**INDEX**

* **INTRODUCTION…………………………………02**
* **SYSTEM REQUIREMENT………………………03**
  + **SOFTWARE REQUIREMENT**
  + **HARDWARE REQUIREMENT**
* **INTRODUCTION TO FRONTEND…………….04**
  + **HTML………………………………………07-09**
  + **CSS……………………………………….....10**
  + **JAVA SCRIPT……………………………..11**
* **INTRODUCTION TO BACK END……………….12-13**
  + **MY SQL………………………………….14-15**
  + **CLIENT SERVER ARCHITECTURE...16**
* **CODING:…………………………………………….17-37**
* **SCREEN SHOT……………………………………...38-39**
* **REFRENCES………………………………………...40**

**CHAPTER-1**

**INTRODUCTION**

**LOGIN & SIGNUP PAGE**

My project is on Login and Signup page I am using in My project HTML, CSS, and JavaScript This is a simple frontend Page in My project I described that How to create a Login & Signup page with the help of HTML, CSS and Javascript.

**My Specialization**

In My website we can register with username, email, and password and after register we can also login with the help of email id and password. My login and signup website store data with the help of php and using mySql in the system.

In my Login and signup page the password stored in hash values therefore it is not easy to understand for any one it use the hash function of the PHP and login time also use hash function and match the store password and user entered password if the email id and password matched then redirect on welcome page. Welcome page show only “welcome – user\_email” Text.

**CHAPTER-2**

**SYSTEM REQUIREMENT**

**SYSTEM SPECIFICATIONS**

**HARDWARE DESCRIPTION**

The selection of hardware is very important in the existence and proper working of any

Software. When selecting hardware, the size and requirements are also important.

Minimum Requirements:

Processor - Pentium II class, 450MHz

RAM - 128MB

Hard Disk Drive - 3GB

Video - 800X600, 256 colors

CD-ROM - Required

The proposed System is developed on:

Processor - INTEL Pentium 4

RAM - 512MB

Hard Disk Drive - 40GB

Key Board - Standard 101/102 or Digi Sync Family

Monitor - Display Panel (1024 X 764)

Display Adapter - Trident Super VGA

Network Adapter SMC Ethernet Card Elite 16 Ultra

Mouse - Logitech Serial Mouse

**SOFTWARE DESCRIPTION**

Operating System - Windows10

Front- End - HTML, CSS, JAVASCRIPT

Back- End - MY SQL SERVE

Browser - BRAVE BROWSER

**CHAPTER-3**

**INTRODUCTION TO FRONTEND**

**Introducing Web Application:-**

Among all technologies, the Internet has been the fastest growing technology. Ever since its inception, the Internet has evolved exponentially. In the recent years, it has changed the way business is conducted. Organizations are increasingly becoming dependent on the Internet for sharing and accessing information. This Internet boom has changed the focus of application development from stand-alone applications to distributed Web applications.

The content of a Web application that consists of only HTML pages is static. It does not respond dynamically to the actions performed by users. To respond dynamically to user requests, you can use client side and server side scripting in addition to HTML pages.

Components of Dynamic Web Applications

A dynamic Web application can have client side scripting, server side scripting, or both.

Client side scripting enables you to develop Web pages that can dynamically respond to user input without having to interact with a Web server. For example, if there is a Web application that requires users to enter the user name and the password before displaying the home page, you need to ensure that a user does not leave the user name and password fields blank. To check whether a user has left the user name and password fields blank, you can write client side scripts that execute on the client computer. A client side script helps reduce network traffic because it does not need to interact with a Web server to provide dynamic response to user input. In addition to providing dynamic content, client side scripting speeds up the response time of a Web application. This happens because a Web server is not overloaded with the job of processing the client side script for every client. Scripting languages, such as VBScript and JavaScript, are used to write client side scripts.

Server side scripting provides dynamic content to users based on the information stored in a remote location, such as a back-end database. It includes code written in server side scripting languages, such as Active Server Pages (ASP) and Java Server Pages (JSP).A server side script is executed on a Web server. When a browser requests for an HTML page containing a server side script, the Web server to which the request is sent, first processes the script and then sends the results to the browser. For this reason, any data that should come from the Web server needs to be processed using server side scripting. For example, if a Web page is to display the current time of the system in which the Web site is hosted, you need to implement server side scripting. If you use client side scripting, then each of the browsers requesting for the file containing the script will display the current time of the system in which the browser is located.

An optimal dynamic Web page combines both client side scripting and server side scripting. The general rules for using client side scripting and server side scripting are:

Any access to data that resides on the client computer or is dependent on the client computer should be implemented using client side scripting. For example, if you need to know the type of browser being used to access a Web site, you should use client side scripting to retrieve the information.

Any access to data that resides on a Web server or a remote computer, such as a back-end database should be implemented using server side scripting. For example, if you want to retrieve the product details of an online shopping store from a back-end database, you should use server side scripting to retrieve the information.

PHP has come a long way since its birth in the mid-1990's. From humble beginnings to becoming one of the most prominent languages powering the web, the evolution of PHP is a geek's fairy tale. Mind you, such explosive growth was no easy task. Those of you interested in briefly seeing how PHP grew to become what it is today, read on. If you'd like to touch a piece of Internet history, you can find old releases of PHP in the

HTML documents. The advantage of using templates is that you can dynamically insert the content retrieved from data sources, such as back-end databases and text files, into an HTML document before the HTML document is displayed to users. For this reason, the information need not be changed manually as and when the contents retrieved from the data source change.

### PHP Tools, FI, Construction Kit, and PHP/FI

PHP as it's known today is actually the successor to a product named PHP/FI. Created in 1994 by Rasmus lerdorf, the very first incarnation of PHP was a simple set of Common Gateway Interface (CGI) binaries written in the C programming language. Originally used for tracking visits to his online resume, he named the suite of scripts "Personal Home Page Tools," more frequently referenced as "PHP Tools." Over time, more functionality was desired, and Rasmus rewrote PHP Tools, producing a much larger and richer implementation. This new model was capable of database interaction and more, providing a framework upon which users could develop simple dynamic web applications such as guest books. In June of 1995, asmus [» released](http://groups.google.com/group/comp.infosystems.www.authoring.cgi/msg/cc7d43454d64d133) the source code for PHP Tools to the public, which allowed developers to use it as they saw fit. This also permitted - and encouraged - users to provide fixes for bugs in the code, and to generally improve upon it.

**PHP Framework:-**

Simple MVC Framework is a PHP 5.3+ MVC Framework. It's designed to be lightweight and modular, allowing developers to build better and easy to maintain code with PHP.

The base framework comes with a range of helper classes, Classes can easily be added at any stage of development.

.**Features of PHP:-**

PHP 5.3.0 offers a wide range of new features:

* Support for [namespaces](http://php.net/manual/en/language.namespaces.php) has been added.
* Support for [Late Static Bindings](http://php.net/manual/en/language.oop5.late-static-bindings.php) has been added.
* Support for [jump labels](http://php.net/manual/en/control-structures.goto.php) (limited goto) has been added.
* Support for native [Closures](http://php.net/manual/en/functions.anonymous.php) (Lambda/Anonymous functions) has been added.
* There are two new magic methods, [\_\_callStatic()](http://php.net/manual/en/language.oop5.overloading.php#object.callstatic) and [\_\_invoke()](http://php.net/manual/en/language.oop5.magic.php#object.invoke).
* [Nowdoc](http://php.net/manual/en/language.types.string.php#language.types.string.syntax.nowdoc) syntax is now supported, similar to [Heredoc](http://php.net/manual/en/language.types.string.php#language.types.string.syntax.heredoc) syntax, but with single quotes.
* It is now possible to use [Heredoc](http://php.net/manual/en/language.types.string.php#language.types.string.syntax.heredoc)s to initialize static variables and class properties/constants.
* [Heredoc](http://php.net/manual/en/language.types.string.php#language.types.string.syntax.heredoc)s may now be declared using double quotes, complementing the [Nowdoc](http://php.net/manual/en/language.types.string.php#language.types.string.syntax.nowdoc) syntax.
* [Constants](http://php.net/manual/en/language.constants.syntax.php) can now be declared outside a class using the const keyword.
* The [ternary](http://php.net/manual/en/language.operators.comparison.php#language.operators.comparison.ternary) operator now has a shorthand form: ?:.
* The HTTP stream wrapper now considers all status codes from 200 to 399 to be successful.
* Dynamic access to static methods is now possible

1. application. For example, you can create custom controls; extend the default Hypertext Markup Language (HTTP) pipeline. Note that the path through which the data flows from the client to the server is called the HTTP pipeline

# HTML  INTRODUCATION

HTML is a **markup** language for **describing** web documents (web pages).

* HTML stands for **H**yper **T**ext **M**arkup **L**anguage
* A markup language is a set of **markup tags**
* HTML documents are described by **HTML tags**
* Each HTML tag **describes** different document content

## HTML Example

### A small HTML document: <html> <head> <title>Page Title</title> </head> <body> <h1>My First Heading</h1> <p>My first paragraph.</p> </body> </html>

* The **DOCTYPE** declaration defines the document type to be HTML
* The text between **<html>** and **</html>** describes an HTML document
* The text between **<head>** and **</head>** provides information about the document
* The text between **<title>** and **</title>** provides a title for the document
* The text between **<body>** and **</body>** describes the visible page content
* The text between **<h1>** and **</h1>** describes a heading
* The text between **<p>** and **</p>** describes a paragraphUsing this description, a web browser can display a document with a heading and a paragraph.

## HTML Tags

HTML tags are **keywords** (tag names) surrounded by **angle brackets**:

<tagname>content</tagname>

* HTML tags normally come **in pairs** like <p> and </p>
* The first tag in a pair is the **start tag,** the second tag is the **end tag**
* The end tag is written like the start tag, but with a **slash** before the tag name.

|  |  |
| --- | --- |
|  | The start tag is often called the **opening tag**. The end tag is often called  the **closing tag**. |

## HTML Page Structure

Below is a visualization of an HTML page structure:

<html>

<head>

<title>Page title</title>

</head>

<body>

<h1>This is a heading</h1>

<p>This is a paragraph.</p>

<p>This is another paragraph.</p>

</body>

</html>.

There are different document types on the web.To display a document correctly, the browser must know both type and version.The doctype declaration is not case sensitive. All cases are acceptable:

### HTML5

<!DOCTYPE html>

### HTML 4.01

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">

### XHTML 1.0

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

## HTML Versions

Since the early days of the web, there have been many versions of HTML:

|  |  |
| --- | --- |
| **Version** | **Year** |
| HTML | 1991 |
| HTML 2.0 | 1995 |
| HTML 3.2 | 1997 |
| XHTML | 2000 |
|  |  |
|  |  |

# CSS INTRODUCATION

# CSS stands for Cascading Style Sheets

* CSS describes **how HTML elements are to be displayed on screen, paper, or in other media**
* CSS **saves a lot of work**. It can control the layout of multiple web pages all at once
* External style sheets are stored in **CSS files**

## CSS Demo - One HTML Page - Multiple Styles!

Here we will show one HTML page displayed with four different style sheets. Click on the "Style sheet 1", "Style sheet 2", "Style sheet 3", "Style sheet 4" links below to see the different styles:

## CSS Solved a Big Problem

HTML was NEVER intended to contain tags for formatting a web page!

HTML was created to **describe the content** of a web page, like:

<h1>This is a heading</h1>

<p>This is a paragraph.</p>

When tags like <font>, and color attributes were added to the HTML 3.2 specification, it started a nightmare for web developers. Development of large websites, where fonts and color information were added to every single page, became a long and expensive process.To solve this problem, the World Wide Web Consortium (W3C) created CSS.

CSS removed the style formatting from the HTML page!

**JAVASCRIPT INTRODUCTION**

JavaScript is the most popular programming language in the world.

This page contains some examples of what JavaScript can do.

## JavaScript Can Change HTML Content

One of many HTML methods is **getElementById()**.

This example uses the method to "find" an HTML element (with id="demo"), and changes the element content (**innerHTML**) to "Hello JavaScript":

### Example

document.getElementById("demo").innerHTML = "Hello JavaScript";

## JavaScript Can Change HTML Attributes

This example changes an HTML image, by changing the src attribute of an <img> tag:

## JavaScript Can Change HTML Styles (CSS)

Changing the style of an HTML element, is a variant of changing an HTML attribute:

## Did You Know

|  |  |
| --- | --- |
|  | JavaScript and Java are completely different languages, both in concept and design. JavaScript was invented by Brendan Eich in 1995, and became an ECMA standard in 1997. ECMA-262 is the official name. ECMAScript 6 (released in June 2015) is  The latest official version of JavaScript. |

**CHAPTER-4**

**BACKEND**

Microsoft SQL Server is an application used to create computer databases for the [Microsoft Windows](http://www.functionx.com/sqlserver/Lesson01.htm) family of server operating systems. It provides an environment used to generate databases that can be accessed from [workstations](http://www.functionx.com/sqlserver/Lesson01.htm), the web, or other media such as a personal digital assistant (PDA).

Probably before using a database, you must first have one. A database is primarily a group of [computer](http://www.functionx.com/sqlserver/Lesson02.htm) files that each has a name and a location. Just as there are different ways to connect to a server, in the same way, there are also different ways to create a database.

SQL is short for Structured Query Language and is a widely used database language, providing means of data manipulation (store, retrieve, update, delete) and database creation. Almost all modern Relational Database Management Systems like MS SQL Server, Microsoft Access, MSDE, Oracle, DB2, Sybase, Myself, Postures and Informix use SQL as standard database language. Now a word of warning here, although all those RDBMS use SQL, they use different SQL dialects. For example MS SQL Server specific version of the SQL is called T-SQL, Oracle version of SQL is called PL/SQL, MS Access version of SQL is called JET SQL, etc.

Microsoft SQL Server is a Relational Database Management System (RDBMS) designed to run on platforms ranging from laptops to large multiprocessor servers. SQL Server is commonly used as the backend system for websites and corporate CRMs and can support thousands of concurrent users.

SQL Server comes with a number of tools to help you with your database administration and programming tasks.

SQL Server is much more robust and scalable than a desktop database management system such as Microsoft Access. Anyone who’s ever tried using Access as a backend to a website will probably be familiar with the errors that were generated when too many users tried to access the database! Although SQL Server can also be run as a desktop database system, it is most commonly used as a server database system.

**Server Database Systems**

Server based database systems are designed to run on a central server, so that multiple users can access the same data simultaneously. The users normally access the database through an application.

For example, a website could store all its content in a database. Whenever a visitor views an article, they are retrieving data from the database. As you know, websites aren't normally limited to just one user. So, at any given moment, a website could be serving up hundreds or even

**Database** A database is a coherent collection of data with some inherent meaning, designed, built and populated with data for a specific purpose. It stores data that is useful to user. This data is only a part of the entire data available in the world around us.

**Database Management System**

A Database Management System (DBMS) consists of a collection of interrelated data and a set of programs to access those data. The primary goal of DBMS is to provide an environment that is both convenient and efficient to use in retrieving and storing database information.

**Objectives of database: -**

The objectives of database are: -

It reduces data redundancy and inconsistency.

* Eliminates difficulty in accessing data.
* Helps in isolation of data so that there is no difficulty in writing new applications due to scattered data.

**View of data (data abstraction) -**

The major purpose of a database system is to provide users with an abstract view of data. For usable system the retrieve of data must be efficient. Since many database-system users are not computer trained, the complexity of data has to be hidden through several levels of abstraction, to simplify users interaction with the system. The data has been classified into following levels of data abstraction -

* Physical level - The lowest level of abstraction describes how the data are actually stored. The record or data is described in term of block of consecutive storage locations like words or bytes.
* Logical level - The next level of abstraction describes what data are stored in the database, and what relationships exist among those data.

**Drawback of DBMS**

In the early days of computing the DBMS, used to manage data, were of the Hierarchic or Network model. When these were placed onto network operating systems and multiple users began to access table data concurrently the DBMS responded to these users very sluggishly and totally stable when the number of users exceeded four or five. This caused commercial application developers to abandon the use of DBMS.

###### INTRODUCTION OF SQL SERVER

* Introduction to SQL and Its Tools:

The SQL SERVER product is primarily divided into:

* SQL Server Tools
* SQL Client Tool
* SQL Server:

SQL SERVER is a company that produces the most widely used, Server based Multi user RDBMS. The SQL SERVER Server is a program installed on the Server’s hard disk driver. This program must be loaded in RAM so that it can process user requests.

The SQL SERVER Server product is either called SQL SERVER Workgroup Server Or SQL SERVER Enterprise Server

The functionality of both these products is identical. However, the SQL SERVER Workgroup Server restricts the number of concurrent users who can query the Server. SQL SERVER Enterprise Server has no such restrictions. Enter product must be loaded on a multi user operating system.

The SQL SERVER Server takes care of the following:

* Updating the database.
* Retrieving information from the database.
* Accepting query language statements.
* Enforcing security specifications.
* Enforcing data integrity specifications.
* Enforcing transaction consistency.
* Managing data sharing.
* Optimizing queries.
* Managing system catalogs.
* **SQL SERVER Client Tools**:-

Once the SQL SERVER engine is loaded into the server’s memory, users would have to log into the engine to get work done. SQL SERVER Corporation has several client-based tools that facilitate this. The client tool most commonly used for Commercial Application Development is called SQL SERVER Developer 2000*.*

SQL\*Plus is separate SQL SERVER client-side tool. It is a product that works on Microsoft Windows 95 and Windows NT both of which are standard client based GUI operating systems.

* **What is SQL Used for:**
* Using SQL one can create and maintain data manipulation objects such as table, views, sequence etc. These data manipulation objects will be created and stored on the server's hard disk drive, in a table space, to which the user has been assigned.

Once these data manipulation objects are created, they are used extensively in commercial applications.

**DML, DCL, DDL:**-

In addition to the creation of data manipulation objects, the actual manipulation of data within these objects is done using SQL.

The SQL sentences that are used to create these objects are called DDL's or Data Definition Language. The SQL sentences used to manipulate data within these objects are called DML's or Data Manipulation Language.

The SQL sentences, which are used to control the behavior of these objects, are called DCL's or Data Control Language.

Hence, once access to the SQL\*Plus tool is available and SQL syntax is known, the creation of data storage and the manipulation of data within the storage system, required by commercial applications, is possible

**SQL SERVER and Client/Server :-**

SQL SERVER Corporation's reputation as a database company is firmly established in its full-featured, high-performance RDBMS server. With the database as the cornerstone of its product line, SQL SERVER has evolved into more than just a database company, complementing its RDBMS server with a rich offering of well-integrated products that are designed specifically for distributed processing and client/server applications. As SQL SERVER's database server has evolved to support large-scale enterprise systems for transaction processing and decision support, so too have its other products, to the extent that SQL SERVER can provide a complete solution for client/server application development and deployment. This chapter presents an overview of client/server database systems and the SQL SERVER product architectures that support their implementation.

* The components of a client/server system are distributed across more varied types of processors. There are many more software components that manage client, network, and server functions, as well as an array of infrastructure layers, all of which must be in
* The complexity of GUI applications far outweighs that of their character-based predecessors. GUIs are capable of presenting much more information to the user and providing many additional navigation paths to elements of the interface.

**Databases in a Client/Server Architecture**

Client/server technologies have changed the look and architecture of application systems in two ways. Not only has the supporting hardware architecture undergone substantial changes, but there have also been significant changes in the approach to designing the application logic of the system. place and configured to be compatible with each other.

Prior to the advent of client/server technology, most SQL SERVER applications ran on a single node. Typically, a character-based SQL\*Forms application would access a database instance on the same machine with the application and the RDBMS competing for the same CPU and memory resources. Not only was the system responsible for supporting the entire database processing, but it was also responsible for executing the application logic. In addition, the system was burdened with all the I/O processing for each terminal on the system; the same processor that processed database requests and application logic controlled each keystroke and display attribute.

Client/server systems change this architecture considerably by splitting the entire interface management and much of the application processing from the host system processor and distributing it to the client processor.

Combined with the advances in hardware infrastructure, the increased capabilities of RDBMS servers have also contributed to changes in the application architecture. Prior to the release of SQL SERVER7, SQL SERVER's RDBMS was less sophisticated in its capability to support the processing logic necessary to maintain the integrity of data in the database. For example, primary and foreign key checking and enforcement was performed by the application. As a result, the database was highly reliant on application code for enforcement of business rules and integrity, making application code bulkier and more complex. Figure 2.1 illustrates the differences between traditional host-based applications and client/server applications. Client/server database applications can take advantage of the SQL SERVER7 server features for implementation of some of the application logic.

**CHAPTER-5**

# CODING

Here we use the Dreamweaver for coding.

# Dreamweaver Tutorial:

# How to Create a Website with Dreamweaver CS3 (Part 1)

Adobe Dreamweaver Creative Suite 3 (CS3), formerly known as Macromedia Dreamweaver, is a fully-featured commercial web editor that allows you to create, build and manage complex websites. The editor is a [What-You-See-Is-What-You-Get (WYSIWYG) web editor](http://www.thefreecountry.com/webmaster/htmleditors.shtml), which means that you can design your web page visually and whatever you see on the screen is what your visitors will see when they visit your website. Dreamweaver generates standards-compliant code for your website which means your website will not become "broken" every time a new version of a web browser is released. For the technically inclined, the HTML and CSS code that it creates for your website will [validate correctly](http://www.thesitewizard.com/webdesign/htmlvalidation.shtml).

This tutorial guides you through the steps of creating your first website using Dreamweaver.

## What You Will Need

### Dreamweaver CS3

You will need Dreamweaver CS3 (obviously). The tutorial assumes that you are using Dreamweaver CS3. There are versions of Dreamweaver for both Windows and Mac OS X — either version will do fine. For the most part, both versions work in the same way.

### A Web Hosting Account

You will need a web host to publish your pages to. For the complete beginner, a web host is (loosely speaking) a company which has computers that are permanently connected to the Internet. After you design your web page(s), you will need to transfer your pages to your web host's computer (called a web server), so that the rest of the world can see it. There are numerous web hosts around — you can find a [list of cheap web hosts](http://www.thefreecountry.com/webhosting/budget1.shtml) on<http://www.thefreecountry.com/webhosting/budget1.shtml>

There are other things involved in getting your first web site up and running, such as [getting your own domain name](http://www.thesitewizard.com/archive/registerdomain.shtml), [making your website search engine friendly](http://www.thesitewizard.com/sitepromotion/search-engine-friendly.shtml) and [promoting your web site](http://www.thesitewizard.com/sitepromotion/index.shtml). This tutorial however does not deal with those matters — it is strictly about designing (creating) and publishing ([uploading](http://www.thesitewizard.com/faqs/glossary.shtml#uploading)) your website using Dreamweaver. If you are a total beginner, you may want to consult my article [How to Start / Create Your Own Website: The Beginner's A-Z Guide](http://www.thesitewizard.com/gettingstarted/startwebsite.shtml) for an overview of the entire process and all the things you will need.

## Overall Goals of the Dreamweaver Tutorial

By the end of this tutorial, you will have set up a working website with multiple pages, including a main page, a feedback form, an About Us page, and a Site Map. Your pages will contain a sophisticated navigation menu bar, images, multiple columns, a form, links to other pages within your site, links to other sites, text in different font sizes, etc. In other words, you will have a fully functional website.

More importantly, you will know how to use Dreamweaver to create, design and publish your site so that you can design new sites any time you want.

## Goal of this Chapter

In this chapter, you will learn to create a rudimentary two-column web page (like [this web page](http://www.thesitewizard.com/gettingstarted/dreamweaver1.shtml) that you are currently reading) and publish it so that it can be accessed on the Internet. By the end of this chapter, you will be viewing your web page on the Internet with your favourite web browser.

Note that this is a hands-on tutorial. To benefit from it, in fact, to even understand it, you need to follow the steps as I describe them. The practical nature of this guide makes it difficult to follow or understand if you're not doing the things mentioned.

## Setting Up Your Website in Dreamweaver

Start up Dreamweaver.

You will be greeted with a window with a top-half looks something like the picture above (without the words "Dreamweaver Tutorial thesitewizard.com" of course). Your picture may b

**REGISTER PAGE CODING**

**<?php**

**require\_once "config.php";**

**$username = $email = $password = "";**

**$username\_err = $email\_err = $password\_err = "";**

**if ($\_SERVER['REQUEST\_METHOD'] == "POST"){**

**//check if username is empty**

**if(empty(trim($\_POST['username']))){**

**$username\_err = "Username cannot be blank";**

**}**

**else {**

**$sql = "SELECT id FROM users WHERE username = ?";**

**$stmt = mysqli\_prepare($conn, $sql);**

**if ($stmt) {**

**mysqli\_stmt\_bind\_param($stmt, "s", $param\_username);**

**//set the value of param username**

**$param\_username =trim($\_POST['username']);**

**// Try to execute this statement**

**if (mysqli\_stmt\_execute($stmt)) {**

**mysqli\_stmt\_store\_result($stmt);**

**if (mysqli\_stmt\_num\_rows($stmt) == 1) {**

**$username\_err = "This username is already taken";**

**}**

**else{**

**$username = trim($\_POST['username']);**

**}**

**}**

**else {**

**echo "something went wrong";**

**}**

**}**

**}**

**mysqli\_stmt\_close($stmt);**

**//check if email is empty**

**if(empty(trim($\_POST['email']))){**

**$email\_err = "email cannot be blank";**

**}**

**else {**

**$sql = "SELECT id FROM users WHERE email = ?";**

**$stmt = mysqli\_prepare($conn, $sql);**

**if ($stmt) {**

**mysqli\_stmt\_bind\_param($stmt, "s", $param\_email);**

**//set the value of param email**

**$param\_email =trim($\_POST['email']);**

**// Try to execute this statement**

**if (mysqli\_stmt\_execute($stmt)) {**

**mysqli\_stmt\_store\_result($stmt);**

**if (mysqli\_stmt\_num\_rows($stmt) == 1) {**

**$email\_err = "This email is already taken";**

**}**

**else{**

**$email = trim($\_POST['email']);**

**}**

**}**

**else {**

**echo "something went wrong";**

**}**

**}**

**}**

**mysqli\_stmt\_close($stmt);**

**// check for password**

**if (empty(trim($\_POST['password']))) {**

**$password\_err = "password cannot be blank";**

**}**

**elseif (strlen(trim($\_POST['password'])) < 5) {**

**$password\_err = "password cannot be less than 5 chatacters";**

**}**

**else {**

**$password = trim($\_POST['password']);**

**}**

**// if there were no errors, go ahead and insert into the database**

**if (empty($username\_err) && empty($email\_err) && empty($password\_err)) {**

**$sql = "INSERT INTO users (username, email, password) VALUES (?,?,?)";**

**$stmt = mysqli\_prepare($conn, $sql);**

**if ($stmt) {**

**mysqli\_stmt\_bind\_param($stmt, "sss", $param\_username, $param\_email,$param\_password);**

**//set the parameters**

**$param\_username = $username;**

**$param\_email = $email;**

**$param\_password = password\_hash($password, PASSWORD\_DEFAULT);**

**//Try to execute the querry**

**if (mysqli\_stmt\_execute($stmt)) {**

**header("location: login.php");**

**}**

**else {**

**echo "something went wrong... cannot redirect!";**

**}**

**}**

**mysqli\_stmt\_close($stmt);**

**}**

**mysqli\_close($conn);**

**}**

**?>**

**<!DOCTYPE html>**

**<html lang="en">**

**<head>**

**<meta charset="UTF-8">**

**<meta http-equiv="X-UA-Compatible" content="IE=edge">**

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

**<title>Login and Signup</title>**

**<link rel="stylesheet" href="istyle.css">**

**</head>**

**<body>**

**<header>**

**<h2 class="logo">Logo</h2>**

**<nav class="navigation">**

**<a href="#">Home</a>**

**<a href="about.html">About</a>**

**<a href="#">Services</a>**

**<a href="#">Contact</a>**

**<button class="btnlogin-popup" onclick="window.location.href = 'login.php'">Login</button>**

**</nav>**

**</header>**

**<div class="wrapper">**

**<!-- <span class="icon-close"><ion-icon name="close"></ion-icon></span> -->**

**<div class="form-box register">**

**<h2>Register</h2>**

**<form action="" method="post">**

**<div class="input-box">**

**<span class="icon"><ion-icon name="person"></ion-icon></span>**

**<input type="text" name="username" required>**

**<label >Username</label>**

**</div>**

**<div class="input-box">**

**<span class="icon"><ion-icon name="mail"></ion-icon></span>**

**<input type="Email" name="email" required>**

**<label >Email</label>**

**</div>**

**<div class="input-box">**

**<span class="icon"><ion-icon name="lock-closed"></ion-icon></span>**

**<input type="password" name="password" required>**

**<label >Password</label>**

**</div>**

**<div class="remember-forgot">**

**<label>**

**<input type="checkbox">I agree to the terms & conditions**

**</label>**

**</div>**

**<button type="submit" class="btn">Register</button>**

**<div class="login-register">**

**<p>Already have an account?<a href="login.php" class="login-link">Login</a></p>**

**</div>**

**</form>**

**</div>**

**</div>**

**<script type="module" src="https://unpkg.com/ionicons@7.1.0/dist/ionicons/ionicons.esm.js"></script>**

**<script nomodule src="https://unpkg.com/ionicons@7.1.0/dist/ionicons/ionicons.js"></script>**

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

**</body>**

**</html>**

**LOGIN PAGE CODING**

**<?php**

**//this script will handle login**

**//check if the user is already logged in**

**if (isset($\_SESSION['email'])) {**

**header("location: welcome.php");**

**exit;**

**}**

**require\_once "config.php";**

**$email = $password = "";**

**$err = "";**

**//if request method is post**

**if ($\_SERVER['REQUEST\_METHOD'] == "POST"){**

**if (empty(trim($\_POST['email'])) || empty(trim($\_POST['password']))) {**

**$err = "Please enter your email + password";**

**}**

**else {**

**$email = trim($\_POST['email']);**

**$password = trim($\_POST['password']);**

**}**

**if (empty($err)) {**

**$sql = "SELECT id, email, password FROM users WHERE email = ?";**

**$stmt = mysqli\_prepare($conn, $sql);**

**mysqli\_stmt\_bind\_param($stmt, "s", $param\_email);**

**$param\_email = $email;**

**}**

**// Try to execute this statement**

**if (mysqli\_stmt\_execute($stmt)) {**

**mysqli\_stmt\_store\_result($stmt);**

**if (mysqli\_stmt\_num\_rows($stmt) == 1) {**

**mysqli\_stmt\_bind\_result($stmt, $id, $email, $hashed\_password);**

**if (mysqli\_stmt\_fetch($stmt)) {**

**if (password\_verify($password, $hashed\_password)) {**

**//this means the password is correct. Allow to user to login**

**session\_start();**

**$\_SESSION["email"] = $email;**

**$\_SESSION["id"] = $id;**

**$\_SESSION["loggedin"] = true;**

**//redirect to user welcome page**

**header("location: welcome.php");**

**}**

**}**

**}**

**}**

**}**

**?>**

**<!DOCTYPE html>**

**<html lang="en">**

**<head>**

**<meta charset="UTF-8">**

**<meta http-equiv="X-UA-Compatible" content="IE=edge">**

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

**<title>Login and Signup</title>**

**<link rel="stylesheet" href="istyle.css">**

**</head>**

**<body>**

**<header>**

**<h2 class="logo">Logo</h2>**

**<nav class="navigation">**

**<a href="#">Home</a>**

**<a href="about.html">About</a>**

**<a href="#">Services</a>**

**<a href="#">Contact</a>**

**<button class="btnlogin-popup" onclick="window.location.href = 'register.php'">Register</button>**

**</nav>**

**</header>**

**<div class="wrapper">**

**<!-- <span class="icon-close"><ion-icon name="close"></ion-icon></span> -->**

**<div class="form-box login">**

**<h2>Login</h2>**

**<form action="" method="post">**

**<div class="input-box">**

**<span class="icon"><ion-icon name="mail"></ion-icon></span>**

**<input type="Email" name="email" required>**

**<label >Email</label>**

**</div>**

**<div class="input-box">**

**<span class="icon"><ion-icon name="lock-closed"></ion-icon></span>**

**<input type="password" name="password" required>**

**<label >Password</label>**

**</div>**

**<div class="remember-forgot">**

**<label>**

**<input type="checkbox">Remember me**

**</label>**

**<a href="#">Forgot Password?</a>**

**</div>**

**<button type="submit" class="btn">Login</button>**

**<div class="login-register">**

**<p>Don't have an account?<a href="register.php" class="register-link">Register</a></p>**

**</div>**

**</form>**

**</div>**

**<!-- <div class="form-box register">**

**<h2>Register</h2>**

**<form action="" method="post">**

**<div class="input-box">**

**<span class="icon"><ion-icon name="person"></ion-icon></span>**

**<input type="text" name="username" required>**

**<label >Username</label>**

**</div>**

**<div class="input-box">**

**<span class="icon"><ion-icon name="mail"></ion-icon></span>**

**<input type="Email" name="email" required>**

**<label >Email</label>**

**</div>**

**<div class="input-box">**

**<span class="icon"><ion-icon name="lock-closed"></ion-icon></span>**

**<input type="password" name="password" required>**

**<label >Password</label>**

**</div>**

**<div class="remember-forgot">**

**<label>**

**<input type="checkbox">I agree to the terms & conditions**

**</label>**

**</div>**

**<button type="submit" class="btn">Register</button>**

**<div class="login-register">**

**<p>Already have an account?<a href="#" class="login-link">Login</a></p>**

**</div>**

**</form>**

**</div> -->**

**</div>**

**<script type="module" src="https://unpkg.com/ionicons@7.1.0/dist/ionicons/ionicons.esm.js"></script>**

**<script nomodule src="https://unpkg.com/ionicons@7.1.0/dist/ionicons/ionicons.js"></script>**

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

**</body>**

**</html>**

**WELCOME PAGE CODING**

**<?php**

**session\_start();**

**if(!isset($\_SESSION['loggedin']) || $\_SESSION['loggedin'] !==true) {**

**header("location: login.php");**

**}**

**?>**

**<!DOCTYPE html>**

**<html lang="en">**

**<head>**

**<meta charset="UTF-8">**

**<meta http-equiv="X-UA-Compatible" content="IE=edge">**

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

**<title>Login and Signup</title>**

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

**</head>**

**<body>**

**<header>**

**<h2 class="logo">Logo</h2>**

**<nav class="navigation">**

**<a href="#">Home</a>**

**<a href="about.html">About</a>**

**<a href="#">Services</a>**

**<a href="#">Contact</a>**

**<button onclick="window.location.href = 'logout.php' " class="btnlogin-popup">Logout</button>**

**</nav>**

**</header>**

**<h1><?php echo "welcome - ". $\_SESSION['email']?></h1>**

**</div>**

**<script type="module" src="https://unpkg.com/ionicons@7.1.0/dist/ionicons/ionicons.esm.js"></script>**

**<script nomodule src="https://unpkg.com/ionicons@7.1.0/dist/ionicons/ionicons.js"></script>**

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

**</body>**

**</html>**

**LOGOUT BUTTON CODING**

**<?php**

**session\_start();**

**$\_SESSION = array();**

**session\_destroy();**

**header("location: login.php");**

**?>**

**CONFIG CODING**

**<?php**

**/\* This file contains database configuration assuming you are running my sql using user "root"and passwrord "" \*/**

**define('DB\_SERVER', 'localhost');**

**define('DB\_USERNAME', 'root');**

**define('DB\_PASSWORD', '');**

**define('DB\_NAME', 'login');**

**$conn = mysqli\_connect(DB\_SERVER, DB\_USERNAME,DB\_PASSWORD,DB\_NAME);**

**// check connection**

**if($conn == false){**

**dir('Error: Cannot connect'); }**

**?>**

**CSS CODING**

**\*{**

**margin: 0;**

**padding: 0;**

**box-sizing: border-box;**

**}**

**h2{**

**text-align: center;**

**margin-top: 30px;**

**}**

**header{**

**position: fixed;**

**top: 0;**

**left: 0;**

**width: 100%;**

**padding: 20px 100px;**

**display: flex;**

**justify-content: space-between;**

**align-items: center;**

**z-index: 99;**

**}**

**body{**

**display: flex;**

**justify-content: center;**

**align-items: center;**

**min-height: 100vh;**

**background: url(background.jpg);**

**background-size: cover;**

**background-position: center;**

**z-index: 99;**

**}**

**.logo{**

**font-size: 30px;**

**color: white;**

**user-select: none;**

**}**

**.navigation a{**

**position: relative;**

**color: #ffffff;**

**font-size: 20px;**

**text-decoration: none;**

**font-weight: 500;**

**margin-left: 40px;**

**}**

**.navigation a::after{**

**content: '';**

**position: absolute;**

**left: 0;**

**bottom: -6px;**

**width: 100%;**

**height: 3px;**

**background: white;**

**border-radius: 5px;**

**transform-origin: right;**

**transform: scaleX(0);**

**transition: transform .5s;**

**}**

**.navigation a:hover::after{**

**transform: scaleX(1);**

**transform-origin: left;**

**}**

**.navigation .btnlogin-popup{**

**width: 130px;**

**height: 50px;**

**background: transparent;**

**border: 2px solid white;**

**outline: none;**

**border-radius: 6px;**

**cursor: pointer;**

**font-size: 20px;**

**color: white;**

**font-weight: 500;**

**margin-left: 40PX;**

**transition: .5s;**

**}**

**.navigation .btnlogin-popup:hover{**

**background: #fff;**

**color: #161616;**

**}**

**.wrapper{**

**position: relative;**

**top: 30px;**

**width: 400px;**

**height: 440px;**

**background: transparent;**

**border: 2px solid rgba(255, 255, 255, .5);**

**border-radius: 20px;**

**backdrop-filter: blur(20px);**

**box-shadow: 0 0 30px rgba(0, 0, 0, .5);**

**display: flex;**

**justify-content: center;**

**align-items: center;**

**overflow: hidden;**

**/\* transform: scale(0); \*/**

**transition: transform .5s ease, height .2s ease;**

**}**

**/\* .wrapper .form-box.register{**

**position: absolute;**

**transition: none;**

**transform: translateX(400px);**

**} \*/**

**.wrapper .form-box{**

**width: 100%;**

**padding: 40px;**

**}**

**.wrapper .icon-close{**

**position: absolute;**

**top: 0;**

**right: 0;**

**width: 45px;**

**height: 45px;**

**background: #162938;**

**font-size: 2em;**

**color: #fff;**

**display: flex;**

**justify-content: center;**

**align-items: center;**

**border-bottom-left-radius: 20px;**

**cursor: pointer;**

**z-index: 1;**

**}**

**.form-box h2{**

**font-size: 2em;**

**color: #161616;**

**}**

**.input-box{**

**position: relative;**

**width: 100%;**

**height: 50px;**

**border-bottom: 2px solid black;**

**margin: 30px 0;**

**}**

**.input-box label{**

**position: absolute;**

**top: 50%;**

**left: 5px;**

**transform: translateY(-50%);**

**font-size: 1em;**

**color: #162938;**

**font-weight: 500;**

**pointer-events: none;**

**transition: .5s;**

**}**

**.input-box input:focus~label,**

**.input-box input:valid~label{**

**top: -5px;**

**}**

**.input-box input {**

**width: 100%;**

**height: 100%;**

**background: transparent;**

**border: none;**

**outline: none;**

**font-size: 1em;**

**color: #162938;**

**font-weight: 600;**

**padding: 0 35px 0 5px;**

**}**

**.input-box .icon{**

**position: absolute;**

**right: 8px;**

**font-size: 1.2em;**

**color: #161616;**

**line-height: 57px;**

**}**

**.remember-forgot{**

**font-size: .9em;**

**color: #161616;**

**font-weight: 500;**

**margin: -15px 0 15px;**

**display: flex;**

**justify-content: space-between;**

**}**

**.remember-forgot label input{**

**accent-color: #162938;**

**margin-right: 3px;**

**}**

**.remember-forgot a{**

**text-decoration: none;**

**color: #162938;**

**}**

**.remember-forgot a:hover{**

**text-decoration: underline;**

**}**

**.btn{**

**width: 100%;**

**height: 45px;**

**background: #162938;**

**outline: none;**

**border: none;**

**color: white;**

**border-radius: 6px;**

**cursor: pointer;**

**font-size: 1em;**

**font-weight: 500;**

**}**

**.login-register{**

**font-size: .9em;**

**color: #162938;**

**text-align: center;**

**font-weight: 500;**

**margin: 25px 0 10px;**

**}**

**.login-register p a{**

**color: #162938;**

**text-decoration: none;**

**font-weight: 600;**

**}**

**.login-register p a:hover{**

**text-decoration: underline;**

**}**

**JAVASCRIPT CODING**

**const wrapper = document.querySelector('.wrapper');**

**const loginLink = document.querySelector('.login-link');**

**const registerLink = document.querySelector('.register-link');**

**const btnPopup = document.querySelector('.btnlogin-popup');**

**const iconClose = document.querySelector('.icon-close');**

**registerLink.addEventListener('click', ()=>{**

**wrapper.classList.add('active');**

**});**

**loginLink.addEventListener('click', ()=>{**

**wrapper.classList.remove('active');**

**});**

**btnPopup.addEventListener('click', ()=>{**

**wrapper.classList.add('active-popup');**

**});**

**iconClose.addEventListener('click', ()=>{**

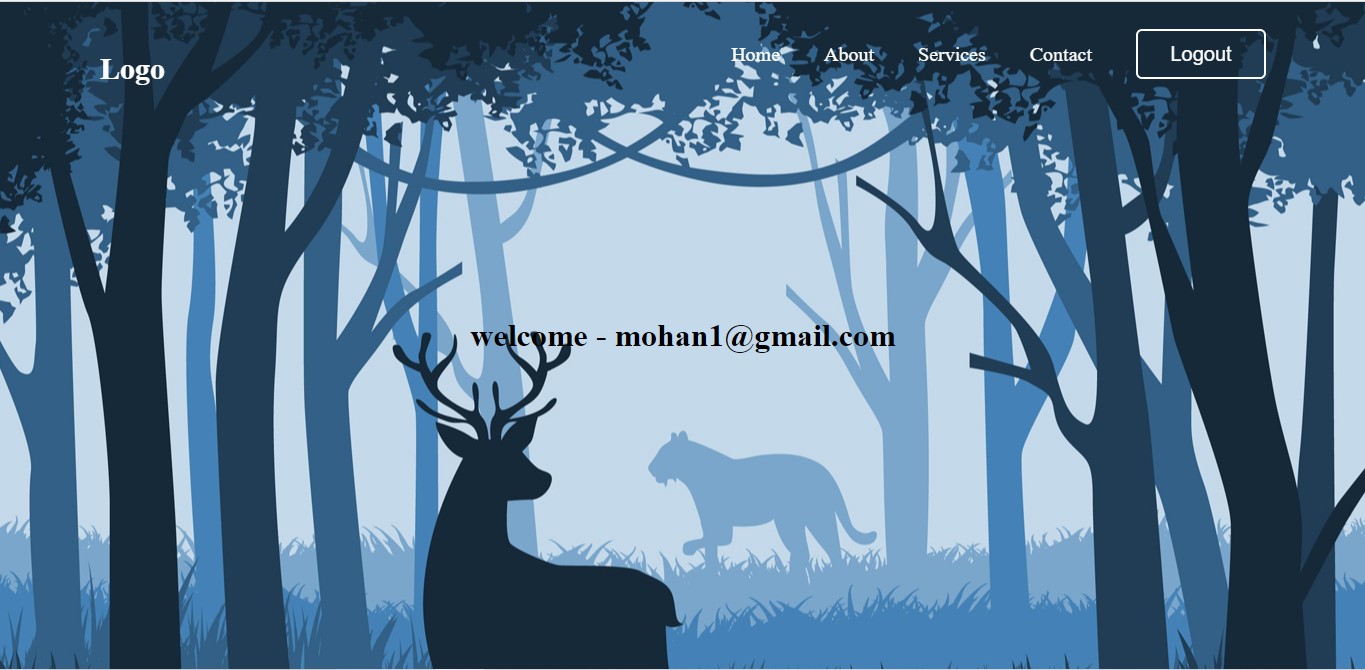
**wrapper.classList.remove('active-popup');**

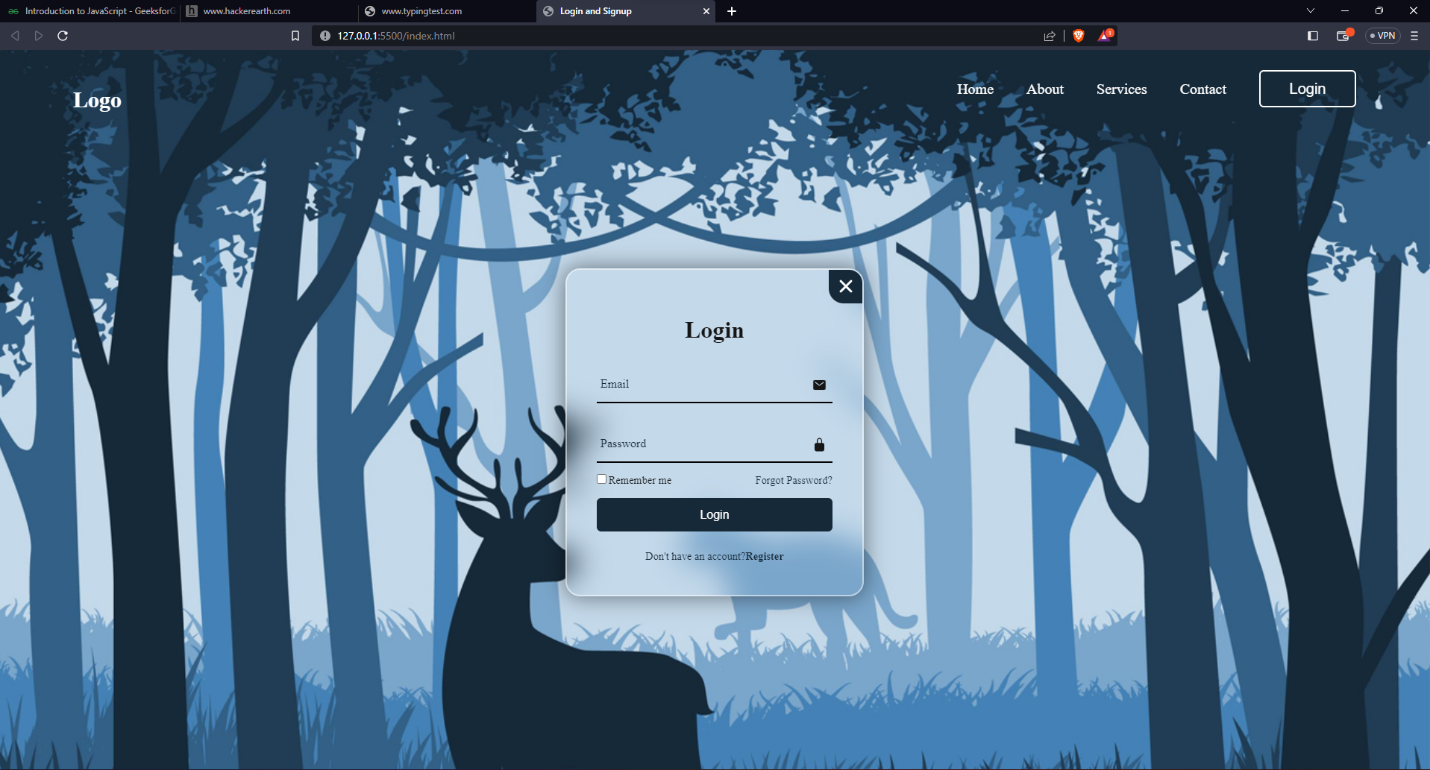
**});**

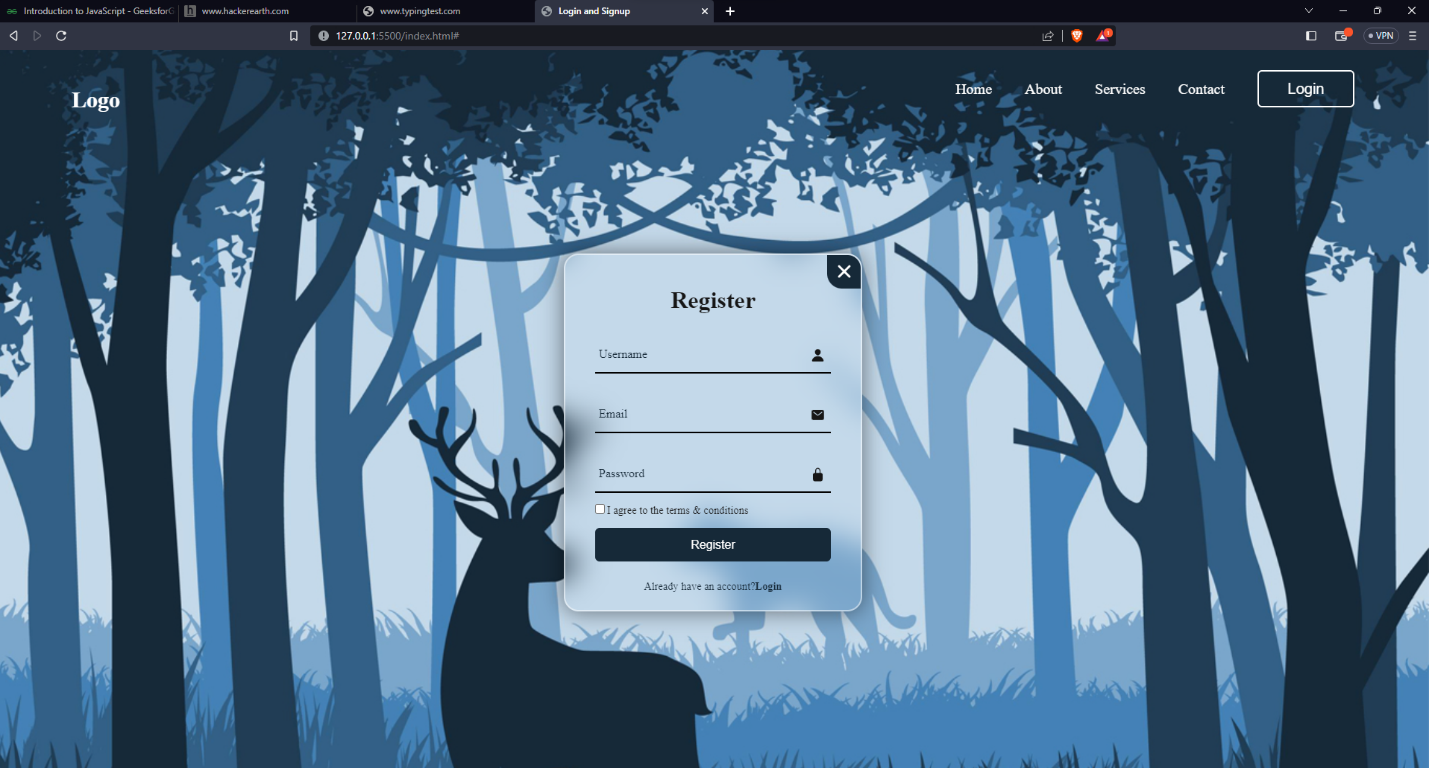
CHAPTER-6

**SCREENSHOTS**

Bottom of Form

****

****

****

# CHAPTER-7

# REFRENCE

* + [www.w3school.com](http://www.w3school.com) site
  + [www.tutorials](http://www.tutorials)point.com
  + [www.w3.org](http://www.w3.org)
  + www.codecademy.com
  + Html, css, java script, php toturial.
  + Refrenced Books:

1).HTML & CSS: Design and Build Web Sites BY:-Jon Duckett

2).HTML FILE BY:-Matthew Macdonald