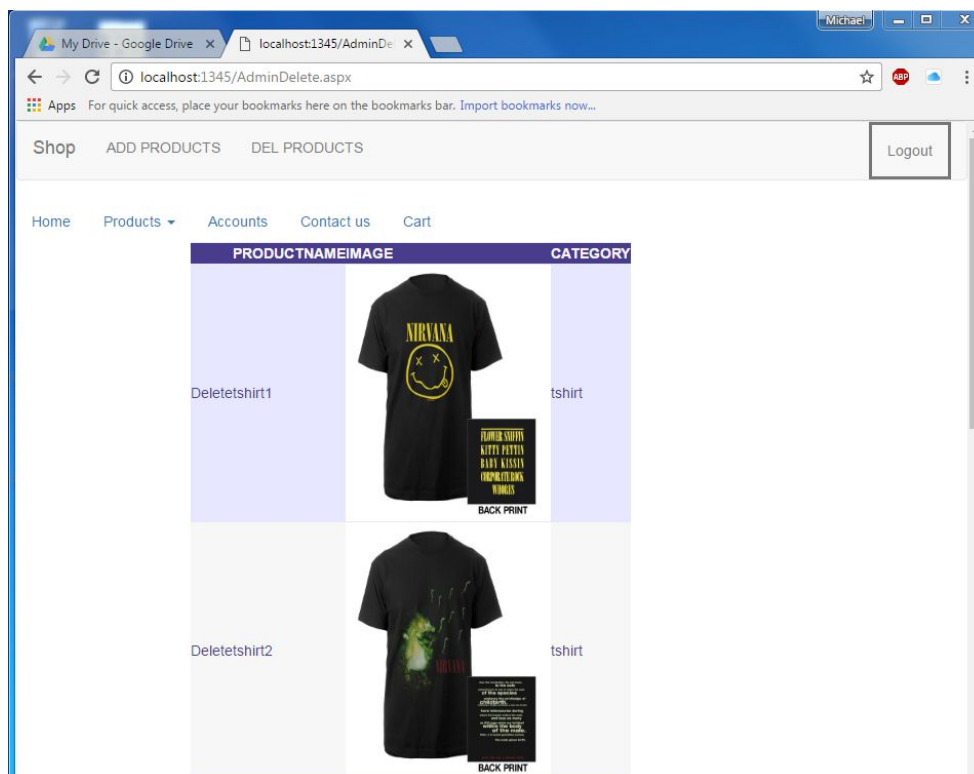
Admin Page/add products :





The screenshot shows a web browser window with the URL `localhost:1345/AdminEntry.aspx`. The page has a navigation bar with links for **Shop**, **ADD PRODUCTS**, and **DEL PRODUCTS**, and a **Logout** button. Below the navigation bar, there are links for **Home**, **Products**, **Accounts**, **Contact us**, and **Cart**. The main form includes a dropdown menu for **Select product CATEGORY** (currently set to **tshirt**), text input fields for **Enter product NAME** and **Enter product PRICE**, and a file selection area for **Browse product IMAGE** with a **Choose File** button and the text **No file chosen**. A **SAVE** button is located at the bottom left of the form.

When an admin logs in. He has the privileges to add products.

Admin Page/delete products :

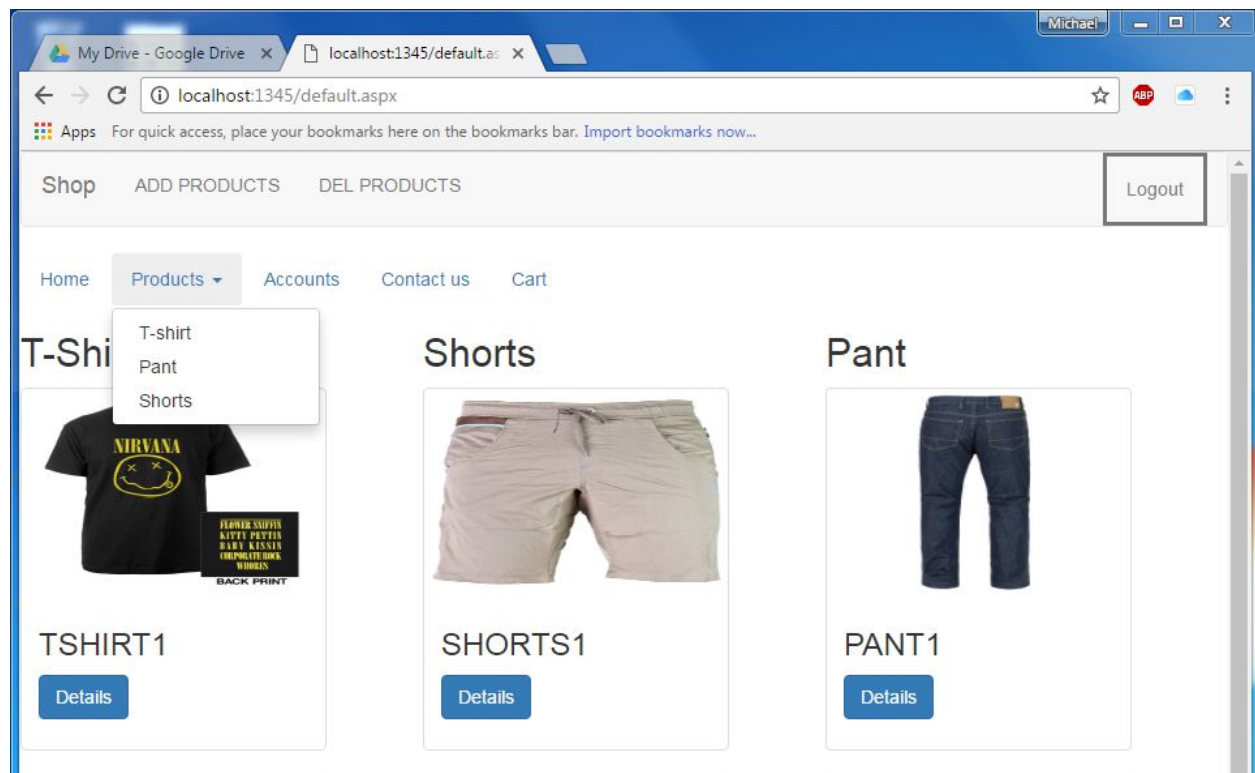


The screenshot shows a web browser window with the URL `localhost:1345/AdminDelete.aspx`. The page has a navigation bar with links for **Shop**, **ADD PRODUCTS**, and **DEL PRODUCTS**, and a **Logout** button. Below the navigation bar, there are links for **Home**, **Products**, **Accounts**, **Contact us**, and **Cart**. The main content area displays a table with two columns: **PRODUCTNAMEIMAGE** and **CATEGORY**.

PRODUCTNAMEIMAGE	CATEGORY
<div>Deletetshirt1</div>  <div>BACK PRINT</div>	tshirt
<div>Deletetshirt2</div>  <div>BACK PRINT</div>	tshirt

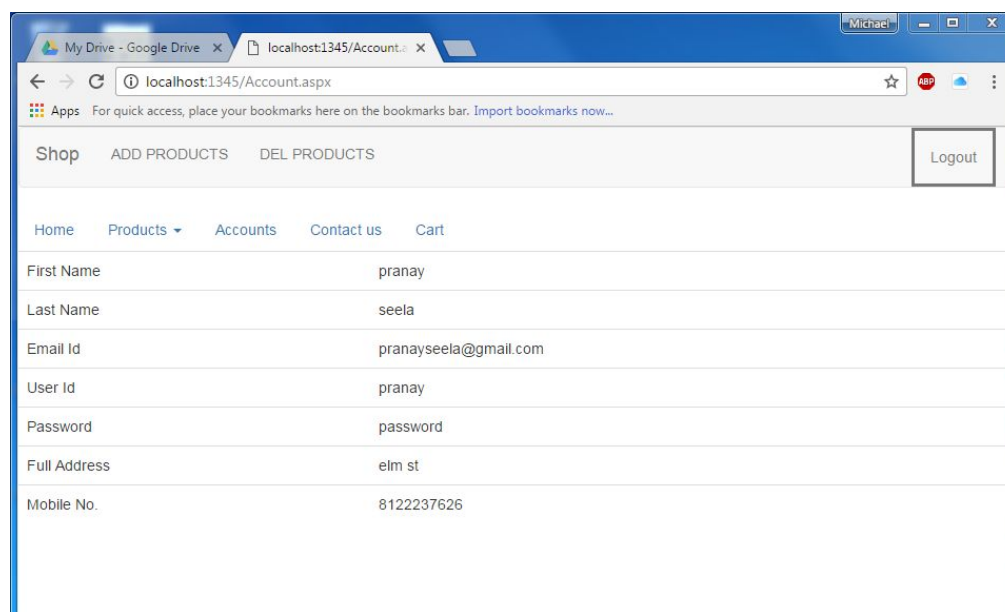
Admin has the privileges to delete a product from the web portal.

Products/category drop down :



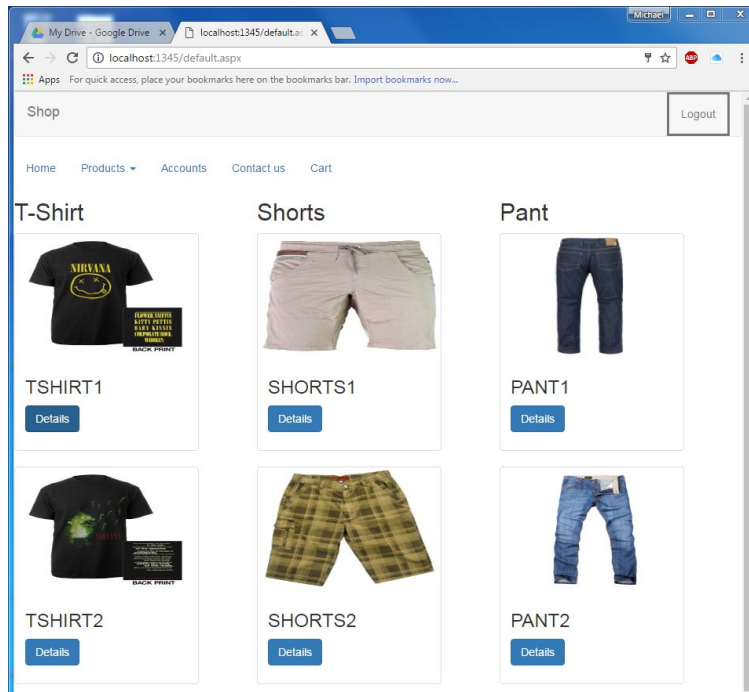
Can select which type of clothing/merchandise you want to shop.

Accounts Page :



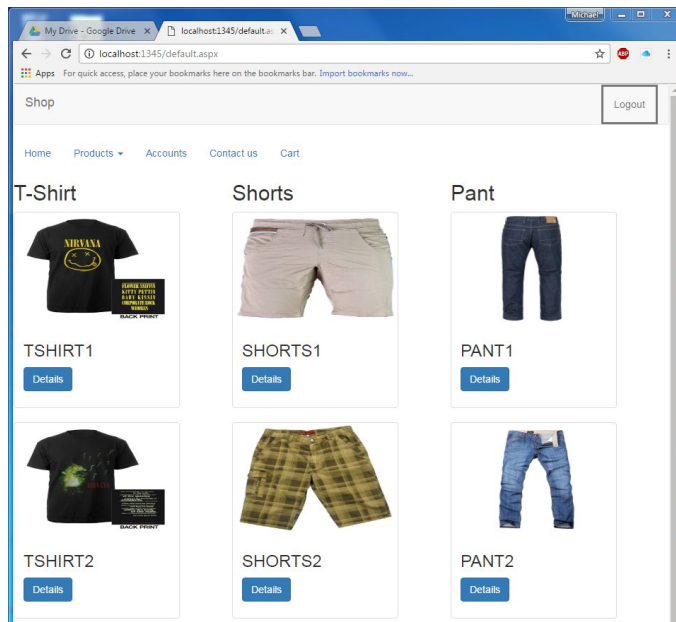
Logged in person either admin/user. The account information of that particular person is displayed in this page.

Logged in as User:



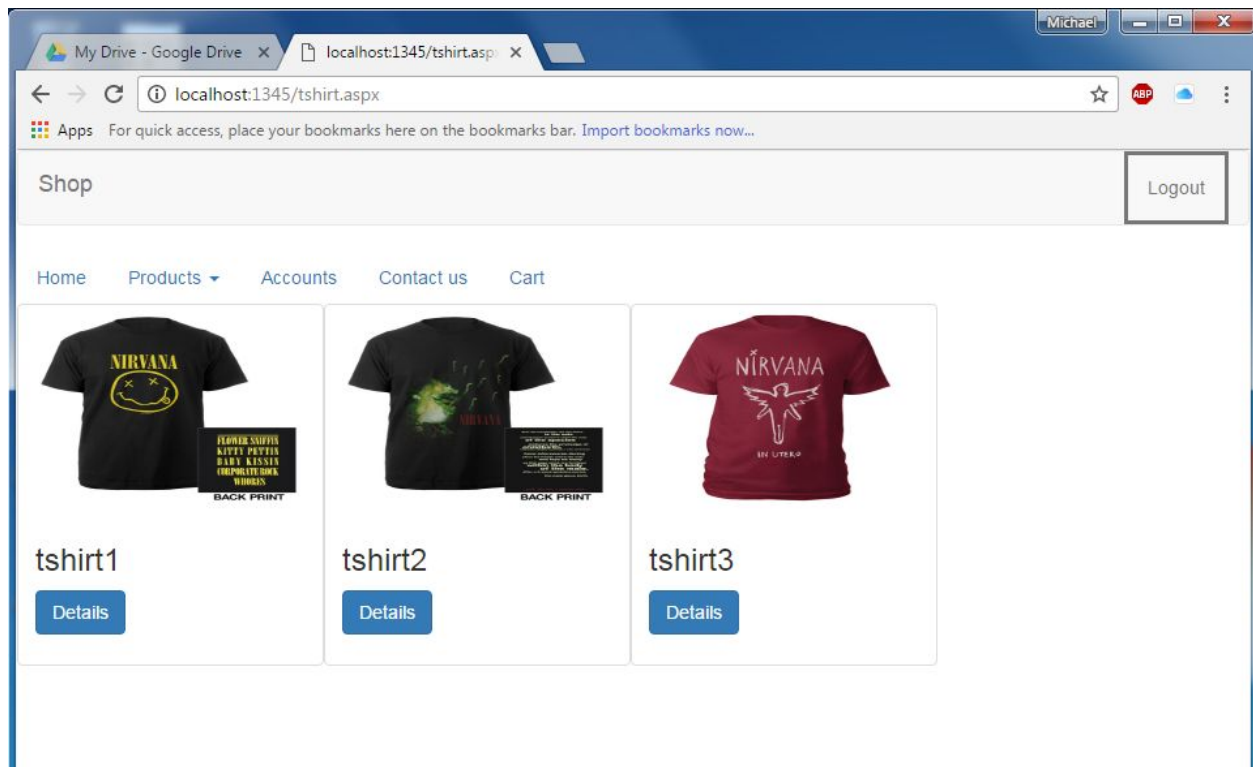
User cannot add/delete the products. So it is hidden from the user login page.

Home/default page :

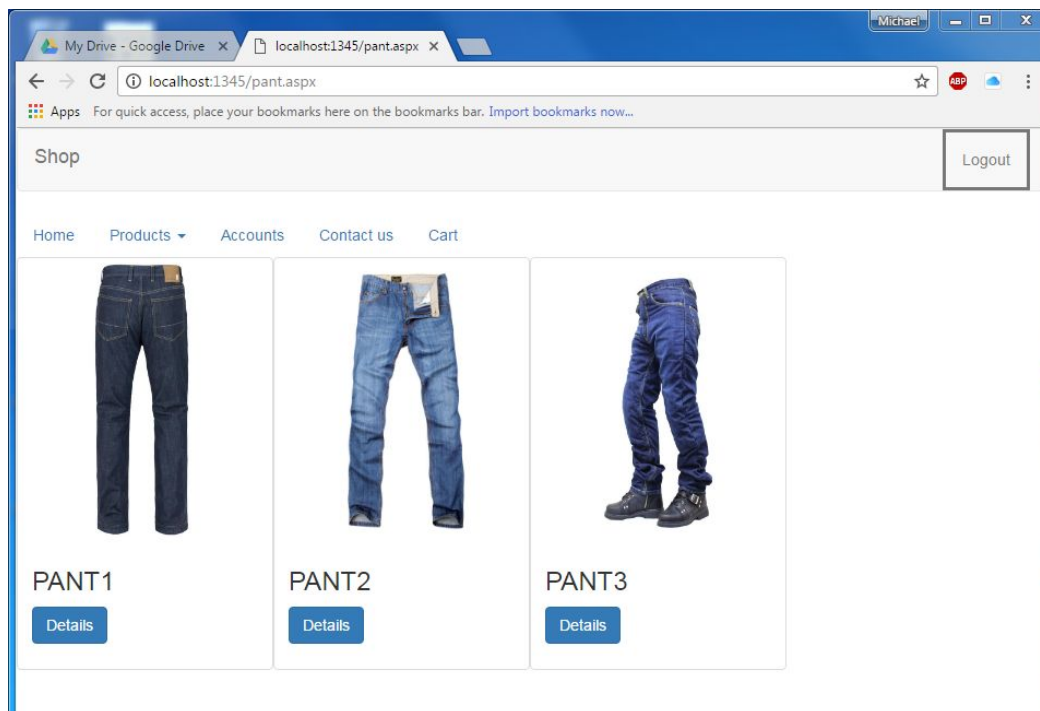


The top products of each type are displayed in the home page. It is something like the featured products page.

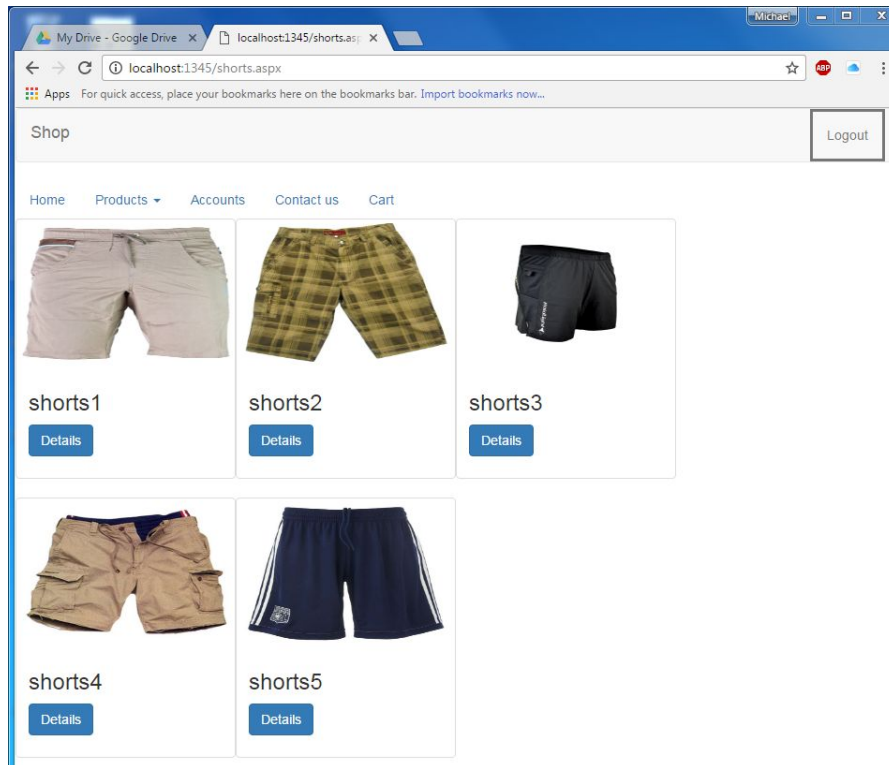
T-Shirts page :



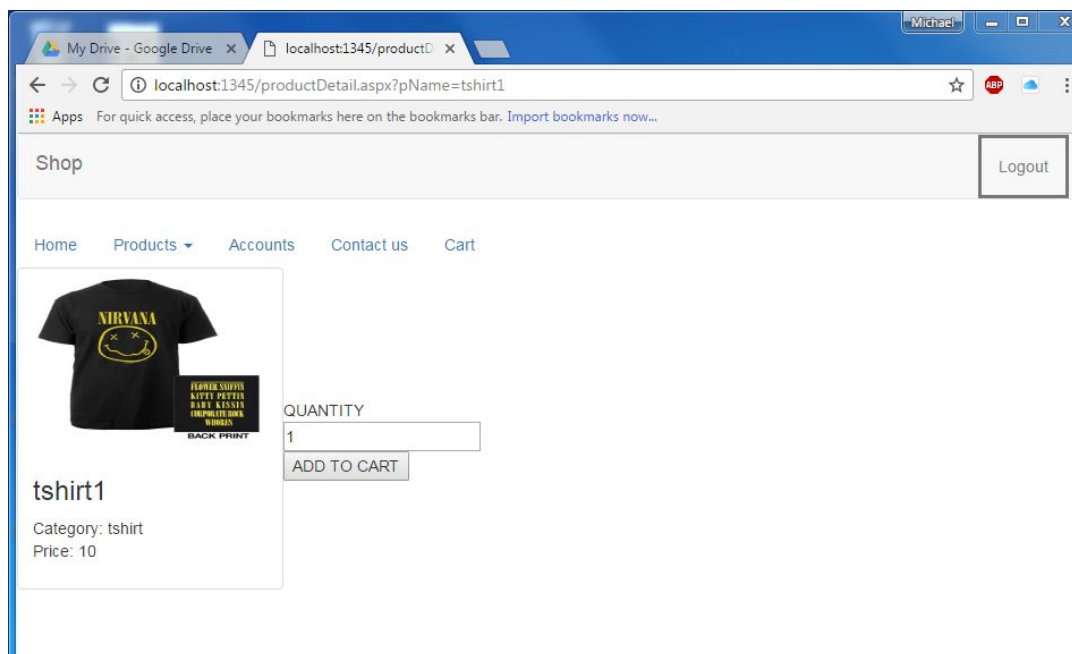
Pants page :



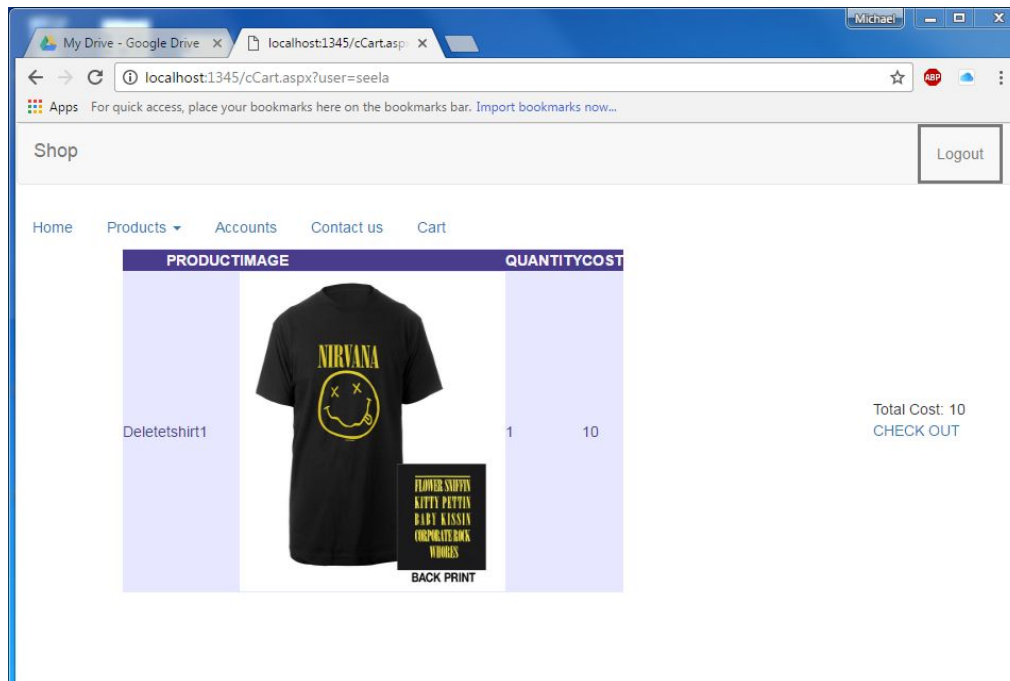
Shorts Page :



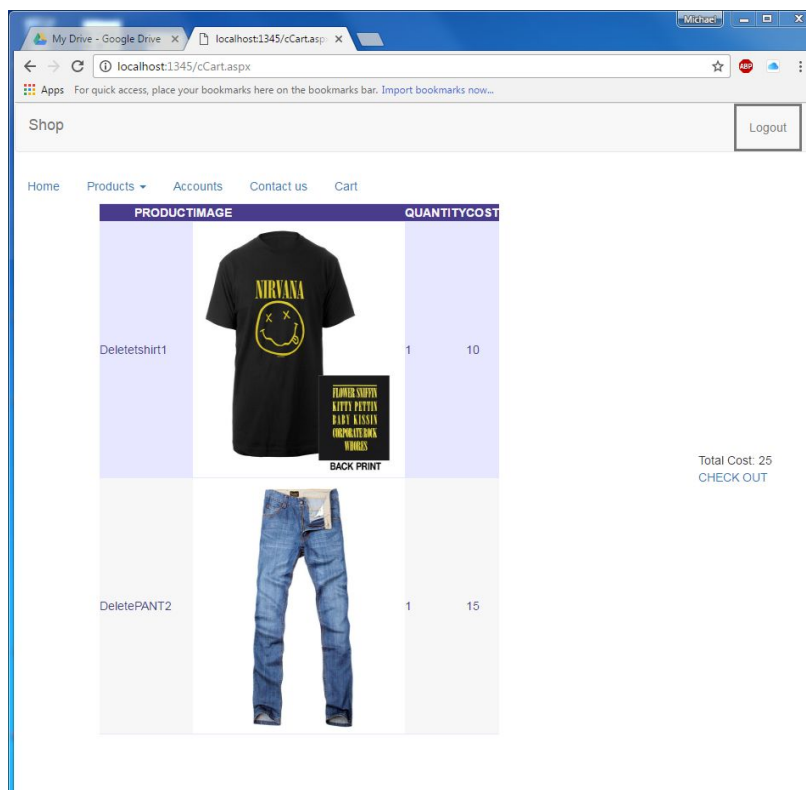
Product detail page :



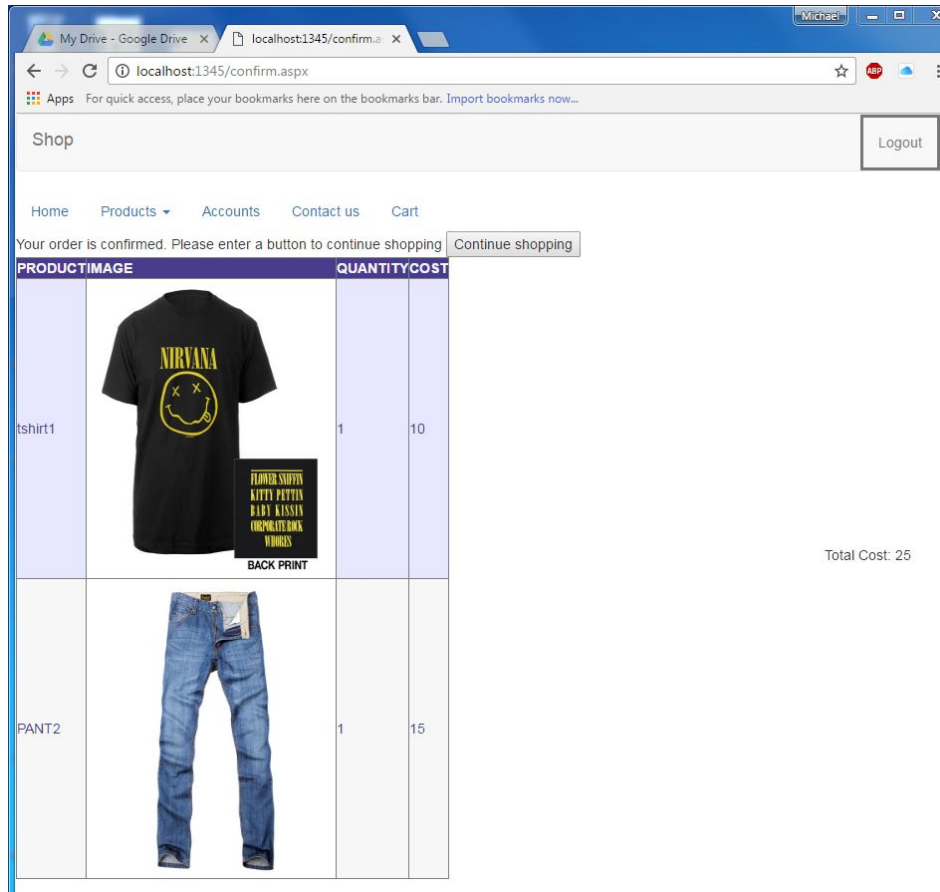
Product cart :



Cart :



Checkout :



11. Limitations and Future Development

There are some limitations for the current system to which solutions can be provided as a future development:

1. The system is not configured for multi-users at this time. The concept of transaction can be used to achieve this.
2. The Website is not accessible to everyone. It can be deployed on a web server so that everybody who is connected to the Internet can use it.
3. Credit Card validation is not done. Third party proprietary software can be used for validation check

As for other future developments, the following can be done:

1. The Administrator of the web site can be given more functionalities, like looking at a specific customer's profile, the books that have to be reordered, etc.
2. Multiple Shopping carts can be allowed

12. Conclusion

The Internet has become a major resource in modern business, thus electronic shopping has gained significance not only from the entrepreneur's but also from the customer's point of view. For the entrepreneur, electronic shopping generates new business opportunities and for the customer, it makes comparative shopping possible. As per a survey, most consumers of online stores are impulsive and usually make a decision to stay on a site within the first few seconds. "Website design is like a shop interior. If the shop looks poor or like hundreds of other shops the customer is most likely to skip to the other site". Hence we have designed the project to provide the user with easy navigation, retrieval of data and necessary feedback as much as possible.

In this project, the user is provided with an e-commerce web site that can be used to buy books online. To implement this as a web application we used ASP.NET as the Technology. ASP.NET has several advantages such as enhanced performance, 66 scalability, built-in security and simplicity. To build any web application using ASP.NET we need a programming language such as C#, VB.NET, J# and so on. C# was the language used to build this application. For the client browser to connect to the ASP.NET engine we used Microsoft's Internet Information Services (IIS) as the Web Server. ASP.NET uses ADO.NET to interact with the database as it provides in-memory caching that eliminates the need to contact the database server frequently and it can easily deploy and maintain an ASP.NET application. MySQL was used as back-end database since it is one of the most popular open source databases, and it provides fast data access, easy installation and simplicity.

A good shopping cart design must be accompanied with user-friendly shopping cart application logic. It should be convenient for the customer to view the contents of their cart and to be able to remove or add items to their cart. The shopping cart application described in this project provides a number of features that are designed to make the customer more comfortable.

This project helps in understanding the creation of an interactive web page and the technologies used to implement it. The design of the project which includes Data Model and Process Model illustrates how the database is built with different tables, how the data is accessed and processed from the tables. The building of the project has given me a precise knowledge about how ASP.NET is used to develop a website, how it connects to the database to access the data and how the data and web pages are modified to provide the user with a shopping cart application.

13. Bibliography

13.1. Websites :

- 13.1.1. <http://encyclopedia.laborlawtalk.com/IIS>
- 13.1.2. <http://aspnet.4guysfromrolla.com/articles/020404-1.aspx>
- 13.1.3. <http://samples.gotdotnet.com/quickstart/aspplus/doc/mtstransactions.aspx>
- 13.1.4. <http://aspnet.4guysfromrolla.com/articles/011404-1.aspx>
- 13.1.5. <http://msdn.microsoft.com/>
- 13.1.6. <https://www.mysql.com/>
- 13.1.7. <http://dev.mysql.com/downloads/windows/>
- 13.1.8. <https://www.visualstudio.com/free-developer-offers/>
- 13.1.9. <https://www.asp.net/web-forms/overview/getting-started/getting-started-with-aspnet-45-web-forms/aspnet-error-handling>
- 13.1.10. <http://asp.net-tutorials.com/mysql/getting-started/>
- 13.1.11. <http://stackoverflow.com/>
- 13.1.12. <https://code.msdn.microsoft.com/>