MURUGAPPA POLYTECHNIC COLLEGE

Sathyamurthy Nagar, Chennai-600 062.

(An Academically Autonomous Institution)

DEPARTMENT OF COMPUTER ENGINEERING

PROJECT REPORT

ON

SQL INJECTION PREVENTION SYSTEM IN ONLINE SHOPPING

Submitted By

NAME	REGISTER NUMBER
MYTHRIYAN R	1912270
NANDHA A	1912272
NAVEEN KUMAR D	1912276
RAJESH S	1912282

In partial fulfilment of the requirements of the award of

DIPLOMA IN COMPUTER ENGINEERING (REGULAR)

Under the guidance of

Mrs. E. Nambirani, B.E.
Lecturer, Department of Computer Engineering

APRIL 2022

MURUGAPPA POLYTECHNIC COLLEGE

Sathyamurthy Nagar, Chennai-600 062.

(An Academically Autonomous Institution)

DEPARTMENT OF COMPUTER ENGINEERING

BONAFIDE CERTIFICATE

Certified that this project report entitled "**SQL INJECTION PREVENTION SYSTEM IN ONLINE SHOPPING**" is the bonafide record of work done by Selvan ___________________ Registration No ___________, student of Sixth Semester Computer Engineering (Regular) Diploma Course, during MARCH 2021 – MAY 2022 in fulfillment of the requirements for the award of **DIPLOMA IN COMPUTER ENGINEERING (REGULAR).**

Project Guide	**Head of the Department**
Name: Mrs. E. Nambirani, B. E	Name: Mr. B. RAJENDRAN, M.E
Date:	Date:

Submitted for the End Semester Examination held on _________________

INTERNAL EXAMINER **EXTERNAL EXAMINER**

ACKNOWLEDGEMENT

Any work completed consists not only the experience and skill of an individual but also it includes an organized body of people. As such, many kind hearts lay behind the completion of our project.

First, we wish our grateful gratitude to our parents who dedicated their life for our studies. We express our deep sense of gratitude to our beloved principal **Dr.K.SUDHAGAR, Ph.D.** whose motivation has helped us to develop this project.

We wish to express our sincere thanks to **HOD** of Computer Engineering **Mr.B.RAJENDRAN, M.E** and our guide **Mrs.E.NAMBIRANI, B.E**. The valuable guidance and encouragement given by have helped us in successfully completing this project.

We also wish to thank all staff members of Computer Engineering Department for their guidance and encouragement.

We place in record all our profound gratitude to helping hands for their courtesy and guidance.

The team members are very thankful to their friends and fellow students for helping us during the course of this project Work.

SYNOPSIS

An online shop that allows users to check for different cloths available at the online store the project consists of a list of cloths display in various materials and designs. The users may browse through these products as per categories. If the user likes a product. He/she can add it to his/her shopping cart.

Once a user wishes to checkout, he must register on the site first. Once the user makes a successfully transaction, the admin will get a report of his bought projects. The objective of this project is to develop a secure path for transaction done by the user.

Using AES (Advance Encryption Standard) encryption technique, the transaction and user account details can be made secure. AES encryption is also used to encrypt the user’s card and password information while transaction and to prevent SQL injections in the website.

1. INTRODUCTION

1.1 AIM OF THE PROJECT

This is very beneficial in case of preventing out the website from the hackers who use SQL Injections to steal the information in the database server.

Given most websites are built on data in a database server, a malicious SQL injection can be lethal. Attackers can access sensitive information, modify web content, and in catastrophic cases, delete your data.

So, the all the developers have a responsibility to save it from the hackers.

In such way we use SQL Injection prevention to save data form the hackers.

1.2 OBJECTIVES

Using AES (Advance Encryption Standard) encryption technique, the transaction and user account details can be made secure. AES encryption is also used to encrypt the user's card and password information while transaction and to prevent SQL injections in the website.

The objective is to deceive the database system into running malicious code that will reveal sensitive information or otherwise compromise the server. By modifying the expected Web application parameters, an attacker can submit SQL queries and pass commands directly to the database.

2. SYSTEM ANALYSIS

FEASIBILITY STUDY

Feasibility studies to aim to objectively and rationally identification systems Uncover the strengths and weakness of the existing business or proposed venture, opportunities and threats as presented by the environment, the resources required to carry through and ultimately the prospects for success.

2.1 EXISTING SYSTEM

This older website is less secure and easy to hack. The attackers use different attacks which the system won't understand. So that the system will allow the SQL Injections and suspicious codes to get into the server. Then the server will be hacked.

2.2 PROPOSED SYSTEM

Technically, the only way to prevent a SQL injection attack is to have input validation in place. This means that inputs entered by the users must be monitored and sanitized to filter out any potential malicious codes.

This is exactly what a web application firewall (WAF) does. It analyzes all inputs entered by the users into the web application to find any matches with suspicious codes.

Once the user input is an SQL Injection the prevention system in the project will automatically triggered and block the SQL Injection and prevent the system. Once this prevention done the system notify the admin about the attack.

3. SYSTEM REQUIREMENTS

3.1 HARDWARE REQUIREMENTS

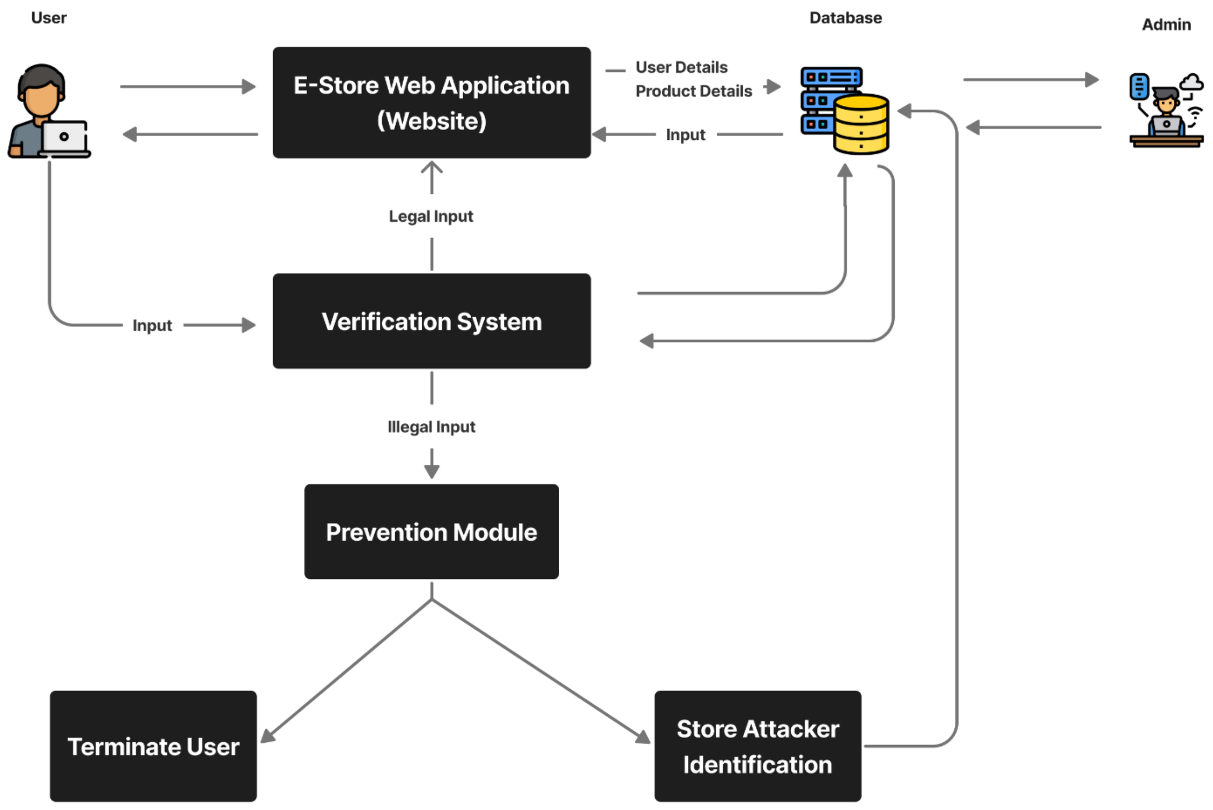
➢ **Processor**	Intel I3
➢ **RAM**	4GB
➢ **Hard Disk**	250GB

3.2 SOFTWARE REQUIREMENTS

➢ **Operating System**	Windows 10
➢ **Languages**	HTML, CSS, JAVASCRIPT, PHP
➢ **Database**	MySQL Version 8.0
➢ **Software**	Xampp, Visual studio code

4. SYSTEM DESIGN

4.1 SYSTEM ARCHITECTURE



4.2 DATABASE DESIGN

Users:

S.No	**Name**	**Type**	**Null**	**Default**	**Extra**
1	users_id	int(11)	No	None	A_I
2	users_name	varchar(255)	No	None	
3	users_email	varchar(255)	No	None	
4	users_pass	varchar(255)	No	None	

Newsletter:

S.No	**Name**	**Type**	**Null**	**Default**	**Extra**
1	newsletter_id	int(11)	No	None	A_I
2	newsletter_email	varchar(255)	No	None	

Prevent:

S.No	**Name**	**Type**	**Null**	**Default**	**Extra**
1	prevent_id	int(11)	No	None	A_I
2	prevent_user_id	int(11)	No	None	
3	prevent_time	varchar(30)	No	None	
4	prevent_attack	Text	No	None	
5	prevent_page	Text	No	None	
6	prevent_section	Text	No	None	
7	prevent_user_name	varchar(255)	No	None	
8	prevent_user_email	varchar(255)	No	None	

Products:

S.No	Name	Type	Null	Default	Extra
1	product_id	int(11)	No	None	A_I
2	product_name	varchar(255)	No	None	
3	product_price	varchar(50)	No	None	

Payment Method:

S.No	Name	Type	Null	Default	Extra
1	payment_methods_id	int(11)	No	None	A_I
2	payment_methods_uid	int(11)	No	None	
3	payment_methods_card_num	varchar(20)	No	None	
4	payment_methods_cname	varchar(50)	No	None	
5	payment_methods_year	varchar(10)	No	None	

Search

S.No	Name	Type	Null	Default	Extra
1	search_id	int(11)	No	None	A_I
2	search_uid	int(11)	No	None	
3	search_content	varchar(255)	No	None	

5. MODULES DESCRIPTION

1. Account Validation
2. Electronic Store Web Application
3. User Input Verification
4. SQL Injection Prevention

1. Account Validation

➢ Users means an individual who is authorized customer to use the web form of the application.

- In this project we have three types of users namely,
    - Legal Users
    - Hacker
    - Admin ○ **Legal Users:**
    - Legal users are the genuine customers of the application and the stick to the work flow of the application and experience the architecture and working.
- **Hacker:**
    - Hacker is the threat to the application they attack the website and they take the information they need and they crack the whole system and make the application unresponsive.
- **Admin:**

  - A website administrator is a technical professional who maintains websites. They typically have a thorough knowledge of web maintenance and web development that help a website function, as well as skills in front-end development that contribute to a website's appearance and usability.

2. Electronic Store Web Application

- ➢ The E-Store web application is a platform that collects information from different sources into a single user interface and presents users with the most information for their content.

- ➢ The E-Store web application allows you a convenient and secure way to view the content they require.

3. User Input Verification

- ➢ Verification is the process for determining whether the user is legal or a hacker who perform some illegal activity in the E-Store web application verification system has a direct contact with the user.

- ➢ The E-Store web application and the database. It determines the user whether they are legal or hacker.

4. SQL Injection Prevention

- ➢ SQL Injection prevention system is a fully backed algorithm that takes action on the hacker and make the prevention to the application by the hackers.

- ➢ SQL Injection prevention system is also responsible for informing the admin about the attack.

5. LANGUAGE SPECIFICATION

6.1 SOFTWARE DESCRIPTION

LAMP stands for Linux, Apache, MySQL, and PHP. Together, they provide a proven set of software for delivering high-performance web applications. Each component contributes essential capabilities to the stack:

- **Linux**: The operating system. Linux is a free and open-source operating system (OS) that has been around since the mid-1990s. Today, it has an extensive worldwide user base that extends across industries. Linux is popular in part because it offers more flexibility and configuration options than some other operating systems.
- **Apache**: The web server. The Apache web server processes requests and serves up web assets via HTTP so that the application is accessible to anyone in the public domain over a simple web URL. Developed and maintained by an open community, Apache is a mature, feature-rich server that runs a large share of the websites currently on the internet.
- **MySQL**: The database. MySQL is an open source relational database management system for storing application data. With My SQL, you can store all your information in a format that is easily queried with the SQL language. SQL is a great choice if you are dealing with a business domain that is well structured, and you want to translate that structure into the backend. MySQL is suitable for running even large and complex sites. See "SQL vs. NoSQL Databases: What's the Difference?" for more information on SQL and NoSQL databases.
- **PHP**: The programming language. The PHP open source scripting language works with Apache to help you create dynamic web pages. You cannot use HTML to perform dynamic processes such as pulling data out of a database. To provide this type of functionality, you simply drop PHP code into the parts of a page that you want to be dynamic.

PHP is designed for efficiency. It makes programming easier—and a bit more fun—by allowing you to write new code, hit refresh, and immediately see the resulting changes without the need for compiling. If you prefer, you can swap out PHP in favor of Perl or the increasingly popular Python language.

HTML

The Hypertext Markup Language or HTML is the standard markup language for documents designed to be displayed in a web browser. It can be assisted by technologies such as Cascading Style Sheets (CSS) and scripting languages such as JavaScript.

Web browsers receive HTML documents from a web server or from local storage and render the documents into multimedia web pages. HTML describes the structure of a web page semantically and originally included cues for the appearance of the document.

HTML elements are the building blocks of HTML pages. With HTML constructs, images and other objects such as interactive forms may be embedded into the rendered page. HTML provides a means to create structured documents by denoting structural semantics for text such as headings, paragraphs, lists, links, quotes and other items. HTML elements are delineated by tags, written using angle brackets. Tags such as <img /> and <input /> directly introduce content into the page. Other tags such as <p> surround and provide information about document text and may include other tags as sub-elements. Browsers do not display the HTML tags but use them to interpret the content of the page.

HTML can embed programs written in a scripting language such as JavaScript, which affects the behavior and content of web pages. Inclusion of CSS defines the look and layout of content. The World Wide Web Consortium (W3C), former maintainer of the HTML and current maintainer of the CSS standards, has encouraged the use of CSS over explicit presentational HTML since 1997.

A form of HTML, known as HTML5, is used to display video and audio, primarily using the <canvas> element, in collaboration with JavaScript.

History

In 1980, physicist Tim Berners-Lee, 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 Robert Cillian 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 Tim 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 Standard

Generalized Markup Language (SGML)-based documentation format at CERN. Eleven of these elements still exist in HTML 4.

HTML version timeline

➢ **HTML 2**

November 24, 1995

HTML 2.0 was published as RFC 1866. Supplemental RFCs added capabilities:

- November 25, 1995: RFC 1867 (form-based file upload)
- May 1996: RFC 1942 (tables)
- August 1996: RFC 1980 (client-side image maps)
- January 1997: RFC 2070 (internationalization)

➢ **HTML 3**

January 14, 1997

HTML 3.2 was published as a W3C Recommendation. It was the first version developed and standardized exclusively by the W3C, as the IETF had closed its HTML Working Group on September 12, 1996.

Initially code-named "Wilbur", HTML 3.2 dropped math formulas entirely, reconciled overlap among various proprietary extensions and adopted most of Netscape's visual markup tags. Netscape's blink element and Microsoft's marquee element were omitted due to a mutual agreement between the two companies. A markup for mathematical formulas similar to that in HTML was not standardized until 14 months later in MathML.

➢ **HTML 4**

December 18, 1997

HTML 4.0 was published as a W3C Recommendation. It offers three variations:

- Strict, in which deprecated elements are forbidden
- Transitional, in which deprecated elements are allowed
- Frameset, in which mostly only frame related elements are allowed.

Initially code-named "Cougar", HTML 4.0 adopted many browser-specific element types and attributes, but at the same time sought to phase out Netscape's visual markup features by marking them as deprecated in favor of style sheets. HTML 4 is an SGML application conforming to ISO 8879 – SGML.

April 24, 1998

HTML 4.0 was reissued with minor edits without incrementing the version number.

December 24, 1999

HTML 4.01 was published as a W3C Recommendation. It offers the same three variations as HTML 4.0 and its last errata were published on May 12, 2001.

May 2000

ISO/IEC 15445:2000 ("ISO HTML", based on HTML 4.01 Strict) was published as an ISO/IEC international standard. In the ISO these standard falls in the domain of the ISO/IEC JTC1/SC34 (ISO/IEC Joint Technical Committee 1, Subcommittee 34 – Document description and processing languages).

After HTML 4.01, there was no new version of HTML for many years as development of the parallel, XML-based language XHTML occupied the W3C's HTML Working Group through the early and mid-2000s.

HTML 5

Main article: HTML5

October 28, 2014

HTML5 was published as a W3C Recommendation.

November 1, 2016

HTML 5.1 was published as a W3C Recommendation.

December 14, 2017

HTML 5.2 was published as a W3C Recommendation.

CSS

Cascading Style Sheets (CSS) is a style sheet language used for describing the presentation of a document written in a markup language such as HTML. CSS is a cornerstone technology of the World Wide Web, alongside HTML and JavaScript.

CSS is designed to enable the separation of presentation and content, including layout, colors, and fonts. This separation can improve content accessibility; provide more flexibility and control in the specification of presentation characteristics; enable multiple web pages to share formatting by specifying the relevant CSS in a separate .CSS file, which reduces complexity and repetition in the structural content; and enable the .CSS file to be cached to improve the page load speed between the pages that share the file and its formatting.

Separation of formatting and content also makes it feasible to present the same markup page in different styles for different rendering methods, such as onscreen, in print, by voice (via speech-based browser or screen reader).

Braille-based tactile devices. CSS also has rules for alternate formatting if the content is accessed on a mobile device.

The name cascading comes from the specified priority scheme to determine which style rule applies if more than one rule matches a particular element. This cascading priority scheme is predictable.

SYNTAX

CSS has a simple syntax and uses a number of English keywords to specify the names of various style properties.

A style sheet consists of a list of rules. Each rule or rule-set consists of one or more selectors, and a declaration block.

Selector

In CSS, *selectors* declare which part of the markup a style applies to by matching tags and attributes in the markup itself.

Selectors may apply to the following:

➢ All elements of a specific type, e.g. the second-level headers h2 ➢ Elements specified by attribute, in particular:
  - id: an identifier unique within the document, identified with a hash prefix e.g., #id
  - class: an identifier that can annotate multiple elements in a document, identified with a period prefix e.g. classname

➢ elements depending on how they are placed relative to others in the document tree.

Classes and IDs are case-sensitive, start with letters, and can include alphanumeric characters, hyphens, and underscores. A class may apply to any number of instances of any elements. An ID may only be applied to a single element.

Pseudo-classes are used in CSS selectors to permit formatting based on information that is not contained in the document tree. One example of a widely used pseudo-class is: however, which identifies content only when the user "points to" the visible element, usually by holding the mouse cursor over it. It is appended to a selector as in a: however, or #elementid: hover. A pseudo-class classifies document elements, such as: link or: visited, whereas a pseudo-element makes a selection that may consist of partial elements, such as: first-line or: first-letter.

Selectors may be combined in many ways to achieve great specificity and flexibility. Multiple selectors may be joined in a spaced list to specify elements by location, element type, id, class, or any combination thereof. The order of the selectors is important. For example, div .my Class {color: red;} applies to all elements of class my Class that are inside div elements, whereas

Class div {color: red;} applies to all div elements that are inside elements of class.

This is not to be confused with concatenated identifiers such as div. my Class {color: red;} which applies to div elements of class my Class.

Use

Before CSS, nearly all presentational attributes of HTML documents were contained within the HTML markup. All font colors, background styles, element alignments, borders and sizes had to be explicitly described, often repeatedly, within the HTML. CSS lets authors move much of that information to another file, the style sheet, resulting in considerably simpler HTML.

For example, headings (h1 elements), sub-headings (h2), sub-sub-headings (h3), etc., are defined structurally using HTML. In print and on the screen, choice of font, size, color and emphasis for these elements is presentational.

Before CSS, document authors who wanted to assign such typographic characteristics to, say, all h2 headings had to repeat HTML presentational markup for each occurrence of that heading type. This made documents more complex, larger, and more error-prone and difficult to maintain. CSS allows the separation of presentation from structure. CSS can define color, font, text alignment, size, borders, spacing, layout and many other typographic characteristics, and can do so independently for on-screen and printed views. CSS also defines non-visual styles, such as reading speed and emphasis for aural text readers. The W3C has now deprecated the use of all presentational HTML markup.

JavaScript

JavaScript often abbreviated JS, is a programming language that is one of the core technologies of the World Wide Web, alongside HTML and CSS. Over 97% of websites use JavaScript on the client side for web page behavior, often incorporating third-party libraries. All major web browsers have a dedicated JavaScript engine to execute the code on users' devices.

JavaScript is a high-level, often just-in-time compiled language that conforms to the ECMAScript standard. It has dynamic typing, prototype-based object-orientation, and first-class functions. It is multi-paradigm, supporting event-driven, functional, and imperative programming styles. It has application programming interfaces (APIs) for working with text, dates, regular expressions, standard data structures, and the Document Object Model (DOM).

The ECMAScript standard does not include any input/output (I/O), such as networking, storage, or graphics facilities. In practice, the web browser or other runtime system provides JavaScript APIs for I/O.

JavaScript engines were originally used only in web browsers, but are now core components of some servers and a variety of applications. The most popular runtime system for this usage is Node.js.

Although Java and JavaScript are similar in name, syntax, and respective standard libraries, the two languages are distinct and differ greatly in design.

History

Creation at Netscape

The first web browser with a graphical user interface, Mosaic, was released in 1993. Accessible to non-technical people, it played a prominent role in the rapid growth of the nascent World Wide Web. The lead developers of Mosaic then founded the Netscape corporation, which released a more polished browser, Netscape Navigator, in 1994. This quickly became the most-used.

Adoption by Microsoft

Microsoft debuted Internet Explorer in 1995, leading to a browser war with Netscape. On the JavaScript front, Microsoft reverse-engineered the Navigator interpreter to create its own, called JScript.

The rise of JScript

In November 1996, Netscape submitted JavaScript to Emma International, as the starting point for a standard specification that all browser vendors could conform to. This led to the official release of the first ECMAScript language specification in June 1997.

The standards process continued for a few years, with the release of ECMAScript 2 in June 1998 and ECMAScript 3 in December 1999. Work on ECMAScript 4 began in 2000.

Meanwhile, Microsoft gained an increasingly dominant position in the browser market. By the early 2000s, Internet Explorer's market share reached 95%. This meant that JScript became the de facto standard for client-side scripting on the Web.

Growth and standardization

During the period of Internet Explorer dominance in the early 2000s, clientside scripting was stagnant. This started to change in 2004, when the successor of Netscape, Mozilla, released the Firefox browser. Firefox was well received by many, taking significant market share from Internet Explorer.

In 2005, Mozilla joined ECMA International, and work started on the ECMAScript for XML (E4X) standard. This led to Mozilla working jointly with Macromedia (later acquired by Adobe Systems), who were implementing E4X in their ActionScript 3 language, which was based on an ECMAScript 4 draft. The goal became standardizing ActionScript 3 as the new ECMAScript 4. To this end, Adobe Systems released the Tamarin implementation as an open source project. However, Tamarin and ActionScript 3 were too different from established client-side scripting, and without cooperation from Microsoft, ECMAScript 4 never reached fruition.

Reaching maturity

Ambitious work on the language continued for several years, culminating in an extensive collection of additions and refinements being formalized with the publication of ECMAScript 6 in 2015.

The creation of Node.js in 2009 by Ryan Dahl sparked a significant increase in the usage of JavaScript outside of web browsers. Node combines the V8 engine, an event loop, and I/O APIs, thereby providing a stand-alone JavaScript runtime system. As of 2018, Node had been used by millions of developers, and name had the most modules of any package manager in the world.

PHP

PHP is a general-purpose scripting language geared toward web development. It was originally created by Danish-Canadian programmer RasmusLerdorf in 1994. The PHP reference implementation is now produced by The PHP Group. PHP originally stood for Personal Home Page, but it now stands for the recursive initialism PHP: Hypertext Preprocessor.

PHP code is usually processed on a web server by a PHP interpreter implemented as a module, a daemon or as a Common Gateway Interface (CGI) executable. On a web server, the result of the interpreted and executed PHP code – which may be any type of data, such as generated HTML or binary image data – would form the whole or part of an HTTP response. Various web template systems, web content management systems, and web frameworks exist which can be employed to orchestrate or facilitate the generation of that response. Additionally, PHP can be used for many programming tasks outside the web context, such as standalone graphical applications and robotic drone control. PHP code can also be directly executed from the command line.

The standard PHP interpreter, powered by the Zend Engine, is free software released under the PHP License. PHP has been widely ported and can be deployed on most web servers on a variety of operating systems and platforms.

The PHP language evolved without a written formal specification or standard until 2014, with the original implementation acting as the de facto standard which other implementations aimed to follow. Since 2014, work has gone on to create a formal PHP specification.

W3Techs reports that, as of January 2022, "PHP is used by 78.1% of all the websites whose server-side programming language we know." PHP version 7.4 is the most used version. Support for version 7.3 was dropped on 6 December 2021.

HISTORY

Early history

PHP development began in 1994 when Rasmus Lerdorf wrote several Common Gateway Interface (CGI) programs in C, which he used to maintain his personal homepage. He extended them to work with web forms and to communicate with databases, and called this implementation "Personal Home Page/Forms Interpreter" or PHP/FI.

PHP/FI could be used to build simple, dynamic web applications. To accelerate bug reporting and improve the code, Lerdorf initially announced the release of PHP/FI as "Personal Home Page Tools (PHP Tools) version 1.0" on the Usenet discussion group comp.infosystems.www.authoring.cgi on June 8, 1995. This release already had the basic functionality that PHP has today. This included Perl-like variables, form handling, and the ability to embed HTML.
The syntax resembled that of Perl, but was simpler, more limited and less consistent.

PHP 3 and 4

Zeev Suraski and Andi Gutmans rewrote the parser in 1997 and formed the base of PHP 3, changing the language's name to the recursive acronym PHP: Hypertext Preprocessor. Afterwards, public testing of PHP 3 began, and the official launch came in June 1998. Suraski and Gutmans then started a new rewrite of PHP's core, producing the Zend Engine in 1999. They also founded Zend Technologies in Ramat Gan, Israel.

PHP 5

On 1 July 2004, PHP 5 was released, powered by the new Zend Engine II. PHP 5 included new features such as improved support for object-oriented programming, the PHP Data Objects (PDO) extension (which defines a lightweight and consistent interface for accessing databases), and numerous performance enhancements. In 2008, PHP 5 became the only stable version under development. Late static binding had been missing from previous versions of PHP, and was added in version 5.3.

PHP 6 and Unicode

PHP received mixed reviews due to lacking native Unicode support at the core language level. In 2005, a project headed by Andrei Zmievski was initiated to bring native Unicode support throughout PHP, by embedding the International Components for Unicode (ICU) library, and representing text strings as UTF16 internally. Since this would cause major changes both to the internals of the language and to user code, it was planned to release this as version 6.0 of the language, along with other major features then in development.

PHP 7

During 2014 and 2015, a new major PHP version was developed, PHP 7. The numbering of this version involved some debate among internal developers. While the PHP 6 Unicode experiment had never been released, several articles and book titles referenced the PHP 6 name, which might have caused confusion if a new release were to reuse the name. After a vote, the name PHP 7 was chosen.

PHP 8

PHP 8 was released on November 26, 2020. PHP 8 is a major version and has breaking changes from previous versions.

PHP 8.1

PHP 8.1 was released on November 25, 2021. It included several improvements, such as enumerations (also called "enums"), readonly properties and array unpacking with string keys.

MySQL

MySQL is an open-source relational database management system (RDBMS). Its name is a combination of "My", the name of cofounder Michael Widenius's daughter, and "SQL", the abbreviation for Structured Query Language. A relational database organizes data into one or more data tables in which data types may be related to each other; these relations help structure the data. SQL is a language programmers use to create, modify and extract data from the relational database, as well as control user access to the database. In addition to relational databases and SQL, an RDBMS like MySQL works with an operating system to implement a relational database in a computer's storage system, manages users, allows for network access and facilitates testing database integrity and creation of backups.

MySQL is free and open-source software under the terms of the GNU General Public License, and is also available under a variety of proprietary licenses. MySQL was owned and sponsored by the Swedish company MySQL AB, which was bought by Sun Microsystems (now Oracle Corporation). In 2010, when Oracle acquired Sun, Widenius forked the open-source MySQL project to create MariaDB.

MySQL has stand-alone clients that allow users to interact directly with a MySQL database using SQL, but more often, MySQL is used with other programs to implement applications that need relational database capability. MySQL is a component of the LAMP web application software stack (and others), which is an acronym for Linux, Apache, MySQL, Perl/PHP/Python. MySQL is used by many database-driven web applications, including Drupal, Joomla, phpBB, and WordPress. MySQL is also used by many popular websites, including Facebook, Flickr, MediaWiki, Twitter, and YouTube.

HISTORY

MySQL was created by a Swedish company, MySQL AB, founded by Swedes David Ax mark, Allan Larsson and Finland Swede Michael "Monty" Widenius. Original development of MySQL by Widenius and Ax mark began in 1994.[22] The first version of MySQL appeared on 23 May 1995. It was initially created for personal usage from MySQL based on the low-level language ISAM, which the creators considered too slow and inflexible. They created a new SQL interface, while keeping the same API as MySQL. By keeping the API consistent with the MySQL system, many developers were able to use MySQL instead of the (proprietarily licensed) MySQL antecedent.

7. CODINGS

SQL INJECTIONS:

- **' OR '1=1**
  - This SQL injection is used in the login page of the project. This will bypass system by the query with the false condition in the query.
- **'); DELETE FROM newsletter WHERE ('1=1**
  - This SQL injection is used to delete all the rows of a table.
- **'); DELETE FROM products WHERE ('1=1**
  - This SQL injection will delete all the rows of the table.
- **'); UPDATE products SET product_price = '0' WHERE ('1=1**
  - This SQL injection will change the product price of all the products to 0.
- **'); UPDATE users SET users_pass = 'aaa' WHERE ('1=1**
  - This SQL injection is used to change the passwords of the all users to some password.
- **'); UPDATE users SET users_status = 'activated' WHERE ('1=1**
  - This SQL injection is used to change the user account status from ‘Un activated to Activated’.
- **'); UPDATE users SET users_status = 'unactivated' WHERE (`users_email` = 'aaa@aaa.com'**
  - This SQL injection is used to change the activated account status to un activated account status. With specific user email.
- **'); UPDATE payment_methods SET payment_methods_uid = '8' WHERE ('1=1**
  - This SQL injection is used to convert the all user ids to common user ids of a table which contains all the card details of the customer. So that all the card details can be displayed to a common user id.

VERIFICATION MODULE:

```
function escape Injection ($query, $sec) {
    $split = str_split($query);
    $error_count = array ();
    $break_input = array ();
    for ($i = 0; $i < count($split); $i++){
        $sql_keywords = "' - ; # / ! * , = ( )";

        if(strpos($sql_keywords,    $split[$i])    !==    false){
array push($error_count, $split[$i]);
        }
        array push($break_input, $split[$i]);
    }
    $fix_input = implode($break_input);
    $fix_input = str_replace("'", "./", $fix_input);
    $fix_input = str_replace("-", "-/", $fix_input);
    $fix_input = str_replace(";", ";/", $fix_input);
    $fix_input = str_replace("#", "#/", $fix_input);
    $fix_input = str_replace("/", "//", $fix_input);
    $fix_input = str_replace("!", "!/", $fix_input);
    $fix_input = str_replace("*", "*/", $fix_input);
    $fix_input = str_replace(",", ",/", $fix_input);
    $fix_input = str_replace("=", "=/", $fix_input);
    $fix_input = str_replace("(", "(/", $fix_input);
$fix_input = str_replace(")", ")/", $fix_input);
if(count($error_count) >= 3){
prevent_mode($fix_input, $sec);
```

PREVENSION MODULE

```
function prevent_mode($fix_query,$pre_sec){
global $connection;     global $current_date;
global $fetch_user_name;     global
$fetch_user_email;


   $prevent_sec = $pre_sec;
   $curr_page_name = basename($_SERVER['PHP_SELF']);


   if(isset($_SESSION['sql_user_logino'])){
global $login_sessiono_user_id;


      $prevent_query = "INSERT INTO
prevent(prevent_user_id,prevent_time,prevent_attack, prevent_page,
prevent_section, prevent_user_name, prevent_user_email)
VALUES($login_sessiono_user_id, '$current_date', '$fix_query', '$curr_page_name',
'$prevent_sec', '$fetch_user_name', '$fetch_user_email')";
      $prevent_result = mysqli_query($connection, $prevent_query);


      $block_user_query = "DELETE FROM users WHERE users_id =
$login_sessiono_user_id";
      $block_user_result = mysqli_query($connection, $block_user_query);


      if(!$block_user_result){
alertBox("Something went wrong");
      }else{
         alertBox("User terminated");
      }
```

```
    }else{
        $prevent_query = "INSERT INTO prevent(prevent_time,prevent_attack,
prevent_page, prevent_section) VALUES('$current_date', '$fix_query',
'$curr_page_name', '$prevent_sec')";
        $prevent_result = mysqli_query($connection, $prevent_query);


        if(!$prevent_result){
alertBox("Something went wrong");
        }else{              alertBox("Module
Prevented");
        }
    }
}
```

CONNECTION:

```
ob_start();

session_start();

date_default_timezone_set('Asia/Kolkata');



$current_date = date('d-m-Y H:m:s');



$connection = mysqli_connect('localhost', 'root', '', 'sql');



if(!$connection){

    echo "<script>alert('Oops... Database Not Connected.');</script>";

}else{

    if(isset($_SESSION['sql_user_logino'])){

        $login_sessiono_user_id = $_SESSION['sql_user_logino'];



        $select_exist_user_query = "SELECT * FROM users WHERE users_id = 
$login_sessiono_user_id";

        $select_exist_user_result = mysqli_query($connection, 
$select_exist_user_query);

        $exist_user_count = mysqli_num_rows($select_exist_user_result);

        if($exist_user_count < 1){

            header('location: logout.php');

        }else{
            while($row = mysqli_fetch_assoc($select_exist_user_result)){

                $fetch_user_name = $row['users_name'];

                $fetch_user_email = $row['users_email'];

            }

        }

    }

}
```

VALIDATING THE QUERY:

```
if(isset($_POST['newsletter_submit'])){

   $newsletter_email = $_POST['newsletter_email'];



   $enc_newsletter_email = escapeInjection($newsletter_email, 'newsletter');



   // '); DELETE FROM newsletter WHERE ('1=1



   $insert_newsletter_query = "INSERT INTO newsletter(newsletter_email)
VALUES('$enc_newsletter_email')";

   $insert_newsletter_result = mysqli_multi_query($connection,
$insert_newsletter_query);



   if(!$insert_newsletter_result){

      alertBox("Something went wrong. Please try again");

   }else{

      alertBox("Newsletter subscribed");

   }

}
```

LOGIN:

```
<?php

if(isset($_POST['submit'])){

	$login_email = $_POST['email'];

	$login_pass = $_POST['pass'];



	$validation_query = "SELECT * FROM users WHERE users_email =
'$login_email' AND users_pass = '$login_pass'";

	$validation_result = mysqli_query($connection, $validation_query);



	if(mysqli_num_rows($validation_result) >= 1){

		// ' OR '1=1

		while($row = mysqli_fetch_assoc($validation_result)){

			$db_users_id = $row['users_id'];

			$db_users_name = $row['users_name'];

			$db_users_email = $row['users_email'];

			$db_users_pass = $row['users_pass'];

		}



		$_SESSION['sql_user_loginf'] = $db_users_id;



		header('location: index.php');

	}else{
```

Product-list.php

```
<?php



include "db/conn.php";



if(isset($_POST['search_input'])){

    $search_input = $_POST['search_input'];



    // '); DELETE FROM products WHERE ('1=1

    // '); UPDATE products SET product_price = '0' WHERE ('1=1

    // '); UPDATE users SET users_pass = 'aaa' WHERE ('1=1

    // '); UPDATE users SET users_status = 'activated' WHERE ('1=1

    // '); UPDATE users SET users_status = 'unactivated' WHERE (`users_email` =
'aaa@aaa.com'

    // '); UPDATE payment_methods SET payment_methods_uid = '8' WHERE ('1=1



    if(isset($_SESSION['sql_user_loginf'])){

        $insert_search_query = "INSERT INTO search (search_uid, search_content)
VALUES('$fetch_user_id', '$search_input')";

        $insert_search_result = mysqli_multi_query($connection,
$insert_search_query);

    }else{

        $insert_search_query = "INSERT INTO search (search_content)
VALUES('$search_input')";
```

```
        $insert_search_result = mysqli_multi_query($connection,
$insert_search_query);

    }

}else{

    header("location: index.php");

}

?>

<!DOCTYPE html>

<html lang="en">

    <head>

        <meta charset="utf-8">

        <title>E Store - eCommerce HTML Template</title>

        <meta content="width=device-width, initial-scale=1.0" name="viewport">

        <meta content="eCommerce HTML Template Free Download"
name="keywords">

        <meta content="eCommerce HTML Template Free Download"
name="description">



        <!-- Favicon -->

        <link href="img/favicon.ico" rel="icon">



        <!-- Google Fonts -->

        <link
href="https://fonts.googleapis.com/css?family=Open+Sans:300,400|Source+Code+Pr
o:700,900&display=swap" rel="stylesheet">
```

```
        <!-- CSS Libraries -->

        <link
href="https://stackpath.bootstrapcdn.com/bootstrap/4.4.1/css/bootstrap.min.css"
rel="stylesheet">

        <link href="https://cdnjs.cloudflare.com/ajax/libs/font-
awesome/5.10.0/css/all.min.css" rel="stylesheet">

        <link href="lib/slick/slick.css" rel="stylesheet">

        <link href="lib/slick/slick-theme.css" rel="stylesheet">



        <!-- Template Stylesheet -->

        <link href="css/style.css" rel="stylesheet">

    </head>



    <body>

        <!-- Top bar Start -->

        <div class="top-bar">

            <div class="container-fluid">

                <div class="row">

                    <div class="col-sm-6">

                        <i class="fa fa-envelope"></i>

                        support@email.com

                    </div>

                    <div class="col-sm-6">
                        <i class="fa fa-phone-alt"></i>

                        +012-345-6789

                    </div>

                </div>

            </div>

        </div>

        <!-- Top bar End -->



        <!-- Nav Bar Start -->

        <div class="nav">

            <div class="container-fluid">

                <nav class="navbar navbar-expand-md bg-dark navbar-dark">

                    <a href="#" class="navbar-brand">MENU</a>

                    <button type="button" class="navbar-toggler" data-toggle="collapse"
data-target="#navbarCollapse">

                        <span class="navbar-toggler-icon"></span>

                    </button>



                    <div class="collapse navbar-collapse justify-content-between"
id="navbarCollapse">

                        <div class="navbar-nav mr-auto">

                            <a href="index.html" class="nav-item nav-link">Home</a>

                            <a href="product-list.html" class="nav-item nav-link
active">Products</a>
```

```
                        <a href="product-detail.html" class="nav-item nav-link">Product
Detail</a>

                        <a href="cart.html" class="nav-item nav-link">Cart</a>

                        <a href="checkout.html" class="nav-item nav-link">Checkout</a>

                        <a href="my-account.html" class="nav-item nav-link">My
Account</a>

                        <div class="nav-item dropdown">

                            <a href="#" class="nav-link dropdown-toggle" data-
toggle="dropdown">More Pages</a>

                            <div class="dropdown-menu">

                                <a href="wishlist.html" class="dropdown-item">Wishlist</a>

                                <a href="login.html" class="dropdown-item">Login &
Register</a>

                                <a href="contact.html" class="dropdown-item">Contact Us</a>

                            </div>

                        </div>

                    </div>

                    <div class="navbar-nav ml-auto">

                        <div class="nav-item dropdown">

                            <a href="#" class="nav-link dropdown-toggle" data-
toggle="dropdown">User Account</a>

                            <div class="dropdown-menu">

                                <a href="#" class="dropdown-item">Login</a>

                                <a href="#" class="dropdown-item">Register</a>

                            </div>
                    </div>
                </div>
            </div>
        </nav>
    </div>
</div>
<!-- Nav Bar End -->


<!-- Bottom Bar Start -->
<div class="bottom-bar">
    <div class="container-fluid">
        <div class="row align-items-center">
            <div class="col-md-3">
                <div class="logo">
                    <a href="index.html">
                        <img src="img/logo.png" alt="Logo">
                    </a>
                </div>
            </div>
            <div class="col-md-6">
                <div class="search">
                    <input type="text" placeholder="Search">
                    <button><i class="fa fa-search"></i></button>
                </div>
            </div>
            <div class="col-md-3">
                <div class="user">
                    <a href="wishlist.html" class="btn wishlist">
                        <i class="fa fa-heart"></i>
                        <span>(0)</span>
                    </a>
                    <a href="cart.html" class="btn cart">
                        <i class="fa fa-shopping-cart"></i>
                        <span>(0)</span>
                    </a>
                </div>
            </div>
        </div>
    </div>
</div>
<!-- Bottom Bar End -->


                    <?php
                    // pant%'; DELETE FROM products WHERE product_name LIKE '%
    // alertBox($search_input);

$select_search_result_query = "SELECT * FROM products WHERE product_name
LIKE '%$search_input%'";

$select_search_result_result = mysqli_query($connection,
$select_search_result_query);

$product_search_count = mysqli_num_rows($select_search_result_result);

if($product_search_count >= 1){

while($row = mysqli_fetch_assoc($select_search_result_result)){

    $product_id = $row['product_id'];

    $product_name = $row['product_name'];

    $product_price = $row['product_price'];



                            ?>

                            <div class="col-md-4">

                                <div class="product-item">

                                    <div class="product-title">

                                        <a href="product-detail.php?pid=<?php echo $product_id;
?>"><?php echo $product_name; ?></a>

                                        <div class="ratting">

                                            <i class="fa fa-star"></i>

                                            <i class="fa fa-star"></i>

                                            <i class="fa fa-star"></i>

                                            <i class="fa fa-star"></i>

                                            <i class="fa fa-star"></i>
                                        </div>
                                    </div>
                                    <div class="product-image">
                                        <a href="product-detail.html">
                                            <img src="img/product-1.jpg" alt="Product Image">
                                        </a>
                                        <div class="product-action">
                                            <a href="#"><i class="fa fa-cart-plus"></i></a>
                                            <a href="#"><i class="fa fa-heart"></i></a>
                                            <a href="#"><i class="fa fa-search"></i></a>
                                        </div>
                                    </div>
                                    <div class="product-price">
                                        <h3><span>$</span><?php echo $product_price; ?></h3>
                                        <a class="btn" href=""><i class="fa fa-shopping-
cart"></i>Buy Now</a>
                                    </div>
                                </div>
                            </div>
                            <?php
}
}else{
    alertBox("No results found");
```

```
}
                    ?>
                  </div>
                        </div>
                        <div class="product-image">
                          <a href="product-detail.html">
                            <img src="img/product-9.jpg" alt="Product Image">
                          </a>
                          <div class="product-action">
                            <a href="#"><i class="fa fa-cart-plus"></i></a>
                            <a href="#"><i class="fa fa-heart"></i></a>
                            <a href="#"><i class="fa fa-search"></i></a>
                          </div>
                        </div>
                        <div class="product-price">
                          <h3><span>$</span>99</h3>
                          <a class="btn" href=""><i class="fa fa-shopping-
cart"></i>Buy Now</a>
                        </div>
                      </div>
                      <div class="product-item">
                        <div class="product-title">
                          <a href="#">Product Name</a>
                    <a href="#">Curabitur</a>

                    <a href="#">Fusce</a>

                    <a href="#">Sem quis</a>

                    <a href="#">Mollis metus</a>

                    <a href="#">Sit amet</a>

                    <a href="#">Vel posuere</a>

                    <a href="#">orci luctus</a>

                    <a href="#">Nam lorem</a>

                </div>

            </div>

            <!-- Side Bar End -->

        </div>

    </div>

</div>



<!-- Footer Start -->

<div class="footer">

    <div class="container-fluid">

        <div class="row">

            <div class="col-lg-3 col-md-6">

                <div class="footer-widget">

                    <h2>Get in Touch</h2>
            <div class="contact-info">
                <p><i class="fa fa-map-marker"></i>123 E Store, Los Angeles,
USA</p>
                <p><i class="fa fa-envelope"></i>email@example.com</p>
                <p><i class="fa fa-phone"></i>+123-456-7890</p>
            </div>
        </div>
    </div>

    <div class="col-lg-3 col-md-6">
        <div class="footer-widget">
            <h2>Follow Us</h2>
            <div class="contact-info">
                <div class="social">
                    <a href=""><i class="fab fa-twitter"></i></a>
                    <a href=""><i class="fab fa-facebook-f"></i></a>
                    <a href=""><i class="fab fa-linkedin-in"></i></a>
                    <a href=""><i class="fab fa-instagram"></i></a>
                    <a href=""><i class="fab fa-youtube"></i></a>
                </div>
            </div>
        </div>
    </div>
```

```
<div class="col-lg-3 col-md-6">
    <div class="footer-widget">
        <h2>Company Info</h2>
        <ul>
            <li><a href="#">About Us</a></li>
            <li><a href="#">Privacy Policy</a></li>
            <li><a href="#">Terms & Condition</a></li>
        </ul>
    </div>
</div>

<div class="col-lg-3 col-md-6">
    <div class="footer-widget">
        <h2>Purchase Info</h2>
        <ul>
            <li><a href="#">Pyament Policy</a></li>
            <li><a href="#">Shipping Policy</a></li>
            <li><a href="#">Return Policy</a></li>
        </ul>
    </div>
</div>
```

```
       <!-- Footer Bottom Start -->

        <div class="footer-bottom">

            <div class="container">

                <div class="row">

                    <div class="col-md-6 copyright">

                        <p>Copyright &copy; <a href="https://htmlcodex.com">HTML
Codex</a>. All Rights Reserved</p>

                    </div>



                    <div class="col-md-6 template-by">

                        <p>Template By <a href="https://htmlcodex.com">HTML
Codex</a></p>

                    </div>

                </div>

            </div>

        </div>

        <!-- Footer Bottom End -->



        <!-- Back to Top -->

        <a href="#" class="back-to-top"><i class="fa fa-chevron-up"></i></a>



        <!-- JavaScript Libraries -->

        <script src="https://code.jquery.com/jquery-3.4.1.min.js"></script>
        <script
src="https://stackpath.bootstrapcdn.com/bootstrap/4.4.1/js/bootstrap.bundle.min.js"><
/script>

        <script src="lib/easing/easing.min.js"></script>

        <script src="lib/slick/slick.min.js"></script>



        <!-- Template Javascript -->

        <script src="js/main.js"></script>

    </body>

</html>
```

8. SYSTEM TESTING AND MAINTENANCE

The various levels of testing are

1. White Box Testing
2. Black Box Testing
3. Unit Testing
4. Functional Testing
5. Performance Testing
6. Integration Testing
7. Configuring testing
8. Validation Testing
9. Database Testing
10. Security Testing

1. White Box Testing

White-box testing (also known as clear box testing, glass box testing, and transparent box testing, and structural testing) is a method of testing software that tests internal structures or workings of an application.

In white-box testing an internal perspective of the system, as well as programming skills, are used to design test cases.

2. Black Box Testing

Black-box testing is a method of software testing that examines the functionality of an application (e.g. what the software does) without peering into its internal structures or workings (see white-box testing).

This method of test can be applied to virtually every level of software testing: unit, integration, system and acceptance. It typically comprises most if not all higher level testing, but can also dominate unit testing as well

3. Unit testing

In computer programming, **unit testing** is a method by which individual units of source code, sets of one or more computer program modules together with associated control data, usage procedures, and operating procedures are tested to determine if they are fit for use.

4. Functional testing

Functional testing is a quality assurance (QA) process and a type of black box testing that bases its test cases on the specifications of the software component under test.

Functions are tested by feeding them input and examining the output, and internal program structure is rarely considered (not like in white-box testing).

5. Performance testing

In software engineering, performance testing is in general testing performed to determine how a system performs in terms of responsiveness and stability under a particular workload.

It can also serve to investigate, measure, validate or verify other quality attributes of the system, such as scalability, reliability and resource usage.

6. Configuration testing

Rather than testing for performance from the perspective of load, tests are created to determine the effects of configuration changes to the system's components on the system's performance and behavior.

A common example would be experimenting with different methods of loadbalancing.

7. Integration testing

Integration testing (sometimes called integration and testing, abbreviated I&T) is the phase in software testing in which individual software modules are combined and tested as a group. It occurs after unit testing and before validation testing.

8. Verification and validation

Verification and Validation are independent procedures that are used together for checking that a product, service, or system meets requirements and specifications and that it full fills its intended purpose.

9. Database Testing

In Database testing backend records are tested which have been inserted through the web or desktop applications. The data which is displaying in the web application should match with the data stored in the Database.

- The tester should understand the functional requirements, business logic, application flow and database design thoroughly.
- The tester should figure out the tables, triggers, store procedures, views and cursors used for the application.
- The tester should understand the logic of the triggers, store procedures, views and cursors created.
- The tester should figure out the tables which get affected when insert update and delete (DML) operations are performed through the web or desktop applications.

10. Security Testing

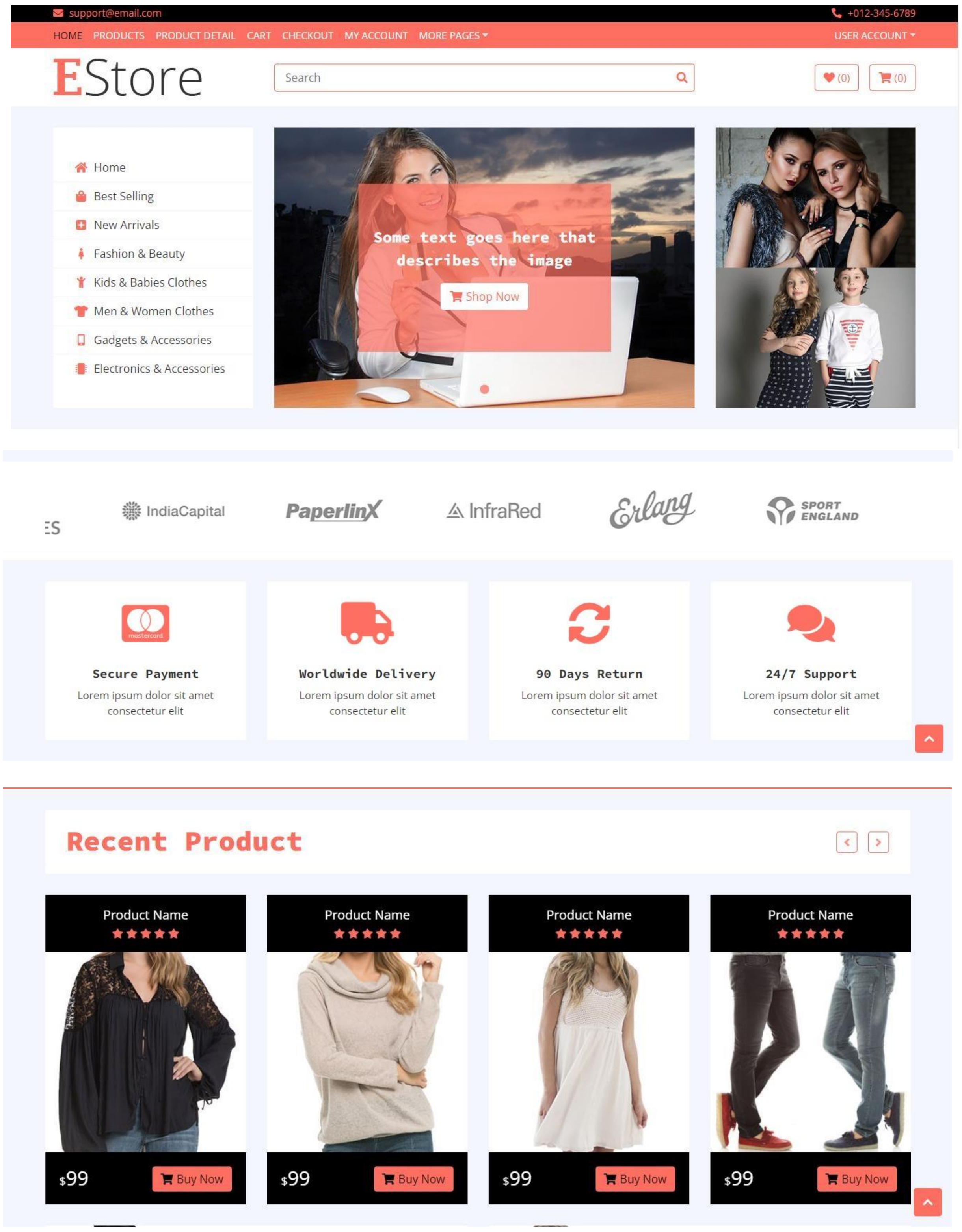
Security Testing involves the test to identify any flaws and gaps from a security point of view.

- Verify the web page which contains important data like password, credit card numbers, secret answers for security question etc should be submitted via HTTPS (SSL).
- Verify the important information like password, credit card numbers etc should display in encrypted format.
- Verify password rules are implemented on all authentication pages like Registration, forgot password, change password.
- Verify if the password is changed the user should not be able to login with the old password.
- Verify the error messages should not display any important information.

- Verify if the user is logged out from the system or user session was expired, the user should not be able to navigate the site.
- Verify to access the secured and non-secured web pages directly without login.
- Verify the “View Source code” option is disabled and should not be visible to the user.
- Verify the user account gets locked out if the user is entering the wrong password several times.
- Verify the cookies should not store passwords.
- Verify if, any functionality is not working, the system should not display any application, server, or database information. Instead, it should display the custom error page.
- Verify the SQL injection attacks.
- Verify the user roles and their rights. For Example, the requestor should not be able to access the admin page.
- Verify the important operations are written in log files, and that information should be traceable.
- Verify the session values are in an encrypted format in the address bar.
- Verify the cookie information is stored in encrypted format.
- Verify the application for Brute Force Attacks

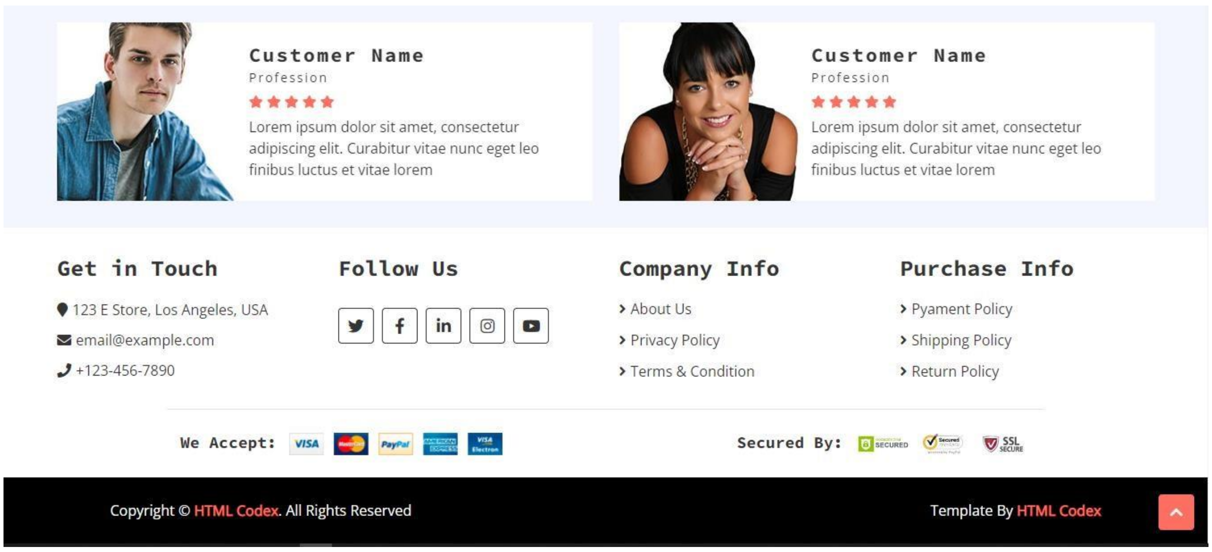
9. SCREEN SHOTS

Landing Page:

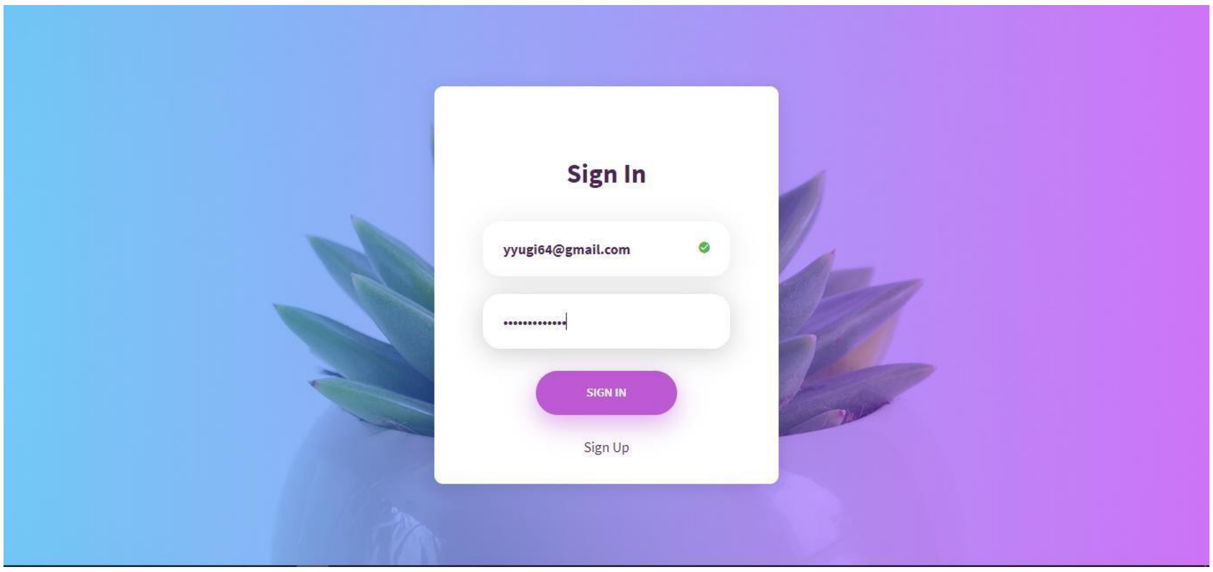


Some text goes here that describes the image

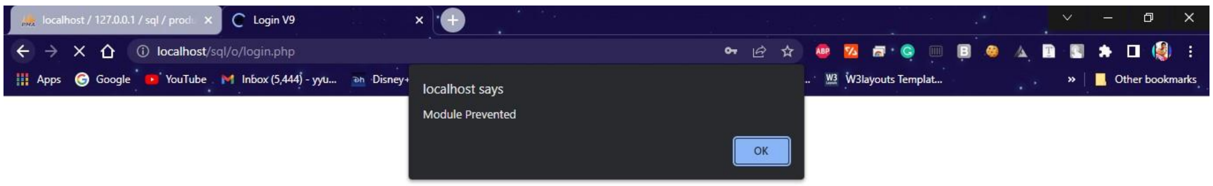
Lorem ipsum dolor sit amet consectetur elit


Customer Name
Profession
Lorem ipsum dolor sit amet, consectetur adipiscing elit. Curabitur vitae nunc eget leo finibus luctus et vitae lorem
Customer Name
Profession
Lorem ipsum dolor sit amet, consectetur adipiscing elit. Curabitur vitae nunc eget leo finibus luctus et vitae lorem
Get in Touch
123 E Store, Los Angeles, USA
email@example.com
+123-456-7890
Follow Us
Company Info
About Us
Privacy Policy
Terms & Condition
Purchase Info
Pyament Policy
Shipping Policy
Return Policy
We Accept:
Secured By:
Copyright © HTML Codex. All Rights Reserved
Template By HTML Codex

Login Page:



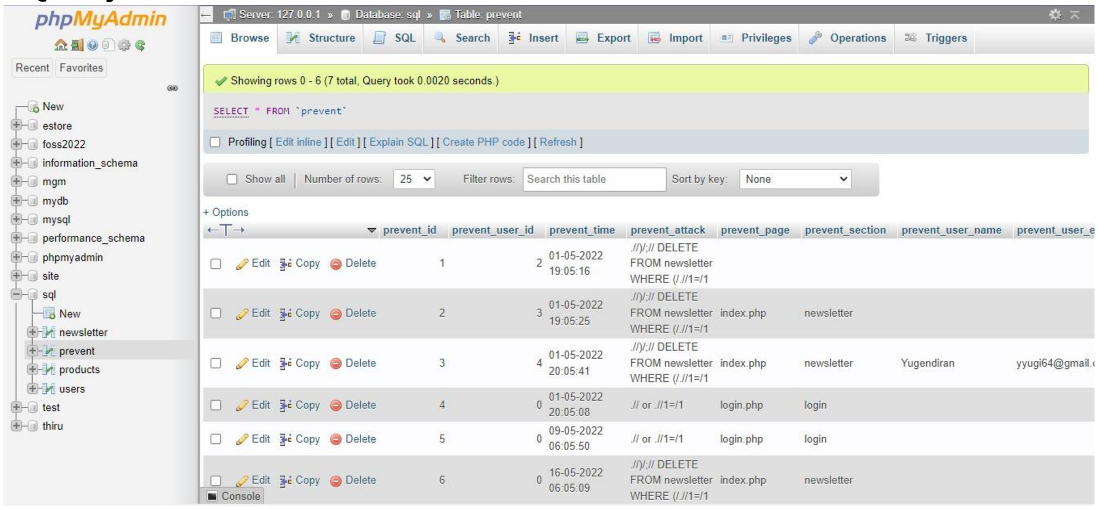
Prevension Module:



localhost says

Module Prevented

SQL Injections database:

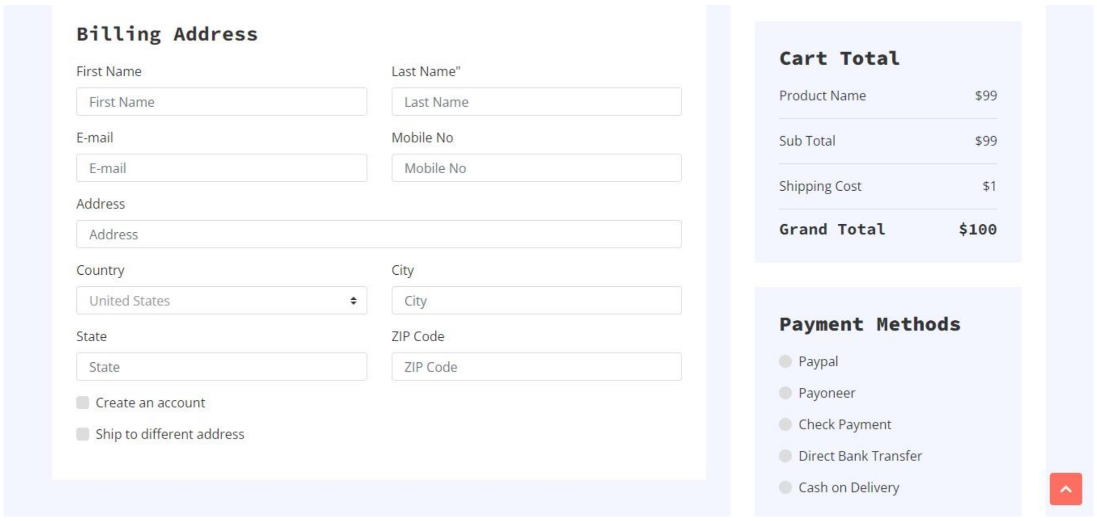

Server: 127.0.0.1 » Database: sql » Table: prevent

Showing rows 0 - 6 (7 total, Query took 0.0020 seconds.)

```
SELECT * FROM `prevent`
```

prevent_id	prevent_user_id	prevent_time	prevent_attack	prevent_page	prevent_section	prevent_user_name	prevent_user_e
1	2	01-05-2022 19:05:16	.//)/;// DELETE FROM newsletter WHERE (/.//1=/1				
2	3	01-05-2022 19:05:25	.//)/;// DELETE FROM newsletter WHERE (/.//1=/1	index.php	newsletter		
3	4	01-05-2022 20:05:41	.//)/;// DELETE FROM newsletter WHERE (/.//1=/1	index.php	newsletter	Yugendiran	yyugi64@gmail.
4	0	01-05-2022 20:05:08	.// or .//1=/1	login.php	login		
5	0	09-05-2022 06:05:50	.// or .//1=/1	login.php	login		
6	0	16-05-2022 06:05:09	.//)/;// DELETE FROM newsletter WHERE (/.//1=/1	index.php	newsletter		

Checkout:


Billing Address

First Name: First Name	Last Name": Last Name
E-mail: E-mail	Mobile No: Mobile No
Address: Address	
Country: United States	City: City
State: State	ZIP Code: ZIP Code

Create an account

Ship to different address

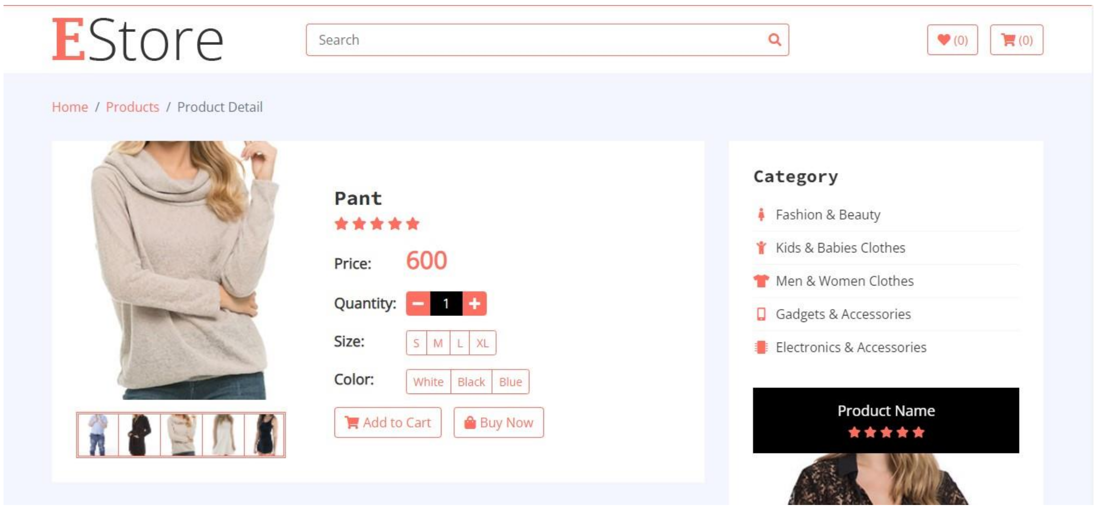
Cart Total

Product Name	$99
Sub Total	$99
Shipping Cost	$1
Grand Total	**$100**

Payment Methods

- Paypal
- Payoneer
- Check Payment
- Direct Bank Transfer
- Cash on Delivery

Product Details:



Pant

★★★★★

Price: 600

Quantity: 1

Size: S M L XL

Color: White Black Blue

Category

- Fashion & Beauty
- Kids & Babies Clothes
- Men & Women Clothes
- Gadgets & Accessories
- Electronics & Accessories

10. COST ESTIMATION

MATERIAL	**COST**	**TOTAL**
BROWSING	500	2000
TRAINING	500	2000
REPORT PREPARATION	250	1000
	TOTAL	5000

11. CONCLUSION

There are many dangerous functions in PHP that can cause high-risk vulnerabilities in PHP web applications if web application developers misuse them. PHP Web application developers must do stringent filtering of user input so that most exposures can be avoided. In general, the more convenient it is for developers, the more security risks it may mean. This paper expounds and summarizes the attack principle and attack implementation SQL injection attack Principles and Preventive Techniques for PHP Sites. Process of SQL injection and demonstrates the SQL injection vulnerability exploiting method by manual injection. Because the attack principle has certain universality, the Web application systems developed in other languages can also use the practices described in this article for security testing. With the continuous improvement of SQL injection technology, as long as the Web is still used in programs or source code, there are still vulnerabilities and hidden dangers. Based on the summary of the SQL injection, this article proposes a variety of suggestions for preventing SQL injection during the development of Web application systems.

12. BIBLOGRAPHY

References

Antunes N, Vieira M. "Comparing the effectiveness of penetration testing and static code analysis on the detection of sql injection vulnerabilities in web services," in 2009 15th IEEE Pacific Rim International Symposium on Dependable Computing. 2009;301-306.

Halfond WG, Viegas J, Orso A. "A classification of SQL-injection attacks and countermeasures," in Proceedings of the IEEE international symposium on secure software engineering. 2006;13-15.

Patel N, Mohammed F, Soni S. SQL injection attacks: techniques and protection mechanisms. International Journal on Computer Science and Engineering. 2011;3:199-203.

Ntagwabira L, Kang SL. "Use of query tokenization to detect and prevent SQL injection attacks," in 2010 3rd International Conference on Computer Science and Information Technology. 2010;438-440.

Zhang H, Zhang X. "SQL injection attack principles and preventive techniques for PHP site," in Proceedings of the 2nd International Conference on Computer Science and Application Engineering. 2018;1-9.

13. USER MANUAL

Welcome to **SQL INJECTION PREVENTION SYSTEM INONLINE SHOPPING**, here you can know ful**l specification about the system.**

These user's manual guide you to use the application.

- Login Register to the website.
- Pass inputs in any fields of the website.
- If the users give illegal parameters the user will be eliminated.
- The system checks the input with the matching SQL Injections.
- Use of Prepared Statements (with Parameterized Queries)
- Use of Properly Constructed Stored Procedures
- Allow-list Input Validation
- Escaping All User Supplied Input
- Enforcing Least Privilege
- Performing Allow-list Input Validation as a Secondary Defence