Implementing Security on E-Commerce Website

A Thesis submitted in partial fulfillment of the requirements for the Degree of

Bachelor of Technology
IN
Computer Science and Engineering

By

Anshul Tayal
101247

Department of Computer Science and Engineering
Jaypee University of Information Technology
Waknaghat, P.O. Waknaghat,
Teh Kandaghat, Distt. Solan
PIN-173 234
(H.P.), India

CERTIFICATE

This is to certify that the thesis entitled, “***Implementing Security on E-Commerce Website***” submitted by ***Anshul Tayal*** in partial fulfillment of the requirement for the award of Bachelor of Technology degree in Computer Science and Engineering at Jaypee University of Information Technology, Waknaghat is an authentic work carried out by him under my supervision and guidance. To the best of my knowledge, the matter embodied in the thesis has not been submitted to any other University/Institute for the award of any Degree or Diploma.

Date:

Mr. Amit Kumar Singh
Assistant Professor (Grade-II)
Dept. of CSE, JUIT

ACKNOWLEDGEMENT

This is a great opportunity to acknowledge and to thank all those persons without whom this project would have been impossible. I would like to add a few heartfelt words for my parents and the people who were a part of this project in numerous ways. I am extremely grateful to Mr. Amit Kumar Singh, for her indefatigable guidance, valuable suggestion, moral support, constant encouragement and contribution of time for the successful completion of project work. I am very thankful to him, for providing all the facilities needed during the project development. I would also like to extend my sincere thanks to all the staff members of the JUIT for providing us with different facilities such as computer support and library infrastructure support etc. required for the project.

Anshul Tayal
101247
CSE

Contents

Abstract

Electronic Commerce is process of doing business through computer networks. A person sitting on his chair in front of a computer can access all the facilities of the Internet to buy or sell the products.

E-commerce Security is a part of the Information Security framework and is specifically applied to the components that affect e-commerce that include Computer Security, Data security and other wider realms of the Information Security framework. E-commerce security has its own particular nuances and is one of the highest visible security components that affect the end user through their daily payment interaction with business.

E-commerce security is the protection of e-commerce assets from unauthorized access, use, alteration, or destruction. Dimensions of e-commerce security are Integrity, Non-repudiation, Authenticity, Confidentiality, Privacy and Availability. E-Commerce offers the banking industry great opportunity, but also creates a set of new risks and vulnerability such as security threats. Information security, therefore, is an essential management and technical requirement for any efficient and effective Payment transaction activities over the internet. Still, its definition is a complex Endeavour due to the constant technological and business change and requires a coordinated match of algorithm and technical solutions.

You may not think that your site has anything worth being hacked for, but web sites are compromised all the time. The majority of website security breaches are not to steal your data or deface your website, but instead attempt to use your server as an E-mail relay for spam, or to setup a temporary web server, normally to serve files of an Illegal nature.

Table Index

Figure Index

Index

Title……………………………………………………..Page No

1. ***Project Design***

a.	Name of the Project	Implement Security on E-Commerce Website
b.	Objective/ Vision	Study the Overview of E-commerce security. Understand the Online Shopping - Steps to place an order. Understand the purpose of Security in E-commerce. Discuss the different security issues in E-commerce. Understand the Secure online shopping guidelines.
c.	Users of the System	• Admin • Customer as Trade Account type • Customer as Personal Account type
d.	Functional Requirements	• User Roles - Check Product Detail, Add Product to Cart, Update Cart, Register, Login, Check Category • Admin Roles - Check User Details, Check Order Details, Change Status of Order (New, Dispatched, Cancel), Delete User, Change User Details, Add New Category, Edit any Category Detail, Delete Added Category, Add New Product, Edit Product Detail, Delete any Product • User Account can be personal or trade. • Browse the product catalog. • Making an order. • Generate an invoice for an order that has been completed. • Product name. • Product description. • Product category. • Product price and tax.
e.	Non-functional requirements	• Site Security in terms of Access Control, Authentication, Confidentiality and Integrity of Data • Performance or speed of the system • Quality

		• Size • Ease of use
f.	Other important issues	Website should be highly customizable and flexible enough to easily deploy.
g.	Technologies to be used	HTML, CSS, JavaScript, PHP, MySQL
h.	Tools to be used	XAMPP
i.	Security Algorithm to be used	Hash Algorithm(SHA-1) and Public Key Encryption Algorithm(RSA)

2. ***Introduction***

Shopcade is an e-commerce website in which we have to mainly implement the security aspects. Here we are going to discuss about the security issues that are mainly happen in any E-Commerce Websites and how to overcome those issues. While many security issues in e-commerce are the same as general security issues, some of them are specific for the kind of software used by e-commerce businesses: databases, in particular databases which are accessed remotely, online forms, and shopping carts.

E-commerce Security is a part of the Information Security framework and is specifically applied to the components that affect e-commerce that include Computer Security, Data security and other wider realms of the Information Security framework. E-commerce security has its own particular nuances and is one of the highest visible security components that affect the end user through their daily payment interaction with business.

Today, privacy and security are a major concern for electronic technologies. M-commerce shares security concerns with other technologies in the field. Privacy concerns have been found, revealing a lack of trust in a variety of contexts, including commerce, electronic health records, e-recruitment technology and social networking, and this has directly influenced users.   Security is one of the principal and continuing concerns that restrict customers and organizations engaging with ecommerce. Web e-commerce applications that handle payments (online banking, electronic transactions or using debit cards, credit cards, PayPal or other tokens) have more compliance issues, are at increased risk from being targeted than other websites and there are greater consequences if there is data loss or alteration.

Online shopping through shopping websites having certain steps to buy a product with safe and secure. The e-commerce industry is slowly addressing security issues on their internal networks. There are guidelines for securing systems and networks available for the ecommerce systems personnel to read and implement. Educating the consumer on security issues is still in the infancy stage but will prove to be the most critical element of the e-commerce security architecture. Trojan horse programs launched against client systems pose the greatest threat to e-commerce because they can bypass or subvert most of the authentication and authorization mechanisms used in an e- commerce transaction.[1]

3. Literature Survey

3.1. Hyper Text Markup Language [2]

In 1980, physicist Tim Berners-Lee, who was a contractor at CERN, proposed and prototyped ENQUIRE, a system for CERN researchers to use and share documents. In 1989, Berners-Lee wrote a memo proposing an Internet-based hypertext system.

Berners-Lee specified HTML and wrote the browser and server software in late 1990. That year, Berners-Lee and CERN data systems engineer Robert Cailliau collaborated on a joint request for funding, but the project was not formally adopted by CERN. In his personal notes from 1990 he listed "some of the many areas in which hypertext is used" and put an encyclopedia first.

The first publicly available description of HTML was a document called "HTML Tags", first mentioned on the Internet by Berners-Lee in late 1991. It describes 18 elements comprising the initial, relatively simple design of HTML. Except for the hyperlink tag, these were strongly influenced by SGMLguid, an in-house SGML-based documentation format at CERN. Eleven of these elements still exist in HTML 4.

Hypertext Markup Language is a markup language that web browsers use to interpret and compose text, images and other material into visual or audible web pages. Default characteristics for every item of HTML markup are defined in the browser, and these characteristics can be altered or enhanced by the web page designer's additional use of CSS. Many of the text elements are found in the 1988 ISO technical report TR 9537Techniques for using SGML, which in turn covers the features of early text formatting languages such as that used by the RUNOFF command developed in the early 1960s for the CTSS (Compatible Time-Sharing System) operating system: these formatting commands were derived from the commands used by typesetters to manually format documents. However, the SGML concept of generalized markup is based on elements (nested annotated ranges with attributes) rather than merely print effects, with also the separation of structure and markup; HTML has been progressively moved in this direction with CSS.

- HTML is a language for describing web pages.
- HTML stands for Hyper Text Markup Language.
- HTML is a markup language. A markup language is a set of markup tags.
- HTML documents contain HTML tags and plain text.
- HTML documents are also called web pages.

HTML5[2] is a core technology markup language of the Internet used for structuring and presenting content for the World Wide Web. It is the fifth revision of the HTML standard (created in 1990 and standardized as HTML 4 as of 1997) and, as of December 2012, is a candidate recommendation of the World Wide Web Consortium (W3C). Its core aims have been to improve the language with support for the latest multimedia while keeping it easily readable by humans and consistently understood by computers and devices (web browsers, parsers, etc.). HTML5 is intended to subsume not only HTML 4, but also XHTML 1 and DOM Level 2 HTML.

Following its immediate predecessors HTML 4.01 and XHTML 1.1, HTML5 is a response to the fact that the HTML and XHTML in common use on the World Wide Web are a mixture of features introduced by various specifications, along with those introduced by software products such as web browsers, those established by common practice, and the many syntax errors in existing web documents. It is also an attempt to define a single markup language that can be written in either HTML or XHTML syntax. It includes detailed processing models to encourage more interoperable implementations; it extends, improves and rationalizes the markup available for documents, and introduces markup and application programming interfaces (APIs) for complex web applications. For the same reasons, HTML5 is also a potential candidate for cross-platform mobile applications. Many features of HTML5 have been built with the consideration of being able to run on low-powered devices such as smartphones and tablets. In December 2011, research firm Strategy Analytics forecast sales of HTML5 compatible phones would top 1 billion in 2013.

In particular, HTML5 adds many new syntactic features. These include the new <video>, <audio> and <canvas> elements, as well as the integration of scalable vector graphics (SVG) content (replacing generic <object> tags), and MathML for mathematical formulas. These features are designed to make it easy to include and handle multimedia and graphical content on the web without having to resort to proprietary plugins and APIs.

Other new elements, such as <section>, <article>, <header> and <nav>, are designed to enrich the semantic content of documents. New attributes have been introduced for the same purpose, while some elements and attributes have been removed. Some elements, such as <a>, <cite> and <menu> have been changed, redefined or standardized. The APIs and Document Object Model (DOM) are no longer afterthoughts, but are fundamental parts of the HTML5 specification. HTML5 also defines in some detail the required processing for invalid documents so that syntax errors will be treated uniformly by all conforming browsers and other user agents.

3.2. Cascading Style Sheet[3]

When a browser reads a style sheet, it will format the document according to it.
There are three ways of inserting a style sheet:

- External style sheet
- Internal style sheet
- Inline style

3.2.1. External Style Sheet

An external style sheet is ideal when the style is applied to many pages. With an external style sheet, you can change the look of an entire Web site by changing one file. Each page must link to the style sheet using the <link> tag. The <link> tag goes inside the head section:
hr {color: sienna ;} p {margin-left: 20px ;} body { background – image : url (" images / back40 .gif ");}

3.2.2. Internal Style Sheet

An internal style sheet should be used when a single document has a unique style. You define internal styles in the head section of an HTML page, by using the <style> tag, like this: <Head> <style> hr {color: sienna ;} p {margin-left: 20px ;} body {background-image: url ("images/back40.gif") ;} </style> </head>. An inline style loses many of the advantages of style sheets by mixing content with presentation. Use this method sparingly! To use inline styles you use the style attribute in the relevant tag. The style attribute can contain any CSS property. The example shows how to change the color and the left margin of a paragraph: <p style="color: sienna; margin-left: 20px ;"> this is a paragraph. </p>

3.2.3. Multiple Style Sheets

If some properties have been set for the same selector in different style sheets, the values will be inherited from the more specific style sheet. For example, an external style sheet has these properties for the h3 selector:
h3 {color: red; text-align: left; font-size: 8pt ;}

3.3. <u>Java Script[4][5]</u>

JavaScript is the world's most popular programming language. It is the language for HTML, for the web, for servers, PCs, laptops, tablets, phones, and more. JavaScript is a Scripting Language. A scripting language is a lightweight programming language. JavaScript is programming code that can be inserted into HTML pages. JavaScript code can be executed by all modern web browsers. JavaScript is easy to learn.

3.3.1. JavaScript: Reacting to Events

The alert () function is not much used in JavaScript, but it is quite handy for trying out code. The onclick event is only one of the many HTML events you will learn about in this tutorial.

3.3.2. JavaScript: Changing HTML Content

Using JavaScript to manipulate the content of HTML elements is a very common. JavaScript Functions and Events The JavaScript statements, in the example above, are executed while the page loads. More often, we want to execute code when an event occurs, like when the user clicks a button. If we put JavaScript code inside a function, we can call that function when an event occurs.

3.3.3. External Java Scripts

Scripts can also be placed in external files. External files often contain code to be used by several different web pages. External JavaScript files have the file extension .js. To use an external script, point to the .js file in the "src" attribute of the <script> tag:
<! DOCTYPE html> <html> <body> <script src="myScript.js"></script> </body> </html>

3.4. PHP[6][7]

PHP is a server-side scripting language designed for web development but also used as a general-purpose programming language. PHP is now installed on more than 244 million websites and 2.1 million web servers. Originally created by Rasmus Lerdorf in 1995, the reference implementation of PHP is now produced by The PHP Group. While PHP originally stood for Personal Home Page, it now stands for PHP: Hypertext Preprocessor, a recursive acronym. PHP code is interpreted by a web server with a PHP processor module, which generates the resulting web page: PHP commands can be embedded directly into an HTML source document rather than calling an external file to process data. It has also evolved to include a command-line interface capability and can be used in standalone graphical applications. PHP is free software released under the PHP License, which is incompatible with the GNU General Public License (GPL) due to restrictions on the usage of the term PHP. PHP can be deployed on most web servers and also as a standalone shell on almost every operating system and platform, free of charge.

- PHP is an acronym for "PHP Hypertext Preprocessor"
- PHP is a widely-used, open source scripting language
- PHP scripts are executed on the server
- PHP costs nothing, it is free to download and use PHP files can contain text, HTML, CSS, JavaScript, and PHP code
- PHP code are executed on the server, and the result is returned to the browser as plain HTML
- PHP files have extension ".php". PHP can generate dynamic page content
- PHP can create, open, read, write, and close files on the server
- PHP can collect form data
- PHP can send and receive cookies
- PHP can add, delete, modify data in your database
- PHP can restrict users to access some pages on your website
- PHP can encrypt data
- PHP runs on various platforms (Windows, Linux, Unix, Mac OS X, etc.)
- PHP is compatible with almost all servers used today (Apache, IIS, etc.)
- PHP supports a wide range of databases
- PHP is easy to learn and runs efficiently on the server side

3.5. ***MY SQL[8]***

- SQL is a standard language for accessing and manipulating databases.
- SQL stands for Structured Query Language
- SQL lets you access and manipulate databases
- SQL is an ANSI (American National Standards Institute) standard
- SQL can execute queries against a database
- SQL can retrieve data from a database
- SQL can insert records in a database
- SQL can update records in a database
- SQL can delete records from a database
- SQL can create new databases
- SQL can create new tables in a database
- SQL can create stored procedures in a database
- SQL can create views in a database
- SQL can set permissions on tables, procedures, and views

Although SQL is an ANSI (American National Standards Institute) standard, there are different versions of the SQL language. However, to be compliant with the ANSI standard, they all support at least the major commands (such as SELECT, UPDATE, DELETE, INSERT, WHERE) in a similar manner.

3.5.1. Using SQL in Your Web Site

To build a web site that shows data from a database, you will need:
An RDBMS database program (i.e. MS Access, SQL Server, MySQL)
To use a server-side scripting language, like PHP or ASP
To use SQL to get the data you want to use HTML / CSS

3.5.2. RDBMS

RDBMS stands for Relational Database Management System. RDBMS is the basis for SQL, and for all modern database systems such as MS SQL Server, IBM DB2, Oracle, MySQL, and Microsoft Access. The data in RDBMS is stored in database objects called tables. A table is a collection of related data entries and it consists of columns and rows.

SQL Statement[9]	**Syntax**

AND / OR	SELECT column_name(s)
	FROM table_name
	WHERE condition
	AND|OR condition

CREATE DATABASE	CREATE DATABASE database_name
CREATE TABLE	CREATE TABLE table_name

	(column_name1 data_type, column_name2 data_type, column_name2 data_type, ...)
CREATE INDEX	CREATE INDEX index_name ON table_name (column_name) or CREATE UNIQUE INDEX index_name ON table_name (column_name)

CREATE VIEW	CREATE VIEW view_name AS
	SELECT column_name(s)
	FROM table_name
	WHERE condition

DELETE	DELETE FROM table_name WHERE some_column=some_value or DELETE FROM table_name (**Note:** Deletes the entire table!!) DELETE * FROM table_name (**Note:** Deletes the entire table!!)
DROP DATABASE	DROP DATABASE database_name
DROP INDEX	DROP INDEX index_name (MySQL)
DROP TABLE	DROP TABLE table_name

INSERT INTO	INSERT INTO table_name
	VALUES (value1, value2, value3,....)
	
	or
	
	INSERT INTO table_name
	(column1, column2, column3,...)
	VALUES (value1, value2, value3,....)

ORDER BY	SELECT column_name(s)
	FROM table_name
	ORDER BY column_name [ASC|DESC]

SELECT	SELECT column_name(s)
	FROM table_name

SELECT *	SELECT *
	FROM table_name

SELECT TOP	SELECT TOP number|percent column_name(s)
	FROM table_name

TRUNCATE TABLE	TRUNCATE TABLE table_name

UPDATE	UPDATE table_name
	SET column1=value, column2=value,...
	WHERE some_column=some_value

WHERE	SELECT column_name(s)
	FROM table_name
	WHERE column_name operator value

Table – 1

Use the PHP mysqli_connect () function to open a new connection to the MySQL server.

3.5.3. Open a Connection to the MySQL Server

Before we can access data in a database, we must open a connection to the MySQL server.

In PHP, this is done with the mysqli_connect () function.

Syntax
mysqli_connect (host, username, password, dbname);

Parameter	**Description**
Host	Optional. Either a host name or an IP address
Username	Optional. The MySQL user name
Password	Optional. The password to log in with
Dbname	Optional. The default database to be used when performing queries

Table - 2

Note: There are more available parameters, but the ones listed above are the most important.

In the following example we store the connection in a variable ($con) for later use in the script:

3.5.4. Create a Database

The CREATE DATABASE statement is used to create a database in MySQL.

We must add the CREATE DATABASE statement to the mysqli_query() function to execute the command.

3.5.5. Create a Table

The CREATE TABLE statement is used to create a table in MySQL.

We must add the CREATE TABLE statement to the mysqli_query () function to execute the command.

Note: When you create a field of type CHAR, you must specify the maximum length of the field, e.g. CHAR(50).

The data type specifies what type of data the column can hold.

3.5.6. Primary Keys and Auto Increment Fields

Each table in a database should have a primary key field.

A primary key is used to uniquely identify the rows in a table. Each primary key value must be unique within the table. Furthermore, the primary key field cannot be null because the database engine requires a value to locate the record.

The primary key field is often an ID number, and is often used with the AUTO_INCREMENT setting. AUTO_INCREMENT automatically increases the value of the field by 1 each time a new record is added. To ensure that the primary key field cannot be null, we must add the NOT NULL setting to the field:

3.6. Site Access Control[10]

One familiar use of authentication and authorization is access control. A computer system that is supposed to be used only by those authorized must attempt to detect and exclude the unauthorized. Access to it is therefore usually controlled by insisting on an authentication procedure to establish with some degree of confidence the identity of the user, granting privileges established for that identity.

Common examples of access control involving authentication include:

- Asking for photo ID when a contractor first arrives at a house to perform work.
- Using captcha as a means of asserting that a user is a human being and not a computer program.
- By using One Time Password (OTP), received on a tele-network enabled device like mobile phone, as an authentication password/PIN
- A computer program using a blind credential to authenticate to another program
- Entering a country with a passport
- Logging in to a computer
- Using a confirmation E-mail to verify ownership of an e-mail address
- Using an Internet banking system
- Withdrawing cash from an ATM

In some cases, ease of access is balanced against the strictness of access checks. For example, the credit card network does not require a personal identification number for authentication of the claimed identity; and a small transaction usually does not even require a signature of the authenticated person for proof of authorization of the transaction. The security of the system is maintained by limiting distribution of credit card numbers, and by the threat of punishment for fraud.

Security experts argue that it is impossible to prove the identity of a computer user with absolute certainty. It is only possible to apply one or more tests which, if passed, have been previously declared to be sufficient to proceed. The problem is to determine which tests are sufficient, and many such are inadequate. Any given test can be spoofed one way or another, with varying degrees of difficulty.

Computer security experts are now also recognizing that despite extensive efforts, as a business, research and network community, we still do not have a secure understanding of the requirements for authentication, in a range of circumstances

3.7. Site Authentication[11]

Authentication is the mechanism you use to verify the identity of visitors to your Web site or Web application. Typically, you do this by assigning a user name and password to a visitor or allowing a visitor to anonymously access public content on your site.

Although you use authentication to confirm the identity of a visitor, you use authorization to control the visitor's access to the different areas of your site or application.

Authentication is the act of confirming the truth of an attribute of a single piece of data or entity. In contrast with identification which refers to the act of stating or otherwise indicating a claim purportedly attesting to a person or thing's identity, authentication is the process of actually confirming that identity. It might involve confirming the identity of a person by validating their identity documents, verifying the validity of a Website with a digital certificate, tracing the age of an artifact by carbon dating, or ensuring that a product is what its packaging and labeling claim to be. In other words, authentication often involves verifying the validity of at least one form of identification.

3.7.1. Factors and identity

The ways in which someone may be authenticated fall into three categories, based on what are known as the factors of authentication: something the user knows, something the user has, and something the user is. Each authentication factor covers a range of elements used to authenticate or verify a person's identity prior to being granted access, approving a transaction request, signing a document or other work product, granting authority to others, and establishing a chain of authority.
Security research has determined that for a positive authentication, elements from at least two, and preferably all three, factors should be verified. The three factors (classes) and some of elements of each factor are:

- **the knowledge factors:** Something the user knows (e.g., a password, pass phrase, or personal identification number(PIN), challenge response (the user must answer a question), pattern)

- **the ownership factors:** Something the user has (e.g., wrist band, ID card, security token, cell phone with built-in hardware token, software token, or cell phone holding a software token)

- **the inherence factors:** Something the user is or does (e.g., fingerprint, retinal pattern, DNA sequence (there are assorted definitions of what is sufficient), signature, face, voice, unique bio-electric signals, or other biometric identifier).

3.8. Site Confidentiality and Integrity [12] [13]

Confidentiality, integrity, and availability (CIA) is a model designed to guide policies for information security within an organization. In this context, confidentiality is a set of rules that limits access to information, integrity is the assurance that the information is trustworthy and accurate, and availability is a guarantee of ready access to the information by authorized people. The model is sometimes known as the CIA triad.

Confidentiality prevents sensitive information from reaching the wrong people, while making sure that the right people can in fact get it. A good example is an account number or routing number when banking online. Data encryption is a common method of ensuring confidentiality. User IDs and passwords constitute a standard procedure; two-factor authentication is becoming the norm and biometric verification is an option as well. In addition, users can take precautions to minimize the number of places where the information appears, and the number of times it is actually transmitted to complete a required transaction.

Integrity involves maintaining the consistency, accuracy, and trustworthiness of data over its entire life cycle. Data must not be changed in transit, and steps must be taken to ensure that data cannot be altered by unauthorized people (for example, in a breach of confidentiality). In addition, some means must be in place to detect any changes in data that might occur as a result of non-human-caused events such as an electromagnetic pulse (EMP) or server crash. If an unexpected change occurs, a backup copy must be available to restore the affected data to its correct state.

Availability is best ensured by rigorously maintaining all hardware, performing hardware repairs immediately when needed, providing a certain measure of redundancy and failover, providing adequate communications bandwidth and preventing the occurrence of bottlenecks, implementing emergency backup power systems, keeping current with all necessary system upgrades, and guarding against malicious actions such as denial-of-service (DoS) attacks.

3.9. Security Techniques

3.9.1. SQL injection

SQL injection attacks are when an attacker uses a web form field or URL parameter to gain access to or manipulate your database. When you use standard Transact SQL it is easy to unknowingly insert rogue code into your query that could be used to change tables, get information and delete data. You can easily prevent this by always using parameterized queries, most web languages have this feature and it is easy to implement.

Consider this query:
"SELECT * FROM table WHERE column = '" + parameter + "';"
If an attacker changed the URL parameter to pass in ' or '1'='1 this will cause the query to look like this:
"SELECT * FROM table WHERE column = '' OR '1'='1';"
Since '1' is equal to '1' this will allow the attacker to add an additional query to the end of the SQL statement which will also be executed.

3.9.2. Error messages

Be careful with how much information you give away in your error messages. For example if you have a login form on your website you should think about the language you use to communicate failure when attempting logins. You should use generic messages like “Incorrect username or password” as not to specify when a user got half of the query right. If an attacker tries a brute force attack to get a username and password and the error message gives away when one of the fields are correct then the attacker knows he has one of the fields and can concentrate on the other field.

3.9.3. Passwords

Everyone knows they should use complex passwords, but that doesn’t mean they always do. It is crucial to use strong passwords to your server and website admin area, but equally also important to insist on good password practices for your users to protect the security of their accounts.

As much as users may not like it, enforcing password requirements such as a minimum of around eight characters, including an uppercase letter and number will help to protect their information in the long run. Passwords should always be stored as encrypted values, preferably using a one way hashing algorithm such as SHA. Using this method means when you are authenticating users you are only ever comparing encrypted values. For extra website security it is a good idea to salt the passwords, using a new salt per password. In the event of someone hacking in and stealing your passwords, using hashed passwords could help damage limitation, as decrypting them is not possible. The best someone can do is a dictionary attack or brute force attack, essentially guessing every combination until it finds a match. When using salted passwords the process of cracking a large number of passwords is even slower as every guess has to be hashed separately for every salt + password which is computationally very expensive.

3.9.4. SSL

SSL is a protocol used to provide security over the Internet. It is a good idea to use a security certificate whenever you are passing personal information between the website and web server or database. Attackers could sniff for this information and if the communication medium is not secure could capture it and use this information to gain access to user accounts and personal data.

SSL allows transferring data in an encrypted form. All information that a customer might want to keep private should be transmitted via SSL. Such information should definitely include credit card number and related information, and may, depending on the type of business, include customer's name, address, and the list of products that the customer is buying. It should also include the customer's password and order ID.

SSL connection (i.e. connection via https, instead of http) should be the default for transmitting customer's information. However, it is often not the default protocol on many commercial web sites. The reason why many businesses don't use SSL as default is because SSL connection is slower than a regular http connection (due to encryption and decryption). eBay has been criticized recently for transmitting passwords in the clear rather than encrypted

How reliable is SSL itself? In 1998 a Bell Labs researcher Daniel Bleichenbacher has discovered a problem with a version of SSL used at that time: it turned out that a long sequence of specifically designed messages allows figuring out the key used by a server in a session by studying error messages returned by the server. Since then SSL has been patched so that no information about the key can be gathered from error messages. The problem, however, was more theoretical than practical, because figuring out the key required around a million messages sent over a designated connection. SSL is considered a secure way of transmitting private information.

3.9.5. Update your web script constantly

Upgrade whenever there is a new version of your script available. Be sure to do it as soon as the upgrade is released, regardless if the upgrade contains new features of not. Even simple point upgrades will fix bugs in the script.

3.9.6. Secure your administrative email address.

Make sure that the admin email address that you use to login to your secure website is secure. This email address should be completely different from any addresses listed on your site's contact page. Keeping this email private will help prevent scammers from sending you phishing emails disguised as email from your host company.

3.9.7. Change your database table prefix

If your website uses a blog or forum script, you can change the default database table prefix. For example, a WordPress blog carries the table prefix "wp." If you change your table prefix, hackers will have a harder time getting data from your website.

3.9.8. Protect your database with a password

In most cases it is not required to assign a password, but having one can act as added security. Having a database password will not slow down the website at all.

3.9.9. Delete your installation folder

Once you have completed the installation, it is not necessary to have the installer folder on your computer. It is possible for a hacker to remotely get into your computer and run the installer again. Once they get in, they can empty your database and control your website and content. Another option is to rename the installation folder rather than delete it.

3.9.10. Restrict Root Access

Be it may FTP or Database, never give root access to everyone Willy nilly. Restrict access to certain non-system folders in the case of FTP uploads by people other than the system administrator.

3.9.11. Ensure the presence of .htaccess file

.htaccess files are often used to specify the security restrictions for the particular directory, and make sure you have not deleted it by accident or if it is there in the first place.

3.9.12. Add robots.txt file

Robots.txt gives special instructions to search engine spiders as to which folders are to be indexed and which ones are not. Folders with documents, images etc. can be kept under wraps from being indexed and displayed in public web searches.

3.10. SHA-1 Security Implementation [14] [15]

In cryptography, SHA-1 is a cryptographic hash function designed by the United States National Security Agency and is a U.S. Federal Information Processing Standard published by the United States NIST. SHA-1 produces a 160-bit (20-byte) hash value. A SHA-1 hash value is typically rendered as a hexadecimal number, 40 digits long. SHA stands for "secure hash algorithm". The four SHA algorithms are structured differently and are named SHA-0, SHA-1, SHA-2, and SHA-3. SHA-0 is the original version of the 160-bit hash function published in 1993 under the name "SHA": it was not adopted by many applications. Published in 1995, SHA-1 is very similar to SHA-0, but alters the original SHA hash specification to correct alleged weaknesses. SHA-2, published in 2001, is significantly different from the SHA-1 hash function. SHA-1 is the most widely used of the existing SHA hash functions, and is employed in several widely used applications and protocols. In 2005, cryptanalysts found attacks on SHA-1 suggesting that the algorithm might not be secure enough for ongoing use. NIST required many applications in federal agencies to move to SHA-2 after 2010 because of the weakness. Although no successful attacks have yet been reported on SHA-2, it is algorithmically similar to SHA-1. In 2012, following a long-running competition, NIST selected an additional algorithm, Keccak, for standardization under SHA-3. In November 2013 Microsoft announced their deprecation policy on SHA-1 according to which Windows will stop accepting SHA-1 certificates in SSL by 2017. In September 2014 Google announced their deprecation policy on SHA-1 according to which Chrome will stop accepting SHA-1 certificates in SSL in a phased way by 2017. Mozilla is also planning to stop accepting SHA-1-based SSL certificates by 2017.

3.10.1. The SHA-1 hash function

SHA-1 produces a message digest based on principles similar to those used by Ronald L. Rivest of MIT in the design of the MD4 and MD5 message digest algorithms, but has a more conservative design.
The original specification of the algorithm was published in 1993 under the title Secure Hash Standard, FIPS PUB 180, by U.S. government standards agency NIST (National Institute of Standards and Technology). This version is now often named SHA-0. It was withdrawn by the NSA shortly after publication and was superseded by the revised version, published in 1995 in FIPS PUB 180-1 and commonly designatedSHA-1. SHA-1 differs from SHA-0 only by a single bitwise rotation in the message schedule of its compression function; this was done, according to the NSA, to correct a flaw in the original algorithm which reduced its cryptographic security. However, the NSA did not provide any further explanation or identify the flaw that was corrected. Weaknesses have subsequently been reported in both SHA-0 and SHA-1. SHA-1 appears to provide greater resistance to attacks, supporting the NSA’s assertion that the change increased the security.

3.10.2. Example hashes

These are examples of SHA-1 message digests in hexadecimal and in Base64 binary to ASCII text encoding.

SHA1 ("The quick brown fox jumps over the lazy dog") gives hexadecimal: 2fd4e1c67a2d28fced849ee1bb76e7391b93eb12 gives Base64 binary to ASCII text encoding: L9ThxnotKPzthJ7hu3bnORuT6xI=

Even a small change in the message will, with overwhelming probability, result in a completely different hash due to the avalanche effect. For example, changing dog to cog produces a hash with different values for 81 of the 160 bits:

SHA1 ("The quick brown fox jumps over the lazy cog") gives hexadecimal: de9f2c7fd25e1b3afad3e85a0bd17d9b100db4b3 gives Base64 binary to ASCII text encoding: 3p8sf9JeGzr60+haC9F9mxANtLM=

The hash of the zero-length string is: SHA1 ("") gives hexadecimal: da39a3ee5e6b4b0d3255bfef95601890afd80709 gives Base64 binary to ASCII text encoding: 2jmj7l5rSw0yVb/vlWAYkK/YBwk=

3.10.3. SHA-1 algorithm

All of the information above could easily be found elsewhere on the internet in a much more thorough and accurate way. In fact, I have intentionally left out many details because there weren't relevant to my main cause. The reason I decided to write this was because I was absolutely fascinated by how these functions actually work, but was completely unable to find a working example online. Wikipedia not only has a great article, but also very wonderful 'pseudocode'. Pseudocode is like the blueprints without any of the details. What I was unable to find was anything that would actually show you step by step exactly how a word gets hashed. It is my plan to walk you through from start to finish exactly what happens inside a hash function

Step 0: Initialize some variables: - There are five variables that now need to be initialized.

- h0 = 01100111010001010010001100000001
- h1 = 11101111110011011010101110001001
- h2 = 10011000101110101101110011111110
- h3 = 00010000001100100101010001110110
- h4 = 11000011110100101110000111110000

Step 1: Pick a string: - In this example I am going to use the string: 'A Test'.

Step 2: Break it into characters: - You must now break the string into characters.

- A
-
- T
- e
- s
- t

Note that spaces count as characters.

Step 3: Convert characters to ASCII codes

Each character should now be converted from text into ASCII. ASCII or the 'American Standard Code for Information Interchange' is a standard that allows computer to communicate different symbols by assigning each one a number. No matter what font or language you use, the same character will always have the same ASCII code.

- 65
- 32
- 84
- 101
- 115
- 116

Note that 'T' and 't' are different ASCII characters.

Step 4: Convert numbers into binary

Binary is simply base two. All base ten numbers are now converted into 8-bit binary numbers. The eight-bit part means that if they don't actually take up a full eight place values, simply append zeros to the beginning so that they do.

- 01000001
- 00100000
- 01010100
- 01100101
- 01110011
- 01110100

Step 5: Add '1' to the end

Put the numbers together: 010000010010000001010100011001010111001101110100

Add the number '1' to the end:

0100000100100000010101000110010101110011011101001

Step 6: Append '0's' to the end

In this step you add zeros to the end until the length of the message is congruent to 448 mod 512. That means that after dividing the message length by 512, the remainder will be 448. In this case, the length of the original message in binary is 48 + 1 from the last step. That means that we will need to add a total of 399 zero's to the end. 0100000100100000010101000110010101110011011101001000000000000000000000-
0000000000000000000000000000000000000000000000000000000000000000000000-

0000000000000000000000000000000000000000000000000000000000000000000000-
0000000000000000000000000000000000000000000000000000000000000000000000-
0000000000000000000000000000000000000000000000000000000000000000000000-
0000000000000000000000000000000000000000000000000000000000000000000000-
0000000000000000000000000000

If your original message were 56 characters long, it would be exactly 448 digits long once converted into binary. After adding the '1' to the end of that, the new message would be 449 characters long. If that were the case you would need to append 575 zero's to make the message congruent to 448%512.

If your original message were 64 characters long, it would be exactly 512 digits long once converted into binary. After adding the '1' to the end of that, the new message would be 513 characters long. If that were the case you would need to append 447 zero's to make the message congruent to 448%512.

Step 6.1: Append original message length

This is the last of the 'message padding' steps. You will now add the 64-bit representation of the original message length, in binary, to the end of the current message. In this case our original message was 48 characters long.
48 in binary is expressed as: 110000. However, since the number must be 64-bits or digits long we must now add 58 zero's to the beginning of that number prior to adding it to the current message.

0100000100100000010101000110010101110011011101001000000000000000000000-
0000000000000000000000000000000000000000000000000000000000000000000000-
0000000000000000000000000000000000000000000000000000000000000000000000-
0000000000000000000000000000000000000000000000000000000000000000000000-
0000000000000000000000000000000000000000000000000000000000000000000000-
0000000000000000000000000000000000000000000000000000000000000000000000-
0000000000000000000000000000000000000000000000000000000000000000000000-
0000000000000000110000

The message length should now be an exact multiple of 512.

Step 7: 'Chunk' the message

We will now break the message up into 512 bit chunks. In this case the message is only 512 bit's long, so there will be only one chunk that will look exactly the same as the last step. 010000010010000001010100011001010111001101110100100000000000000000-
0000000000000000000000000000000000000000000000000000000000000000000000-
0000000000000000000000000000000000000000000000000000000000000000000000-
0000000000000000000000000000000000000000000000000000000000000000000000-
0000000000000000000000000000000000000000000000000000000000000000000000-
0000000000000000000000000000000000000000000000000000000000000000000000-

0000000000000000000000000000000000000000000000000000000000000000000000-
00000000000000000000110000

Step 8: Break the 'Chunk' into 'Words'

Break each chunk up into sixteen 32-bit words

- 0: 01000001001000000101010001100101
- 1: 01110011011101001000000000000000
- 2: 00000000000000000000000000000000
- 3: 00000000000000000000000000000000
- 4: 00000000000000000000000000000000
- 5: 00000000000000000000000000000000
- 6: 00000000000000000000000000000000
- 7: 00000000000000000000000000000000
- 8: 00000000000000000000000000000000
- 9: 00000000000000000000000000000000
- 10: 00000000000000000000000000000000
- 11: 00000000000000000000000000000000
- 12: 00000000000000000000000000000000
- 13: 00000000000000000000000000000000
- 14: 00000000000000000000000000000000
- 15: 00000000000000000000000000110000

Step 9: 'Extend' into 80 words

This is the first sub-step. Each chunk will be put through a little function that will create 80 words from the 16 current ones. This step is a loop. What that means is that every step after this will be repeated until a certain condition is true. In this case we will start by setting the variable 'i' equal to 16. After each run through of the loop we will add 1 to 'i' until 'i' is equal to 79.

Step 9.1: XOR

We begin by selecting four of the current words. The ones we want are: [i-3], [i-8], [i-14] and [i-16]. That means for the first time through the loop we want the words numbered: 13, 8, 2 and 0.

- 0: 01000001001000000101010001100101
- 2: 00000000000000000000000000000000
- 8: 00000000000000000000000000000000
- 13: 00000000000000000000000000000000

However, the next time through the loop we want words: 14, 9, 3 and 1. After the fifth time through the loop we will want words: 17, 12, 6 and 4. Note that word 17 doesn't exist yet, but it will after the first run of the loop. That means that after sixteen passes through the loop we will be using entirely words that aren't part of the original sixteen.

Now that we have our words selected we will start by performing what's known as an 'XOR' or 'Exclusive OR' on them. In the end all four words will be XOR'ed together, but you can think of it as first doing [i-3]XOR[i-8] then XOR'ing that by [i-14] and that again by [i-16]. XOR is one of a few simple logical operators. All it means is that you compare the two numbers bit by bit and if exactly one of them has the value '1', output a '1'. However, if both numbers have a '0' for that bit, or they both have a '1', output a '0'. This works very similar to a logical 'OR' which will be used later. The only difference is that an 'OR' will return a '1' so long as either column has a '1' even if both of them do. So for our example we begin by XOR'ing:

13: 00000000000000000000000000000000

8: 00000000000000000000000000000000

13 XOR 8

Out: 00000000000000000000000000000000

Now we XOR that with word number

[i-14]: 00000000000000000000000000000000

2: 00000000000000000000000000000000

(13 XOR 8) XOR 2

Out: 00000000000000000000000000000000

Now we XOR that with word number [i-16]: 00000000000000000000000000000000

0: 01000001001000000101010001100101

((13 XOR 8) XOR 2) XOR 0

Out: 01000001001000000101010001100101

Step 9.2: Left rotate

Perform a left bit rotation by a factor of one. This is very simple. All you do is remove the first digit on the left and add a '0' to the end. This effectively shifts the number over to the left by one. Here's how it looks:

Output from the last step: 01000001001000000101010001100101

Left Rotate 1: 10000010010000001010100011001010

Once you are done with that, you can store the variable as a new word. In this case it will be word number 16(keep in mind that we start counting at 0).

After step nine is complete we will now have 80 words that look like this:

- 0: 01000001001000000101010001100101
- 1: 01110011011101001000000000000000
- 2: 00000000000000000000000000000000
- 3: 00000000000000000000000000000000
- 4: 00000000000000000000000000000000
- 5: 00000000000000000000000000000000
- 6: 00000000000000000000000000000000
- 7: 00000000000000000000000000000000
- 8: 00000000000000000000000000000000
- 9: 00000000000000000000000000000000
- 10: 00000000000000000000000000000000
- 11: 00000000000000000000000000000000
- 12: 00000000000000000000000000000000
- 13: 00000000000000000000000000000000
- 14: 00000000000000000000000000000000
- 15: 00000000000000000000000000110000
- 16: 10000010010000001010100011001010
- 17: 11100110111010010000000000000000
- 18: 00000000000000000000000001100000
- 19: 00000100100000010101000110010101
- 20: 11001101110100100000000000000001
- 21: 00000000000000000000000011000000
- 22: 00001001000000101010001100101010
- 23: 10011011101001000000000001100011
- 24: 00000100100000010101000000010101
- 25: 11011111110101110100011001010101
- 26: 00110111010010000000000000000111
- 27: 00000000000000000000001100000000
- 28: 00100100000010101000110010101000
- 29: 01101110100100000000000111101110
- 30: 00010110100001000001000111000001
- 31: 10110010100011110001100111110110
- 32: 11010000101000111111001010100011
- 33: 01010110011101100000110000000010
- 34: 10010000001010100011001100100000
- 35: 10101000010001010100000111101101
- 36: 01101101010110000100011100000011
- 37: 11001010001111000110010011011010
- 38: 01100110100001010100011000100111
- 39: 00110111010010000011000110000111

- 40: 01010010101011011000110011010110
- 41: 110111100100100000001111011100001
- 42: 01101000010000010001110000010001
- 43: 00101000111100011001111110101011
- 44: 00000011001111011000100100010111
- 45: 11111100110001001100000110100110
- 46: 00010000101001100111010111011101
- 47: 10100001000110010101101011001001
- 48: 11011101110000010001100111100111
- 49: 01100001101110100100110110100110
- 50: 01101000010101000110010111110110
- 51: 00101110100100110100011011110111
- 52: 11010010101101011000101100101010
- 53: 11010011110010011110001000011010
- 54: 00010100001110111111001110110110
- 55: 00110101010110011111110100001011
- 56: 01101001110010001101011001110100
- 57: 00000110011100000111111010110101
- 58: 01101100111000100001101111110110
- 59: 00100110110111011001110100011101
- 60: 10001110101111000001001010101011
- 61: 11000101111011001100010100000111
- 62: 11111111000000100000010100100011
- 63: 11110110100011011111000110011110
- 64: 00110011011000101101111011000100
- 65: 01101100101101101110000110001111
- 66: 01000001000111000000100101101000
- 67: 11010001110010111100111001101001
- 68: 01001001000010010001011101110000
- 69: 11000100110000011010011011111100
- 70: 10100110011101011101110100010000
- 71: 00011001010110101100101010100001
- 72: 11100101000100110110101101110101
- 73: 11010100110111011011111001101111
- 74: 01110100001100011001010100101001
- 75: 10101111110100111111101000001101
- 76: 11011000110101010111110100101111
- 77: 00000111001000100000111010011001
- 78: 10001011100011011111100111110101
- 79: 10110111011010010100111100111110

Step 10: Initialize some variables

Set the letters A-->E equal to the variables h0-->h4.

- A = h0
- B = h1
- C = h2
- D = h3
- E = h4

Step 11: The main loop

This loop will be run once for each word in succession.

Step 11.1: Four choices

Depending on what number word is being input, one of four functions will be run on it.

Words 0-19 go to function 1.

Words 20-39 go to function 2

Words 40-59 go to function 3

Words 60-79 go to function 4

Function 1

Remember that 'OR' function I mentioned earlier, we're going to be using that and a logical 'AND' for this function. Just to refresh: A logical 'OR' will output a '1' if either or both of the inputs are '1'. A logical 'AND' will output a '1' if the first input and the other is a '1'. For all logical operations a '0' will be output if the conditions are not met. The only other logical operator we will be using is called a 'NOT'. A logical not only takes one input and outputs the opposite. If you put in a '1' you will get a '0', if you put in a '0' you will get a '1'. A logical 'NOT' is often represented as an exclamation point(!).

The first step of function 1 is to set the variable 'f' equal to: (B AND C) or (!B AND D)
B: 11101111110011011010101110001001

C: 10011000101110101101110011111110

B AND C

Out: 10001000100010001000100010001000

!B: 00010000001100100101010001110110

D: 00010000001100100101010001110110

!B AND D

Out: 00010000001100100101010001110110

B AND C: 10001000100010001000100010001000

!B AND D: 00010000001100100101010001110110

(B AND C) OR (!B AND D)

F: 10011000101110101101110011111110

The second step of function 1 is to set the variable 'k' equal to: 01011010100000100111100110011001

Function 2 For this function we will be using the 'XOR' operation exclusively. Just to refresh: A logical 'XOR' will output a '1' if either the first or the second of the inputs are '1' but not both. The first step of function 2 is to set the variable 'f' equal to: B XOR C XOR D Of course by this time our variables have changed. Here's what this step will actually look like by the time we get to it:

B: 11011100100010001001111001110101

C: 10110101000101000100100011110000

B XOR C

Out: 01101001100111001101011010000101

B XOR C: 01101001100111001101011010000101

D: 01001110101011001011101010110111

(B XOR C) XOR D

F: 00100111001100000110110000110010

The second step of function 2 is to set the variable 'k' equal to: 01101110110110011110101110100001

Function 3 For this function we will be using the 'AND' and 'OR' operations. The first step of function 3 is to set the variable 'f' equal to: (B AND C) OR (B AND D) OR (C AND D) By this time our variables have changed again. Here's what this step will actually look like by the time we get to it:

B: 01000100110000000111111001110111

C: 00011010100110110011101010111011

B AND C

Out: 00000000100000000011101000110011

B: 01000100110000000111111001110111

D: 01010011011001010110101011100100

B AND D

Out: 01000000010000000110101001100100

C: 00011010100110110011101010111011

D: 01010011011001010110101011100100

C AND D

Out: 00010010000000010010101010100000

B AND C: 00000000100000000011101000110011

B AND D: 01000000010000000110101001100100

(B AND C) OR (B AND D)

Out: 01000000110000000111101001110111

(B AND C) OR (B AND D): 01000000110000000111101001110111

C AND D: 00010010000000010010101010100000

((B AND C) OR (B AND D)) OR (C AND D)

F: 01010010110000010111101011110111

The second step of function 3 is to set the variable 'k' equal to: 10001111000110111011110011011100

Function 4 Function 4 is exactly the same as function 2 except that we will set 'k' equal to 11001010011000101100000111010110.

Step 11.2: Put them together

After completing one of the four functions above, each variable will move on to this step before restarting the loop with the next word. For this step we are going to create a new

variable called 'temp' and set it equal to: (A left rotate 5) + F + E + K + (the current word).
Notice that other than the left rotate the only operation we're doing is basic addition. Addition in binary is about as simple as it can be.
We'll use the results from the last word(79) as an example for this step.

A lrot 5:   00110001000100010000101101110100

F: 10001011110000011101111100100001

A lrot 5 + F

Out: 110111100110100101110101010010101

Notice that the result of this operation is one bit longer than the two inputs. This is just like adding 5 and 6, you will need a new place value to represent the answer. For everything to work out properly we will need to truncate that extra bit eventually. However, we do not want to do that until the end!
A lrot 5 + F: 110111100110100101110101010010101

E: 11101001001001111110100110101011

A lrot 5 + F + E

Out: 1010100101111110101101010001000000

A lrot 5 + F + E: 1010100101111110101101010001000000

K: 11001010011000101100000111010110

A lrot 5 + F + E + K

Out: 1110111000001011101100101100001 0110

A lrot 5 + F + E + K: 11101110000010111011001011000010110

Word 79:   10110111011010010100111100111110

A lrot 5 + F + E

Out: 100000100111110001101110010101010100

Now we need to truncate the result so that the next operations will work smoothly. We will remove as much of the beginning(left) until the number is 32 bits or 'digits' long.

32-bit temp: 00100111110001101110010101010100

The only thing left to do at this point is 're-set' some variables then start the loop over. We will be setting the following variables as such:

- E = D
- D = C
- C = B Left Rotate 30
- B = A
- A = temp

Step 12: The end

Once the main loop has finished there is very little left to do. All that's left is to set:

- h0 = h0 + A
- h1 = h1 + B
- h2 = h2 + C
- h3 = h3 + D
- h4 = h4 + E

If these variables are longer than 32 bits they should be truncated. For our example the 'h' variables will now have these values:

- h0 = 10001111000011000000100001010101
- h1 = 10010001010101100011001111100100
- h2 = 10100111110111100001100101000110
- h3 = 10001011001110000111010011001000
- h4 = 10010000000111011111000001000011

Finally the variables are converted into base 16 (hex) and joined together.

- 8f0c0855
- 915633e4
- a7de1946
- 8b3874c8
- 901df043

8f0c0855915633e4a7de19468b3874c8901df043

3.10.4. Comparison of Hash functions

In the table below, internal state means the “internal hash sum” after each compression of a data block.

Algorithm and variant		Output Size (bits)	Internal state size (bits)	Block size (bits)	Max message size (bits)	Rounds	Operations	Security (bits)	Example Performance (MiB/s)
MD5 (as reference)		128	128 (4×32)	512	$2^{64} – 1$	64	add mod 2^{32} , and, or, xor, rot	<64 (collisions found)	335
SHA-0		160	160 (5×32)	512	$2^{64} – 1$	80	add mod 2^{32} , and, or, xor, rot	<80 (collisions found)	-
SHA-1		***160***	***160 (5×32)***	***512***	***$2^{64} – 1$***	***80***	***add mod 2^{32} , and, or, xor, rot***	***<80 (theoretical attack in 2^{61})***	***192***
SHA-2	SHA-224 SHA-256	224 256	256 (8×32)	512	$2^{64} – 1$	64	add mod 2^{32} , and, or, xor, shr, rot	112 128	139
	SHA-384 SHA-512 SHA-512/224 SHA-512/256	384 512 224 256	512 (8×64)	1024	$2^{128} – 1$	80	add mod 2^{64} , and, or, xor, shr, rot	192 256 112 128	154

SHA-3	SHA3-224	224	1600 (5×5×64)	1152	∞	24	and, xor, not, rot	112	
	SHA3-256	256		1088				128	
	SHA3-384	384		832				192	
	SHA3-512	512		576				256	
	SHAKE128	d (arbitrary)		1344				min(d/2, 128)	
	SHAKE256	d (arbitrary)		1088				min(d/2, 256)	

Table – 3

3.11. __Digital Certificate__

On the surface Digital Certificates are not as complicated as they are occasionally perceived to be. A trusted public body, such as a Certificate Authority (CA) like GlobalSign, verifies a specified selection of evidence to produce an electronic identification for future presentation, indicating that the authentication of the individual or organization has taken place.

The Digital Certificate contains information about who the certificate was issued to, as well as the certifying authority that issued it. Additionally, some certifying authorities may themselves be certified by a hierarchy of one or more certifying authorities, and this information is also part of the certificate chain. When for example a Digital Certificate is used to sign documents and software, this information is stored with the signed item in a secure and verifiable format, so that it can be displayed to a user to establish a trust relationship.

As is usually the case, when examining a little deeper, mechanisms are not quite that simple. In the case of Digital Certificates there are a number of other factors. What are the qualifications of the third-party, their practices and which cryptographic algorithms did they use to produce the Digital Certificate?

From a CISOs perspective, using Digital Certificates such as SSL raises concerns that may impact on the organisation’s operational environment. By using a certificate from a Certificate Authority, the individual/organisation will need to fully trust the CA’s practices.

This is especially true when it comes to decisions relating to what cryptographic algorithms and key lengths are acceptable in this ever changing industry. Thankfully you do not need to be a cryptographer to make good decisions on this topic, but you will need to have a basic understanding of the history, advances promoted for future use, and carefully consider algorithms provided by a number of Certificate Authorities operating in the security market at present.

3.11.1. History

Digital signatures are composed of two different algorithms, the hashing algorithm (SHA-1 for example) and the other the signing algorithm (RSA for example). Over time these algorithms, or the parameters they use, need to be updated to improve security.

RSA's strength is directly related to the key size, the larger the key the stronger the signature. Advances in cryptanalysis have driven the increase in the key size used with this algorithm. While this requires some additional computing power, microprocessors have kept pace with the requirements and there is minimal impact to the entities creating or validating signatures. Each time we double the size of an RSA key, decryption operations require 6-7 times more processing power.

As a result of this, since January 2011, Certificate Authorities have aimed to comply with NIST (National Institute of Standards and Technology) recommendations, by ensuring all new RSA certificates have keys of 2048 bits in length or longer. GlobalSign was one of the first Certificate Authorities to implement 2048 bit key strength within its Root CA Certificates, back in 1998 and other Certification Authorities have since followed suit based on these new requirements.

As computational power increases the hashing algorithms start to become susceptible to hashing collisions. MD5 was used for a number of years until it was found to have a security flaw in 2004 which set the stage for SHA-1. Hash algorithms take a variable length input string and reduce it to a typically shorter and fixed length output (160 bits for SHA-1), the goal of which being to provide a unique identifier for that input. The important thing to understand is that hash algorithms can be susceptible to collisions and the advances in the cryptanalysis have made it more likely to create such a collision. The problem here is that there is no parameter to tweak, the only way to address this issue is to change what algorithm one uses to produce the hash. In this case the next evolutionary step is to one of the SHA-2 family of algorithms.

3.12. <u>*RSA and it's Algorithm*</u>

The growth of the Internet and electronic commerce has brought to the forefront the issue of privacy in electronic communication. Large volumes of personal and sensitive information are electronically transmitted and stored every day. What guarantees do one have that a message sent to another person is not intercepted and read without their knowledge or consent? Tools to ensure the privacy and confidentiality of paper-based communication have existed for a long time. Similar tools exist in the electronic communications arena.

Encryption is the standard method for making a communication private. Anyone wanting to send a private message to another user encrypts (enciphers) the message before transmitting it. Only the intended recipient knows how to correctly decrypt (decipher) the message. Anyone who was "eavesdropping" on the communication would only see the encrypted message. Because they would not know how to decrypt it successfully, the message would make no sense to them. As such, privacy can be ensured in electronic communication.

3.12.1. Definitions

The following definitions are provided in Advanced Concepts in Operating Systems:

- Plaintext(Cleartext) - The intelligible message which will be converted into an unintelligible (encrypted) message
- Ciphertext - A message in encrypted form
- Encryption - The process of converting a plaintext message into a ciphertext message
- Decryption - The process of converting a ciphertext message into a plaintext message
- Key - A parameter used in the encryption and decryption process.
- Cryptosystem - A system to encrypt and decrypt information
- Symmetric Cryptosystem - A cryptosystem that uses the same key to encrypt and decrypt information
- Asymmetric Cryptosystem - A cryptosystem that uses one key to encrypt and a different key to decrypt
- Cryptography - The use of cryptosystems to maintain the confidentiality of information
- Cryptanalysis - The study of breaking cryptosystems

3.12.2. Encryption and Public-Key Cryptosystems

Modern cryptosystems are typically classified as either public-key or private-key. Private-key encryption methods, such as the Data Encryption Standard (DES), use the same key to both encrypt and decrypt data. The key must be known only to the parties who are authorized to encrypt and decrypt a particular message. Public-key cryptosystems, on the other hand, use different keys to encrypt and decrypt data. The public-key is globally available. The private-key is kept confidential.

3.12.3. The Key Distribution Problem

Private-key systems suffer from the key distribution problem. In order for a secure communication to occur, the key must first be securely sent to the other party. An unsecure channel such as a data network cannot be used. Couriers or other secure means are typically used. Public-key systems do not suffer from this problem because of their use of two different keys. Messages are encrypted with a public key and decrypted with a private key. No keys need to be distributed for a secure communication to occur.

3.12.4. Public-Key Cryptosystems

A user wishing to exchange encrypted messages using a public-key cryptosystem would place their public encryption procedure, E, in a public file. The user's corresponding decryption procedure, D, is kept confidential. Rivest, Shamir, and Adleman provide four properties that the encryption and decryption procedures have:

1. Deciphering the enciphered form of a message M yields M. That is, D(E(M)) = M
2. E and D are easy to compute.
3. Publicly revealing E does not reveal an easy way to compute D. As such, only the user can decrypt messages which were encrypted with E. Likewise, only the user can compute D efficiently.
4. Deciphering a message M and then enciphering it results in M. That is, E(D(M)) = M

As Rivest, Shamir, and Adleman point out, if a procedure satisfying property (3) is used, it is extremely impractical for another user to try to decipher the message by trying all possible messages until they find one such that E(M) = C.
A function satisfying properties (1) - (3) is called a "trap-door one-way function". It is called "one-way" because it is easy to compute in one direction but not the other. It is called "trap-door" because the inverse functions are easy to compute once certain private, "trap-door" information is known.

3.12.5. Encryption and Decryption Process

Suppose user A wants to send a private message, M, to user B.

- User A gets User B's public key from some public source.
- User A encrypts message M using B's public key. This produces a ciphertext message, C
- Ciphertext message C is sent over some communication channel
- Upon receipt, user B decrypts message C using their private key. This results in the original message M.

3.12.6. Digital Signatures

Property (4) of public-key cryptosystems allows a user to digitally "sign" a message they send. This digital signature provides proof that the message originated from the designated sender. In order to be effective, digital signature need to be both *message*-dependent as well as *signer*-dependent. This would prevent electronic "cutting and pasting" as well as modification of the original message by the recipient.

Suppose user A wanted to send a "digitally-signed" message, M, to user B:

- User A applies their decryption procedure to M. This results in ciphertext C.
- User A applies the encryption procedure of user B to C. This results in message S.
- Ciphertext message S is sent over some communication channel
- Upon receipt, user B applies their decryption procedure to S. This results in ciphertext message C.
- User B applies user A's encryption procedure to message C. This results in the original message, M.

User B cannot alter the original message or use the signature with any other message. To do so would require user B to know how to decrypt a message using A's decryption procedure.

3.12.7. The RSA Algorithm

The Rivest-Shamir-Adleman (RSA) algorithm is one of the most popular and secures public-key encryption methods. The algorithm capitalizes on the fact that there is no efficient way to factor very large (100-200 digit) numbers.
Using an encryption key (e, n), the algorithm is as follows:

a. Represent the message as an integer between 0 and (n-1). Large messages can be broken up into a number of blocks. Each block would then be represented by an integer in the same range.
b. Encrypt the message by raising it to the eth power modulo n. The result is a ciphertext message C.
c. To decrypt ciphertext message C, raise it to another power d modulo n

The encryption key (e, n) is made public. The decryption key (d, n) is kept private by the user.

How to Determine Appropriate Values for e, d, and n

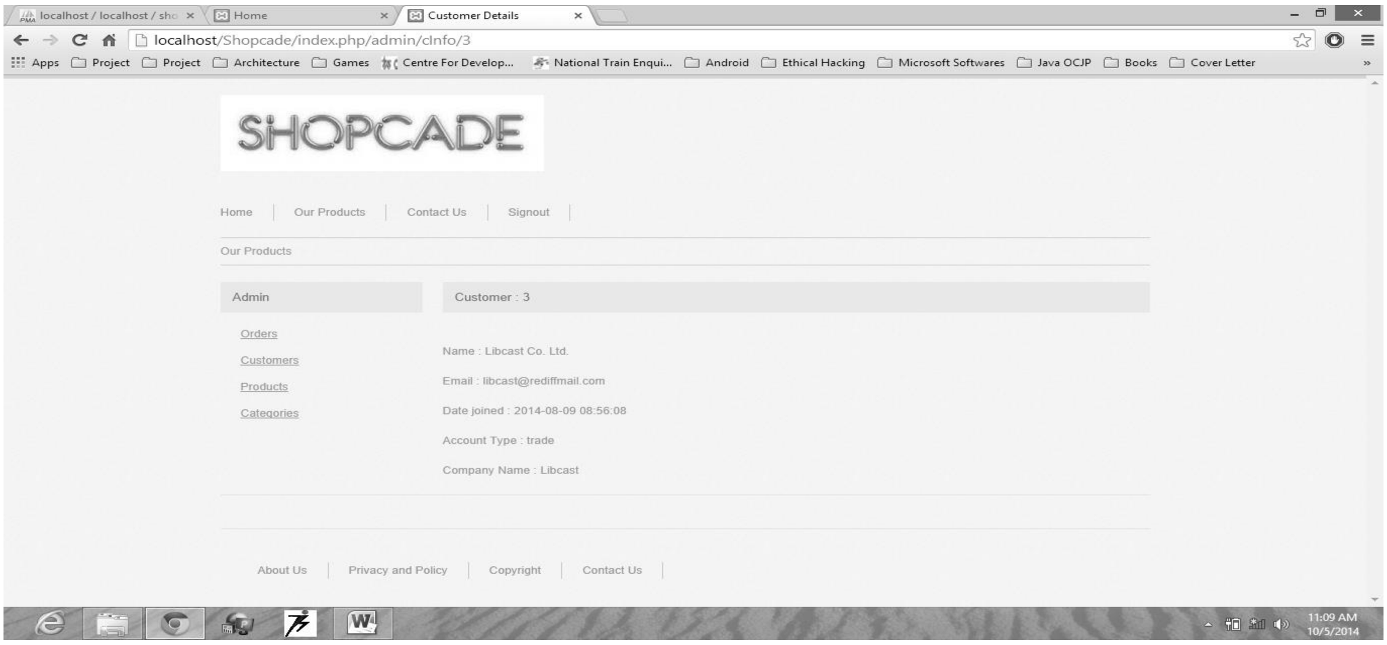
a) Choose two very large (100+ digit) prime numbers. Denote these numbers as p and q.
b) Set n equal to p * q.
c) Choose any large integer, d, such that GCD(d, ((p-1) * (q-1))) = 1
d) Find e such that e * d = 1 (**mod** ((p-1) * (q-1)))

Rivest, Shamir, and Adleman provide efficient algorithms for each required operation.

4. *Site Code and Functionality*

4.1. Admin Functionality

• ***View Customer Details*: -** An admin can View any Customer detail which has sign up to the site either he has placed any order or not.


localhost/Shopcade/index.php/admin/cInfo/3

Home | Our Products | Contact Us | Signout

Our Products

Admin: Orders, Customers, Products, Categories

Customer : 3

Name : Libcast Co. Ltd.
Email : libcast@rediffmail.com
Date joined : 2014-08-09 08:56:08
Account Type : trade
Company Name : Libcast

About Us | Privacy and Policy | Copyright | Contact Us

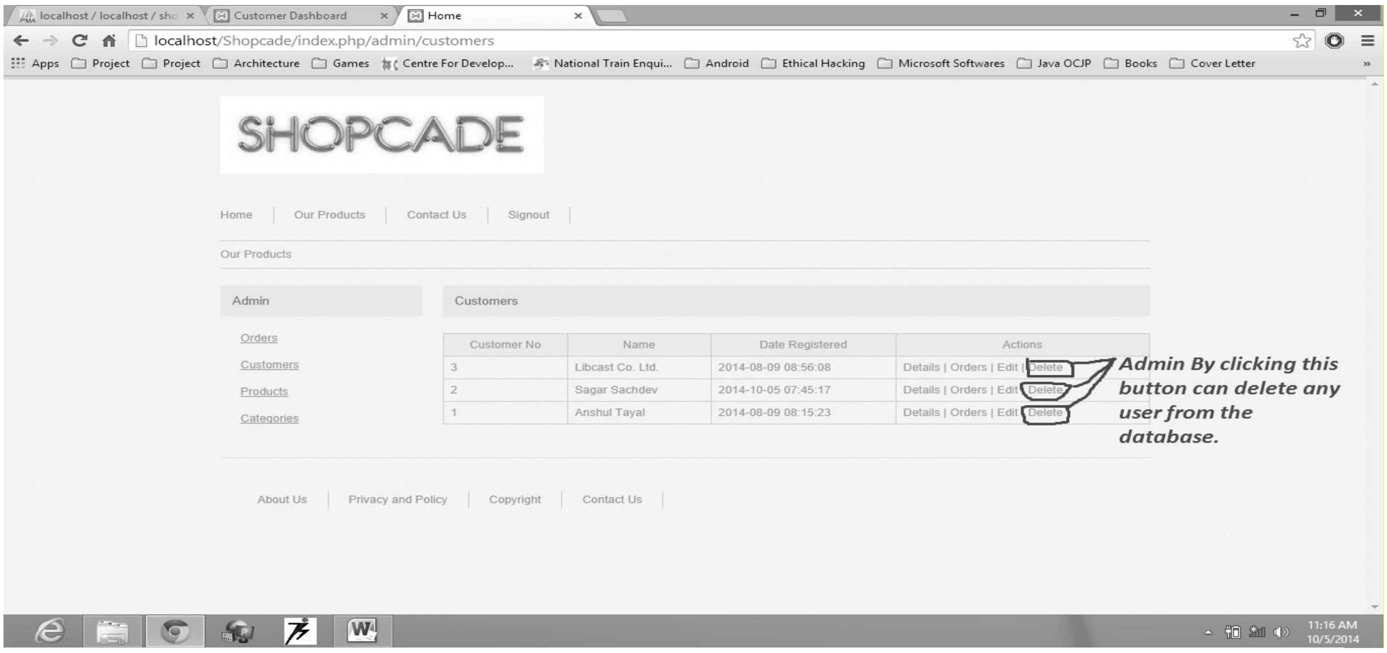
Fig 1: View Customer Details

HTML Code for Customer Details:-

```
<html>
<head>
<meta http-equiv=”Content-Type” content=”text/html; charset=utf-8” />
<title>Customer Details</title>
<link rel=”stylesheet” href=”<?php echo base_url();?>css/main.css” type=”text/css”>
</head>
<body>
<div id=”wrapper”>
<?php include(“header.php”); ?>
<div id=”content” class=”checkout”>
<div id=”breadcrumb”>
<a href=”index.php/admin/products”>Our Products</a>
</div>
<?php include(“left.php”); ?>
<div id=”right”>
<h1 class=”bar”>Customer : <?=$this->uri->segment(3);?></h1>
<p>
<br>Name          :  <?=$customer->full_name;?></br>
<br>Email         :  <?=$customer->email;?></br>
<br>Date joined   :  <?=$customer->date_joined;?></br>
<br>Account Type  :  <?=$customer->account_type;?></br>
```

```
<br>Company Name   :  <?=$customer->company_name;?></br>
</p>   </div>
<div class=”clear”></div>
<div id=”footer”>
<?php include(“footer.php”);?>
</div>         </div>         </div>         </body>       </html>
```

- ***Delete User***: - An admin can delete any user whom admin think has placed wrong details while ordering any order or while signing up.


localhost/Shopcade/index.php/admin/customers

SHOPCADE

Home | Our Products | Contact Us | Signout

Our Products

Admin: Orders, Customers, Products, Categories

Customers

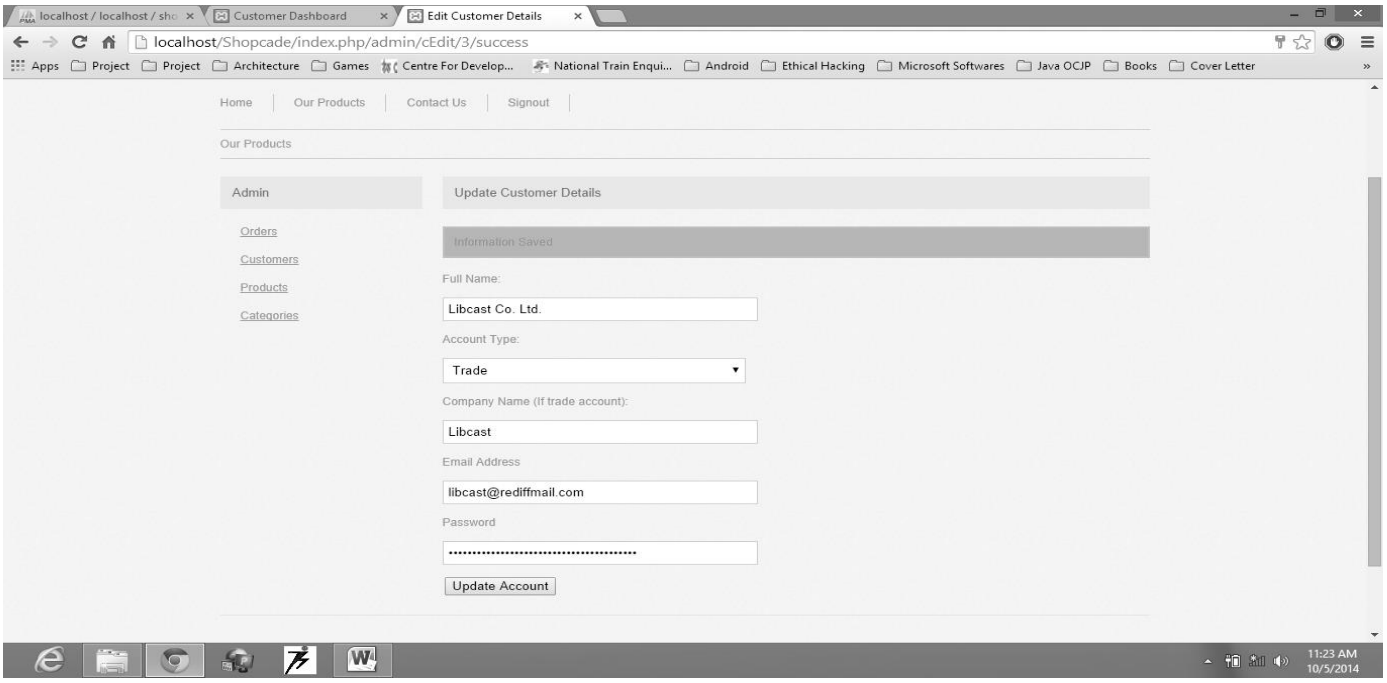
Customer No	Name	Date Registered	Actions
3	Libcast Co. Ltd.	2014-08-09 08:56:08	Details | Orders | Edit | Delete
2	Sagar Sachdev	2014-10-05 07:45:17	Details | Orders | Edit | Delete
1	Anshul Tayal	2014-08-09 08:15:23	Details | Orders | Edit | Delete

Admin By clicking this button can delete any user from the database.

About Us | Privacy and Policy | Copyright | Contact Us

Fig 2: Delete User

- ***Edit User Details***: - An admin can edit user details also.


localhost/Shopcade/index.php/admin/cEdit/3/success

Home | Our Products | Contact Us | Signout

Our Products

Admin: Orders, Customers, Products, Categories

Update Customer Details

Information Saved

Full Name: Libcast Co. Ltd.

Account Type: Trade

Company Name (If trade account): Libcast

Email Address: libcast@rediffmail.com

Password: ••

Update Account

Fig 3: Edit User Details

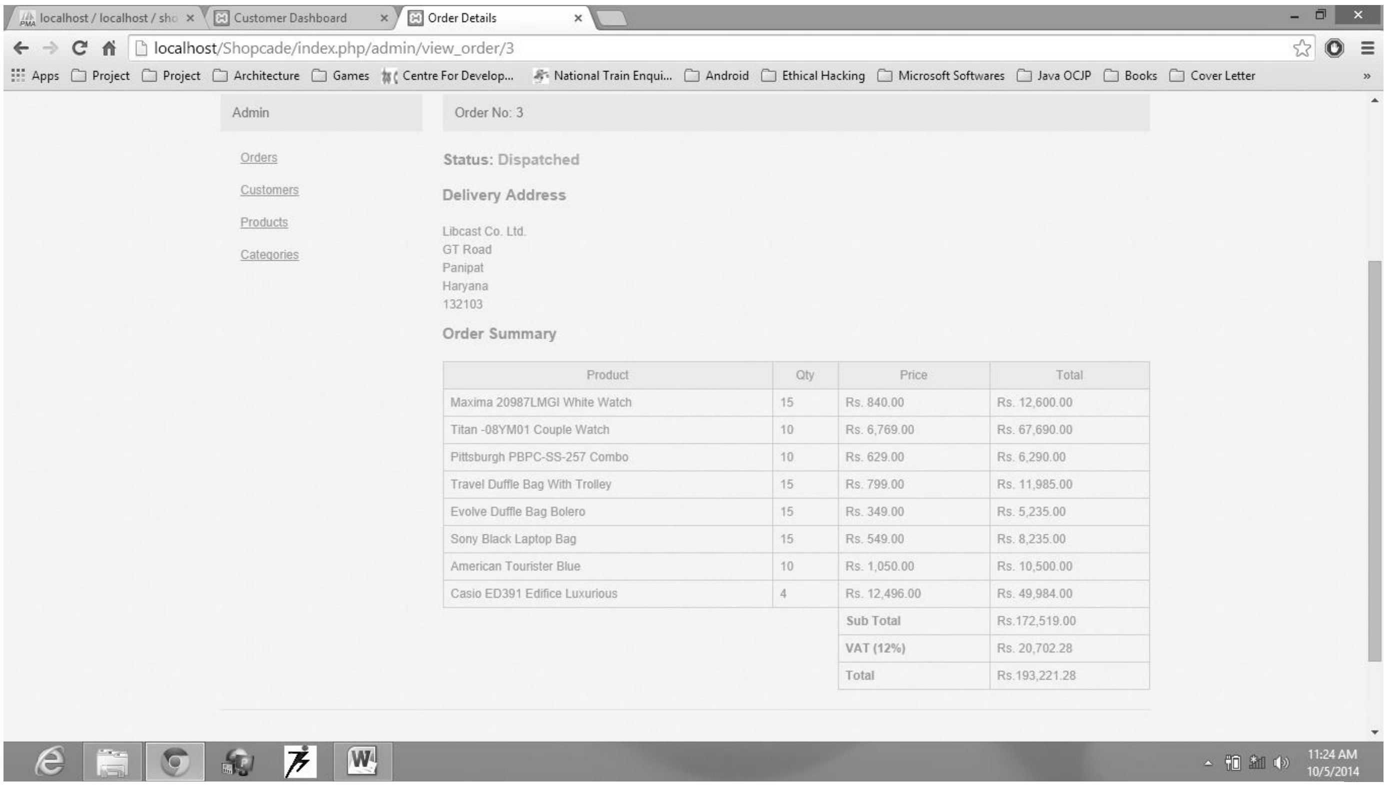
HTML Code for Edit Customer Details:-

```
<head>
<meta http-equiv=”Content-Type” content=”text/html; charset=utf-8” />
```

```
<title>Edit Customer Details</title>
<link rel=”stylesheet” href=”<?php echo base_url();?>css/main.css” type=”text/css”>
</head>
<body>
<div id=”wrapper”>
<?php include(“header.php”); ?>
<div id=”content” class=”checkout”>
<div id=”breadcrumb”>
<a href=”index.php/admin/products”>Our Products</a>
</div>
<?php include(“left.php”); ?>
<div id=”right”>
<h1 class=”bar”>Update Customer Details</h1>
<?php if(validation_errors()) { ?><div id=”errors”><?php echo
validation_errors(); ?></div> <?php } ?>
<?php if($this->uri->segment(4)==’success’) { ?><div id=”success”> Information
Saved</div> <?php } ?>
<form action=”<?=base_url();?>index.php/admin/cEdit/<?php echo $this->uri-
>segment(3);?>” method=”post” enctype=”multipart/form-data” id=”admin”>
<p>
<label>Full Name:</label>
<input name=”full_name” type=”text” id=”full_name” value=”<?php echo $customer-
>full_name;?>”>
</p>
<p>
<label>Account Type:</label>
<select name=”account_type” id=”account_type”>
<option value=””>Select Account Type</option>
<option value=”personal” <?php if($customer->account_type==’personal’) echo
‘selected=”selected”’;?>>Personal</option>
<option value=”trade” <?php if($customer->account_type==’trade’) echo
‘selected=”selected”’;?>>Trade</option>
</select>
</p>   <p>
<label>Company Name (If trade account):</label>
<input name=”company_name” type=”text” id=”company_name” value=”<?php echo
$customer->company_name;?>”>
</p>
<p>
<label>Email Address</label>
<input name=”email” type=”text” id=”email” value=”<?php echo $customer-
>email;?>”>
</p>
<p>
<label>Password</label>
<input name=”password” type=”password” id=”password” value=”<?php echo
$customer->user_pass;?>” >
```

```
</p>
<p>
<input name=”update_action” type=”hidden” id=”update_action” value=”true” />
<input name=”customer_id” type=”hidden” id=”customer_id” value=”<?php echo $this->uri->segment(3);?>” />
</p>
<input type=”submit” name=”submit” value=”Update Account” />
</form>
</div>
<div class=”clear”></div>
<div id=”footer”>
<?php include(“footer.php”);?>
</div>        </div>        </div>        </body>
```

- ***View Order Details***: - An admin can review any order placed by the customer on the site.



Order No: 3

Status: Dispatched

Delivery Address

Libcast Co. Ltd.
GT Road
Panipat
Haryana
132103

Order Summary

Product	Qty	Price	Total
Maxima 20987LMGI White Watch	15	Rs. 840.00	Rs. 12,600.00
Titan -08YM01 Couple Watch	10	Rs. 6,769.00	Rs. 67,690.00
Pittsburgh PBPC-SS-257 Combo	10	Rs. 629.00	Rs. 6,290.00
Travel Duffle Bag With Trolley	15	Rs. 799.00	Rs. 11,985.00
Evolve Duffle Bag Bolero	15	Rs. 349.00	Rs. 5,235.00
Sony Black Laptop Bag	15	Rs. 549.00	Rs. 8,235.00
American Tourister Blue	10	Rs. 1,050.00	Rs. 10,500.00
Casio ED391 Edifice Luxurious	4	Rs. 12,496.00	Rs. 49,984.00
		Sub Total	Rs.172,519.00
		VAT (12%)	Rs. 20,702.28
		Total	Rs.193,221.28

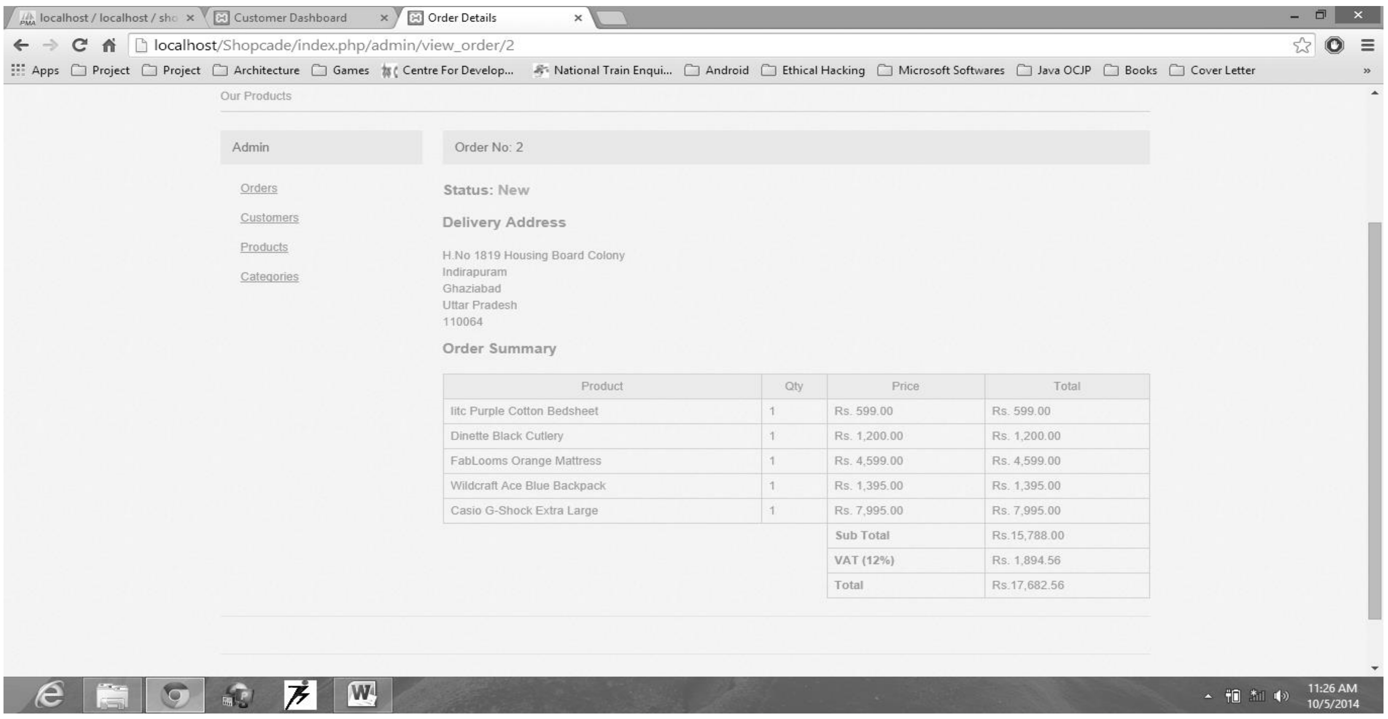
Fig 4: View Order Details

HTML Code for View Customer Order Details:-

```
<head>
<meta http-equiv=”Content-Type” content=”text/html; charset=utf-8” />
<title>Order Details</title>
<link rel=”stylesheet” href=”<?php echo base_url();?>css/main.css” type=”text/css”>
</head>
<body>
<div id=”wrapper”>
<?php include(“header.php”); ?>
```

```
<div id=”content” class=”checkout”>
<div id=”breadcrumb”>
<a href=”index.php/admin/products”>Our Products</a>
</div>
<?php include(“left.php”); ?>
<div id=”right”>
<h1 class=”bar”>Order No: <?=$this->uri->segment(3);?></h1>
<?php
if($order->order_status==’new’)
$color=’green’;
if($order->order_status==’cancelled’)
$color=’red’;
else
$color=’orange’;
?>
<h2>Status: <span style=”color:<?=$color;?>”> <?=ucwords($order-
>order_status);?></span></h2>
<h2>Delivery Address</h2>
<p><?=$address->address1;?><br /><?=$address->address2;?><br /><?=$address-
>city;?><br /><?=$address->county;?><br /><?=$address->post_code;?></p>
<br />
<h2>Order Summary</h2>
<?php
if($cart_products!=’empty’)
{
?>
<table id=”cart”>
<thead>
<th>Product</th>
<th class=”qty-column”>Qty</th>
<th>Price</th>
<th>Total</th>
</thead>
<tbody>
<?php
$total_price = 0;
foreach($cart_products as $product) {
$total_price += $product->item_total_price;
?>
<tr>
<td><?=$product->item_name;?></td>
<td><?=$product->item_quantity;?></td>
<td>Rs. <?=number_format($product->item_price,2);?></td>
<td>Rs. <?=number_format($product->item_total_price,2);?></td>
</tr>
<?php } ?>
<tr>
<td colspan=”2” class=”hidden”></td>
<td><strong>Sub Total</strong></td>
<td>Rs.<?=number_format($total_price,2);?></td>
<tr>
<tr>
<td colspan=”2” class=”hidden”></td>
<td><strong>VAT (12%)</strong></td>
<td>Rs. <?php $vat = (($total_price*(0.12))); echo number_format($vat,2);?></td>
<tr>
<tr>
<td colspan=”2” class=”hidden”></td>
<td><strong>Total</strong></td>
<td>Rs.<?=number_format($total_price+$vat,2);?></td>
<tr>
</tbody>
</table>
<?php  } ?>
</div>
<div class=”clear”></div>
<div id=”footer”>
<?php include(“footer.php”);?>
</div>        </div>        </div>        </body>
```

- ***Change Status of Order (New, Dispatched and Cancel)***: - An admin can change the status of order accordingly **New, Dispatched and Cancel**. Whenever an order is placed the default status is **New**.



Order No: 2

Status: New

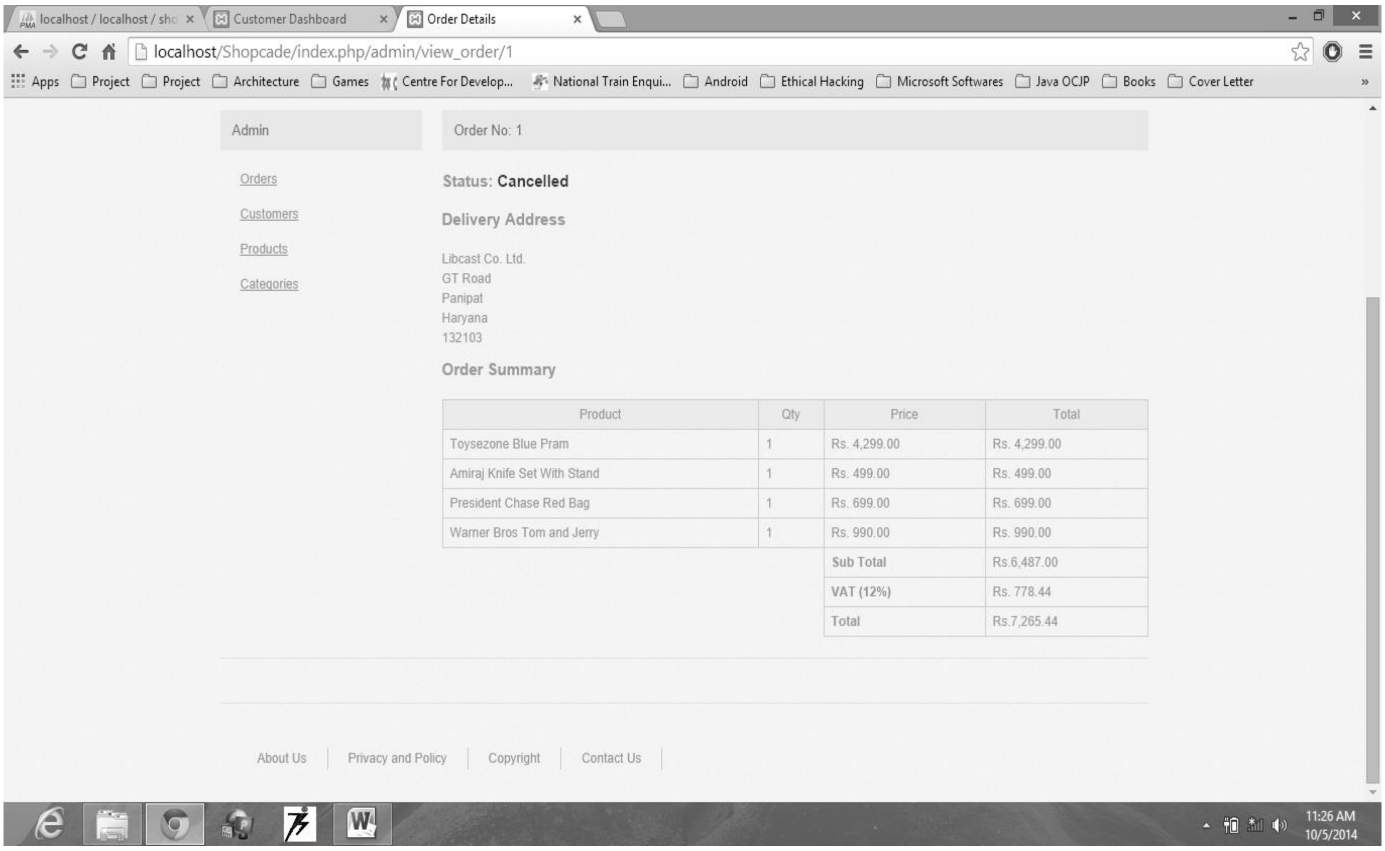
Delivery Address

H.No 1819 Housing Board Colony
Indirapuram
Ghaziabad
Uttar Pradesh
110064

Order Summary

Product	Qty	Price	Total
Iitc Purple Cotton Bedsheet	1	Rs. 599.00	Rs. 599.00
Dinette Black Cutlery	1	Rs. 1,200.00	Rs. 1,200.00
FabLooms Orange Mattress	1	Rs. 4,599.00	Rs. 4,599.00
Wildcraft Ace Blue Backpack	1	Rs. 1,395.00	Rs. 1,395.00
Casio G-Shock Extra Large	1	Rs. 7,995.00	Rs. 7,995.00
		Sub Total	Rs.15,788.00
		VAT (12%)	Rs. 1,894.56
		Total	Rs.17,682.56

Fig 5: Status is New



localhost/Shopcade/index.php/admin/view_order/1

Admin

- Orders
- Customers
- Products
- Categories

Order No: 1

Status: **Cancelled**

Delivery Address

Libcast Co. Ltd.
GT Road
Panipat
Haryana
132103

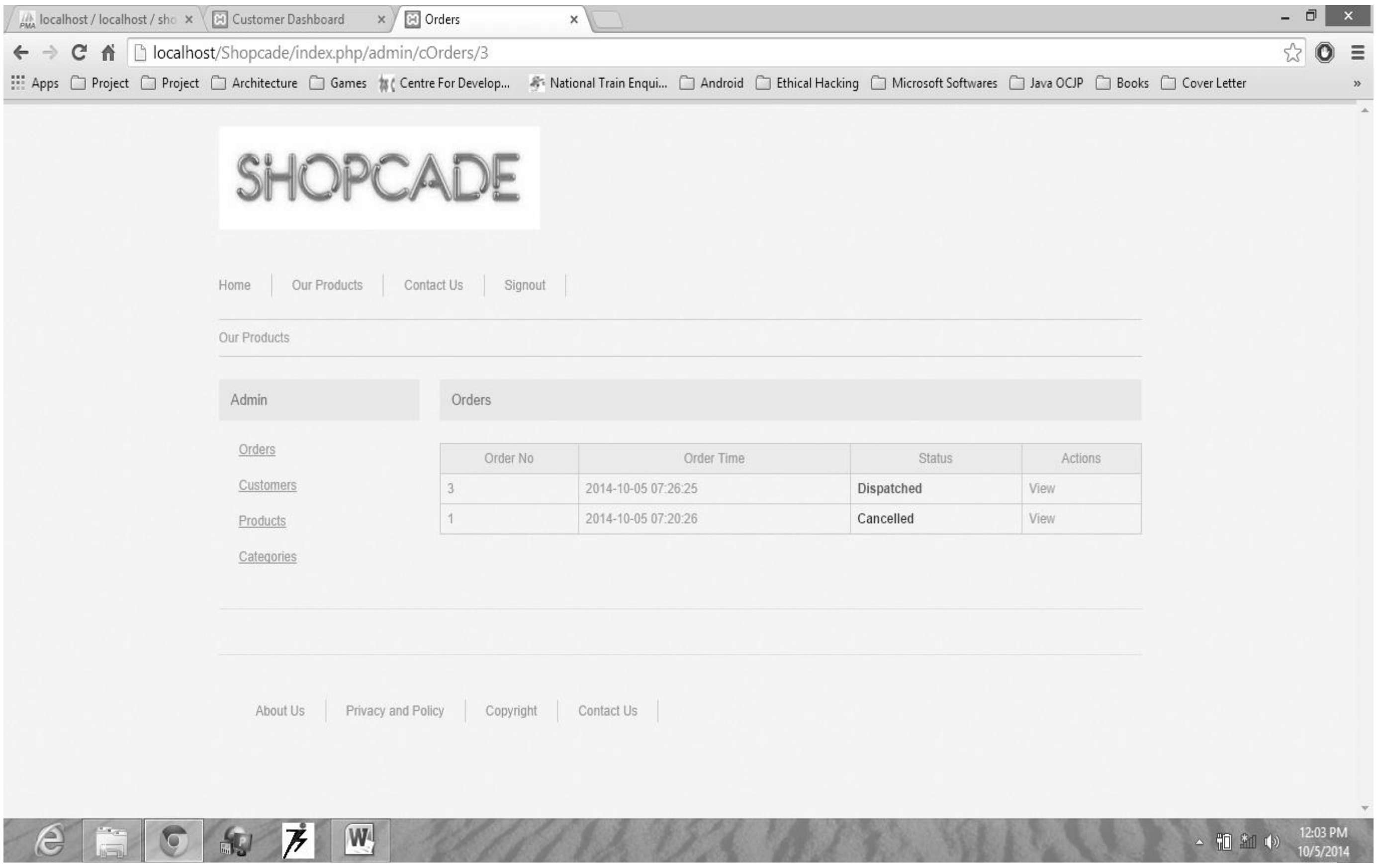
Order Summary

Product	Qty	Price	Total
Toysezone Blue Pram	1	Rs. 4,299.00	Rs. 4,299.00
Amiraj Knife Set With Stand	1	Rs. 499.00	Rs. 499.00
President Chase Red Bag	1	Rs. 699.00	Rs. 699.00
Warner Bros Tom and Jerry	1	Rs. 990.00	Rs. 990.00
		Sub Total	Rs.6,487.00
		VAT (12%)	Rs. 778.44
		Total	Rs.7,265.44

About Us | Privacy and Policy | Copyright | Contact Us

Fig 6: Status is canceled

- ***View Customer Order***: - An admin can view any customer full order details and their status as per that time.



localhost/Shopcade/index.php/admin/cOrders/3

Home | Our Products | Contact Us | Signout

Our Products

Admin

- Orders
- Customers
- Products
- Categories

Orders

Order No	Order Time	Status	Actions
3	2014-10-05 07:26:25	**Dispatched**	View
1	2014-10-05 07:20:26	**Cancelled**	View

About Us | Privacy and Policy | Copyright | Contact Us

Fig 7: View Customer Order

• ***Add New Category***: - An admin can enter any new category any time whenever company has started trading in that field.

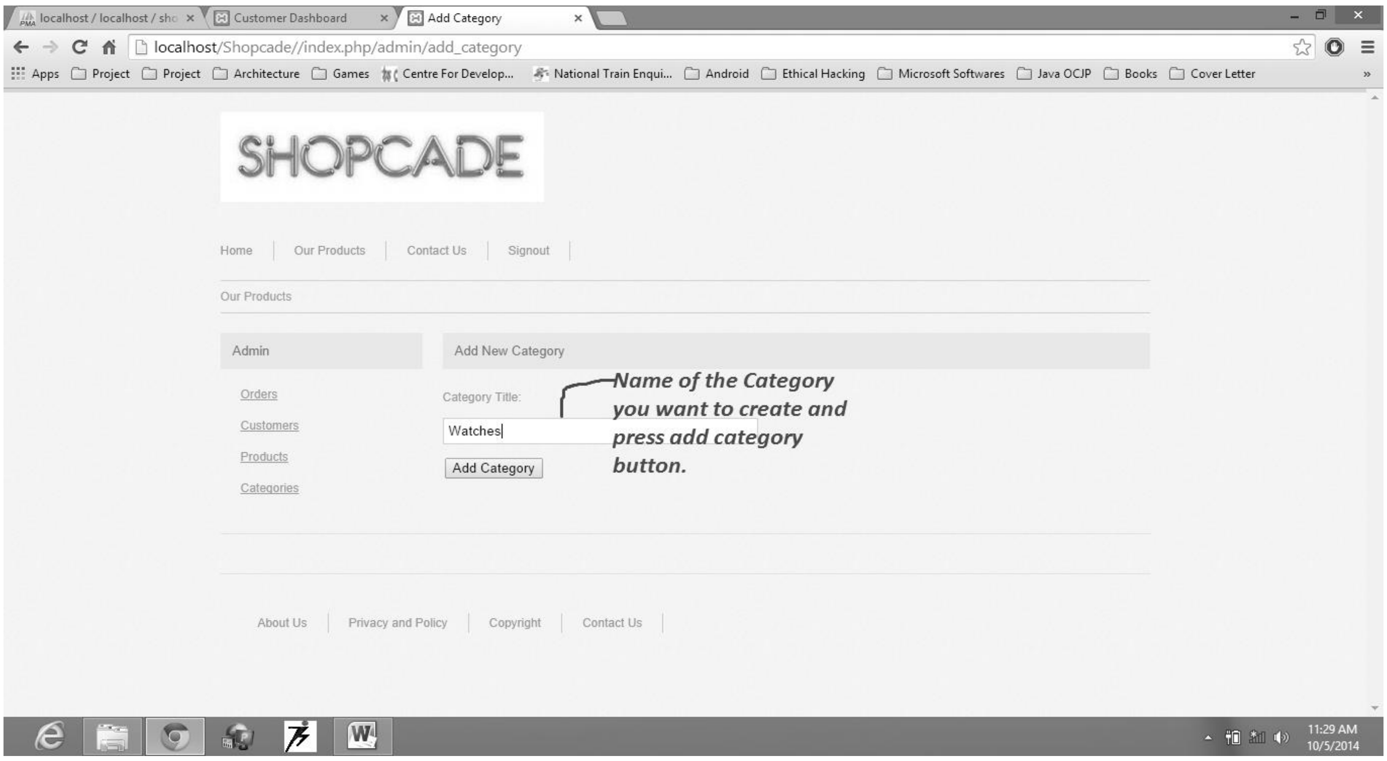


Fig 8: Add New Category

HTML Code for Add new Category:-

```
<head>
<meta http-equiv=”Content-Type” content=”text/html; charset=utf-8” />
<title>Add Category</title>
<link rel=”stylesheet” href=”<?php echo base_url();?>css/main.css” type=”text/css”>
</head>
<body>
<div id=”wrapper”>
<?php include(“header.php”); ?>
<div id=”content” class=”checkout”>
<div id=”breadcrumb”>
<a href=”index.php/admin/products”>Our Products</a>
</div>
<?php include(“left.php”); ?>
<div id=”right”>
<h1 class=”bar”>Add New Category</h1>
<?php if(validation_errors()) { ?><div id=”errors”><?php echo
validation_errors(); ?></div> <?php } ?>
<?php if($this->uri->segment(3)!=’’) { ?><div id=”errors”> Sorry – Category Already
Exists</div> <?php } ?>
<form method=”post” id=”admin” action=”<?php echo
base_url();?>index.php/admin/add_category”>
<p>    <label>Category Title:</label>
```

```
<input name=”name” type=”text” id=”name” value=”<?=set_value(‘name’);?>” />
</p>   <input type=”submit” name=”submit” value=”Add Category” />
<input type=”hidden” name=”action” value=”1” />
</form>       </div>          <div class=”clear”></div>
<div id=”footer”>
<?php include(“footer.php”);?>
</div>         </div>         </div>         </body>
```

• ***Edit Category Detail***: - An admin can also edit the category detail as per the requirement.

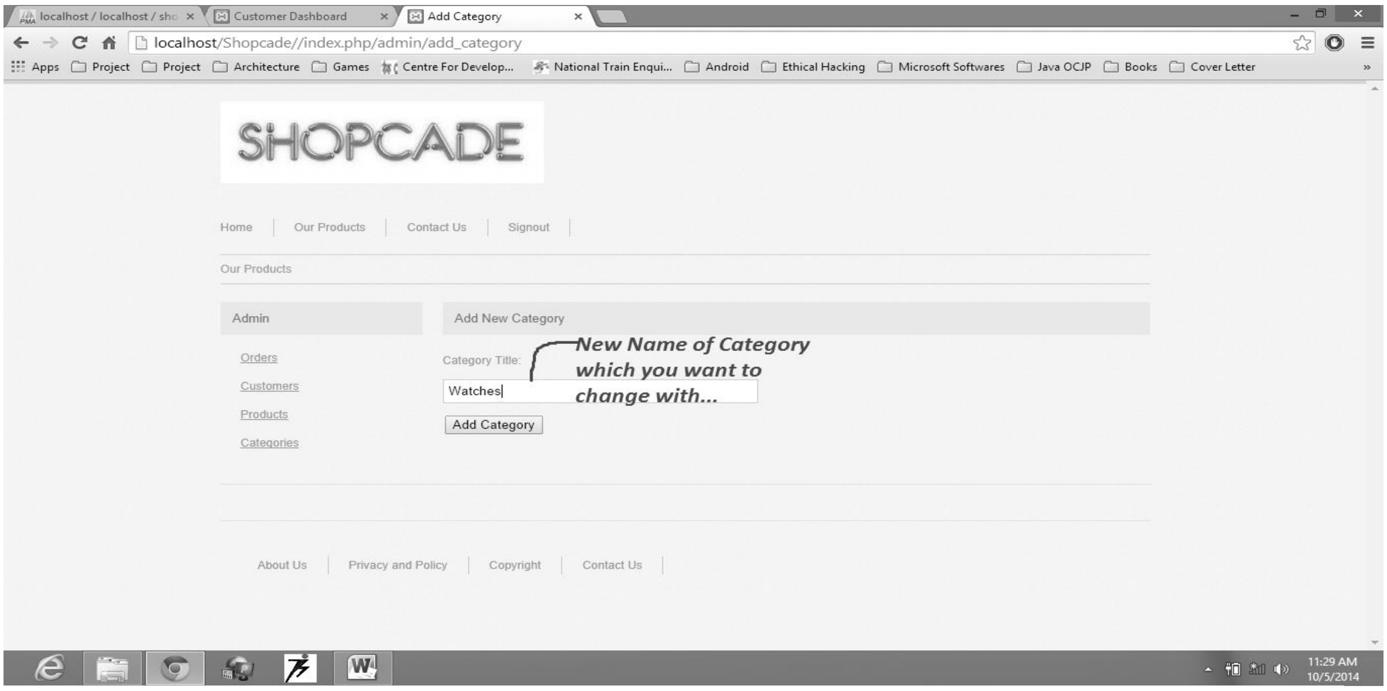
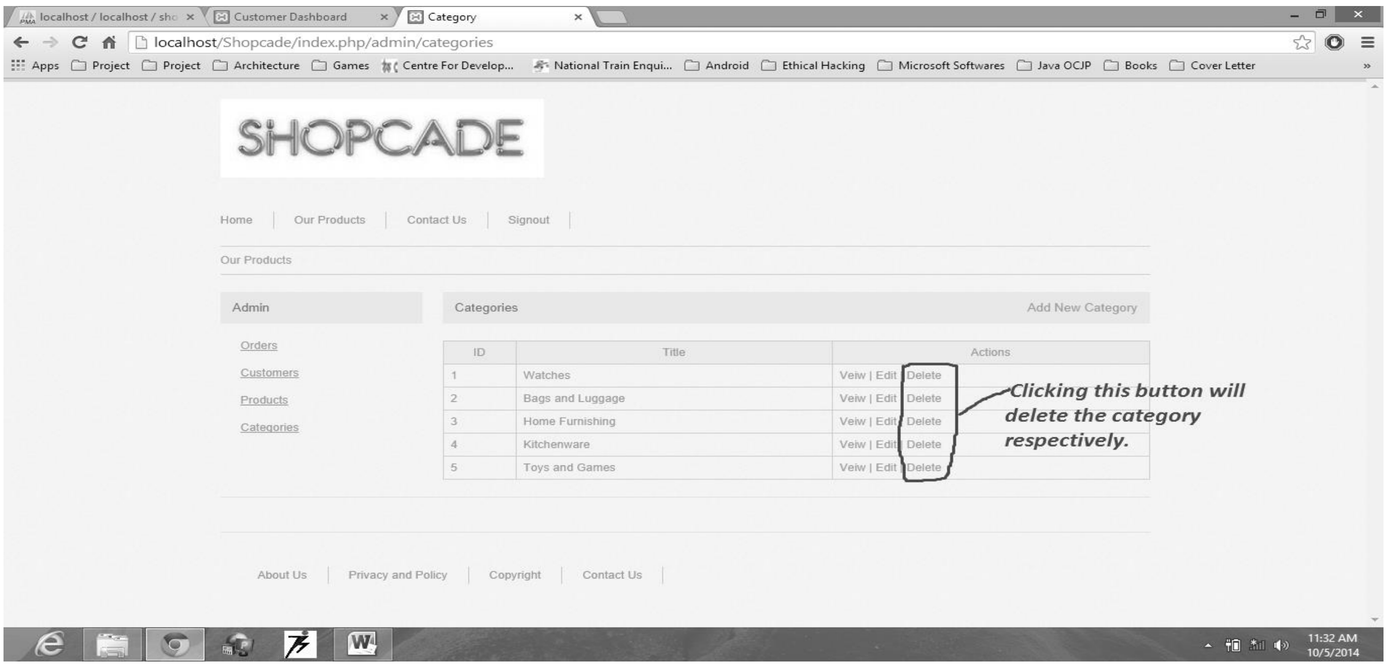


Fig 9: Edit Category Detail

• ***Delete Added Category***: - An admin can also delete any category as if he thinks that there is no need of that category.



ID	Title	Actions
1	Watches	Veiw | Edit | Delete
2	Bags and Luggage	Veiw | Edit | Delete
3	Home Furnishing	Veiw | Edit | Delete
4	Kitchenware	Veiw | Edit | Delete
5	Toys and Games	Veiw | Edit | Delete

Fig 10: Delete Added Category

• ***Add New Product***: - An admin can add any product related to any category whenever a new product is going to instantiate in a market.

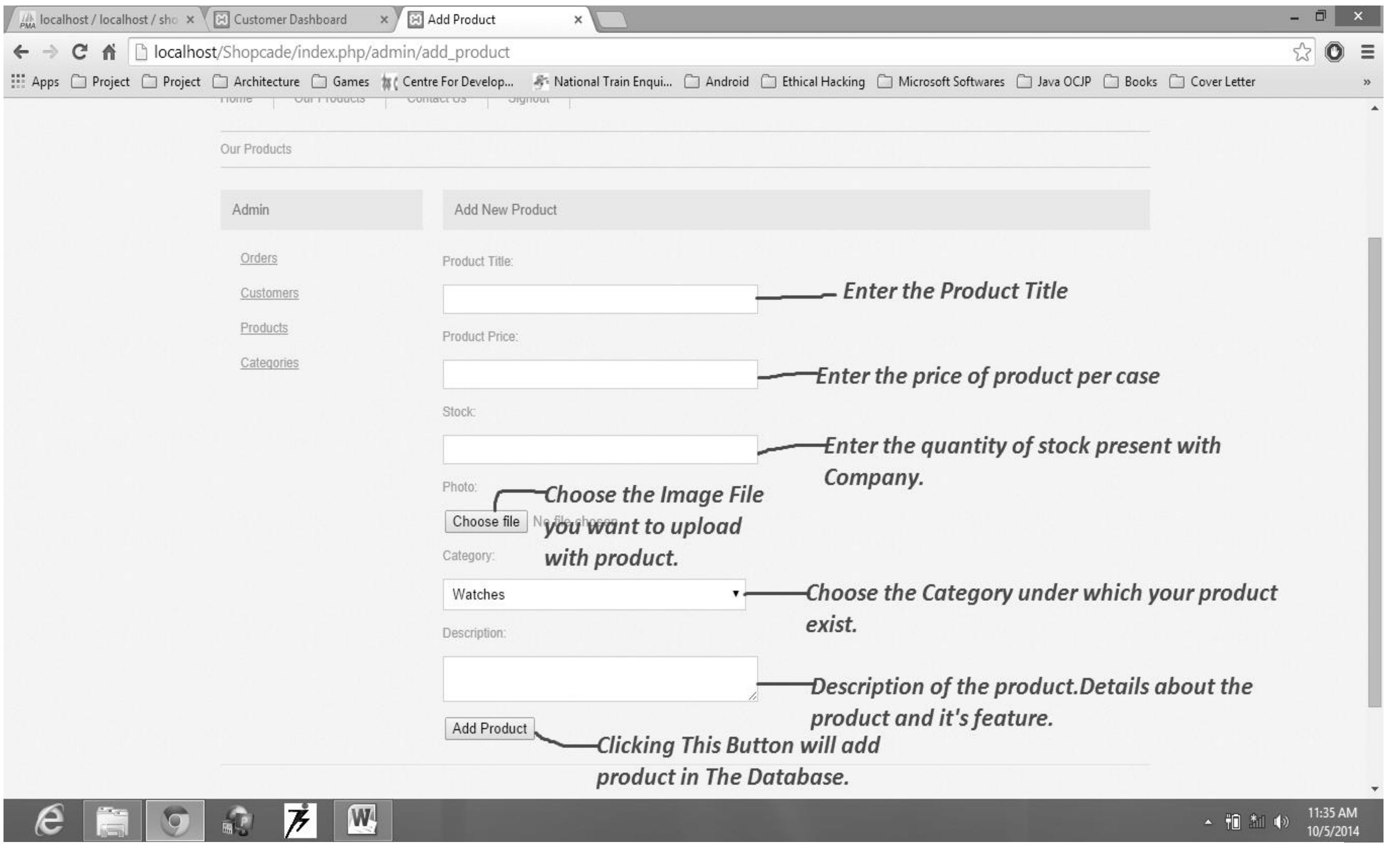

localhost / localhost / sho
Customer Dashboard
Add Product
localhost/Shopcade/index.php/admin/add_product
Apps
Project
Project
Architecture
Games
Centre For Develop...
National Train Enqui...
Android
Ethical Hacking
Microsoft Softwares
Java OCJP
Books
Cover Letter
Our Products
Admin
Add New Product
Orders
Customers
Products
Categories
Product Title:
Enter the Product Title
Product Price:
Enter the price of product per case
Stock:
Enter the quantity of stock present with Company.
Photo:
Choose the Image File you want to upload with product.
Choose file
Category:
Watches
Choose the Category under which your product exist.
Description:
Description of the product.Details about the product and it's feature.
Add Product
Clicking This Button will add product in The Database.
11:35 AM
10/5/2014

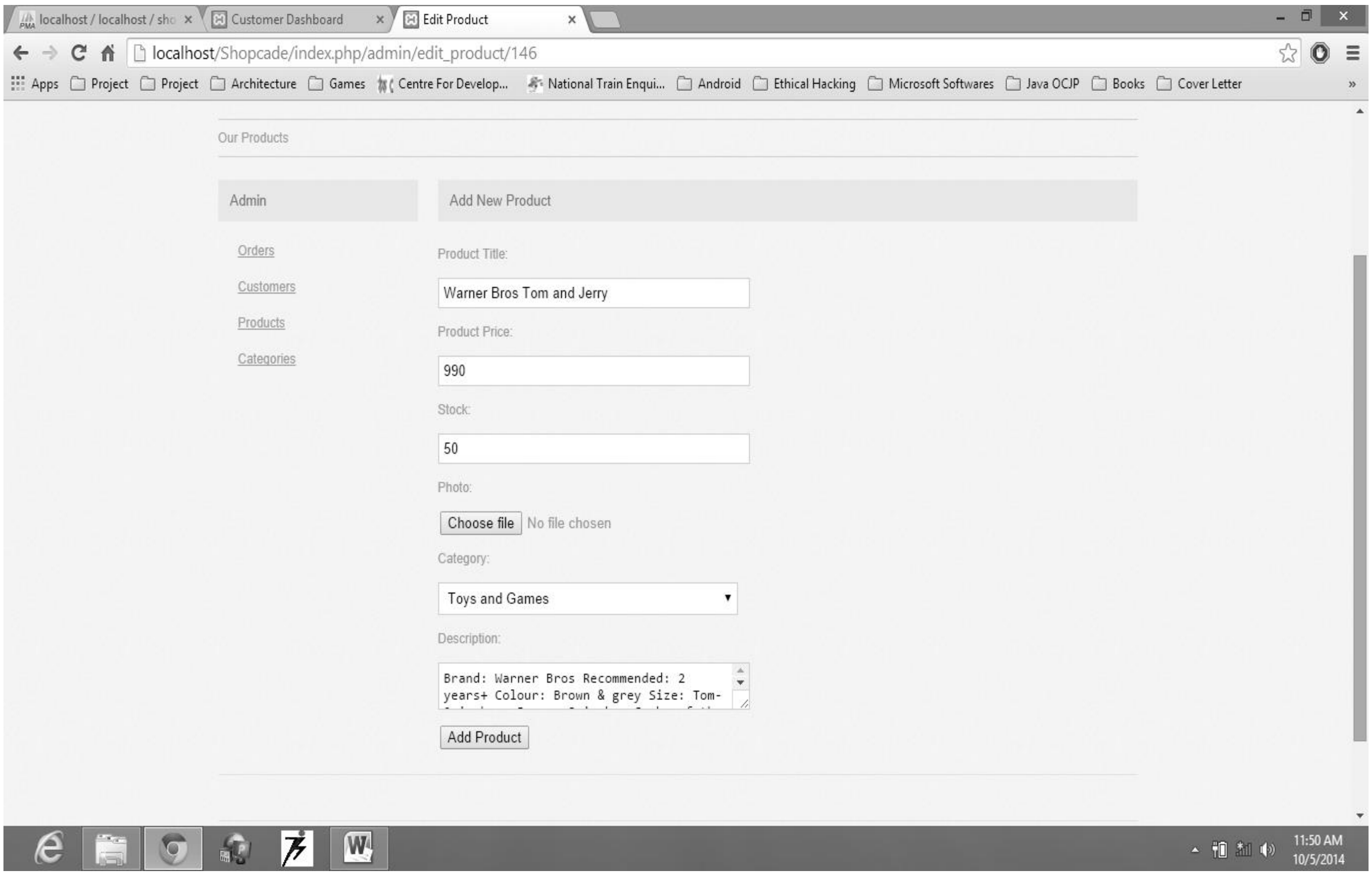
Fig 11: Add New Product

HTML Code for Add new Product:-

```
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<title>Add Product</title>
<link rel="stylesheet" href="<?php echo base_url();?>css/main.css" type="text/css">
</head>
<body>
<div id="wrapper">
<?php include("header.php"); ?>
<div id="content" class="checkout">
<div id="breadcrumb">
<a href="index.php/admin/products">Our Products</a>
</div>
<?php include("left.php"); ?>
<div id="right">
<h1 class="bar">Add New Product</h1>
<?php if(validation_errors()) { ?><div id="errors"><?php echo
validation_errors(); ?></div> <?php } ?>
<?php if($this->uri->segment(4)!=") { ?><div id="errors"> Sorry - Category Already
Exists</div> <?php } ?>
```

```
<form action="<?php echo base_url();?>index.php/admin/add_product" method="post"
enctype="multipart/form-data" id="admin">
<p>
<label>Product Title:</label>
<input name="title" type="text" id="title"  value="<?=set_value('title');?>"/>
</p>
<p>
<label>Product Price:</label>
<input name="price" type="text" id="price" value="<?=set_value('price');?>"/>
</p>
<p>
<label>Stock:</label>
<input name="stock" type="text" id="stock" value="<?=set_value('stock');?>" />
</p>
<p>
<label>Photo:</label>
<input type="file" name="file" id="file" />
</p>
<p>
<label>Category:</label>
<select name="cat_id" id="cat_id">
<?php
$categories = $this->admin_model->getAllCategories();
foreach($categories as $category)
{
?>
<option value="<?php echo $category->cat_id; ?>"><?php echo $category-
>cat_name; ?></option>
<?php
}
?>
</select>
</p>
<p>
<label>Description:</label>
<textarea name="desc" id="desc"><?=set_value('company_name');?></textarea>
</p>
<input type="submit" name="submit" value="Add Product" />
<input type="hidden" name="action" value="1" />
</form>      </div>
<div class="clear"></div>        <div id="footer">  <?php include("footer.php");?>
</div>         </div>         </div>         </body>
```

• ***Edit Product Detail***: - An admin can also edit any product detail.


localhost/Shopcade/index.php/admin/edit_product/146

Our Products

Admin

- Orders
- Customers
- Products
- Categories

Add New Product

Product Title:
Warner Bros Tom and Jerry

Product Price:
990

Stock:
50

Photo:
Choose file No file chosen

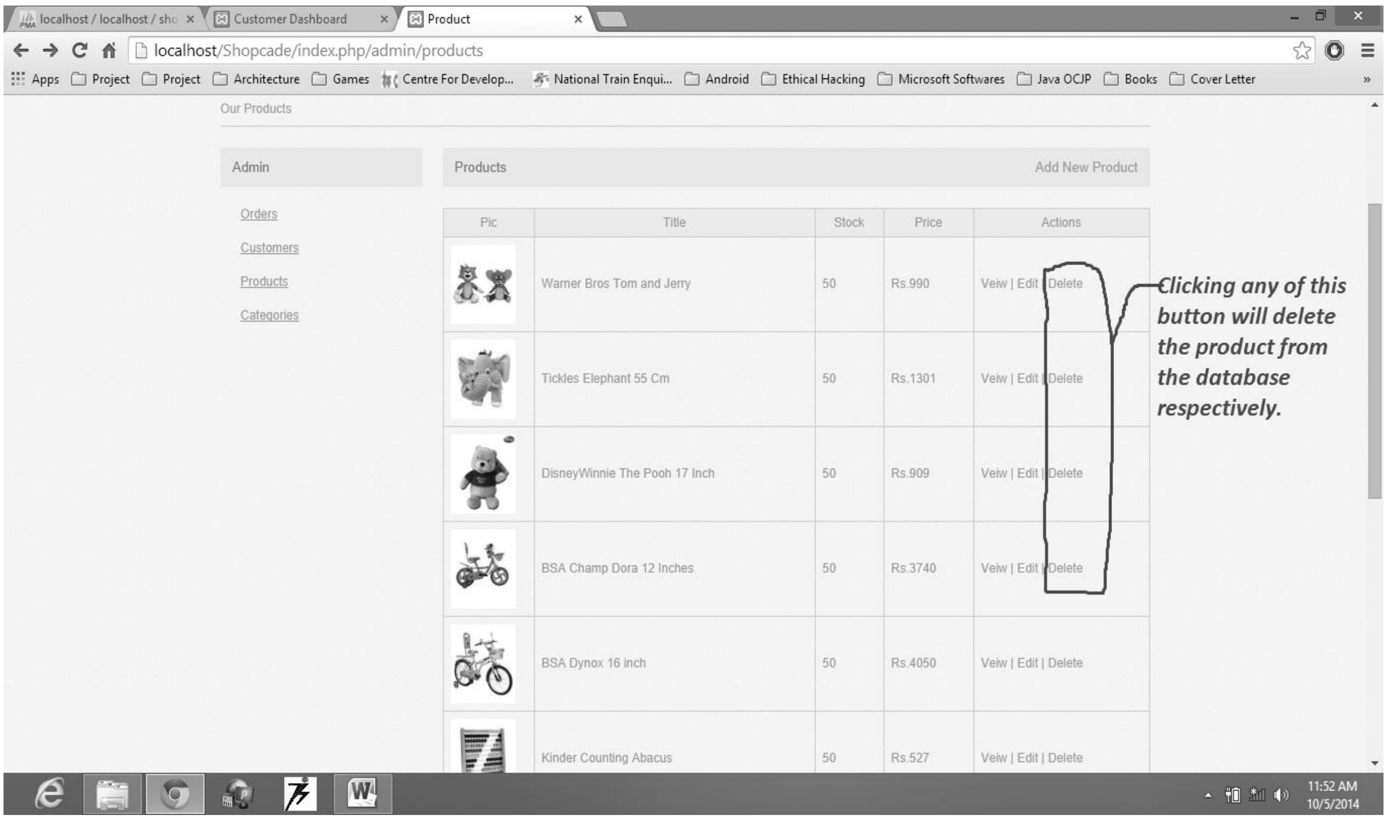
Category:
Toys and Games

Description:
Brand: Warner Bros Recommended: 2 years+ Colour: Brown & grey Size: Tom-

Add Product

Fig 12: Edit Product Detail

• ***Delete any Product***: - An admin can delete any product as soon as there is no requirement of the product or no stock left in the workspace.


localhost/Shopcade/index.php/admin/products

Our Products

Admin

- Orders
- Customers
- Products
- Categories

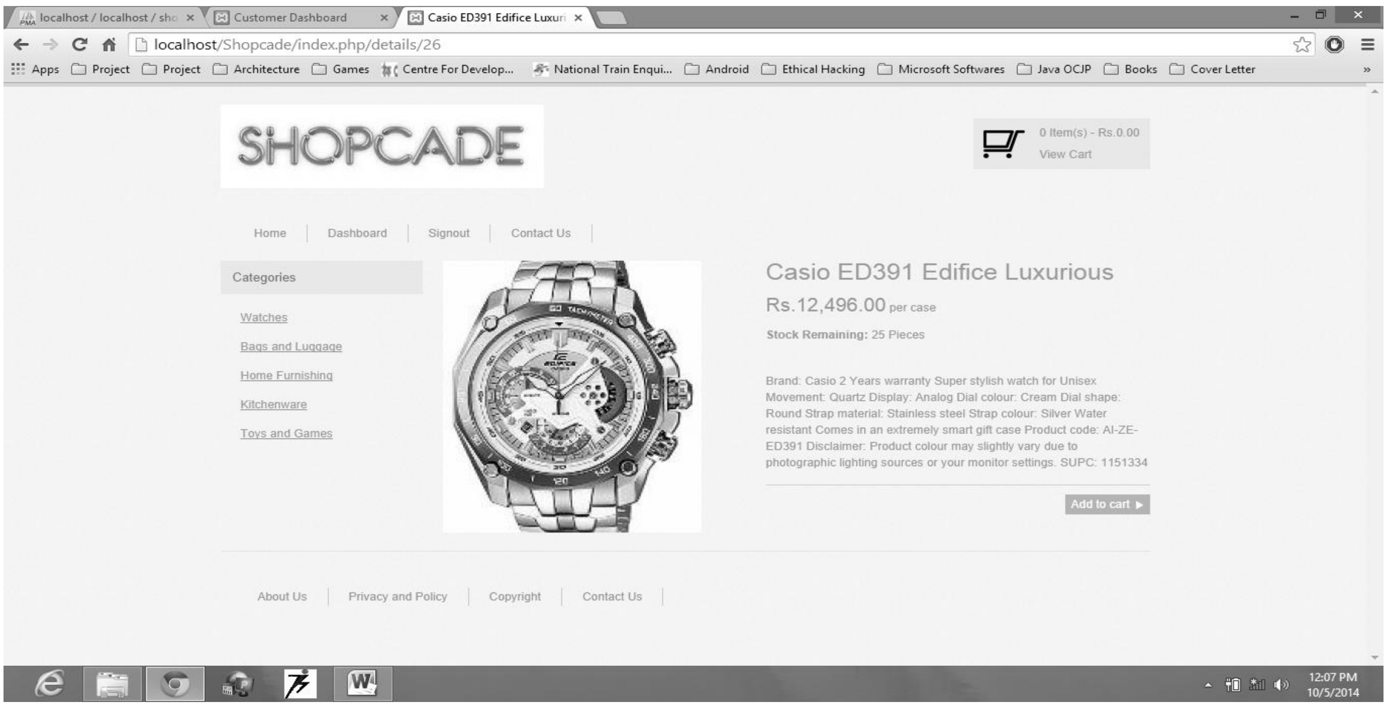
Products | Add New Product

Pic	Title	Stock	Price	Actions
	Warner Bros Tom and Jerry	50	Rs.990	Veiw | Edit | Delete
	Tickles Elephant 55 Cm	50	Rs.1301	Veiw | Edit | Delete
	DisneyWinnie The Pooh 17 Inch	50	Rs.909	Veiw | Edit | Delete
	BSA Champ Dora 12 Inches	50	Rs.3740	Veiw | Edit | Delete
	BSA Dynox 16 inch	50	Rs.4050	Veiw | Edit | Delete
	Kinder Counting Abacus	50	Rs.527	Veiw | Edit | Delete

Clicking any of this button will delete the product from the database respectively.

Fig 13: Delete any Product

- ***View Product***: - An admin can review the detail of any product and check about the status of that product.



Casio ED391 Edifice Luxurious

Rs.12,496.00 per case

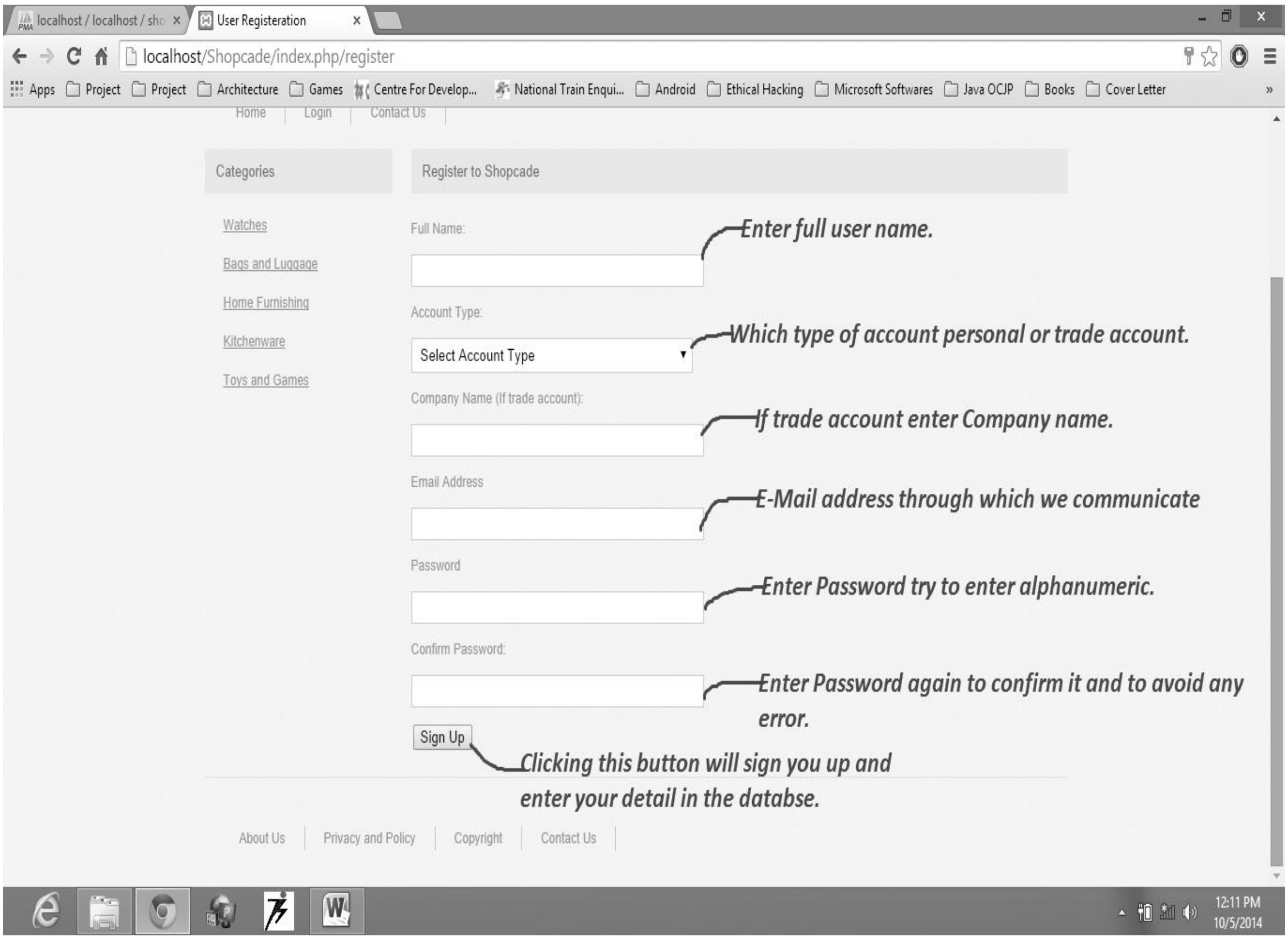
Stock Remaining: 25 Pieces

Brand: Casio 2 Years warranty Super stylish watch for Unisex Movement: Quartz Display: Analog Dial colour: Cream Dial shape: Round Strap material: Stainless steel Strap colour: Silver Water resistant Comes in an extremely smart gift case Product code: AI-ZE-ED391 Disclaimer: Product colour may slightly vary due to photographic lighting sources or your monitor settings. SUPC: 1151334

Fig 14: View Product

4.2. User Functionality

- ***Register***: - A new User can register to the site as when he has to place his first order.


Register to Shopcade

Full Name: — *Enter full user name.*

Account Type: Select Account Type — *Which type of account personal or trade account.*

Company Name (If trade account): — *If trade account enter Company name.*

Email Address — *E-Mail address through which we communicate*

Password — *Enter Password try to enter alphanumeric.*

Confirm Password: — *Enter Password again to confirm it and to avoid any error.*

Sign Up — *Clicking this button will sign you up and enter your detail in the databse.*

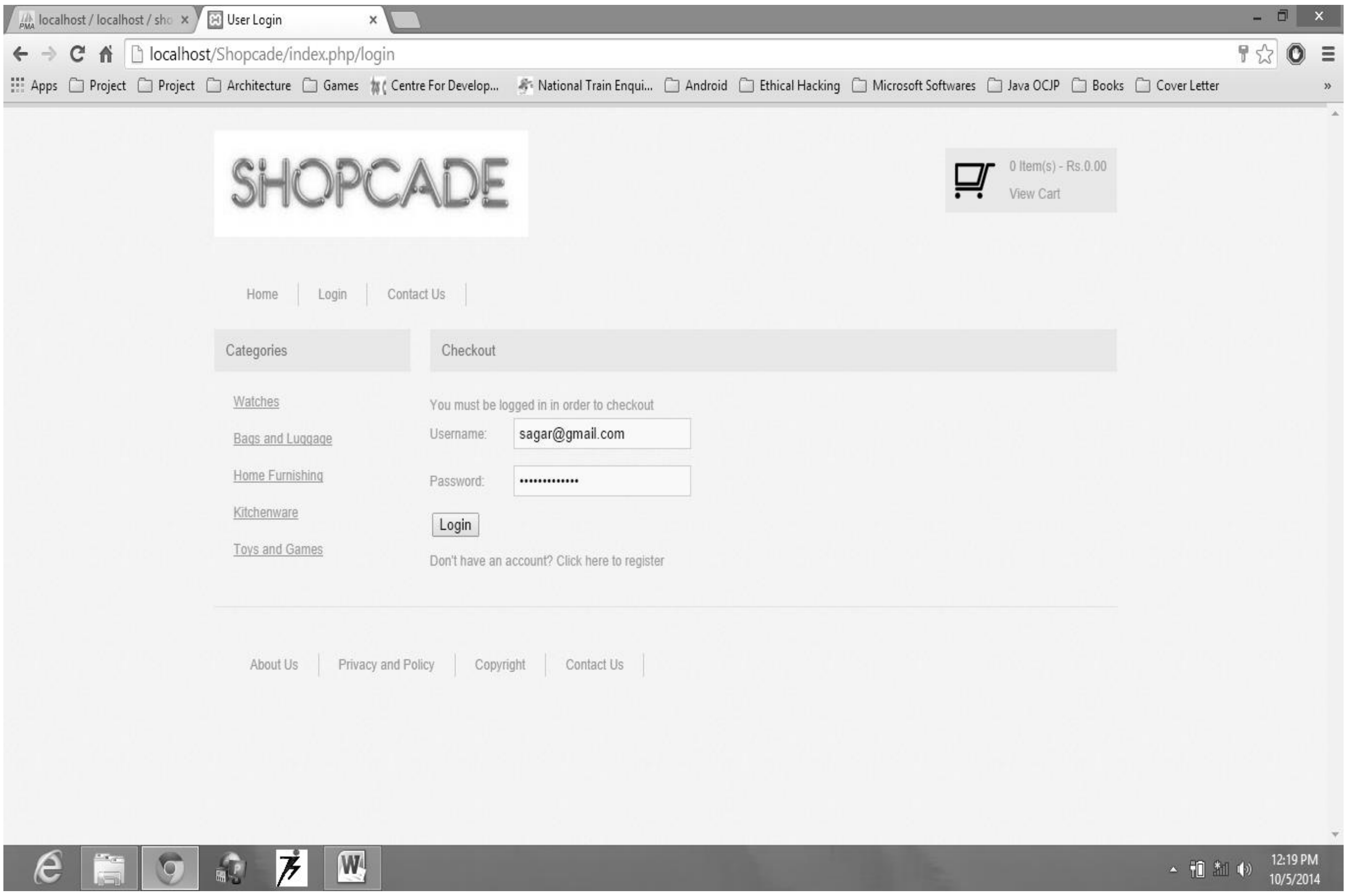
Fig 15: Register

HTML Code for Register:-

```
<head>
<meta http-equiv=”Content-Type” content=”text/html; charset=utf-8” />
<title>User Registeration</title>
<link rel=”stylesheet” href=”<?php echo base_url();?>css/main.css” type=”text/css”>
</head>
<body>
<div id=”wrapper”>
<?php include(“header.php”); ?>
<div id=”content”>
<?php include(“left.php”); ?>
<div id=”right”>
<h1 class=”bar”>Register to Shopcade</h1>
<?php if(validation_errors()) { ?><div id=”errors”><?php echo
validation_errors(); ?></div> <?php } ?>
```

```
<?php if($this->uri->segment(2)!='') { ?><div id="errors"> Sorry – Email is already in
use!</div> <?php } ?>
<form action="<?=base_url();?>index.php/register" method="post"
enctype="multipart/form-data" id="admin">
<p>
<label>Full Name:</label>
<input name="full_name" type="text" id="full_name"
value="<?=set_value('full_name');?>">
</p>
<p>
<label>Account Type:</label>
<select name="account_type" id="account_type">
<option value="">Select Account Type</option>
<option value="personal">Personal</option>
<option value="trade">Trade</option>
</select>
</p>
<p>
<label>Company Name (If trade account):</label>
<input name="company_name" type="text" id="company_name"
value="<?=set_value('company_name');?>">
</p>
<p>
<label>Email Address</label>
<input name="email" type="text" id="email" value="<?=set_value('email');?>">
</p>
<p>
<label>Password</label>
<input name="password" type="password" id="password"
value="<?=set_value('password');?>">
</p>
<p>
<label>Confirm Password:</label>
<input name="password2" type="password" id="password2"
value="<?=set_value('password2');?>">
<input name="register_action" type="hidden" id="register_action" value="true" />
</p>
<input type="submit" name="submit" value="Sign Up">
</form>
</div>
<div class="clear"></div>
<?php include("footer.php"); ?>
</div>
</div>
</body>
```

- ***Login***: - A registered user can login to the site with e-mail address as his username and password he has provide during register him.



You must be logged in in order to checkout

Username: sagar@gmail.com

Password: ••••••••••••••

Don't have an account? Click here to register

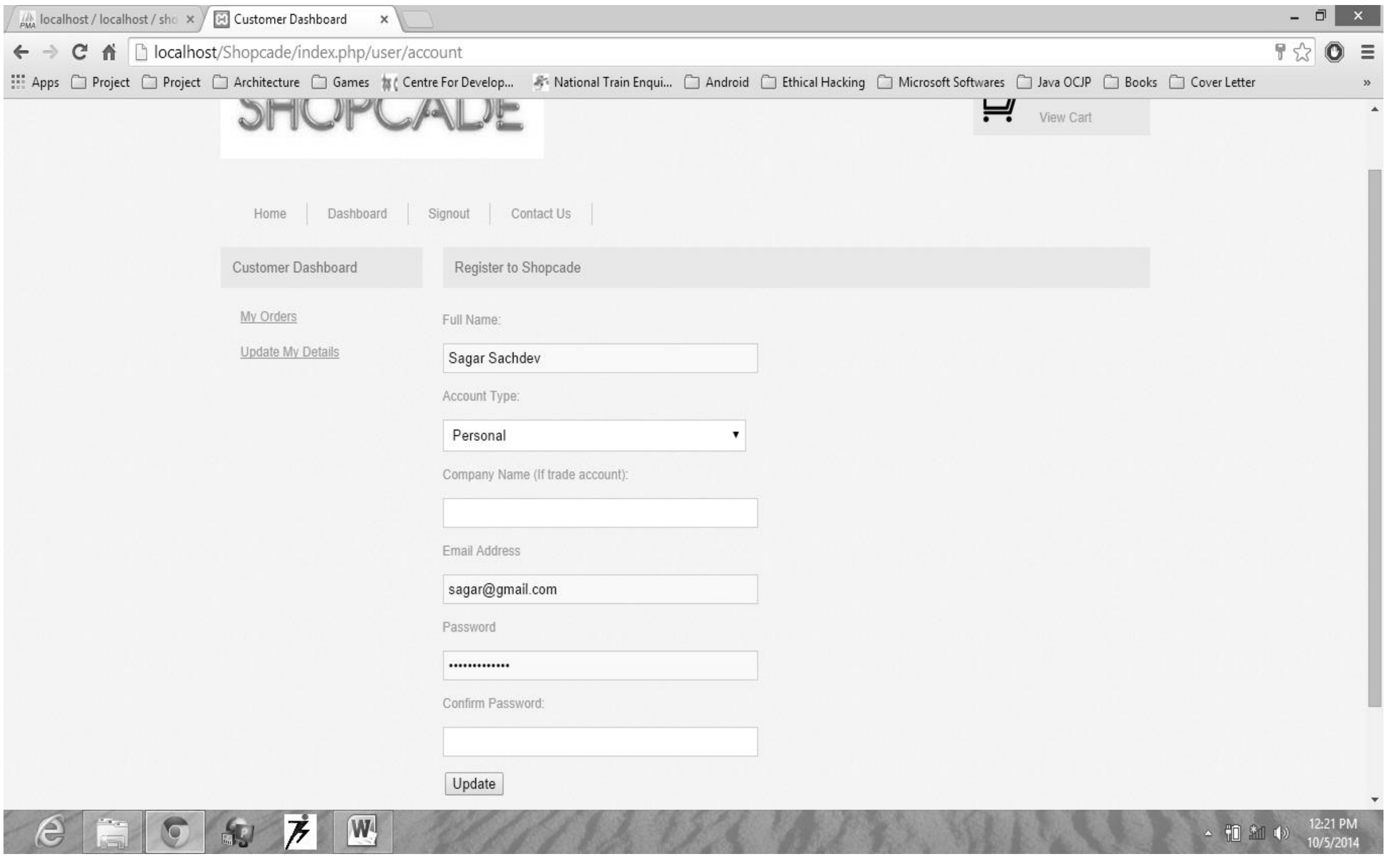
Fig 16: Login

HTML Code for Login:-

```
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<title>User Login</title>
<link rel="stylesheet" href="<?php echo base_url();?>css/main.css" type="text/css">
</head>
<body>
<div id="wrapper">
<?php include("header.php"); ?>
<div id="content">
<?php include("left.php"); ?>
<div id="right">
<h1 class="bar">Checkout</h1>
<p>You must be logged in in order to checkout</p>
<?php if(validation_errors()) { ?><div id="errors"><?php echo
validation_errors(); ?></div> <?php } ?>
<?php if($this->uri->segment(2)=='failed') { ?><div id="errors"> Sorry - Invalid
Username/Password</div> <?php } ?>
```

```
<form action="<?=base_url();?>index.php/login" method="post"
enctype="multipart/form-data" id="login">
<p>
<label>Username:</label>
<input type="text" name="username" value="<?=set_value('username');?>">
</p>
<p>
<label>Password:</label>
<input type="password" name="password" value="<?=set_value('password');?>">
<input name="login_action" type="hidden" id="login_action" value="true" />
</p>
<p>
<input type="submit" value="Login">
<p>Don't have an account? <a href="<?=base_url();?>index.php/register">Click here to
register</a></p>
</form>
</div>
<div class="clear"></div>
<?php include("footer.php"); ?>
</div>
</div>
</body>
```

- ***Edit His Own Details***: - A user can edit his own detail according to his requirement.



Full Name: Sagar Sachdev

Account Type: Personal

Company Name (If trade account):

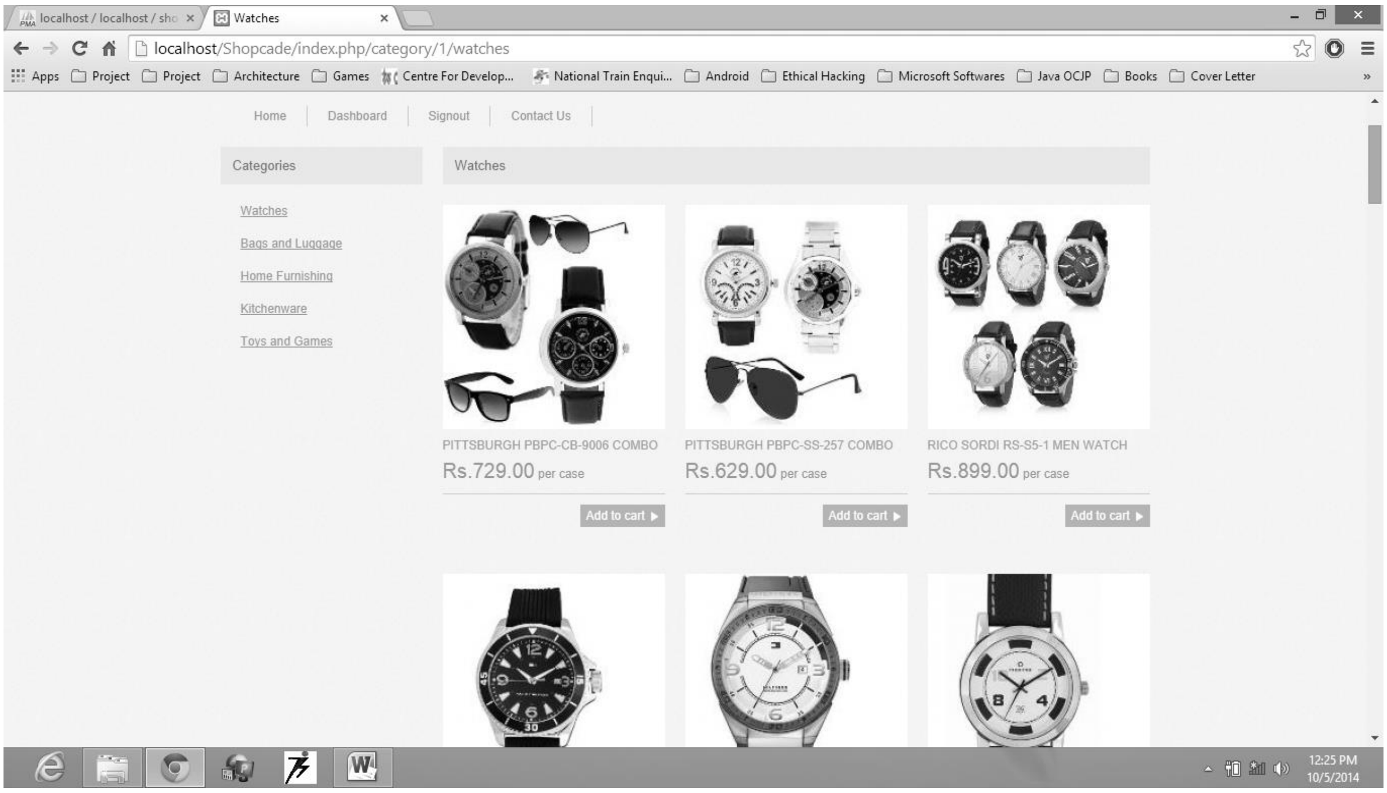
Email Address: sagar@gmail.com

Password: •••••••••••••

Confirm Password:

Fig 17: Edit User Details

- ***View Category Product***: - A User can view the entire product listed in the same category at the time.



Watches

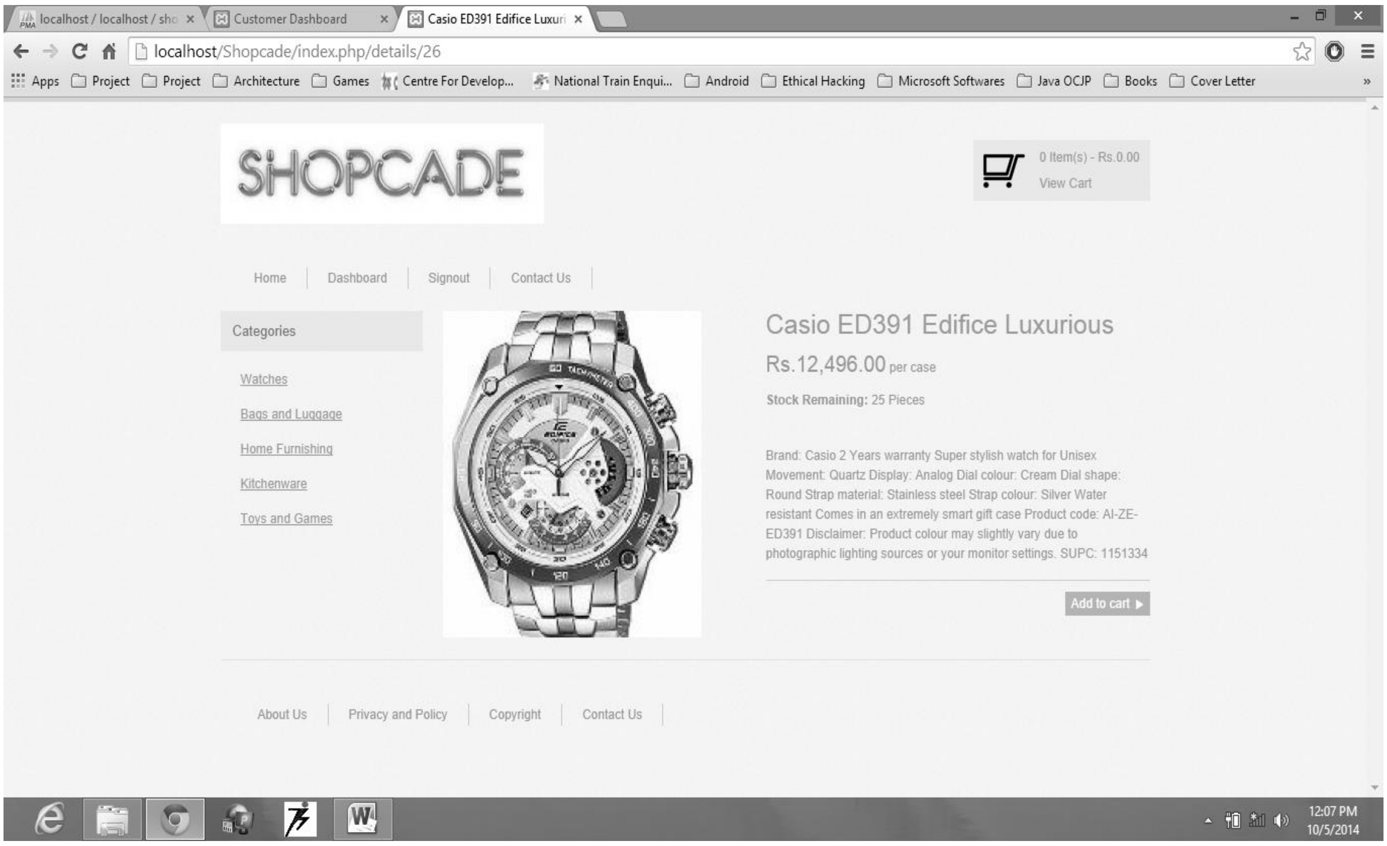
PITTSBURGH PBPC-CB-9006 COMBO
Rs.729.00 per case

PITTSBURGH PBPC-SS-257 COMBO
Rs.629.00 per case

RICO SORDI RS-S5-1 MEN WATCH
Rs.899.00 per case

Fig 18: View Category wise Product

- ***View Product Detail***: - A user can view the detail of any product and check about the status of that product.



Casio ED391 Edifice Luxurious

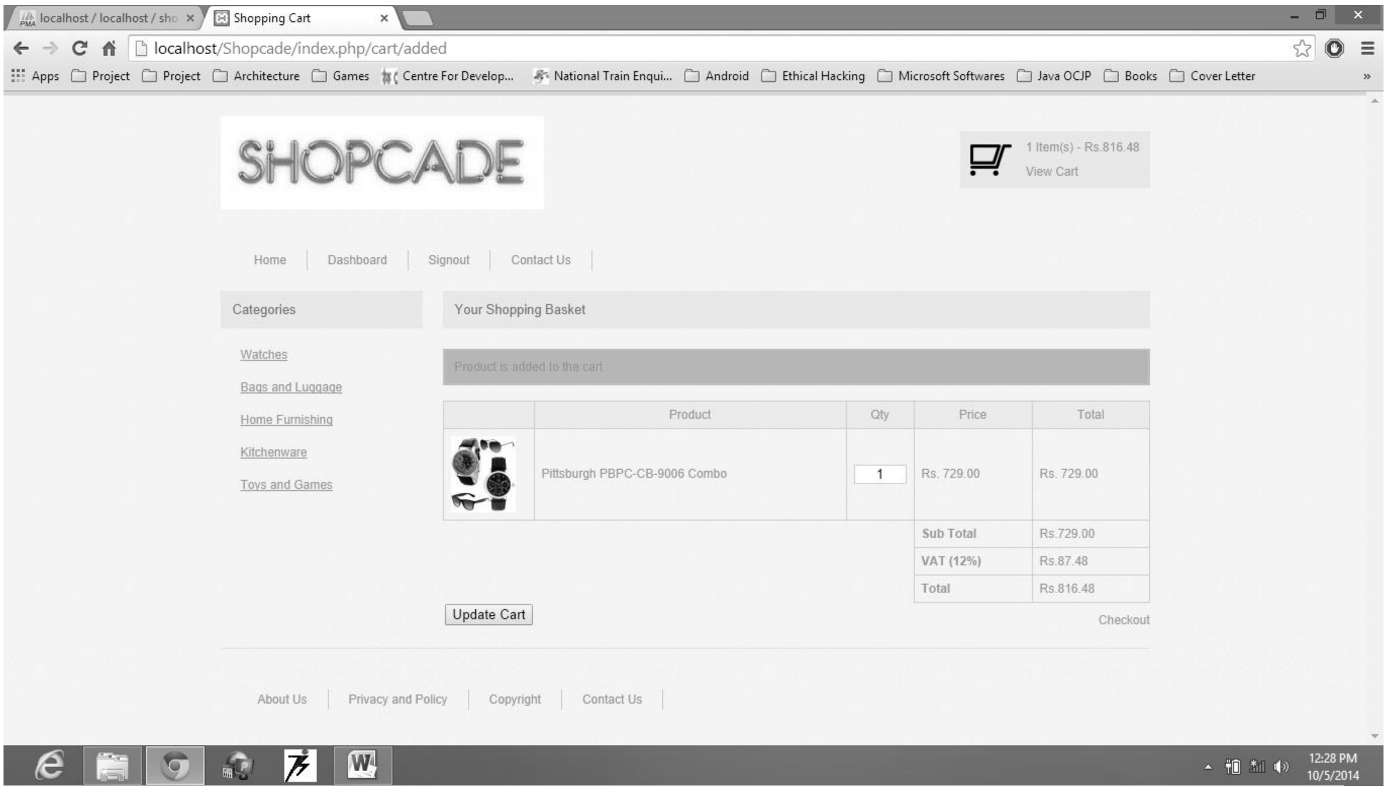
Rs.12,496.00 per case

Stock Remaining: 25 Pieces

Brand: Casio 2 Years warranty Super stylish watch for Unisex Movement: Quartz Display: Analog Dial colour: Cream Dial shape: Round Strap material: Stainless steel Strap colour: Silver Water resistant Comes in an extremely smart gift case Product code: AI-ZE-ED391 Disclaimer: Product colour may slightly vary due to photographic lighting sources or your monitor settings. SUPC: 1151334

Fig 19: View Product Detail

- ***Add Product to Cart***: - A user can add Product to the cart whichever product he wants to buy default quantity is one.



localhost/Shopcade/index.php/cart/added

SHOPCADE

1 Item(s) - Rs.816.48
View Cart

Home | Dashboard | Signout | Contact Us

Categories: Watches, Bags and Luggage, Home Furnishing, Kitchenware, Toys and Games

Your Shopping Basket

Product is added to the cart

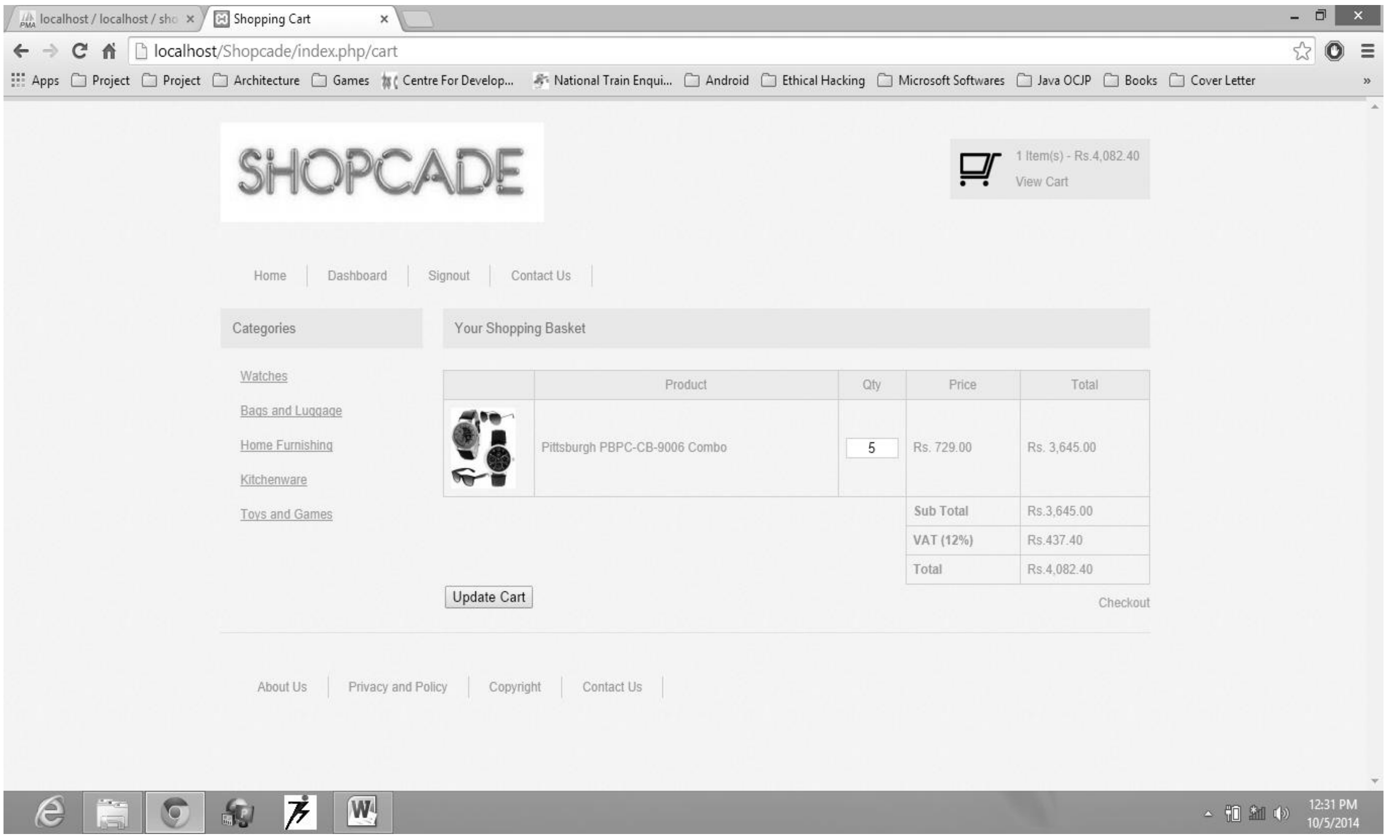
	Product	Qty	Price	Total
	Pittsburgh PBPC-CB-9006 Combo	1	Rs. 729.00	Rs. 729.00
			Sub Total	Rs.729.00
			VAT (12%)	Rs.87.48
			Total	Rs.816.48

Update Cart   Checkout

About Us | Privacy and Policy | Copyright | Contact Us

Fig 20: Add Product to Cart

- ***Update Cart***: - A user can update his cart also by changing his product quantity to more than one or make it zero to remove it from the cart.



localhost/Shopcade/index.php/cart

SHOPCADE

1 Item(s) - Rs.4,082.40
View Cart

Home | Dashboard | Signout | Contact Us

Categories: Watches, Bags and Luggage, Home Furnishing, Kitchenware, Toys and Games

Your Shopping Basket

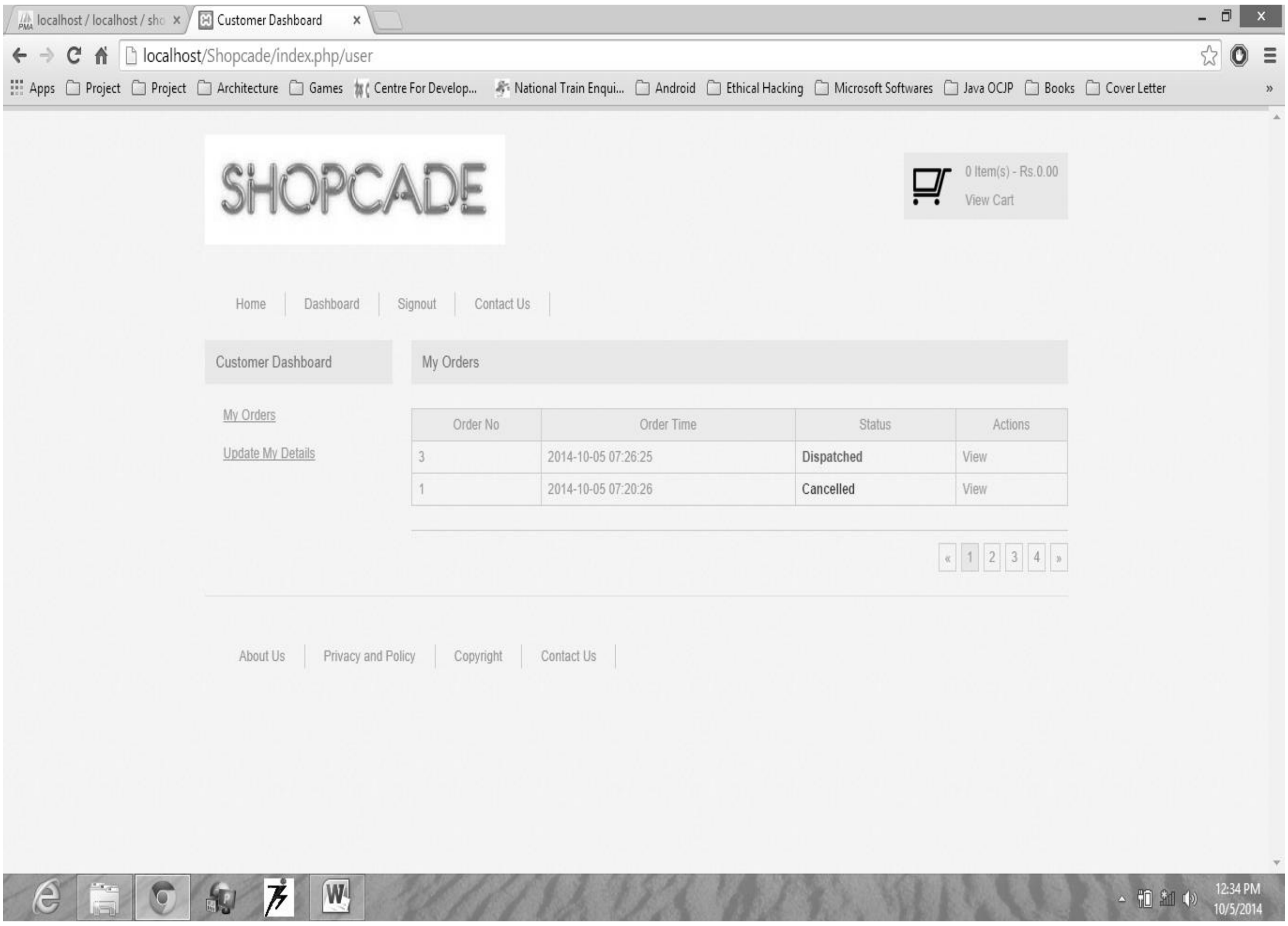
	Product	Qty	Price	Total
	Pittsburgh PBPC-CB-9006 Combo	5	Rs. 729.00	Rs. 3,645.00
			Sub Total	Rs.3,645.00
			VAT (12%)	Rs.437.40
			Total	Rs.4,082.40

Update Cart   Checkout

About Us | Privacy and Policy | Copyright | Contact Us

Fig 21: Update Cart

- ***View Status of his Order***: - A user can review the status of his product whether it is dispatched or not.


localhost/Shopcade/index.php/user

0 Item(s) - Rs.0.00
View Cart

Home | Dashboard | Signout | Contact Us

Customer Dashboard

My Orders

Update My Details

My Orders

Order No	Order Time	Status	Actions
3	2014-10-05 07:26:25	**Dispatched**	View
1	2014-10-05 07:20:26	**Cancelled**	View

« 1 2 3 4 »

About Us | Privacy and Policy | Copyright | Contact Us

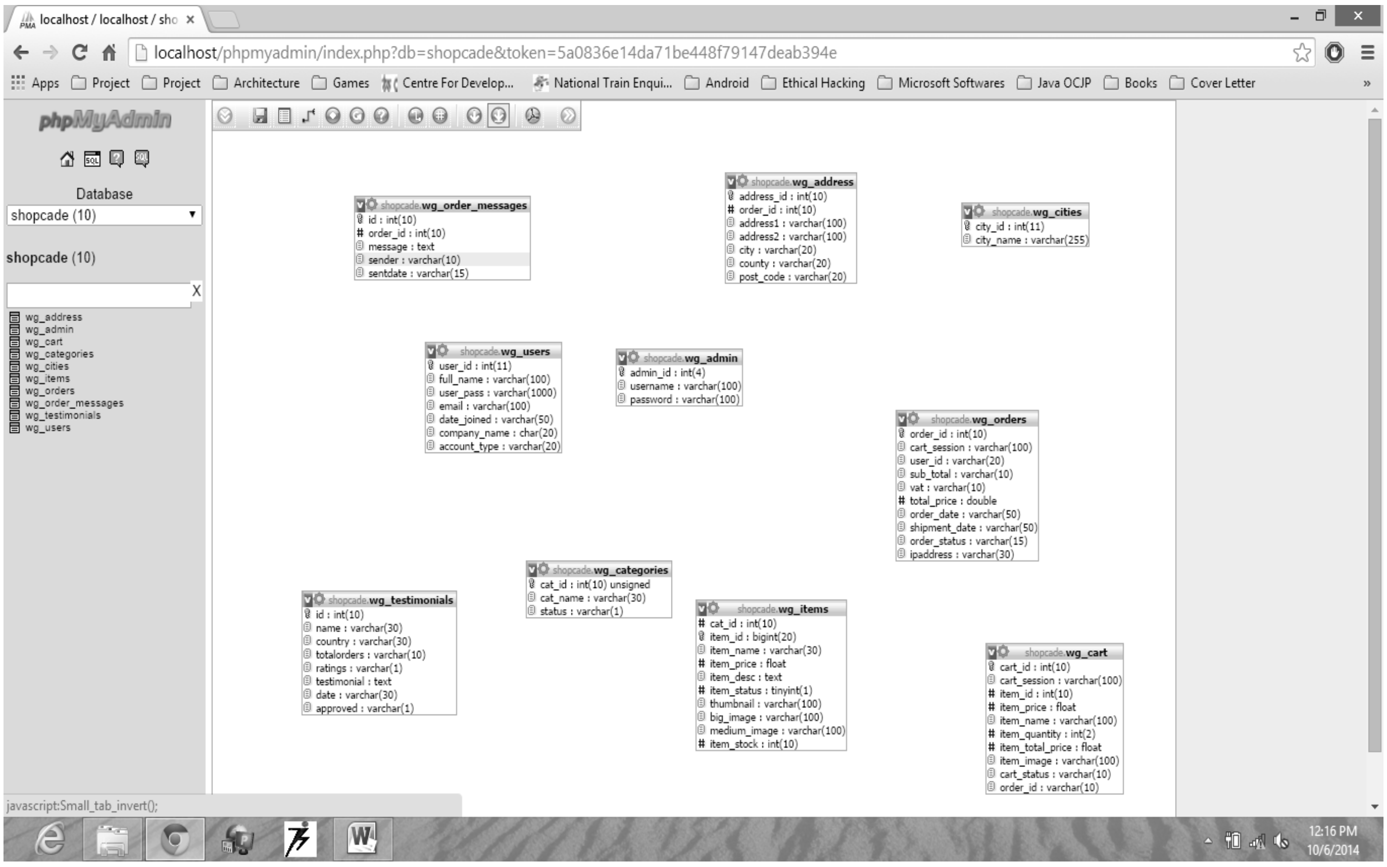
Fig 22: View Status of His Order

4.3. <u>Database Details</u>

In this website we have used MySQL Database in which we have 10 tables with their respective functionality.

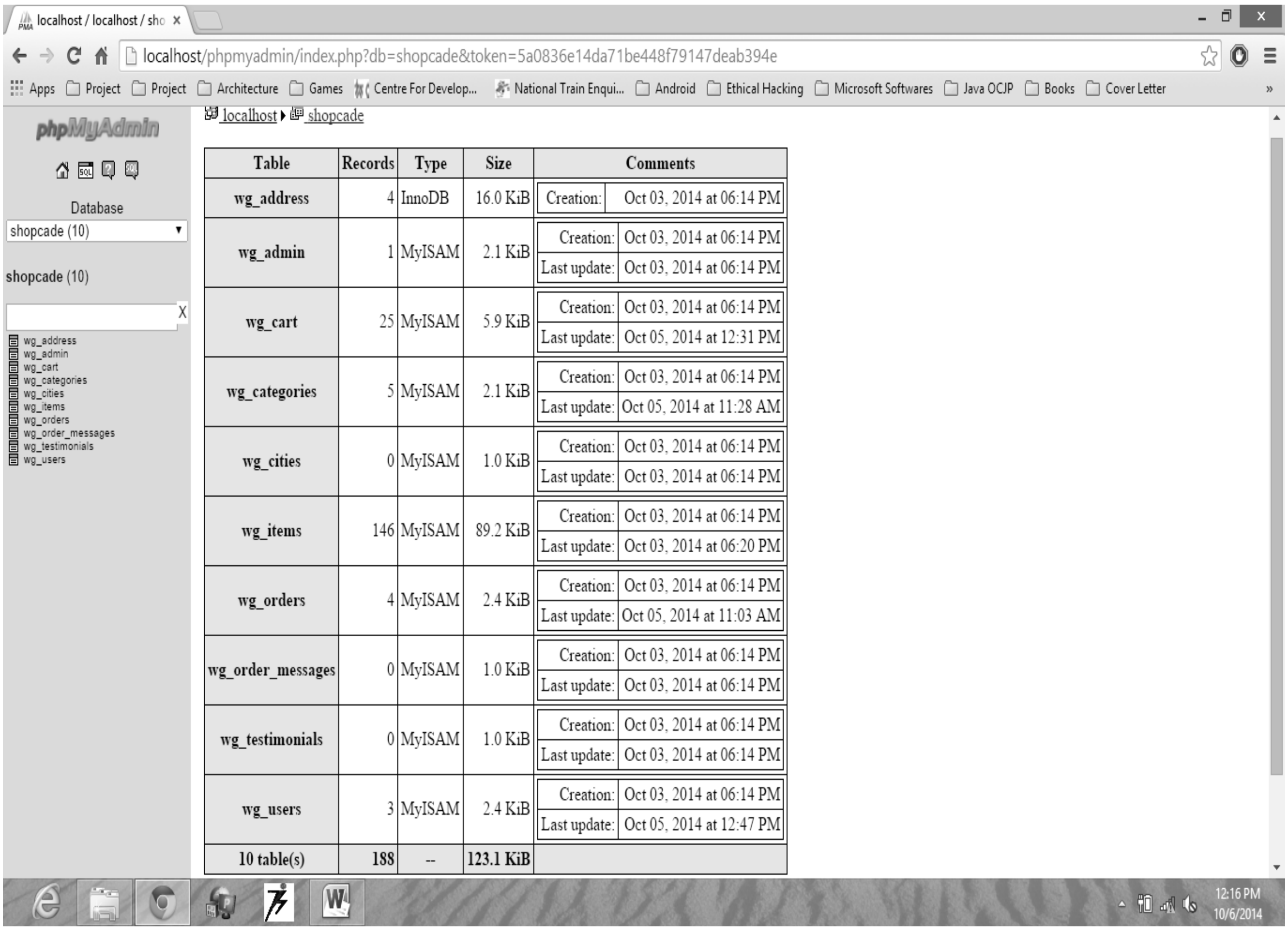
Sr. No.	Table Name	Functionality
1	wg_address	It contains detail about the addresses that are used to place the order. Where the particular delivery has to be happen.
2	wg_admin	It contains detail about the admin login username and password.
3	wg_cart	It contains detail about the cart which product is added to the cart and what is the cart session Id.
4	wg_categories	It contains detail about the category. What are the id and name.
5	wg_cities	It contains detail about the cities in which order has to be provided.
6	wg_items	It contains detail about the items that are present on the site with their particular item id and cart id.
7	wg_order	It contains detail about the order that which order has happened when and by whom.
8	wg_order_messages	If anyone has send any message field it will be stored in this particular table with sender details.
9	wg_testimonials	It contains detail about the testimonial and no of order they have placed and their ratings.
10	wg_users	It contains detail about the user like their username and password.

Table - *4*



```
shopcade.wg_order_messages
id : int(10)
order_id : int(10)
message : text
sender : varchar(10)
sentdate : varchar(15)

shopcade.wg_address
address_id : int(10)
order_id : int(10)
address1 : varchar(100)
address2 : varchar(100)
city : varchar(20)
county : varchar(20)
post_code : varchar(20)

shopcade.wg_cities
city_id : int(11)
city_name : varchar(255)

shopcade.wg_users
user_id : int(11)
full_name : varchar(100)
user_pass : varchar(1000)
email : varchar(100)
date_joined : varchar(50)
company_name : char(20)
account_type : varchar(20)

shopcade.wg_admin
admin_id : int(4)
username : varchar(100)
password : varchar(100)

shopcade.wg_orders
order_id : int(10)
cart_session : varchar(100)
user_id : varchar(20)
sub_total : varchar(10)
vat : varchar(10)
total_price : double
order_date : varchar(50)
shipment_date : varchar(50)
order_status : varchar(15)
ipaddress : varchar(30)

shopcade.wg_categories
cat_id : int(10) unsigned
cat_name : varchar(30)
status : varchar(1)

shopcade.wg_testimonials
id : int(10)
name : varchar(30)
country : varchar(30)
totalorders : varchar(10)
ratings : varchar(1)
testimonial : text
date : varchar(30)
approved : varchar(1)

shopcade.wg_items
cat_id : int(10)
item_id : bigint(20)
item_name : varchar(30)
item_price : float
item_desc : text
item_status : tinyint(1)
thumbnail : varchar(100)
big_image : varchar(100)
medium_image : varchar(100)
item_stock : int(10)

shopcade.wg_cart
cart_id : int(10)
cart_session : varchar(100)
item_id : int(10)
item_price : float
item_name : varchar(100)
item_quantity : int(2)
item_total_price : float
item_image : varchar(100)
cart_status : varchar(10)
order_id : varchar(10)
```

Fig 23: Database Shopcade Design



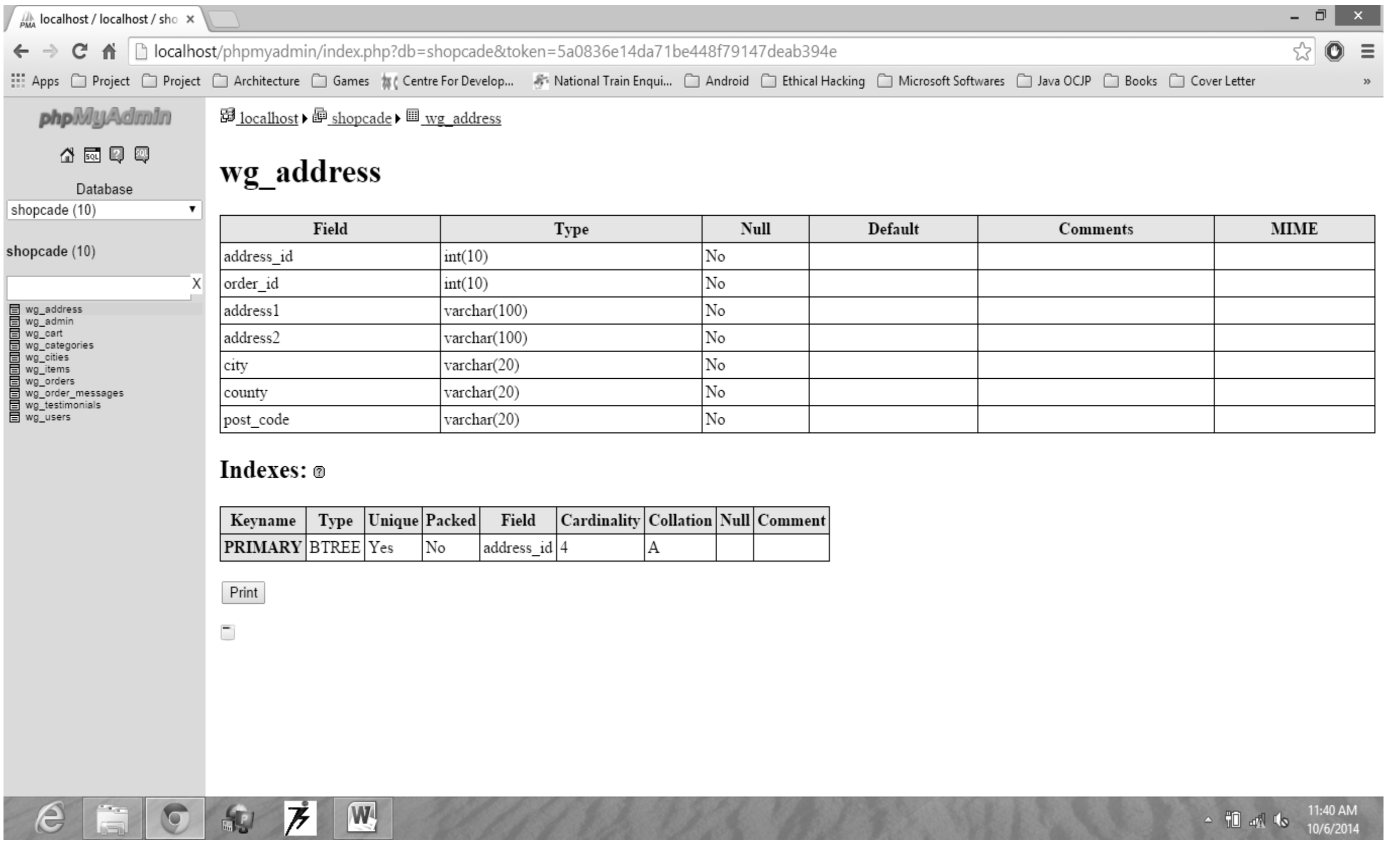
localhost ▸ shopcade

Table	Records	Type	Size	Comments
wg_address	4	InnoDB	16.0 KiB	Creation: Oct 03, 2014 at 06:14 PM
wg_admin	1	MyISAM	2.1 KiB	Creation: Oct 03, 2014 at 06:14 PM; Last update: Oct 03, 2014 at 06:14 PM
wg_cart	25	MyISAM	5.9 KiB	Creation: Oct 03, 2014 at 06:14 PM; Last update: Oct 05, 2014 at 12:31 PM
wg_categories	5	MyISAM	2.1 KiB	Creation: Oct 03, 2014 at 06:14 PM; Last update: Oct 05, 2014 at 11:28 AM
wg_cities	0	MyISAM	1.0 KiB	Creation: Oct 03, 2014 at 06:14 PM; Last update: Oct 03, 2014 at 06:14 PM
wg_items	146	MyISAM	89.2 KiB	Creation: Oct 03, 2014 at 06:14 PM; Last update: Oct 03, 2014 at 06:20 PM
wg_orders	4	MyISAM	2.4 KiB	Creation: Oct 03, 2014 at 06:14 PM; Last update: Oct 05, 2014 at 11:03 AM
wg_order_messages	0	MyISAM	1.0 KiB	Creation: Oct 03, 2014 at 06:14 PM; Last update: Oct 03, 2014 at 06:14 PM
wg_testimonials	0	MyISAM	1.0 KiB	Creation: Oct 03, 2014 at 06:14 PM; Last update: Oct 03, 2014 at 06:14 PM
wg_users	3	MyISAM	2.4 KiB	Creation: Oct 03, 2014 at 06:14 PM; Last update: Oct 05, 2014 at 12:47 PM
10 table(s)	188	--	123.1 KiB	

Fig 24: Database Shopcade Structure View

4.3.1. wg_address:-

This table contains detail about the addresses at which shipping has to be done. It contains attributes like address_id, order_id, address1, address2, city, country, post_code.



localhost ▸ shopcade ▸ wg_address

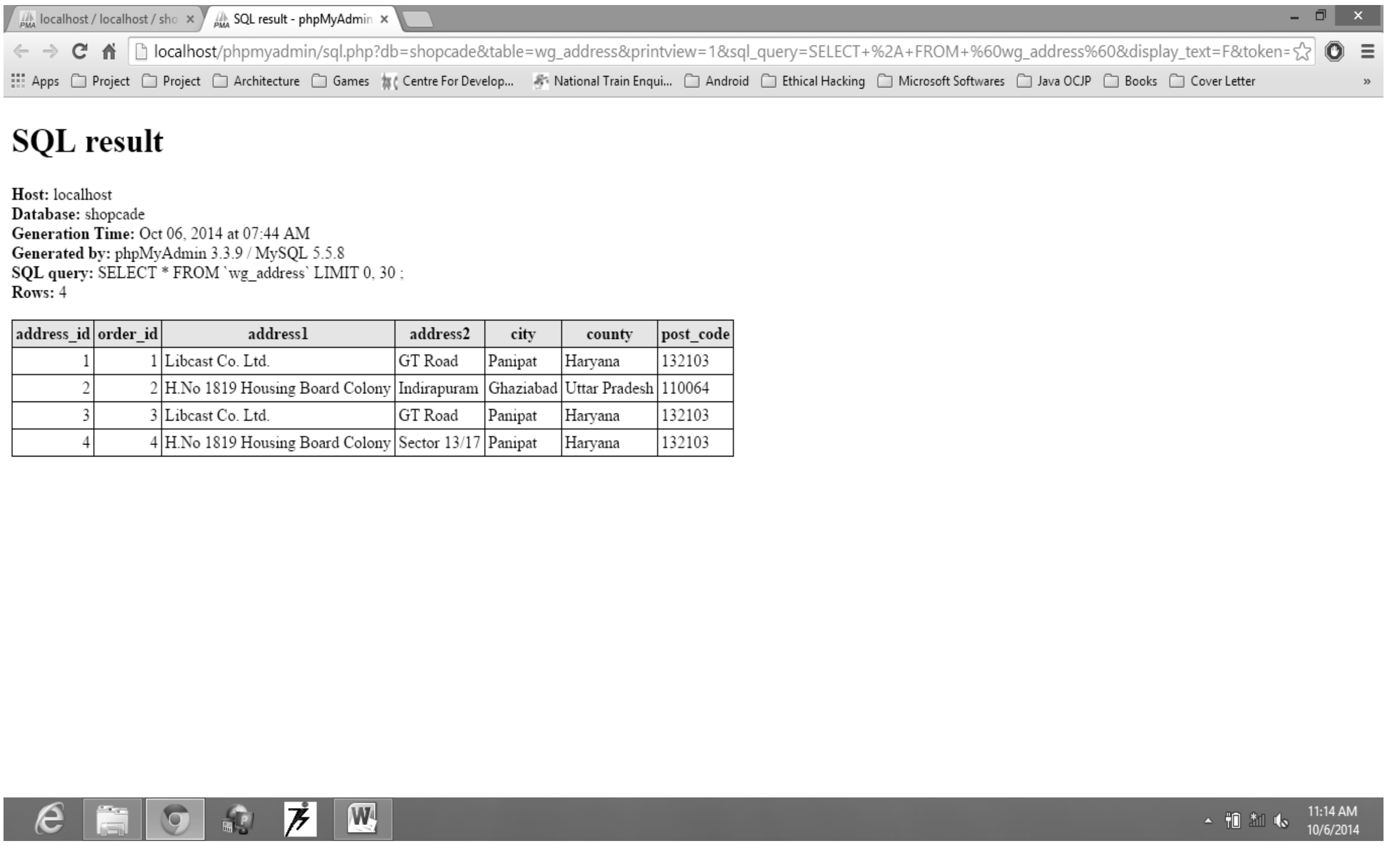
wg_address

Field	Type	Null	Default	Comments	MIME
address_id	int(10)	No			
order_id	int(10)	No			
address1	varchar(100)	No			
address2	varchar(100)	No			
city	varchar(20)	No			
county	varchar(20)	No			
post_code	varchar(20)	No			

Indexes:

Keyname	Type	Unique	Packed	Field	Cardinality	Collation	Null	Comment
PRIMARY	BTREE	Yes	No	address_id	4	A		

Fig 25: Table wg_address Structure View



SQL result

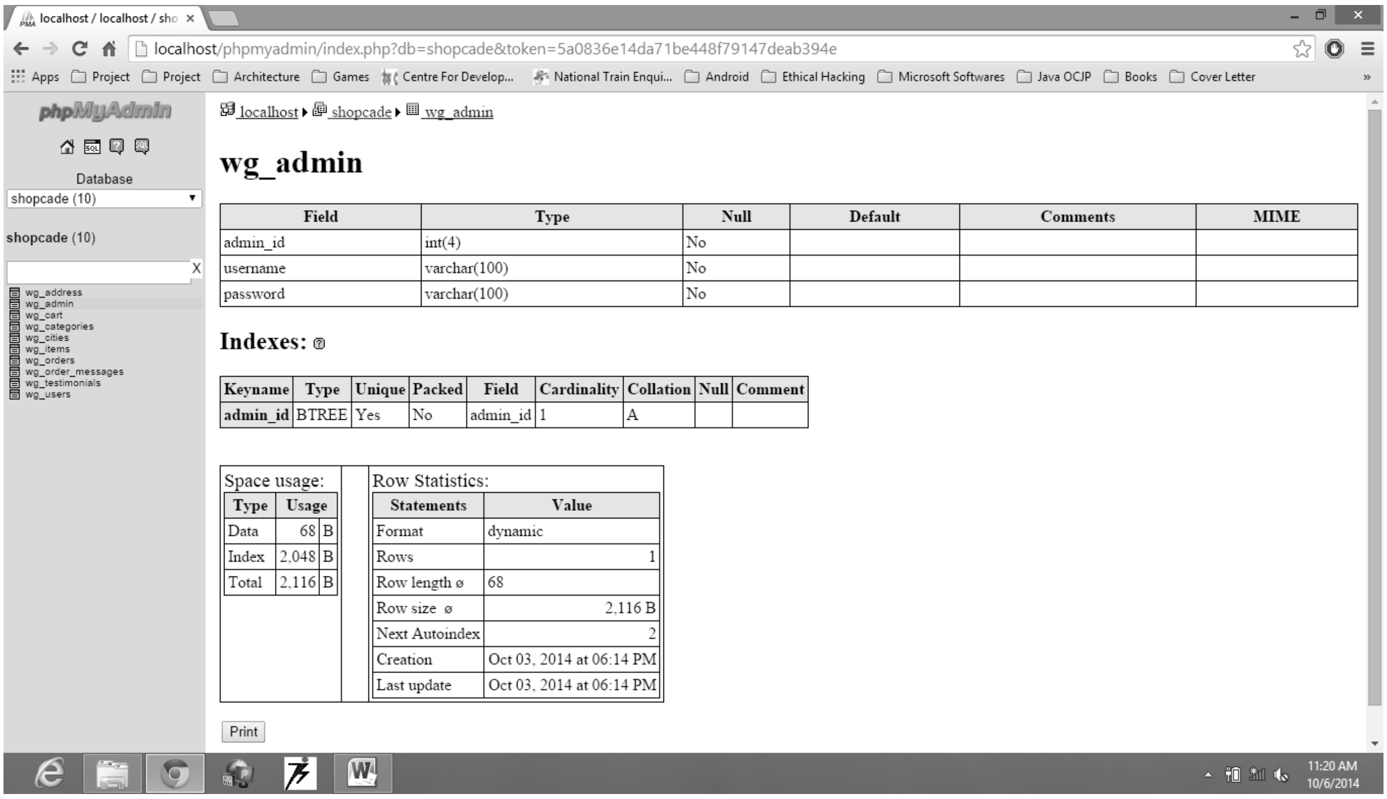
Host: localhost
Database: shopcade
Generation Time: Oct 06, 2014 at 07:44 AM
Generated by: phpMyAdmin 3.3.9 / MySQL 5.5.8
SQL query: SELECT * FROM `wg_address` LIMIT 0, 30 ;
Rows: 4

address_id	order_id	address1	address2	city	county	post_code
1	1	Libcast Co. Ltd.	GT Road	Panipat	Haryana	132103
2	2	H.No 1819 Housing Board Colony	Indirapuram	Ghaziabad	Uttar Pradesh	110064
3	3	Libcast Co. Ltd.	GT Road	Panipat	Haryana	132103
4	4	H.No 1819 Housing Board Colony	Sector 13/17	Panipat	Haryana	132103

Fig 26: Table wg_address Full Text View

4.3.2. <u>wg_admin: -</u>

It contains attribute like admin_id, username and password. In this it is indexes on the bases of admin_id and indexes type is BTREE.



localhost / shopcade / wg_admin

wg_admin

Field	Type	Null	Default	Comments	MIME
admin_id	int(4)	No			
username	varchar(100)	No			
password	varchar(100)	No			

Indexes:

Keyname	Type	Unique	Packed	Field	Cardinality	Collation	Null	Comment
admin_id	BTREE	Yes	No	admin_id	1	A		

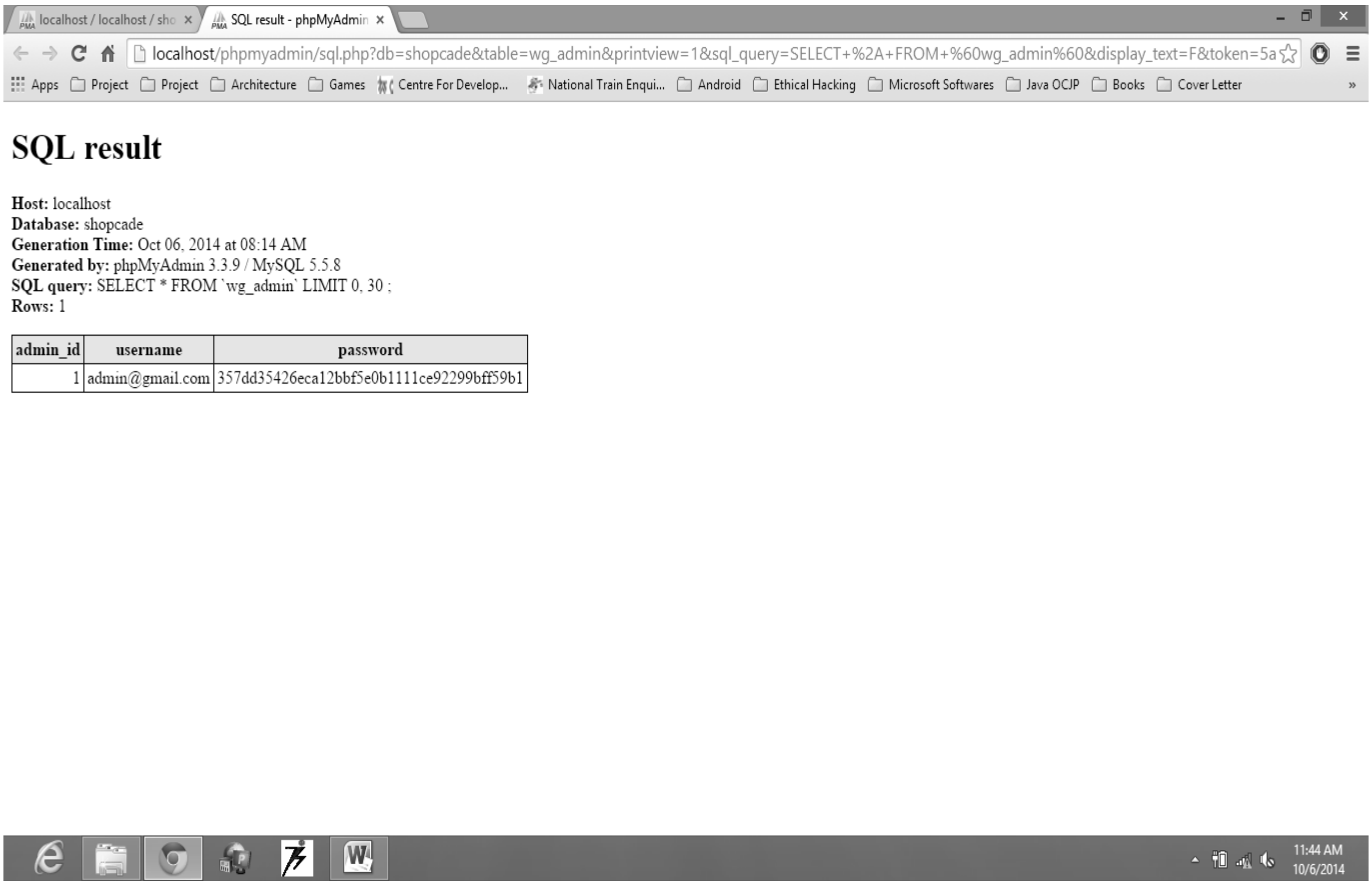
Space usage:

Type	Usage	
Data	68	B
Index	2,048	B
Total	2,116	B

Row Statistics:

Statements	Value
Format	dynamic
Rows	1
Row length ø	68
Row size ø	2,116 B
Next Autoindex	2
Creation	Oct 03, 2014 at 06:14 PM
Last update	Oct 03, 2014 at 06:14 PM

Fig 27: Table wg_admin Structure View



SQL result

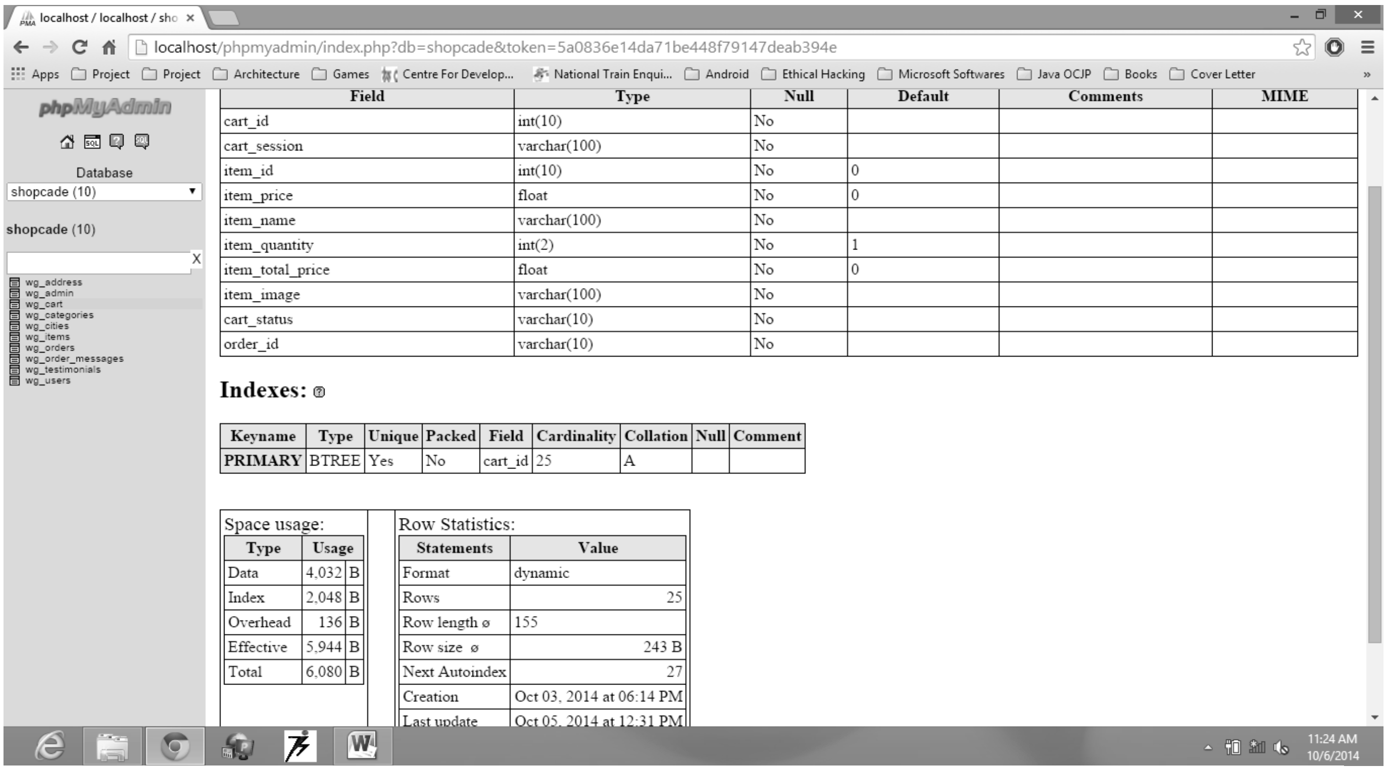
Host: localhost
Database: shopcade
Generation Time: Oct 06, 2014 at 08:14 AM
Generated by: phpMyAdmin 3.3.9 / MySQL 5.5.8
SQL query: SELECT * FROM `wg_admin` LIMIT 0, 30 ;
Rows: 1

admin_id	username	password
1	admin@gmail.com	357dd35426eca12bbf5e0b1111ce92299bff59b1

Fig 28: Table wg_admin Full Text View

4.3.3. wg_cart: -

It contains attribute like cart_id, cart_session, item_id, item_price, item_name, item_quantity, item_total_price, item_image, cart_status and order_id.



localhost/phpmyadmin/index.php?db=shopcade&token=5a0836e14da71be448f79147deab394e

Database: shopcade (10)

wg_address, wg_admin, wg_cart, wg_categories, wg_cities, wg_items, wg_orders, wg_order_messages, wg_testimonials, wg_users

Field	Type	Null	Default	Comments	MIME
cart_id	int(10)	No			
cart_session	varchar(100)	No			
item_id	int(10)	No	0		
item_price	float	No	0		
item_name	varchar(100)	No			
item_quantity	int(2)	No	1		
item_total_price	float	No	0		
item_image	varchar(100)	No			
cart_status	varchar(10)	No			
order_id	varchar(10)	No			

Indexes:

Keyname	Type	Unique	Packed	Field	Cardinality	Collation	Null	Comment
PRIMARY	BTREE	Yes	No	cart_id	25	A		

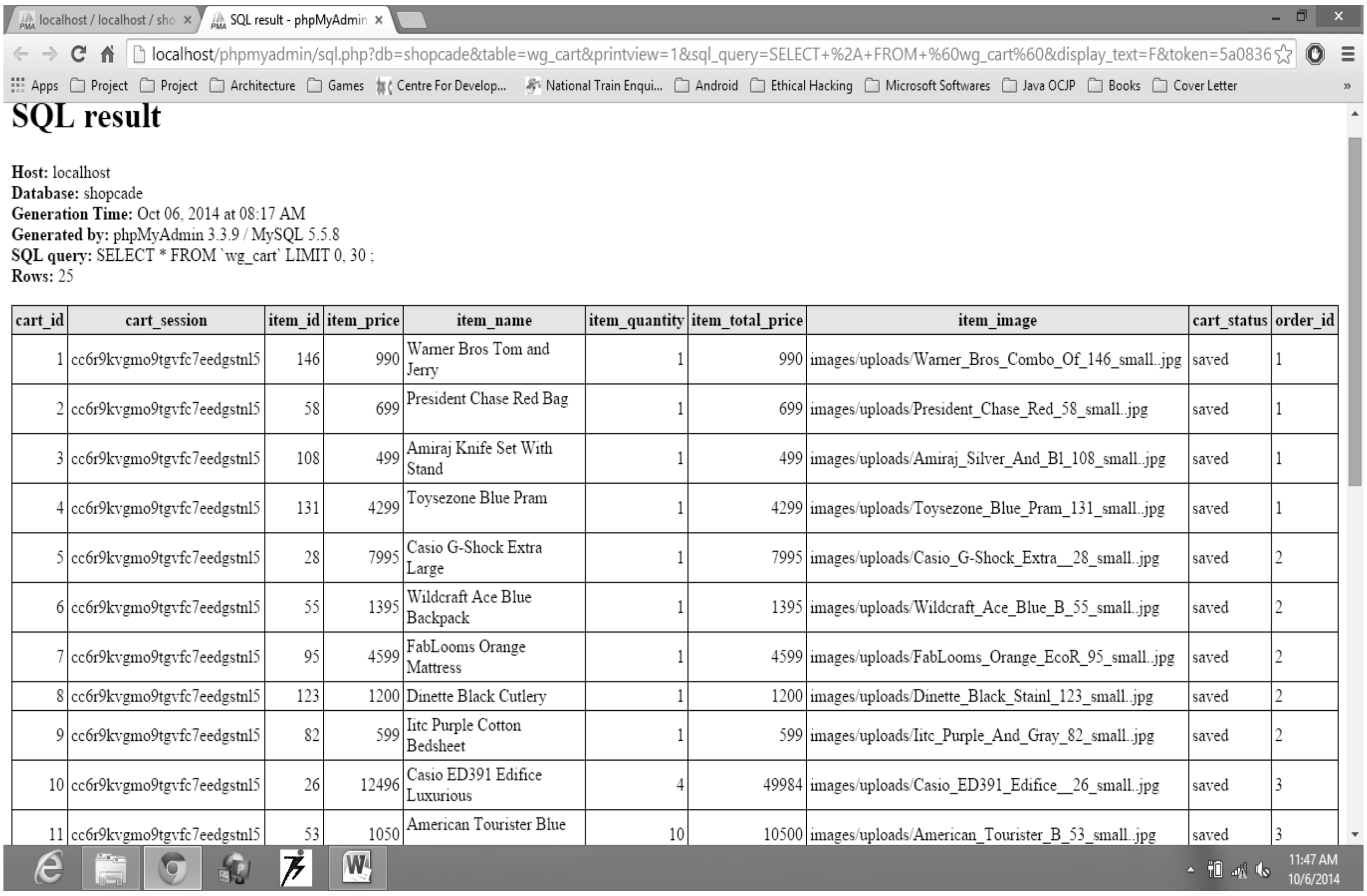
Space usage:

Type	Usage	
Data	4,032	B
Index	2,048	B
Overhead	136	B
Effective	5,944	B
Total	6,080	B

Row Statistics:

Statements	Value
Format	dynamic
Rows	25
Row length ø	155
Row size ø	243 B
Next Autoindex	27
Creation	Oct 03, 2014 at 06:14 PM
Last update	Oct 05, 2014 at 12:31 PM

Fig 29: Table wg_cart Structure View



localhost/phpmyadmin/sql.php?db=shopcade&table=wg_cart&printview=1&sql_query=SELECT+%2A+FROM+%60wg_cart%60&display_text=F&token=5a0836

SQL result

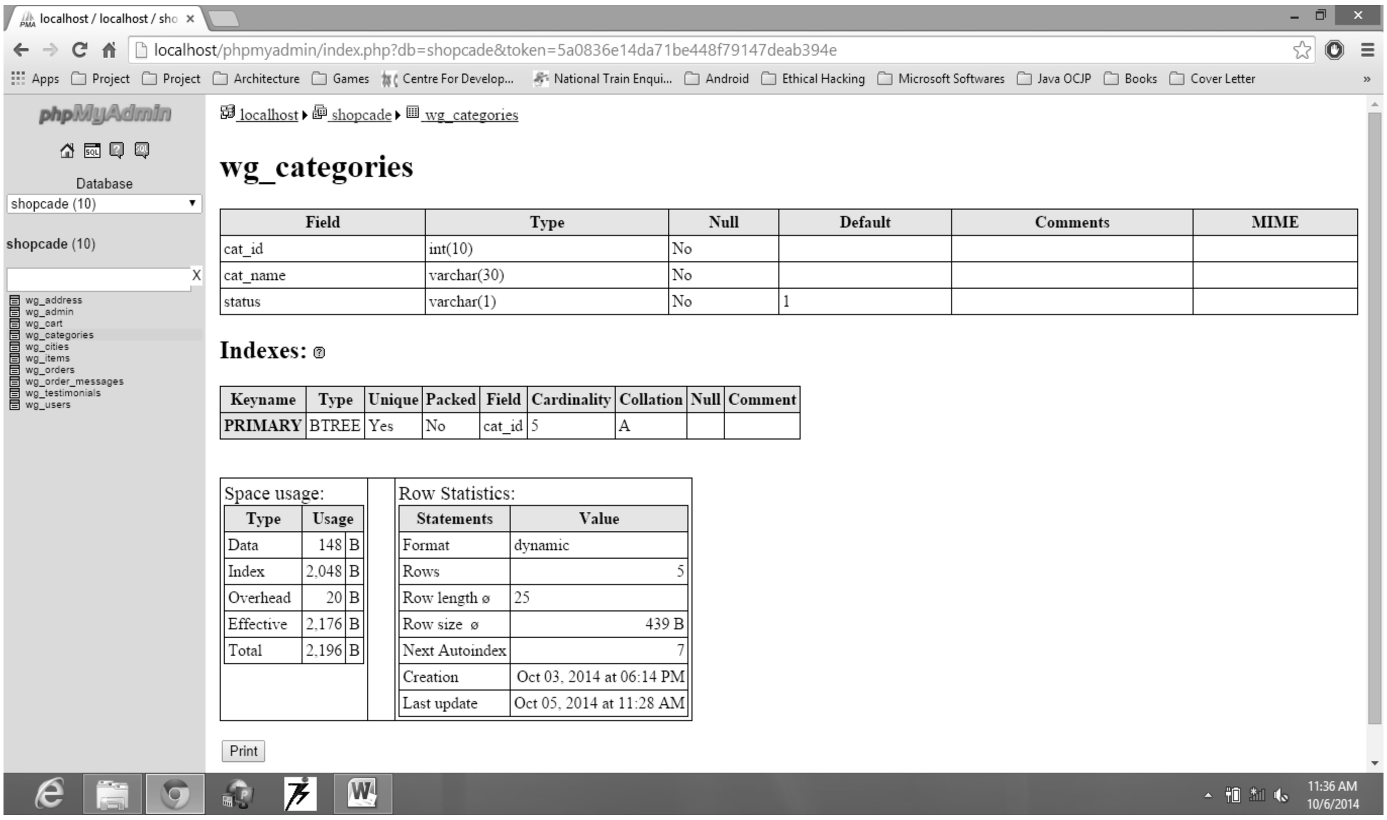
Host: localhost
Database: shopcade
Generation Time: Oct 06, 2014 at 08:17 AM
Generated by: phpMyAdmin 3.3.9 / MySQL 5.5.8
SQL query: SELECT * FROM `wg_cart` LIMIT 0, 30 ;
Rows: 25

cart_id	cart_session	item_id	item_price	item_name	item_quantity	item_total_price	item_image	cart_status	order_id
1	cc6r9kvgmo9tgvfc7eedgstnl5	146	990	Warner Bros Tom and Jerry	1	990	images/uploads/Warner_Bros_Combo_Of_146_small..jpg	saved	1
2	cc6r9kvgmo9tgvfc7eedgstnl5	58	699	President Chase Red Bag	1	699	images/uploads/President_Chase_Red_58_small..jpg	saved	1
3	cc6r9kvgmo9tgvfc7eedgstnl5	108	499	Amiraj Knife Set With Stand	1	499	images/uploads/Amiraj_Silver_And_Bl_108_small..jpg	saved	1
4	cc6r9kvgmo9tgvfc7eedgstnl5	131	4299	Toysezone Blue Pram	1	4299	images/uploads/Toysezone_Blue_Pram_131_small..jpg	saved	1
5	cc6r9kvgmo9tgvfc7eedgstnl5	28	7995	Casio G-Shock Extra Large	1	7995	images/uploads/Casio_G-Shock_Extra__28_small..jpg	saved	2
6	cc6r9kvgmo9tgvfc7eedgstnl5	55	1395	Wildcraft Ace Blue Backpack	1	1395	images/uploads/Wildcraft_Ace_Blue_B_55_small..jpg	saved	2
7	cc6r9kvgmo9tgvfc7eedgstnl5	95	4599	FabLooms Orange Mattress	1	4599	images/uploads/FabLooms_Orange_EcoR_95_small..jpg	saved	2
8	cc6r9kvgmo9tgvfc7eedgstnl5	123	1200	Dinette Black Cutlery	1	1200	images/uploads/Dinette_Black_Stainl_123_small..jpg	saved	2
9	cc6r9kvgmo9tgvfc7eedgstnl5	82	599	Iitc Purple Cotton Bedsheet	1	599	images/uploads/Iitc_Purple_And_Gray_82_small..jpg	saved	2
10	cc6r9kvgmo9tgvfc7eedgstnl5	26	12496	Casio ED391 Edifice Luxurious	4	49984	images/uploads/Casio_ED391_Edifice__26_small..jpg	saved	3
11	cc6r9kvgmo9tgvfc7eedgstnl5	53	1050	American Tourister Blue	10	10500	images/uploads/American_Tourister_B_53_small..jpg	saved	3

Fig 30: Table wg_cart Full Text View

4.3.4. <u>wg_categories: -</u>

It contains attribute like cat_id, cat_name and status. Its Index keyname is cat_id, index type is BTREE and key type is primary.



localhost/phpmyadmin/index.php?db=shopcade&token=5a0836e14da71be448f79147deab394e

localhost ▸ shopcade ▸ wg_categories

wg_categories

Field	Type	Null	Default	Comments	MIME
cat_id	int(10)	No			
cat_name	varchar(30)	No			
status	varchar(1)	No	1		

Indexes:

Keyname	Type	Unique	Packed	Field	Cardinality	Collation	Null	Comment
PRIMARY	BTREE	Yes	No	cat_id	5	A		

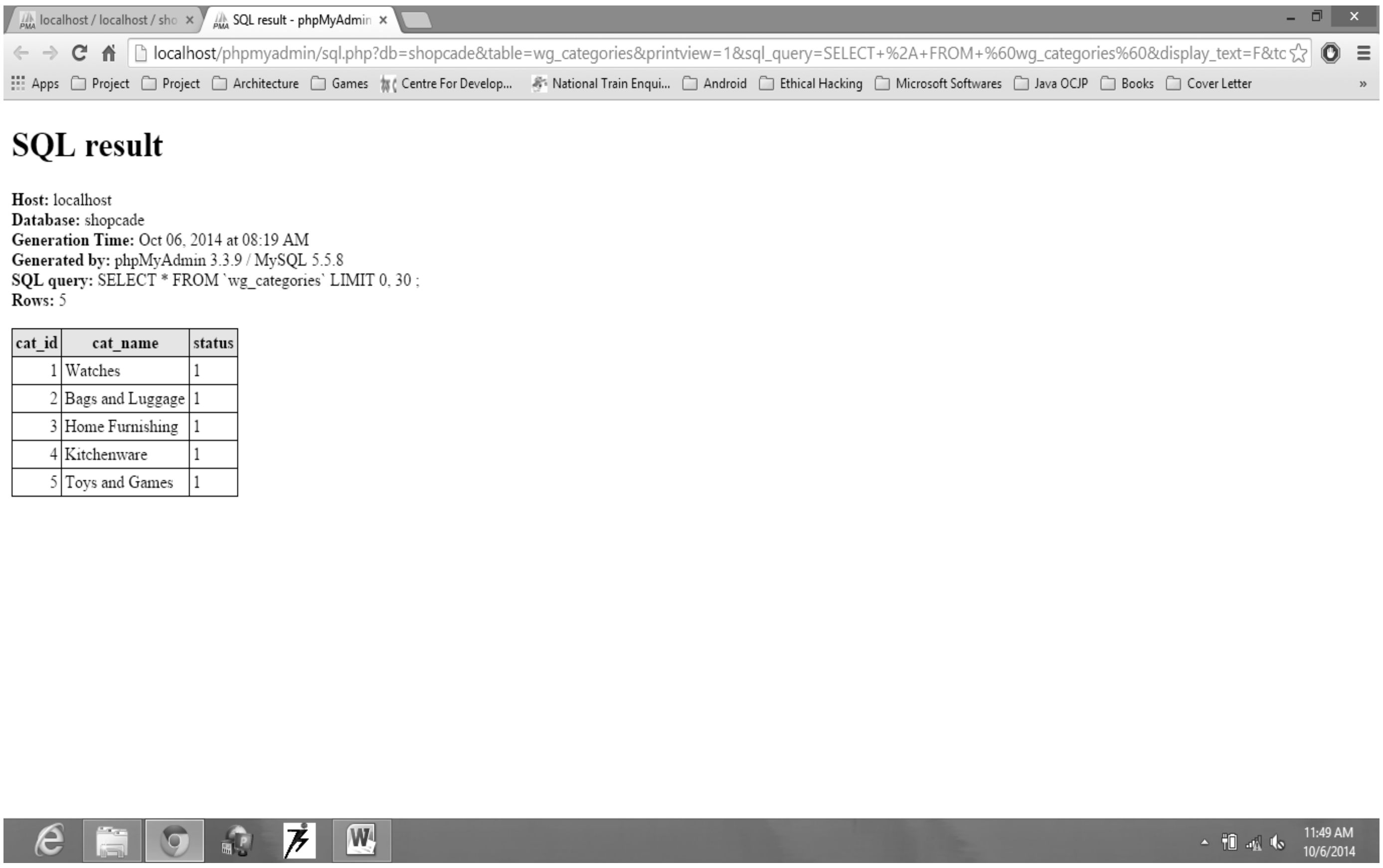
Space usage:

Type	Usage	
Data	148	B
Index	2,048	B
Overhead	20	B
Effective	2,176	B
Total	2,196	B

Row Statistics:

Statements	Value
Format	dynamic
Rows	5
Row length ø	25
Row size ø	439 B
Next Autoindex	7
Creation	Oct 03, 2014 at 06:14 PM
Last update	Oct 05, 2014 at 11:28 AM

Fig 31: Table wg_categories Structure View



localhost/phpmyadmin/sql.php?db=shopcade&table=wg_categories&printview=1&sql_query=SELECT+%2A+FROM+%60wg_categories%60&display_text=F&tc

SQL result

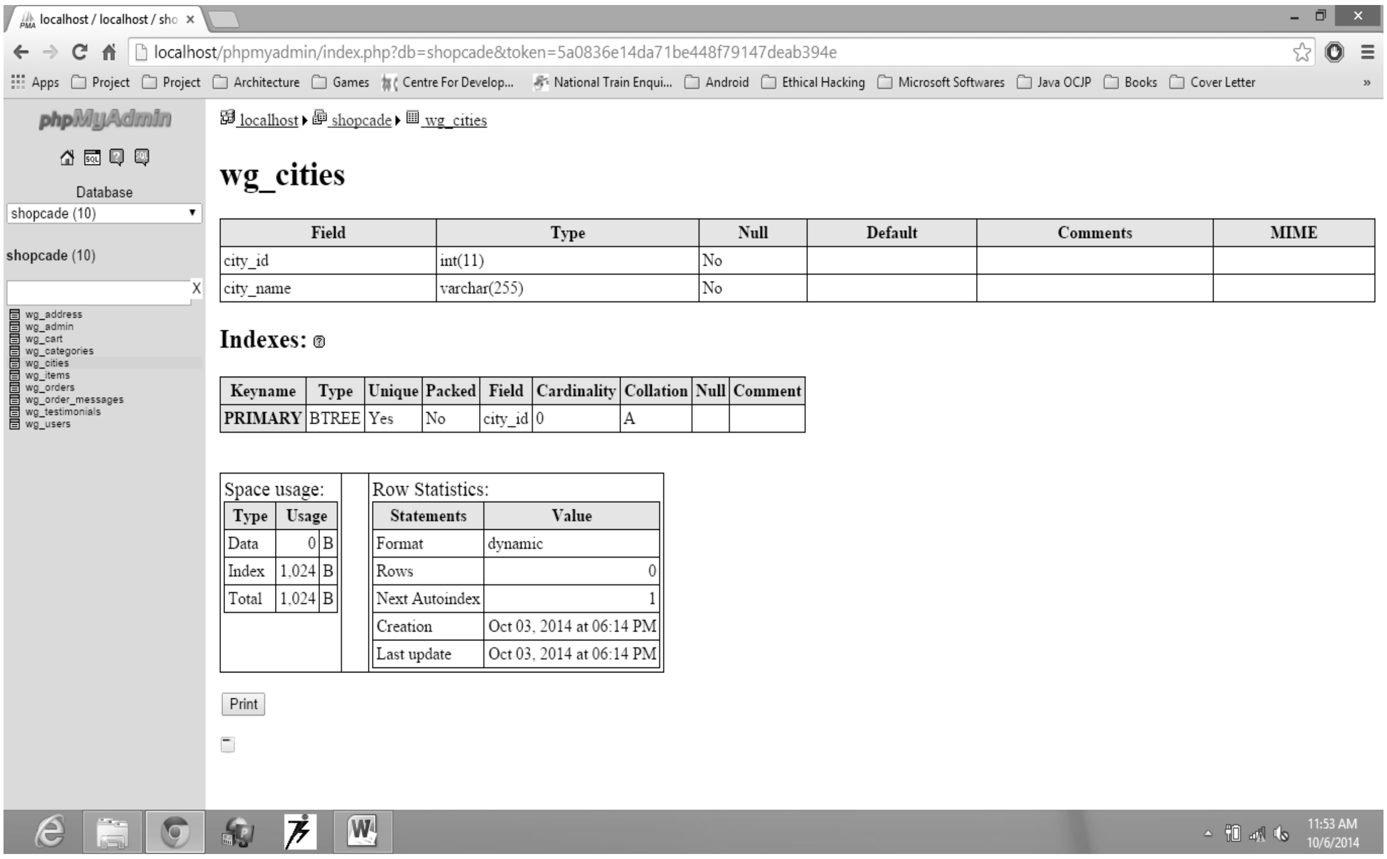
Host: localhost
Database: shopcade
Generation Time: Oct 06, 2014 at 08:19 AM
Generated by: phpMyAdmin 3.3.9 / MySQL 5.5.8
SQL query: SELECT * FROM `wg_categories` LIMIT 0, 30 ;
Rows: 5

cat_id	cat_name	status
1	Watches	1
2	Bags and Luggage	1
3	Home Furnishing	1
4	Kitchenware	1
5	Toys and Games	1

Fig 32: Table wg_categories Full Text View

4.3.5. wg_cities: -

It contains attribute like city_id and city_name. It index keyname is city_id, index type is BTREE and key type is primary.



localhost / shopcade / wg_cities

wg_cities

Field	Type	Null	Default	Comments	MIME
city_id	int(11)	No			
city_name	varchar(255)	No			

Indexes:

Keyname	Type	Unique	Packed	Field	Cardinality	Collation	Null	Comment
PRIMARY	BTREE	Yes	No	city_id	0	A		

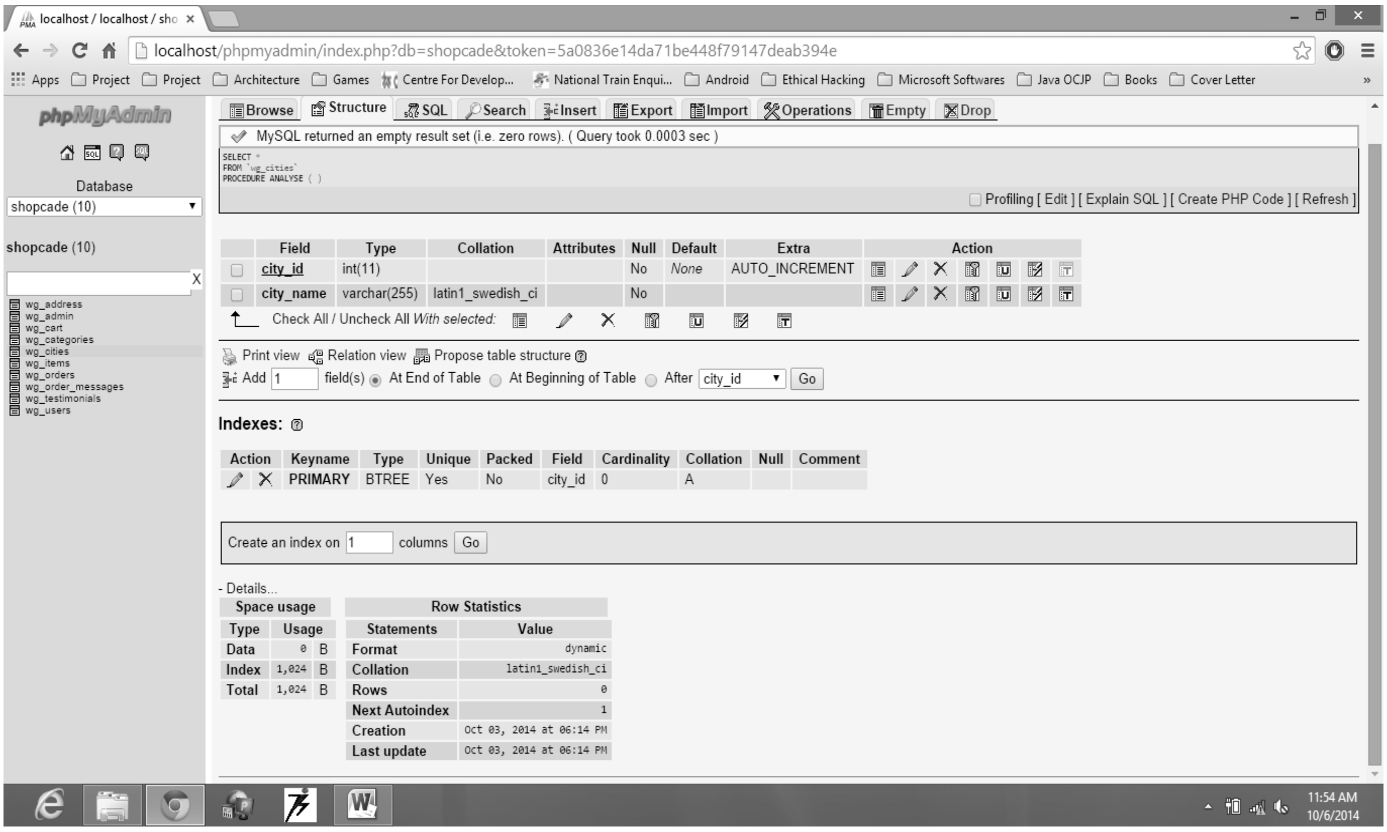
Space usage:

Type	Usage	
Data	0	B
Index	1,024	B
Total	1,024	B

Row Statistics:

Statements	Value
Format	dynamic
Rows	0
Next Autoindex	1
Creation	Oct 03, 2014 at 06:14 PM
Last update	Oct 03, 2014 at 06:14 PM

Fig 33: Table wg_cities Structure View



MySQL returned an empty result set (i.e. zero rows). (Query took 0.0003 sec)

```
SELECT *
FROM `wg_cities`
PROCEDURE ANALYSE ( )
```

Field	Type	Collation	Attributes	Null	Default	Extra
city_id	int(11)			No	None	AUTO_INCREMENT
city_name	varchar(255)	latin1_swedish_ci		No		

Indexes:

Keyname	Type	Unique	Packed	Field	Cardinality	Collation	Null	Comment
PRIMARY	BTREE	Yes	No	city_id	0	A		

Space usage

Type	Usage	
Data	0	B
Index	1,024	B
Total	1,024	B

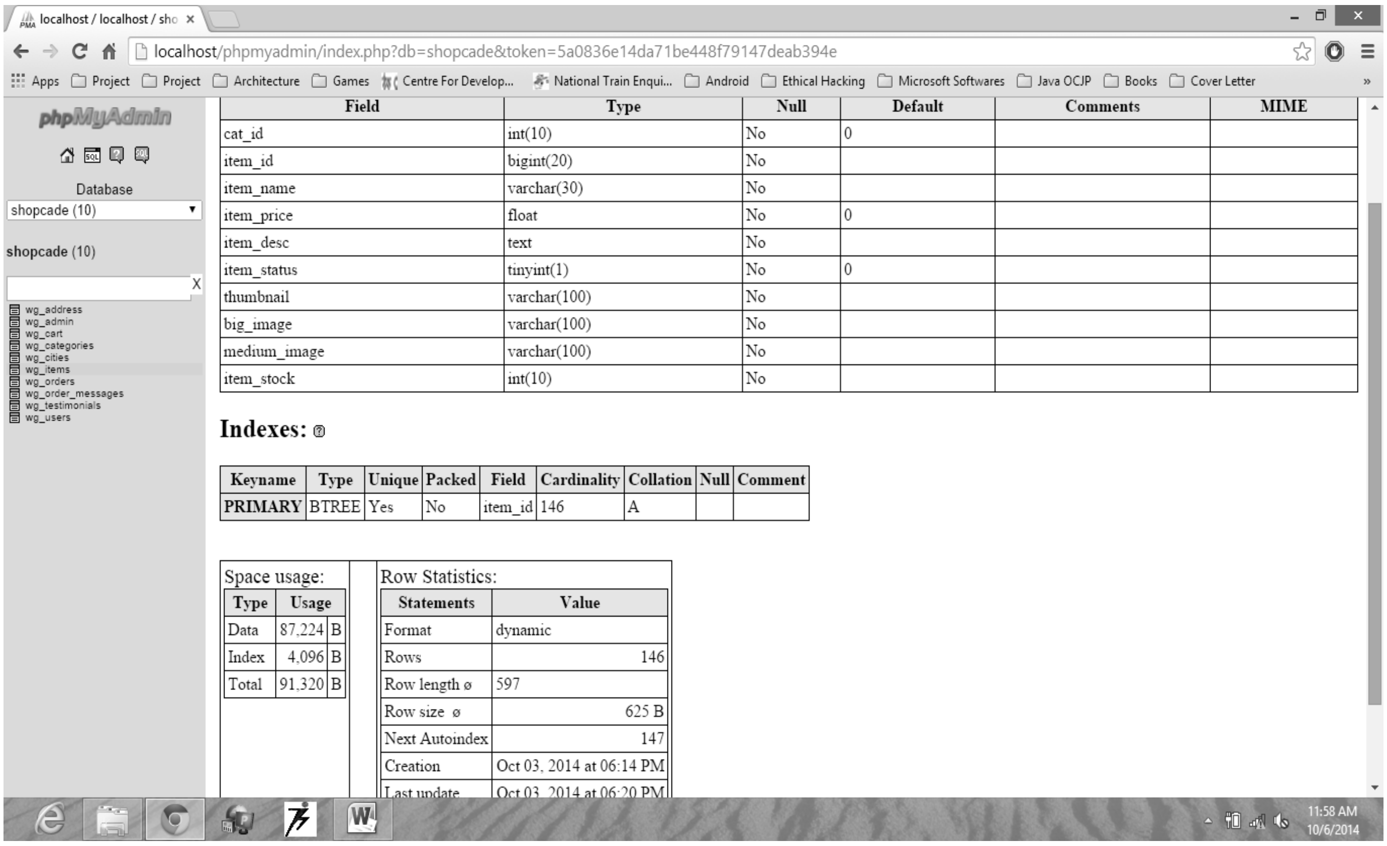
Row Statistics

Statements	Value
Format	dynamic
Collation	latin1_swedish_ci
Rows	0
Next Autoindex	1
Creation	Oct 03, 2014 at 06:14 PM
Last update	Oct 03, 2014 at 06:14 PM

Fig 34: Table wg_cities Full Text View

4.3.6. <u>wg_items: -</u>

It contains attribute like cat_id, item_id, item_name, item_price, item_desc, item_status, thumbnail, big_image, medium_image and item_stock.



Field	Type	Null	Default	Comments	MIME
cat_id	int(10)	No	0		
item_id	bigint(20)	No			
item_name	varchar(30)	No			
item_price	float	No	0		
item_desc	text	No			
item_status	tinyint(1)	No	0		
thumbnail	varchar(100)	No			
big_image	varchar(100)	No			
medium_image	varchar(100)	No			
item_stock	int(10)	No			

Indexes:

Keyname	Type	Unique	Packed	Field	Cardinality	Collation	Null	Comment
PRIMARY	BTREE	Yes	No	item_id	146	A		

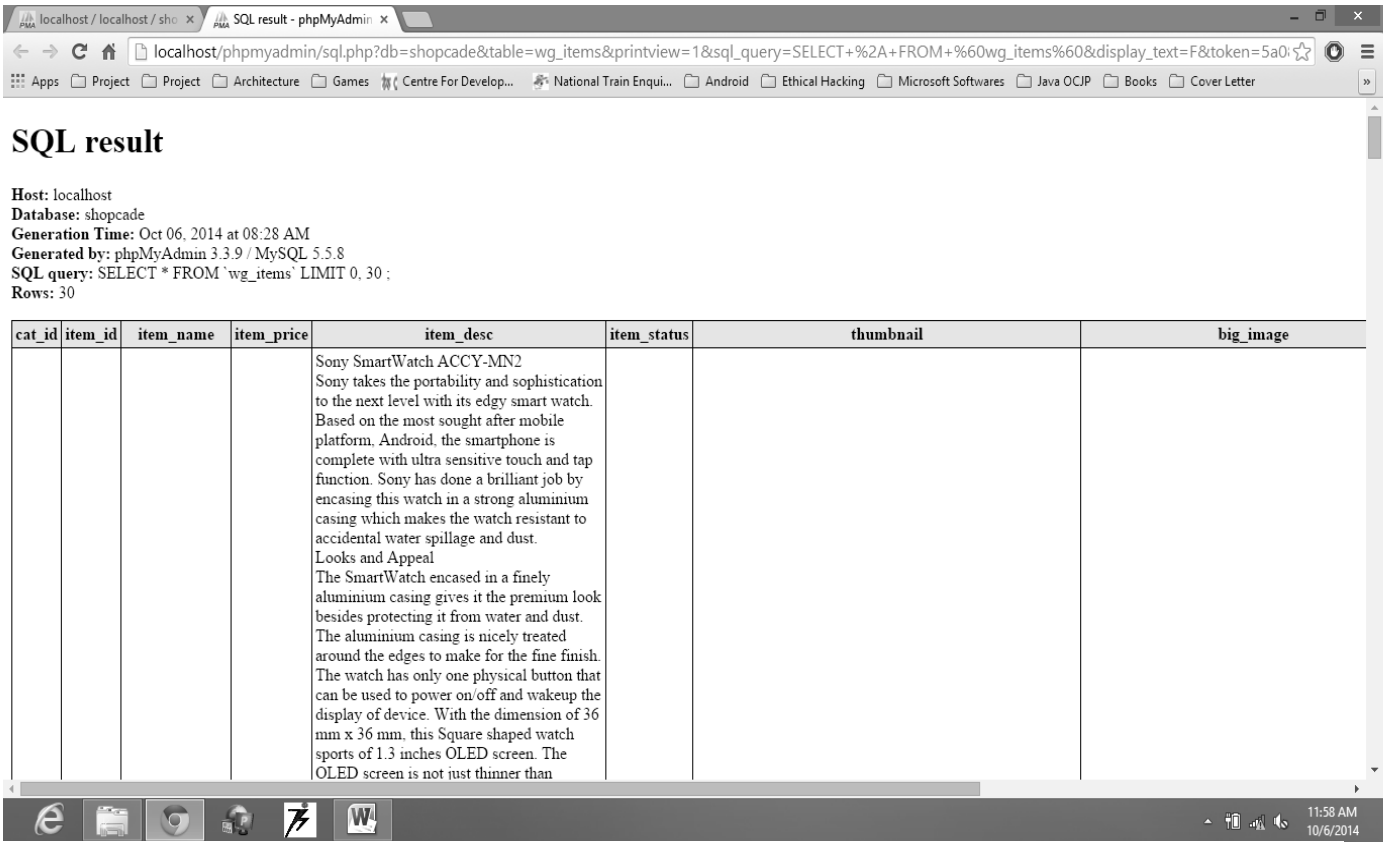
Space usage:

Type	Usage	
Data	87,224	B
Index	4,096	B
Total	91,320	B

Row Statistics:

Statements	Value
Format	dynamic
Rows	146
Row length ø	597
Row size ø	625 B
Next Autoindex	147
Creation	Oct 03, 2014 at 06:14 PM
Last update	Oct 03, 2014 at 06:20 PM

Fig 35: Table wg_items Structure View


SQL result - phpMyAdmin
localhost/phpmyadmin/sql.php?db=shopcade&table=wg_items&printview=1&sql_query=SELECT+%2A+FROM+%60wg_items%60&display_text=F&token=5a0

SQL result

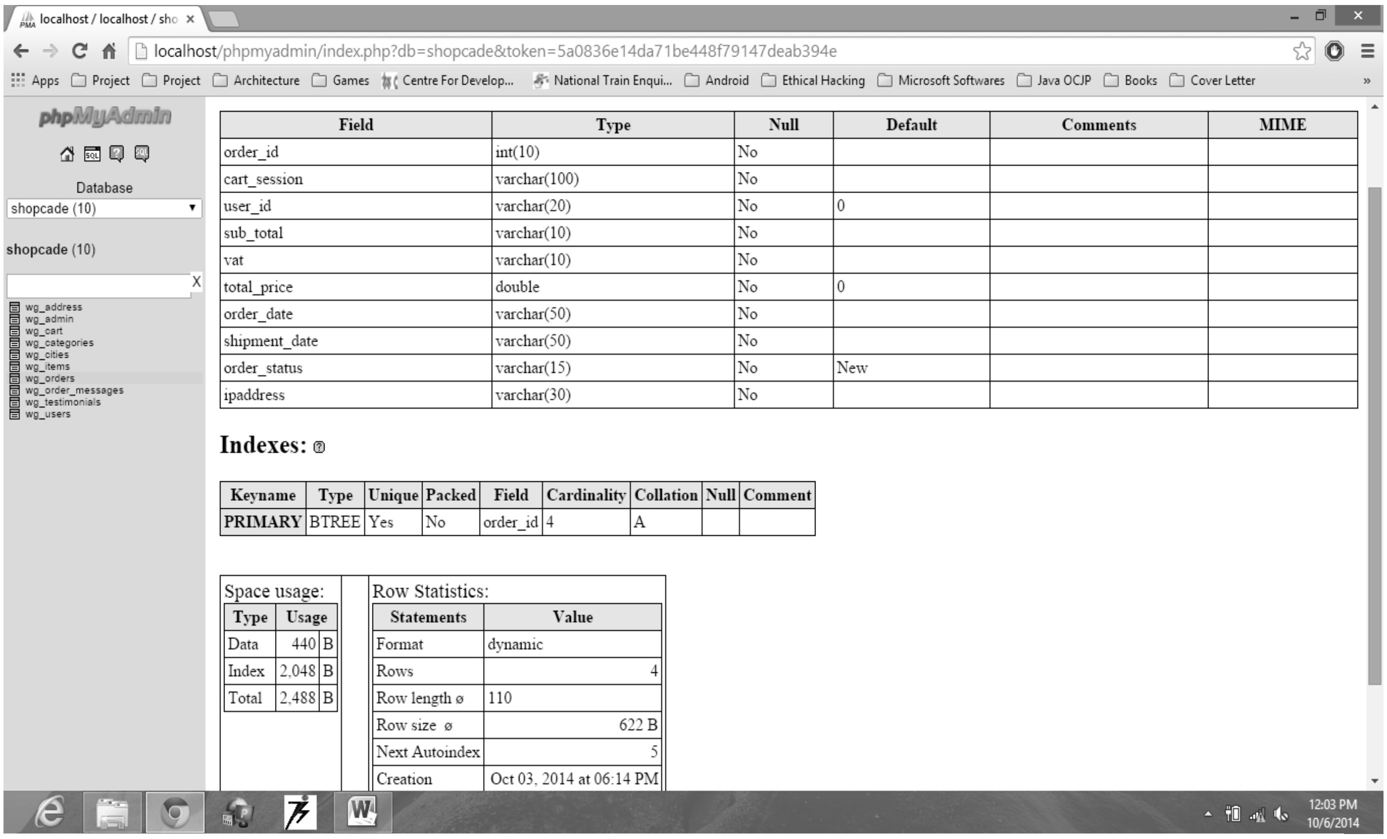
Host: localhost
Database: shopcade
Generation Time: Oct 06, 2014 at 08:28 AM
Generated by: phpMyAdmin 3.3.9 / MySQL 5.5.8
SQL query: SELECT * FROM `wg_items` LIMIT 0, 30 ;
Rows: 30

cat_id	item_id	item_name	item_price	item_desc	item_status	thumbnail	big_image
				Sony SmartWatch ACCY-MN2 Sony takes the portability and sophistication to the next level with its edgy smart watch. Based on the most sought after mobile platform, Android, the smartphone is complete with ultra sensitive touch and tap function. Sony has done a brilliant job by encasing this watch in a strong aluminium casing which makes the watch resistant to accidental water spillage and dust. Looks and Appeal The SmartWatch encased in a finely aluminium casing gives it the premium look besides protecting it from water and dust. The aluminium casing is nicely treated around the edges to make for the fine finish. The watch has only one physical button that can be used to power on/off and wakeup the display of device. With the dimension of 36 mm x 36 mm, this Square shaped watch sports of 1.3 inches OLED screen. The OLED screen is not just thinner than			

Fig 36: Table wg_items Full Text View

4.3.7. wg_order: -

It contains attribute like order_id, cart_session, user_id, sub_total, total_price, order_date, shipment_date, order_status, vat and ipaddress.


localhost / localhost / sho
localhost/phpmyadmin/index.php?db=shopcade&token=5a0836e14da71be448f79147deab394e
Apps | Project | Project | Architecture | Games | Centre For Develop... | National Train Enqui... | Android | Ethical Hacking | Microsoft Softwares | Java OCJP | Books | Cover Letter
phpMyAdmin
Database
shopcade (10)
shopcade (10)
wg_address
wg_admin
wg_cart
wg_categories
wg_cities
wg_items
wg_orders
wg_order_messages
wg_testimonials
wg_users
12:03 PM
10/6/2014

Field	Type	Null	Default	Comments	MIME
order_id	int(10)	No			
cart_session	varchar(100)	No			
user_id	varchar(20)	No	0		
sub_total	varchar(10)	No			
vat	varchar(10)	No			
total_price	double	No	0		
order_date	varchar(50)	No			
shipment_date	varchar(50)	No			
order_status	varchar(15)	No	New		
ipaddress	varchar(30)	No			

Indexes:

Keyname	Type	Unique	Packed	Field	Cardinality	Collation	Null	Comment
PRIMARY	BTREE	Yes	No	order_id	4	A		

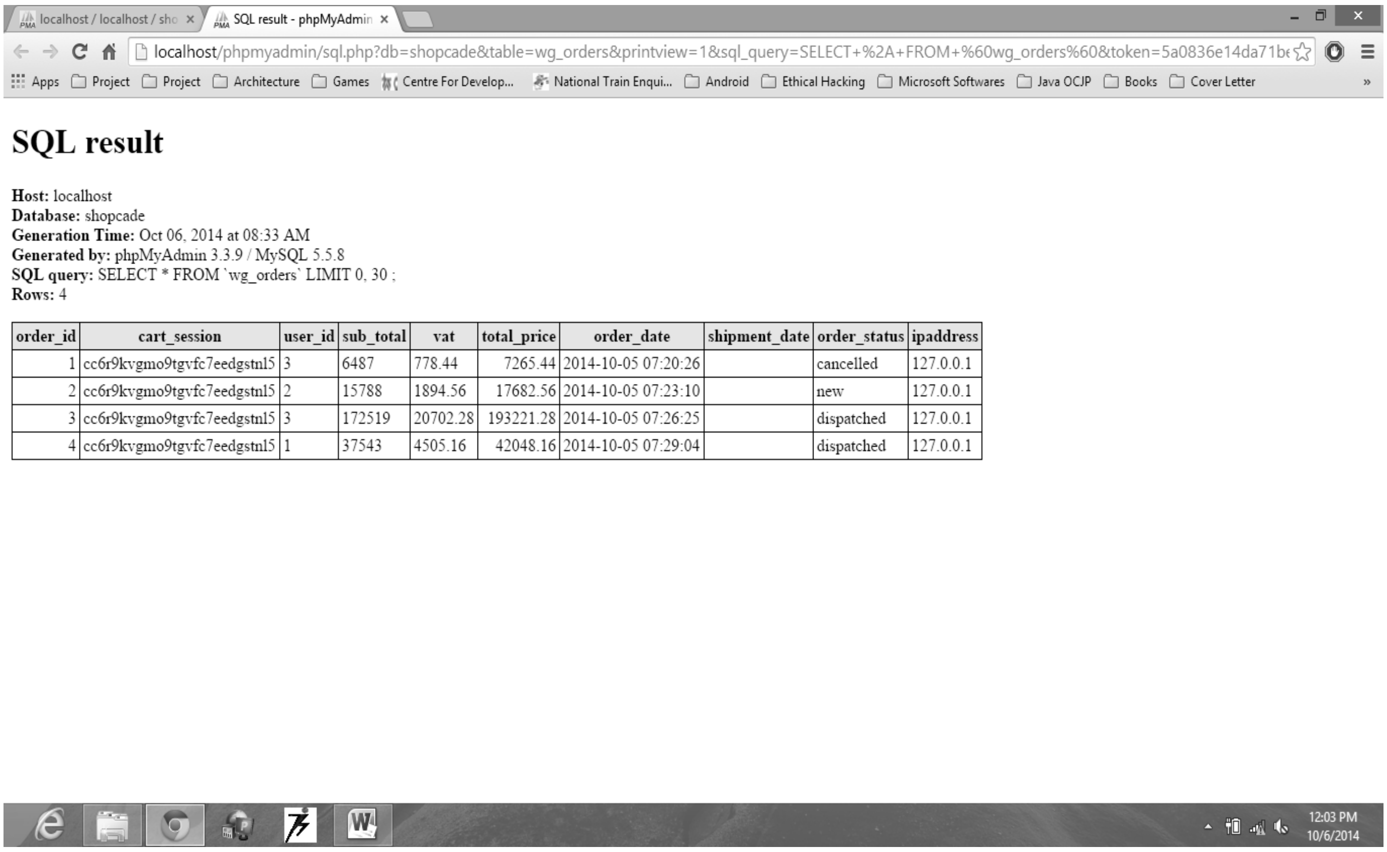
Space usage:

Type	Usage	
Data	440	B
Index	2,048	B
Total	2,488	B

Row Statistics:

Statements	Value
Format	dynamic
Rows	4
Row length ø	110
Row size ø	622 B
Next Autoindex	5
Creation	Oct 03, 2014 at 06:14 PM

Fig 37: Table wg_order Structure View


localhost / localhost / sho
SQL result - phpMyAdmin
localhost/phpmyadmin/sql.php?db=shopcade&table=wg_orders&printview=1&sql_query=SELECT+%2A+FROM+%60wg_orders%60&token=5a0836e14da71be
Apps | Project | Project | Architecture | Games | Centre For Develop... | National Train Enqui... | Android | Ethical Hacking | Microsoft Softwares | Java OCJP | Books | Cover Letter
12:03 PM
10/6/2014

SQL result

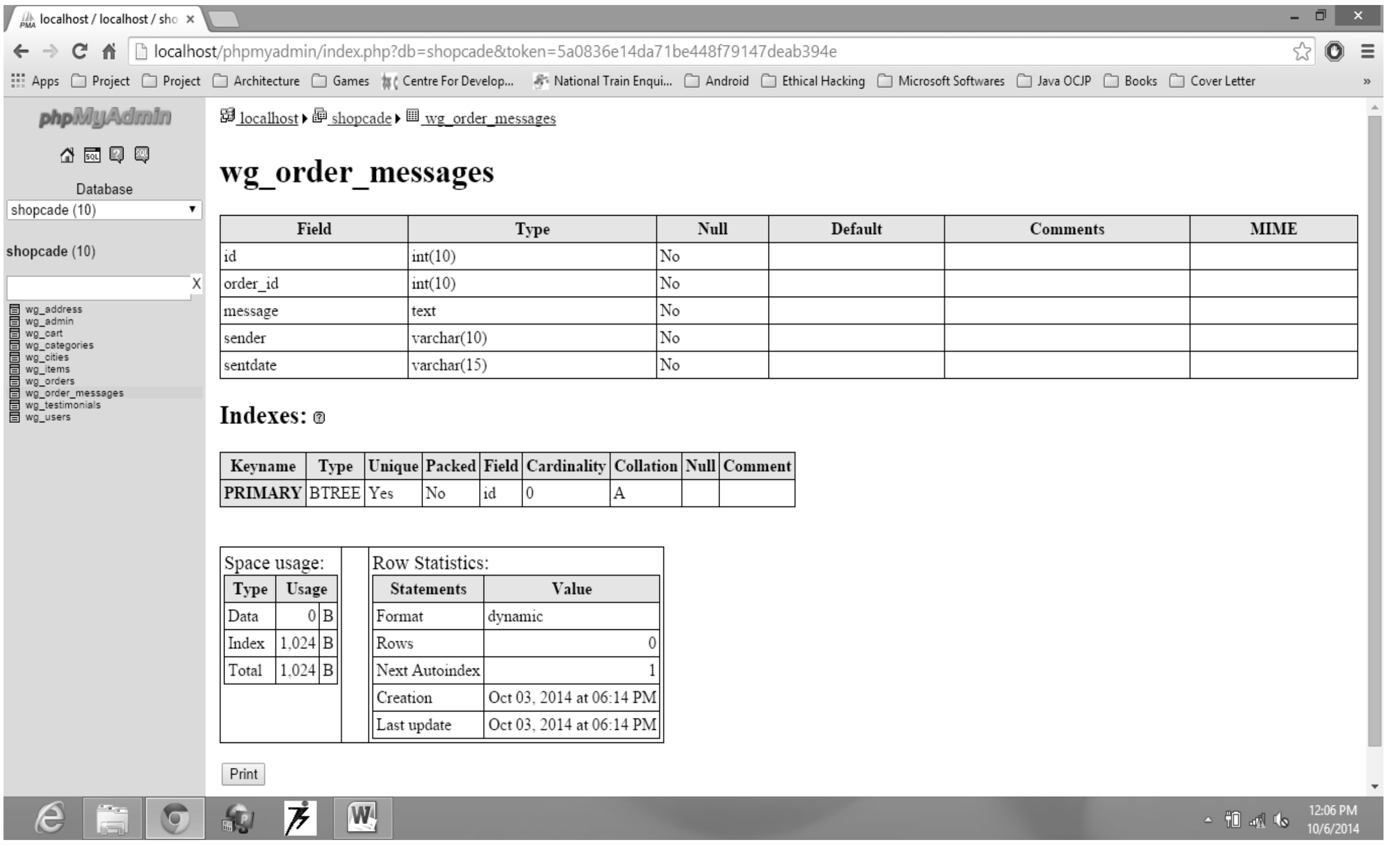
Host: localhost
Database: shopcade
Generation Time: Oct 06, 2014 at 08:33 AM
Generated by: phpMyAdmin 3.3.9 / MySQL 5.5.8
SQL query: SELECT * FROM `wg_orders` LIMIT 0, 30 ;
Rows: 4

order_id	cart_session	user_id	sub_total	vat	total_price	order_date	shipment_date	order_status	ipaddress
1	cc6r9kvgmo9tgvfc7eedgstnl5	3	6487	778.44	7265.44	2014-10-05 07:20:26		cancelled	127.0.0.1
2	cc6r9kvgmo9tgvfc7eedgstnl5	2	15788	1894.56	17682.56	2014-10-05 07:23:10		new	127.0.0.1
3	cc6r9kvgmo9tgvfc7eedgstnl5	3	172519	20702.28	193221.28	2014-10-05 07:26:25		dispatched	127.0.0.1
4	cc6r9kvgmo9tgvfc7eedgstnl5	1	37543	4505.16	42048.16	2014-10-05 07:29:04		dispatched	127.0.0.1

Fig 38: Table wg_order Full Text View

4.3.8. wg_order_messages: -

It contains attribute like id, order_id, message, sender, senddate. Its Index keyname is id, index type is BTREE and key type is primary.



localhost / shopcade / wg_order_messages

wg_order_messages

Field	Type	Null	Default	Comments	MIME
id	int(10)	No			
order_id	int(10)	No			
message	text	No			
sender	varchar(10)	No			
sentdate	varchar(15)	No			

Indexes:

Keyname	Type	Unique	Packed	Field	Cardinality	Collation	Null	Comment
PRIMARY	BTREE	Yes	No	id	0	A		

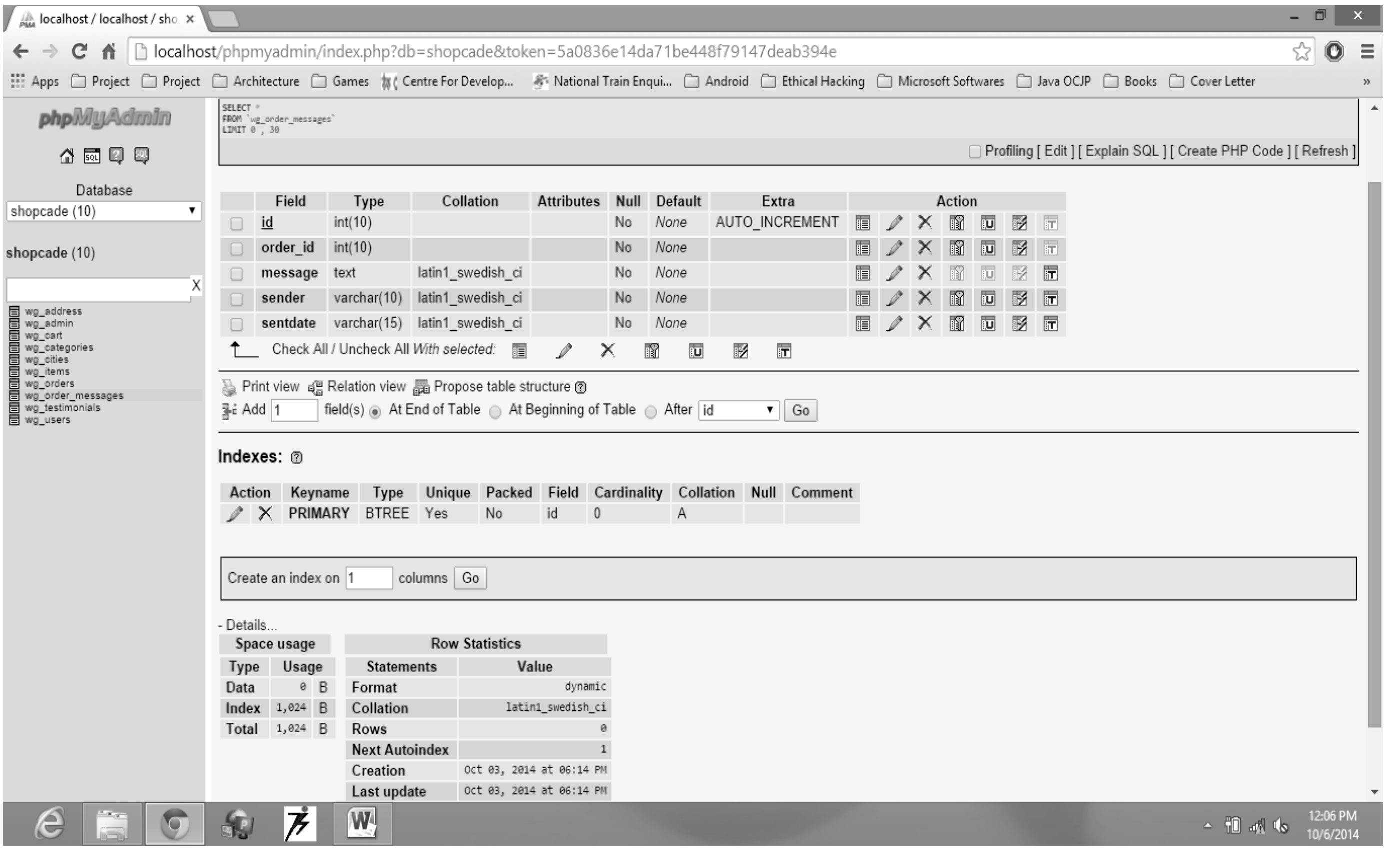
Space usage:

Type	Usage	
Data	0	B
Index	1,024	B
Total	1,024	B

Row Statistics:

Statements	Value
Format	dynamic
Rows	0
Next Autoindex	1
Creation	Oct 03, 2014 at 06:14 PM
Last update	Oct 03, 2014 at 06:14 PM

Fig 39: Table wg_order_messages Structure View



```
SELECT *
FROM `wg_order_messages`
LIMIT 0 , 30
```

	Field	Type	Collation	Attributes	Null	Default	Extra
☐	id	int(10)			No	None	AUTO_INCREMENT
☐	order_id	int(10)			No	None	
☐	message	text	latin1_swedish_ci		No	None	
☐	sender	varchar(10)	latin1_swedish_ci		No	None	
☐	sentdate	varchar(15)	latin1_swedish_ci		No	None	

Check All / Uncheck All *With selected:*

Add 1 field(s) [✓ At End of Table] At Beginning of Table After id Go

Indexes:

Keyname	Type	Unique	Packed	Field	Cardinality	Collation	Null	Comment
PRIMARY	BTREE	Yes	No	id	0	A		

Create an index on 1 columns Go

- Details...

Space usage

Type	Usage	
Data	0	B
Index	1,024	B
Total	1,024	B

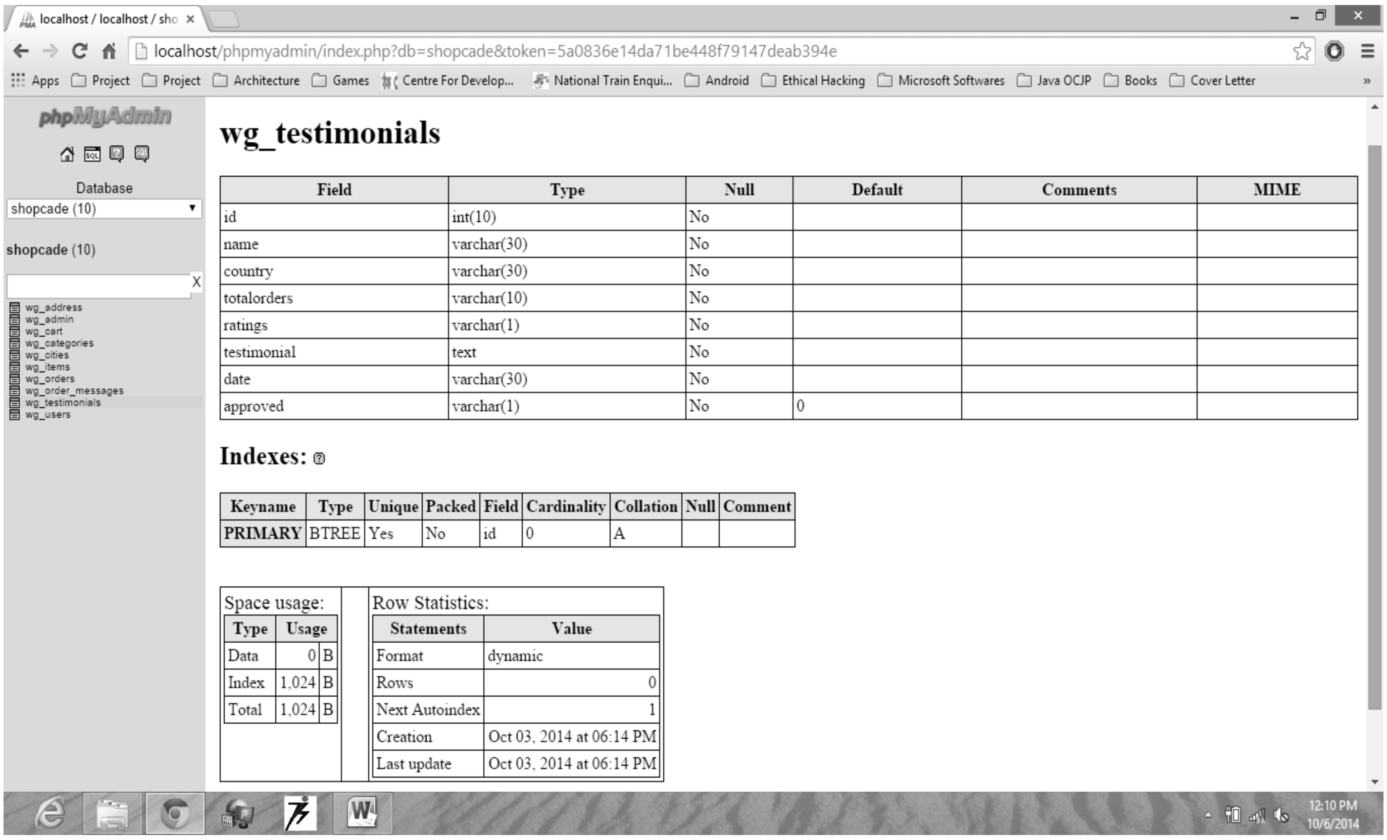
Row Statistics

Statements	Value
Format	dynamic
Collation	latin1_swedish_ci
Rows	0
Next Autoindex	1
Creation	Oct 03, 2014 at 06:14 PM
Last update	Oct 03, 2014 at 06:14 PM

Fig 40: Table wg_order_messages Full Text View

4.3.9. ***wg_testimonials: -***

It contains attribute like id, name, country, totalorders, ratings, testimonial, date and approved. Its Index keyname is id, index type is BTREE and key type is primary.



localhost/phpmyadmin/index.php?db=shopcade&token=5a0836e14da71be448f79147deab394e

Database: shopcade (10)

wg_address, wg_admin, wg_cart, wg_categories, wg_cities, wg_items, wg_orders, wg_order_messages, wg_testimonials, wg_users

wg_testimonials

Field	Type	Null	Default	Comments	MIME
id	int(10)	No			
name	varchar(30)	No			
country	varchar(30)	No			
totalorders	varchar(10)	No			
ratings	varchar(1)	No			
testimonial	text	No			
date	varchar(30)	No			
approved	varchar(1)	No	0		

Indexes:

Keyname	Type	Unique	Packed	Field	Cardinality	Collation	Null	Comment
PRIMARY	BTREE	Yes	No	id	0	A		

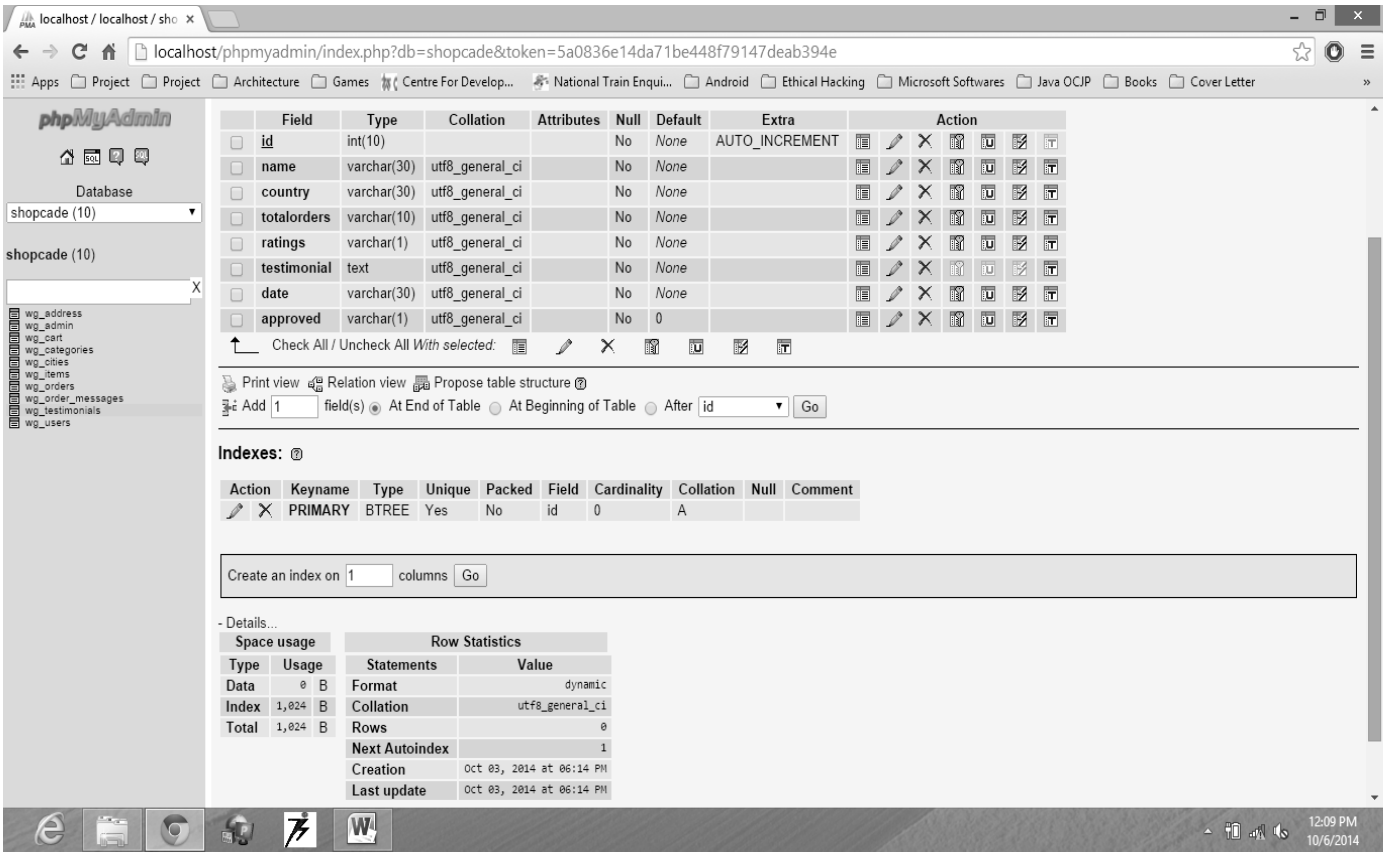
Space usage:

Type	Usage	
Data	0	B
Index	1,024	B
Total	1,024	B

Row Statistics:

Statements	Value
Format	dynamic
Rows	0
Next Autoindex	1
Creation	Oct 03, 2014 at 06:14 PM
Last update	Oct 03, 2014 at 06:14 PM

Fig 41: Table wg_testimonials Structure View



localhost/phpmyadmin/index.php?db=shopcade&token=5a0836e14da71be448f79147deab394e

Database: shopcade (10)

Field	Type	Collation	Attributes	Null	Default	Extra
id	int(10)			No	None	AUTO_INCREMENT
name	varchar(30)	utf8_general_ci		No	None	
country	varchar(30)	utf8_general_ci		No	None	
totalorders	varchar(10)	utf8_general_ci		No	None	
ratings	varchar(1)	utf8_general_ci		No	None	
testimonial	text	utf8_general_ci		No	None	
date	varchar(30)	utf8_general_ci		No	None	
approved	varchar(1)	utf8_general_ci		No	0	

Check All / Uncheck All *With selected:*

Print view Relation view Propose table structure

Add 1 field(s) At End of Table At Beginning of Table After id Go

Indexes:

Keyname	Type	Unique	Packed	Field	Cardinality	Collation	Null	Comment
PRIMARY	BTREE	Yes	No	id	0	A		

Create an index on 1 columns Go

- Details...

Space usage

Type	Usage	
Data	0	B
Index	1,024	B
Total	1,024	B

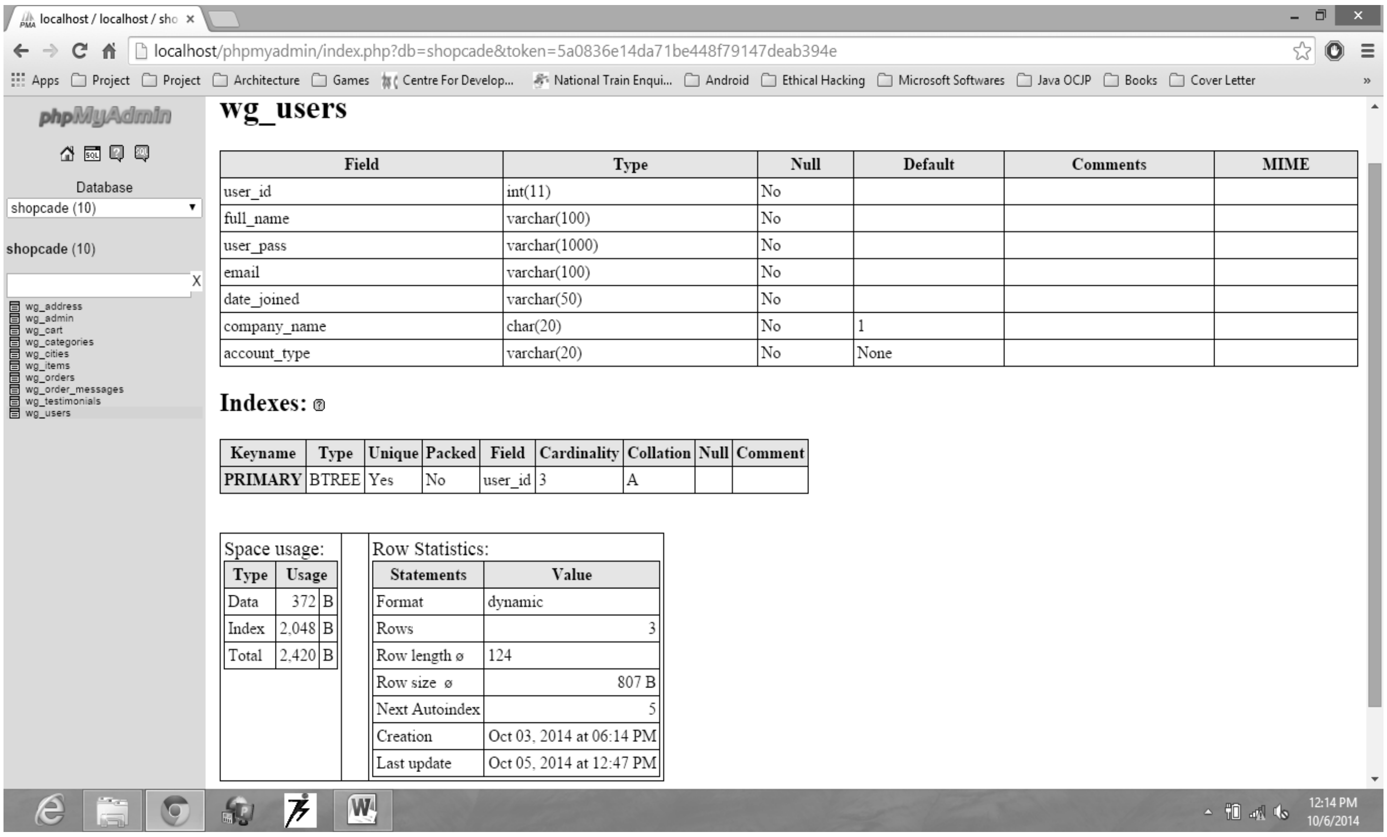
Row Statistics

Statements	Value
Format	dynamic
Collation	utf8_general_ci
Rows	0
Next Autoindex	1
Creation	Oct 03, 2014 at 06:14 PM
Last update	Oct 03, 2014 at 06:14 PM

Fig 42: Table wg_testimonials Full Text View

4.3.10. wg_users: -

It contains attribute like user_id, full_name, user_pass, email, date_joined, company_name and account_type. In this it is indexes on the bases of user_id and indexes type is BTREE.



localhost/phpmyadmin/index.php?db=shopcade&token=5a0836e14da71be448f79147deab394e

Database: shopcade (10)

wg_address, wg_admin, wg_cart, wg_categories, wg_cities, wg_items, wg_orders, wg_order_messages, wg_testimonials, wg_users

wg_users

Field	Type	Null	Default	Comments	MIME
user_id	int(11)	No			
full_name	varchar(100)	No			
user_pass	varchar(1000)	No			
email	varchar(100)	No			
date_joined	varchar(50)	No			
company_name	char(20)	No	1		
account_type	varchar(20)	No	None		

Indexes:

Keyname	Type	Unique	Packed	Field	Cardinality	Collation	Null	Comment
PRIMARY	BTREE	Yes	No	user_id	3	A		

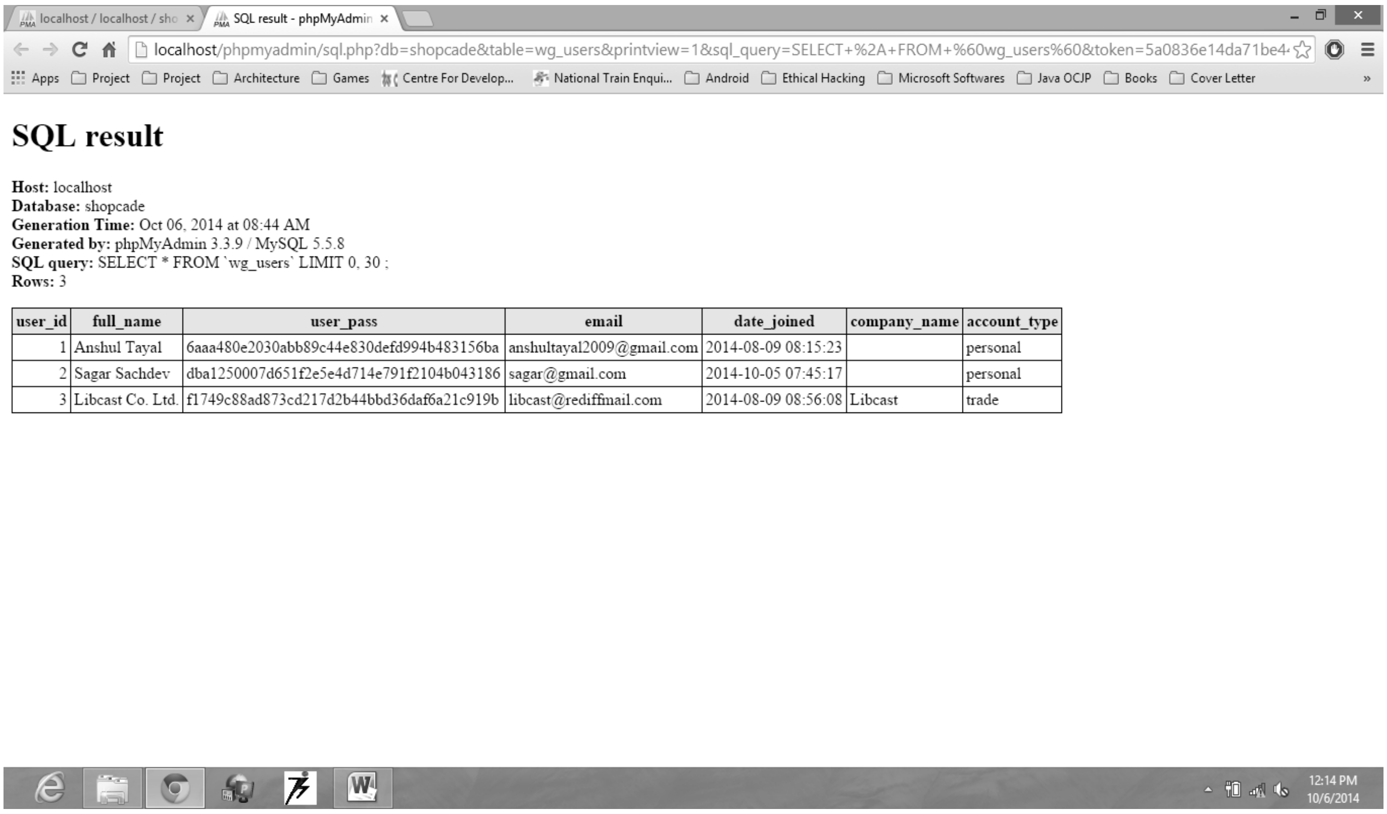
Space usage:

Type	Usage	
Data	372	B
Index	2,048	B
Total	2,420	B

Row Statistics:

Statements	Value
Format	dynamic
Rows	3
Row length ø	124
Row size ø	807 B
Next Autoindex	5
Creation	Oct 03, 2014 at 06:14 PM
Last update	Oct 05, 2014 at 12:47 PM

Fig 43: Table wg_users Structure View



localhost/phpmyadmin/sql.php?db=shopcade&table=wg_users&printview=1&sql_query=SELECT+%2A+FROM+%60wg_users%60&token=5a0836e14da71be4

SQL result

Host: localhost
Database: shopcade
Generation Time: Oct 06, 2014 at 08:44 AM
Generated by: phpMyAdmin 3.3.9 / MySQL 5.5.8
SQL query: SELECT * FROM `wg_users` LIMIT 0, 30 ;
Rows: 3

user_id	full_name	user_pass	email	date_joined	company_name	account_type
1	Anshul Tayal	6aaa480e2030abb89c44e830defd994b483156ba	anshultayal2009@gmail.com	2014-08-09 08:15:23		personal
2	Sagar Sachdev	dba1250007d651f2e5e4d714e791f2104b043186	sagar@gmail.com	2014-10-05 07:45:17		personal
3	Libcast Co. Ltd.	f1749c88ad873cd217d2b44bbd36daf6a21c919b	libcast@rediffmail.com	2014-08-09 08:56:08	Libcast	trade

Fig 44: Table wg_users Full Text View

4.4. InnoDB Table and Index Structures[16]

4.4.1. Role of the .frm File

MySQL stores its data dictionary information for tables in .frm files in database directories. This is true for all MySQL storage engines, but every **InnoDB** table also has its own entry in the **InnoDB** internal data dictionary inside the table space. When MySQL drops a table or a database, it has to delete one or more .frm files as well as the corresponding entries inside the **InnoDB** data dictionary. Consequently, you cannot move **InnoDB** tables between databases simply by moving the .frm files.

4.5. The MyISAM Storage Engine[17]

MyISAM is the default storage engine. It is based on the older (and no longer available) ISAM storage engine but has many useful extensions.
Each MyISAM table is stored on disk in three files. The files have names that begin with the table name and have an extension to indicate the file type. An .frm file stores the table format. The data file has an .MYD (MYData) extension. The index file has an .MYI (MYIndex) extension.

4.5.1. Space Needed for Keys

MyISAM tables use B-tree indexes. You can roughly calculate the size for the index file as (key_length+4)/0.67, summed over all keys. This is for the worst case when all keys are inserted in sorted order and the table doesn't have any compressed keys. String indexes are space compressed. If the first index part is a string, it is also prefix compressed. Space compression makes the index file smaller than the worst-case figure if a string column has a lot of trailing space or is a varchar column that is not always used to the full length. Prefix compression is used on keys that start with a string. Prefix compression helps if there are many strings with an identical prefix.
In MyISAM tables, you can also prefix compress numbers by specifying the PACK_KEYS=1 table option when you create the table. Numbers are stored with the high byte first, so this helps when you have many integer keys that have an identical prefix.

4.6. BTREE Algorithms[18]

4.6.1. Search

Searching is similar to searching a binary search tree. Starting at the root, the tree is recursively traversed from top to bottom. At each level, the search chooses the child pointer (subtree) whose separation values are on either side of the search value.
Binary search is typically (but not necessarily) used within nodes to find the separation values and child tree of interest.

4.6.2. Insertion

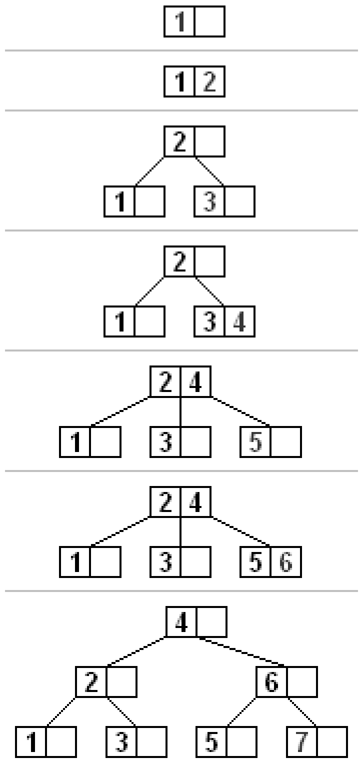


Fig 45: A B Tree insertion example with each iteration. The nodes of this B tree have at most 3 children (Knuth order 3).

All insertions start at a leaf node. To insert a new element, search the tree to find the leaf node where the new element should be added. Insert the new element into that node with the following steps:

1. If the node contains fewer than the maximum legal number of elements, then there is room for the new element. Insert the new element in the node, keeping the node's elements ordered.
2. Otherwise the node is full, evenly split it into two nodes so:
    1. A single median is chosen from among the leaf's elements and the new element.
    2. Values less than the median are put in the new left node and values greater than the median are put in the new right node, with the median acting as a separation value.

3. The separation value is inserted in the node's parent, which may cause it to be split, and so on. If the node has no parent (i.e., the node was the root), create a new root above this node (increasing the height of the tree).

If the splitting goes all the way up to the root, it creates a new root with a single separator value and two children, which is why the lower bound on the size of internal nodes does not apply to the root. The maximum number of elements per node is U−1. When a node is split, one element moves to the parent, but one element is added. So, it must be possible to divide the maximum number U−1 of elements into two legal nodes. If this number is odd, then U=2L and one of the new nodes contains (U−2)/2 = L−1 elements, and hence is a legal node, and the other contains one more element, and hence it is legal too. If U−1 is even, then U=2L−1, so there are 2L−2 elements in the node. Half of this number is L−1, which is the minimum number of elements allowed per node.
An improved algorithm (Mond & Raz 1985) supports a single pass down the tree from the root to the node where the insertion will take place, splitting any full nodes encountered on the way. This prevents the need to recall the parent nodes into memory, which may be expensive if the nodes are on secondary storage. However, to use this improved algorithm, we must be able to send one element to the parent and split the remaining U−2 elements into two legal nodes, without adding a new element. This requires U = 2L rather than U = 2L−1, which accounts for why some textbooks impose this requirement in defining B-trees.

4.6.3. <u>Deletion</u>

There are two popular strategies for deletion from a B-tree.
1. Locate and delete the item, then restructure the tree to regain its invariants, OR
2. Do a single pass down the tree, but before entering (visiting) a node, restructure the tree so that once the key to be deleted is encountered, it can be deleted without triggering the need for any further restructuring

The algorithm below uses the former strategy.
There are two special cases to consider when deleting an element:
1. The element in an internal node is a separator for its child nodes
2. Deleting an element may put its node under the minimum number of elements and children

The procedures for these cases are in order below.
Deletion from a leaf node[edit]
1. Search for the value to delete.
2. If the value is in a leaf node, simply delete it from the node.
3. If underflow happens, rebalance the tree as described in section "Rebalancing after deletion" below.

Deletion from an internal node[edit]
Each element in an internal node acts as a separation value for two subtrees, therefore we need to find a replacement for separation. Note that the largest element in the left subtree is still less than the separator. Likewise, the smallest element in the right subtree is still greater than the separator. Both of those elements are in leaf nodes, and either one can be the new separator for the two sub trees. Algorithmically described below:

1. Choose a new separator (either the largest element in the left subtree or the smallest element in the right subtree), remove it from the leaf node it is in, and replace the element to be deleted with the new separator.
2. The previous step deleted an element (the new separator) from a leaf node. If that leaf node is now deficient (has fewer than the required number of nodes), then rebalance the tree starting from the leaf node.

4.6.4. ***<u>Rebalancing after deletion</u>***

Rebalancing starts from a leaf and proceeds toward the root until the tree is balanced. If deleting an element from a node has brought it under the minimum size, then some elements must be redistributed to bring all nodes up to the minimum. Usually, the redistribution involves moving an element from a sibling node that has more than the minimum number of nodes. That redistribution operation is called a rotation. If no sibling can spare a node, then the deficient node must be merged with a sibling. The merge causes the parent to lose a separator element, so the parent may become deficient and need rebalancing. The merging and rebalancing may continue all the way to the root. Since the minimum element count doesn't apply to the root, making the root be the only deficient node is not a problem. The algorithm to rebalance the tree is as follows:[citation needed]

- If the deficient node's right sibling exists and has more than the minimum number of elements, then rotate left
    1. Copy the separator from the parent to the end of the deficient node (the separator moves down; the deficient node now has the minimum number of elements)
    2. Replace the separator in the parent with the first element of the right sibling (right sibling loses one node but still has at least the minimum number of elements)
    3. The tree is now balanced
- Otherwise, if the deficient node's left sibling exists and has more than the minimum number of elements, then rotate right
    1. Copy the separator from the parent to the start of the deficient node (the separator moves down; deficient node now has the minimum number of elements)
    2. Replace the separator in the parent with the last element of the left sibling (left sibling loses one node but still has at least the minimum number of elements)
    3. The tree is now balanced
- Otherwise, if both immediate siblings have only the minimum number of elements, then merge with a sibling sandwiching their separator taken off from their parent
    1. Copy the separator to the end of the left node (the left node may be the deficient node or it may be the sibling with the minimum number of elements)
    2. Move all elements from the right node to the left node (the left node now has the maximum number of elements, and the right node – empty)
    3. Remove the separator from the parent along with its empty right child (the parent loses an element)

- If the parent is the root and now has no elements, then free it and make the merged node the new root (tree becomes shallower)
- Otherwise, if the parent has fewer than the required number of elements, then rebalance the parent

4.7. SHA-1 Code

```
<? php if (! defined('BASEPATH')) exit('No direct script access allowed');
/**
 * SHA1 Encoding Class
 *
 * Purpose: Provides 160 bit hashing using The Secure Hash Algorithm
 * developed at the National Institute of Standards and Technology. The 40
 * character SHA1 message hash is computationally infeasible to crack.
 *
 * This class is a fallback for servers that are not running PHP greater than
 * 4.3, or do not have the MHASH library.
 *
 *
 */
class CI_SHA1 {
        public function __construct()
        {
                log_message('debug', "SHA1 Class Initialized");
        }
        /**
         * Generate the Hash
         *
         * @access   public
         * @param    string
         * @return   string
         */
        function generate($str)
        {
                $n = ((strlen($str) + 8) >> 6) + 1;
                for ($i = 0; $i < $n * 16; $i++)
                {
                        $x[$i] = 0;
                }
                for ($i = 0; $i < strlen($str); $i++)
                {
                        $x[$i >> 2] |= ord(substr($str, $i, 1)) << (24 – ($i % 4) * 8);
                }
                $x[$i >> 2] |= 0x80 << (24 – ($i % 4) * 8);
                $x[$n * 16 – 1] = strlen($str) * 8;
                $a =  1732584193;
                $b = -271733879;
                $c = -1732584194;
                $d =  271733878;
                $e = -1009589776;
                for ($i = 0; $i < count($x); $i += 16)
                {
```

```
                $olda = $a;
                $oldb = $b;
                $oldc = $c;
                $oldd = $d;
                $olde = $e;
                for ($j = 0; $j < 80; $j++)
                {
                    if ($j < 16)
                    {
                        $w[$j] = $x[$i + $j];
                    }
                    else
                    {
$w[$j] = $this->_rol($w[$j – 3] ^ $w[$j – 8] ^ $w[$j – 14] ^ $w[$j – 16], 1);
                    }
$t  =  $this->_safe_add($this->_safe_add($this->_rol($a,  5),  $this->_ft($j,  $b,  $c,  $d)),
$this->_safe_add($this->_safe_add($e, $w[$j]), $this->_kt($j)));
                    $e = $d;
                    $d = $c;
                    $c = $this->_rol($b, 30);
                    $b = $a;
                    $a = $t;
                }
                $a = $this->_safe_add($a, $olda);
                $b = $this->_safe_add($b, $oldb);
                $c = $this->_safe_add($c, $oldc);
                $d = $this->_safe_add($d, $oldd);
                $e = $this->_safe_add($e, $olde);
            }
return $this->_hex($a).$this->_hex($b).$this->_hex($c).$this->_hex($d).$this->_hex($e);
    }
    // ------------------------------------------------------------------
    /**
     * Convert a decimal to hex
     *
     * @access   private
     * @param    string
     * @return   string
     */
    function _hex($str)
    {
        $str = dechex($str);
        if (strlen($str) == 7)
        {
            $str = ‘0’.$str;
        }
        return $str;
```

```
}
// ------------------------------------------------------------------
/**
 *  Return result based on iteration
 *
 * @access   private
 * @return   string
 */
function _ft($t, $b, $c, $d)
{
        if ($t < 20)
                return ($b & $c) | ((~$b) & $d);
        if ($t < 40)
                return $b ^ $c ^ $d;
        if ($t < 60)
                return ($b & $c) | ($b & $d) | ($c & $d);
        return $b ^ $c ^ $d;
}
// ------------------------------------------------------------------
/**
 * Determine the additive constant
 *
 * @access   private
 * @return   string
 */
function _kt($t)
{
        if ($t < 20)
        {
                return 1518500249;
        }
        else if ($t < 40)
        {
                return 1859775393;
        }
        else if ($t < 60)
        {
                return -1894007588;
        }
        else
        {
                return -899497514;
        }
}
// ------------------------------------------------------------------
/**
 * Add integers, wrapping at 2^32
```

```
         *
         * @access   private
         * @return   string
         */
        function _safe_add($x, $y)
        {
                $lsw = ($x & 0xFFFF) + ($y & 0xFFFF);
                $msw = ($x >> 16) + ($y >> 16) + ($lsw >> 16);
                return ($msw << 16) | ($lsw & 0xFFFF);
        }
        // --------------------------------------------------------------------
        /**
         * Bitwise rotate a 32-bit number
         *
         * @access   private
         * @return   integer
         */
        function _rol($num, $cnt)
        {
                return ($num << $cnt) | $this->_zero_fill($num, 32 - $cnt);
        }
        // --------------------------------------------------------------------
        /**
         * Pad string with zero
         *
         * @access   private
         * @return   string
         */
        function _zero_fill($a, $b)
        {
                $bin = decbin($a);
                if (strlen($bin) < $b)
                {
                        $bin = 0;
                }
                else
                {
                        $bin = substr($bin, 0, strlen($bin) - $b);
                }
                for ($i=0; $i < $b; $i++)
                {
                        $bin = “0”.$bin;
                }
                return bindec($bin);
        }
}
```

4.8. Admin Model Code

```
<?php
class admin_model extends CI_Model {
   function __construct()
   {
      parent::__construct();
                $this->load->helper('security_helper');
   }
                function check_login($options)
                {
                         $username          = $options['username'];
                         $password          = do_hash(do_hash($options['password']));
                         $data = array(
                                                  'username'           => $username,
                                                  'password'           => $password);
                         $result       =       $this->db->get_where('wg_admin', $data);
                        if($result->num_rows()>0)
                        {
                                $row =        $result->row();
                                $sess_array = array(
                                'admin'                        =>     $row->username,
                                );
                                $this->session->set_userdata($sess_array);
                                return 'true';
                        }
                        else
                        return 'failed';
                }
                function check_email_exists($email)
                {
                        $this->db->select('*');
                        $this->db->where('member_email',$email);
                        $result = $this->db->get('members');
                        if($result->num_rows()>0)
                        return "1";
                        else
                        return "0";
                }
                function add_category($options)
                {
                        $cat_name = $options['name'];
                        $data = array(
                        "cat_name" => $cat_name);
                                if($this->check_category_exists($cat_name)=='0')
                                {
                                        $this->db->insert('wg_categories',$data);
                        $url = base_url()."index.php/admin/categories";
                        header("Location:$url");
                        exit();
                }
                else
                {
                        $url= base_url()."index.php/admin/add_category/failed";
                        header("Location:$url");
                }
        }
    function check_category_exists($cat_name)
    {
        $this->db->select('*');
        $this->db->where('cat_name',$cat_name);
        $result = $this->db->get('wg_categories');
        if($result->num_rows()>0)
        return "1";
        else
        return "0";
    }
    function getAllCategories()
    {
                $this->db->select("*");
                $result = $this->db->get('wg_categories');
                if($result->num_rows()>0)
                return $result->result();
                else
                return 'empty';
    }
    function getAllProducts()
    {
                $page = $this->uri->segment(3);
         if($page=='')
            $page=1;
         $start = ($page-1)*RECORDS_PER_PAGE;
         $end = RECORDS_PER_PAGE;
                $this->db->select("*");
         $this->db->limit($end,$start);
         $this->db->order_by('item_id','desc');
                $result = $this->db->get('wg_items');
                if($result->num_rows()>0)
                return $result->result();
                else
                return 'empty';
    }
  function getAllProductsCount()
    {
```

```
                    $this->db->select("*");
                    $result = $this->db->get('wg_items');
                    if($result->num_rows()>0)
                    return $result->num_rows();
                    else
                    return 'empty';
        }
function getCategoryDetails($id)
        {
                    $this->db->select("*");
                    $this->db->where('cat_id',$id);
                    $result = $this->db->get('wg_categories');
                    if($result->num_rows()>0)
                    return $result->row();
                    else
                    return 'empty';
        }
        function getProductDetails($id)
        {
                    $this->db->select("*");
                    $this->db->where('item_id',$id);
                    $result = $this->db->get('wg_items');
                    if($result->num_rows()>0)
                    return $result->row();
                    else
                    return 'empty';
        }
function update_category($options)
{
        $cat_id        =        $options['cat_id'];
        $cat_name      =        $options['name'];
        if($this->check_category_exists($cat_name)=='0')
        {
                $data = array("cat_name" => $cat_name);
                $this->db->where('cat_id',$cat_id);
                $this->db->update('wg_categories',$data);
        }
        else
        {
                $url = base_url()."index.php/admin/edit_category/$cat_id/error";
                header("Location:$url");
                exit();
        }
}
function delete_category($cat_id)
{
        $this->db->where('cat_id',$cat_id);
```

```
            $this->db->delete('wg_categories');
            $url = base_url()."index.php/admin/categories";
            header("Location:$url");
    }
    function add_product($options)
            {
                    $data = array(
                    "cat_id"            => $options['cat_id'],
                    "item_name"         => $options['title'],
                    "item_price"        => $options['price'],
                    "item_desc" => $options['desc'],
                    "item_stock"        => $options['stock']);
                            $this->db->insert('wg_items',$data);
                            return $this->db->insert_id();
            }
    function delete_product($product_id)
    {
            $this->db->where('item_id',$product_id);
            $this->db->delete('wg_items');
            $url = base_url()."index.php/admin/products";
            header("Location:$url");

    }
    function update_product($options)
    {
            $item_id = $options['item_id'];
            $data = array(      "cat_id"            => $options['cat_id'],
                    "item_name"         => $options['title'],
                    "item_price"        => $options['price'],
                    "item_desc" => $options['desc'],
                    "item_stock"        => $options['stock']);
                    $this->db->where('item_id',$item_id);
                    $this->db->update('wg_items',$data);
    }
    function update_image_links($big_image,$medium, $thumbnail,$id)
    {
                    $data = array(
                    "thumbnail" => $thumbnail,
                    "medium_image" => $medium,
                    "big_image" => $big_image);
                    $this->db->where('item_id',$id);
                    $this->db->update('wg_items',$data);
    }
  function getCustomerList()
    {
      $page = $this->uri->segment(3);
                     if($page=='')
                    $page=1;
                 $start = ($page-1)*RECORDS_PER_PAGE;
                 $end = RECORDS_PER_PAGE;
                $this->db->select('*');
           $this->db->limit($end,$start);
                $this->db->order_by("user_id","desc");
                $result = $this->db->get('wg_users');
                if($result->num_rows()>0)
                return $result->result();
                else
                return "empty";
        }
   function getCustomerCountt()
        {
                $this->db->select('*');
                $this->db->order_by("user_id","desc");
                $result = $this->db->get('wg_users');
                if($result->num_rows()>0)
              return $result->num_rows();
                else
                return "empty";
        }
 function getOrderList()
        {
                 $page = $this->uri->segment(3);
                 if($page=='')
                    $page=1;
                 $start = ($page-1)*RECORDS_PER_PAGE;
                 $end = RECORDS_PER_PAGE;
                        $this->db->select("*");
                 $this->db->limit($end,$start);
                        $this->db->order_by('order_id','DESC');
                        $result = $this->db->get('wg_orders');
                        if($result->num_rows()>0)
                        return $result->result();
                        else
                        return 'empty';
        }
   function getOrderCount()
        {
                        $this->db->select("*");
                        $this->db->order_by('order_id','DESC');
                        $result = $this->db->get('wg_orders');
                        if($result->num_rows()>0)
                        return $result->num_rows();
                        else
                        return 'empty';
```

```
          }
 function getOrderDetails($order_id)
          {
                         $this->db->select("*");
                         $this->db->where("order_id",$order_id);
                         $result = $this->db->get('wg_orders');
                         if($result->num_rows()>0)
                         return $result->row();
                         else
                         return 'empty';
          }
  function cancel_order($order_id)
          {
                    $this->db->where('order_id',$order_id);
                    $this->db->set('order_status','cancelled');
                    $this->db->update('wg_orders');
                    $url = base_url()."index.php/admin/orders";
                    header("Location:$url");
          }
 function dispatch_order($order_id)
          {
                    $this->db->where('order_id',$order_id);
                    $this->db->set('order_status','dispatched');
                    $this->db->update('wg_orders');
                    $url = base_url()."index.php/admin/orders";
                    header("Location:$url");
          }
 function getCustomerOrders($user_id)
          {
                         $this->db->select("*");
                         $this->db->where('user_id',$user_id);
                         $this->db->order_by('order_id','DESC');
                         $result = $this->db->get('wg_orders');
                         if($result->num_rows()>0)
                         return $result->result();
                         else
                         return 'empty';
          }
   function getCustomerDetails($user_id)
          {
                         $this->db->select("*");
                         $this->db->where('user_id',$user_id);
                         $result = $this->db->get('wg_users');
                         if($result->num_rows()>0)
                         return $result->row();
                         else
                         return 'empty';
```

```
                }
        function delete_customer($user_id)
                {
                                $this->db->select("*"); $this->db->where('user_id',$user_id);
                                $this->db->delete('wg_users');
                                header("Location:".  base_url()."index.php/admin/");
                }
      function update_customer($options)
       {       $data = array("full_name" => $options['full_name'],
                "email"                        => $options['email'],
                "account_type"                 => $options['account_type'],
                "company_name"                 => $options['company_name'],
                "user_pass"            => do_hash(do_hash($options['password'])),);
                $this->db->where('user_id',$options['customer_id']);
                $this->db->update('wg_users',$data);
                $this->session->set_userdata($data);
header("Location:".base_url()."index.php/admin/cEdit/".$options['customer_id']."/succes
s");
}       }
```

4.9. Cart Model Code

```
<?php
class cart_model extends CI_Model {
   function __construct()
   {    parent::__construct();      }
                function getCartProducts()
                {
                                if($this->session->userdata('cart_session')!='')
                                {
                                $sess = $this->session->userdata('cart_session');
                                $this->db->select("*");
                                $this->db->where("cart_session",$sess);
                                $this->db->where("order_id",'');
                                $this->db->order_by('cart_id','DESC');
                                $result = $this->db->get('wg_cart');
                                if($result->num_rows()>0)
                                          return $result->result();
                                else
                                         return 'empty';
                                }
                                else
                                return 'empty';
                }
                function getCheckoutDetails()
                {
                                if($this->session->userdata('cart_session')!='')
                                {
                                $sess = $this->session->userdata('cart_session');
                                $this->db->select("*");
                                $this->db->where("cart_session",$sess);
                                $this->db->where("cart_status",'saved');
                                $this->db->where("order_id",'');
                                $this->db->order_by('cart_id','DESC');
                                $result = $this->db->get('wg_cart');
                                if($result->num_rows()>0)
                                return $result->result();
                                else
                                return 'empty';
                                }
                                else
                                return 'empty';
                }
                function save_order($options)
                {
                        $data = array(
                        "user_id"            =>     $this->session->userdata('user_id'),
                        "order_date"    =>  date("Y-m-d H:i:s"),
                        "cart_session"  =>  $this->session->userdata('cart_session'),
                        "sub_total"          =>      $options['sub_total'],
                        "vat"                    =>      $options['vat'],
                        "total_price"   =>  $options['total_price'],
                        "order_status"  =>  'new',
                        "ipaddress"          =>      $this->input->ip_address()        );
                        $this->db->insert('wg_orders',$data);
                        $this->session->set_userdata('order_id',$this->db->insert_id());
                    $this->update_cart_order();
                }
                function update_cart_order()
            {
                $this->db->where('cart_session',$this->session->userdata('cart_session'));
                $this->db->where('order_id','');
                $data   = array('order_id'   =>  $this->session->userdata('order_id'));
                $this->db->update('wg_cart',$data);
                $this->session->set_userdata('cart_items_count','');
                $this->session->set_userdata('total_price','');
            }
                function save_address($options)
                {
                        $data = array(
                        "order_id"           =>      $this->session->userdata('order_id'),
                        "address1"           =>      $options['address1'],
                        "address2"           =>      $options['address2'],
                        "city"                   =>      $options['city'],
                        "county"              =>      $options['county'],
                        "post_code"         =>      $options['post_code']);
                        $this->db->insert('wg_address',$data);
                        return $this->db->insert_id();
                }
                function add2cart($pid)
                {
                         $product_exists = $this->checkProductInCart($pid);
                                if($product_exists=='No')
                                        $this->add_product_in_cart($pid);
                                else
                         {
                                 $qty = $this->get_cart_product_quantity($pid);
                                 $this->update_quantity_in_cart($pid,($qty+1));
                         }
                }
            function get_cart_product_quantity($pid)
                {
                        $session_cart        =       $this->session->userdata('cart_session');
                        $this->db->select('*');
```

```
		$this->db->where('item_id',$pid);
		$this->db->where('cart_session',$session_cart);
	  $this->db->where('order_id','');
		$result		=	$this->db->get('wg_cart');
		if($result->num_rows()>0)
		return $result->row()->item_quantity;
		else
		return "No";
	}
	function checkProductInCart($pid)
	{
		$session_cart		=	$this->session->userdata('cart_session');
		$this->db->select('*');
		$this->db->where('item_id',$pid);
		$this->db->where('cart_session',$session_cart);
		$this->db->where('order_id','');
		$result		=	$this->db->get('wg_cart');
		if($result->num_rows()>0)
		return "Yes";
		else
		return "No";
	}
	function add_product_in_cart($pid)
	{
		$product = $this->admin_model->getProductDetails($pid);
		$data = array(
		"item_id" => $product->item_id,
		"item_price" => $product->item_price,
		"item_name" => $product->item_name,
		"item_quantity" => '1',
		"item_total_price" => $product->item_price,
		"item_image" => $product->thumbnail,
		"cart_session" => $this->session->userdata('cart_session')		);
		$this->db->insert('wg_cart',$data);
	}
	function getTotalCartProducts()
	{
		$session_cart		=	$this->session->userdata('cart_session');
		$this->db->select('*');
		$this->db->where('cart_session',$session_cart);
		$this->db->where('order_id','');
		$result = $this->db->get('wg_cart');
		if($result->num_rows()>0)
		return $result->num_rows();
		else
		return "0";
	}
```

```
                function getTotalCartPrice()
                {
                        $session_cart       =       $this->session->userdata('cart_session');
                        $this->db->select('sum(item_total_price) as total');
                        $this->db->where('cart_session',$session_cart);
                        $this->db->where('order_id','');
                        $result = $this->db->get('wg_cart');
                        if($result->num_rows()>0)
                        {
                                $total= $result->row()->total;
                                $vat = $total*0.12;
                                $total = $total + $vat;
                                return $total;
                        }
                        else
                        return "0";
                }
                function update_cart($options)
                {
                        $items         =       $options['items'];
                        $qty   =       $options['qty'];
                                for($i=0;$i<sizeof($items); $i++)
                                {
                            if($qty[$i]>=0)
                            {
                                        if($qty[$i]=='0')
                                        $this->remove_product_from_cart($items[$i]);
                                        else
                                        $this->update_quantity_in_cart($items[$i],$qty[$i]);
                                }
                        }
                }
function remove_product_from_cart($pid)
        {
                $this->db->where('item_id',$pid);
        $this->db->where('cart_session',$this->session->userdata('cart_session'));
                $this->db->delete('wg_cart');
                $total_items = $this->session->userdata('cart_items_count');
                if($total_items!='0')
                $total_items = $total_items-1;
                $this->session->set_userdata('cart_items_count', $total_items);
        }
function update_quantity_in_cart($item_id,$qty)
        {
$query = "update wg_cart set item_total_price = (item_price*$qty),item_quantity='$qty'
where item_id = '$item_id' and cart_session='".$this->session-
>userdata('cart_session')."'";
```

```
                $this->db->query($query);
        }
function saveCart()
        {
                if($this->session->userdata('cart_session')=='')
                 header("Location:".base_url());
                $this->db->where('cart_session', $this->session->userdata('cart_session '));
                $this->db->set('cart_status',"saved");
                $this->db->update('wg_cart');
                 if($this->session->userdata('user_id')=='')
                 $url = base_url()."index.php/login";
                 else
                 $url = base_url()."index.php/order_step2";
                header("Location:$url");
        }
function getUserOrders()
        {
                $this->db->select("*");
                $this->db->where("user_id",$this->session->userdata('user_id'));
                $this->db->order_by('order_id','DESC');
                $result = $this->db->get('wg_orders');
                if($result->num_rows()>0)
                return $result->result();
                else
                return 'empty';
                }
                function getOrderDetails($order_id)
                {
                        $this->db->select("*");
                        $this->db->where("user_id",$this->session->userdata('user_id'));
                        $this->db->where("order_id",$order_id);
                        $result = $this->db->get('wg_orders');
                        if($result->num_rows()>0)
                        return $result->row();
                        else
                        return 'empty';
                }
                function getOrderAddress($order_id)
                {
                                $this->db->select("*");
                                $this->db->where("order_id",$order_id);
                                $result = $this->db->get('wg_address');
                                if($result->num_rows()>0)
                                return $result->row();
                                else
                                return 'empty';
                }
                function getOrderProducts($cart_session)
                {
                                $this->db->select("*");
                                $this->db->where("cart_session",$cart_session);
                                $this->db->where('order_id',$this->uri->segment(3));
                                $this->db->order_by('cart_id','DESC');
                                $result = $this->db->get('wg_cart');
                                if($result->num_rows()>0)
                                return $result->result();
                                else
                                return 'empty';
                }
                function cancel_order($order_id)
                {
                        $this->db->where('user_id',$this->session->userdata('user_id'));
                        $this->db->where('order_id',$order_id);
                        $this->db->set('order_status','cancelled');
                        $this->db->update('wg_orders');
                        $url = base_url()."index.php/user";
                        header("Location:$url");
                }
}
```

4.10. ***Main Model Code***

```
<?php
class main_model extends CI_Model
{
   function __construct()
   {
      parent::__construct();
                $this->load->helper('security_helper');
   }
            function getAllCategories()
               {
                                $this->db->select("*");
                                $result = $this->db->get('wg_categories');
                                if($result->num_rows()>0)
                                return $result->result();
                                else
                                return 'empty';
               }
               function getLatestProducts()
               {
                                $this->db->select("*");
                                $this->db->limit(12,0);
                                $this->db->order_by('item_id','DESC');
                                $result = $this->db->get('wg_items');
                                if($result->num_rows()>0)
                                return $result->result();
                                else
                                return 'empty';
               }
               function getCategoryProducts($cat_id)
               {
                                $this->db->select("*");
                                $this->db->where("cat_id",$cat_id);
                                $this->db->limit(50,0);
                                $this->db->order_by('item_id','DESC');
                                $result = $this->db->get('wg_items');
                                if($result->num_rows()>0)
                                return $result->result();
                                else
                                return 'empty';
               }
               function check_login($options)
               {
                                $this->db->select("*");
                                $this->db->where("email",$options['username']);
               $this->db->where("user_pass",do_hash(do_hash($options['password'])));
```

```
                    $result = $this->db->get('wg_users');
                    if($result->num_rows()>0)
                    {
                    $row = $result->row();
                            $sess_array = array(
                            "user_name"     =>  $row->full_name,
                            "email"              =>      $row->email,
                            "company_name"  =>      $row->company_name,
                            "user_pass"          =>      $row->user_pass,
                            "account_type"  =>         $row->account_type,
                            "user_id"             =>      $row->user_id);
                        $this->session->set_userdata($sess_array);
                        if($this->session->userdata('total_price')!='')
                            $url = base_url()."index.php/order_step2";
                        else
                            $url = base_url()."index.php/user";
                        header("Location:$url");
                        exit();
                    }
                    else
                    {
                        $url = base_url()."index.php/login/failed";
                        header("Location:$url");
                        exit();
                    }
            }
        function register($options)
        {
            $data = array(
            "full_name"          => $options['full_name'],
            "email"                          => $options['email'],
            "account_type"      => $options['account_type'],
            "date_joined"   =>  date("Y-m-d H:i:s"),
            "company_name"  => $options['company_name'],
            "user_pass"  => do_hash(do_hash($options['password'])),          );
            $email_exist = $this->check_if_email_exists($options['email']);
            if($email_exist=='0')
            {
                $this->db->insert('wg_users',$data);
                $this->session->set_userdata($data);
                $this->session->set_userdata('user_id',$this->db->insert_id());
                $url = base_url()."index.php/order_step2";
                header("Location:$url");
            }
            else
            {
                    $url = base_url()."index.php/register/failed";
```

```
                            header("Location:$url");
                   }
          }
          function check_if_email_exists($email)
          {
                            $this->db->select("*");
                            $this->db->where("email",$email);
                            $result = $this->db->get('wg_users');
                            if($result->num_rows()>0)
                            return "1";
                            else
                            return "0";
          }
          function update_user($options)
          {
                   if($_POST['password']!='')
                   $data = array(
                   "full_name"          => $options['full_name'],
                   "email"                        => $options['email'],
                   "account_type"      => $options['account_type'],
                   "company_name"  => $options['company_name'],
                   "user_pass"  => do_hash(do_hash($options['password'])),          );
                   else
                   $data = array(
                   "full_name"          => $options['full_name'],
                   "email"                        => $options['email'],
                   "account_type"      => $options['account_type'],
                   "company_name"  => $options['company_name']);
                   $this->db->where('user_id',$this->session->userdata('user_id'));
                   $this->db->update('wg_users',$data);
                   $this->session->set_userdata($data);
                   header("Location:".base_url()."index.php/user/account/success");

          }
}
```

5. ***Testing Site***

Test Condition	***Test Date***	***Pass/Fail***
Home page	11/10/2014	Pass
Admin Check User Details	11/10/2014	Pass
Admin Check Order Details	11/10/2014	Pass
Admin Change Status of Order (New, Dispatched, Cancel)	11/10/2014	Pass
Admin Delete User	11/10/2014	Pass
Admin Change User Details	11/10/2014	Pass
Admin Add New Category	11/10/2014	Pass
Admin Edit any Category Detail	11/10/2014	Pass
Admin Delete Added Category	11/10/2014	Pass
Admin Add New Product	11/10/2014	Pass
Admin Edit Product Detail	11/10/2014	Pass
Admin Delete any Product	11/10/2014	Pass
Customer Register	11/10/2014	Pass
Customer Login	11/10/2014	Pass
Customer Check Category	11/10/2014	Pass
Customer Check Product Detail	11/10/2014	Pass
Customer Add Product to Cart	11/10/2014	Pass
Customer Update Cart	11/10/2014	Pass
Customer Check product title	11/10/2014	Pass
Customer Check product images	11/10/2014	Pass
Customer Check product description	11/10/2014	Pass
The cart: adding items	11/10/2014	Pass
The cart: changing item quantity	11/10/2014	Pass
The cart: removing items	11/10/2014	Pass
Check VAT and delivery costs add up correctly	11/10/2014	Pass
Move into checkout process	11/10/2014	Pass
Test checkout process.	11/10/2014	Pass
Final amount to pay make sure that this value is correct, after the price of the products, VAT, delivery and any other charges.	11/10/2014	Pass
Test making changes to the products being ordered, changing delivery options, etc. and make sure that the final amount updates correctly.	11/10/2014	Pass

6. ***Conclusion***

E-commerce is widely considered the buying and selling of products over the internet, but any transaction that is completed solely through electronic measures can be considered e-commerce. Day by day E-commerce and M- commerce playing very good role in online retail marketing and peoples using this technology day by day increasing all over the world.

E-commerce security is the protection of e-commerce assets from unauthorized access, use, alteration, or destruction. Dimensions of e-commerce security; Integrity: prevention against unauthorized data modification, No repudiation: prevention against any one party from reneging on an agreement after the fact. Authenticity: authentication of data source. Confidentiality: protection against unauthorized data disclosure. Privacy: provision of data control and disclosure. Availability: prevention against data delays or removal.

Fraudsters are constantly looking to take advantage of online shoppers prone to making novice errors. Common mistakes that leave people vulnerable include shopping on websites that aren't secure, giving out too much personal information, and leaving computers open to viruses. In this report we discussed E-commerce Security Issues, Security measures, Digital E-commerce cycle/Online Shopping, Security Threats and guidelines for safe and secure online shopping through shopping web sites.

7. References

1 Niranjanamurthy M, DR. Dharmendra Chahar “The study of E-Commerce Security Issues and Solutions”. International Journal of Advanced Research in Computer and Communication Engineering Vol. 2, Issue 7, July 2013

2 "HTML5", World Wide Web Consortium, June 10, 2008

3 Håkon Wium Lie and Bert Bos” Cascading Style Sheets: Designing for the Web” (2005) ISBN 0-321-19312-1.

4 Harris, Andy “JavaScript Programming for the Absolute Beginner” (2001), Premier Press, ISBN 0-7615-3410-5.

5 “Java Script”. http://en.wikipedia.org/wiki/JavaScript.

6 "History of PHP". php.net.

7 “PHP 5 Tutorial”. http://www.w3schools.com/php/.

8 Guy Harrison, Steven Feuerstein “MySQL Stored Procedure Programming” (2008) O'Reilly Media. p. 49. ISBN 978-0-596-10089-6.

9 “MySQL Quick Reference”. http://www.w3schools.com/sql/ sql_quickref.asp

10 Harris, Shon “All-in-one CISSP Exam Guide” 6th Edition, McGraw Hill Osborne, Emeryville, California, 2012.

11 “Authentication”. http://en.wikipedia.org/wiki/Authentication.

12 Layton, Timothy P “Information Security: Design, Implementation, Measurement, and Compliance” (2007), Boca Raton, FL: Auerbach publications. ISBN 978-0-8493-7087-8.

13 Dhillon, Gurpreet “Principles of Information Systems Security: text and cases” (2007), NY: John Wiley & Sons. ISBN 978-0-471-45056-6.

14 Xiaoyun Wang, Yiqun Lisa Yin and Hongbo Yu “Finding Collisions in the Full SHA-1” Crypto 2005.

15 “SHA-1 Algorithm”. http://en.wikipedia.org/wiki/SHA-1.

16 “INNO DB Search Engines”. http://dev.mysql.com/doc/refman/5.5/en/ optimization-indexes.html.

17 “MyISAM Storage Engines”. http://dev.mysql.com/doc/internals/en/ myisam-introduction.html.

18 “BTREE Introduction”. http://en.wikipedia.org/wiki/B%2B_tree.