**1. TY Time Table using HTML and CSS/Bootstrap**

HTML Code (ty-timetable.html)

html

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>TY Time Table</title>

<link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.5.2/css/bootstrap.min.css">

</head>

<body>

<div class="container mt-5">

<h2 class="text-center">TY Time Table</h2>

<table class="table table-bordered">

<thead>

<tr>

<th>Subject Name</th>

<th>Faculty Initials</th>

<th>Time Slot</th>

<th>Day of the Week</th>

</tr>

</thead>

<tbody>

<tr>

<td>Mathematics</td>

<td>AB</td>

<td>09:00 - 10:00</td>

<td>Monday</td>

</tr>

<tr>

<td>Physics</td>

<td>CD</td>

<td>10:15 - 11:15</td>

<td>Monday</td>

</tr>

<tr>

<td>Chemistry</td>

<td>EF</td>

<td>11:30 - 12:30</td>

<td>Tuesday</td>

</tr>

<tr>

<td>Biology</td>

<td>GH</td>

<td>09:00 - 10:00</td>

<td>Wednesday</td>

</tr>

<tr>

<td>Computer Science</td>

<td>IJ</td>

<td>10:15 - 11:15</td>

<td>Thursday</td>

</tr>

</tbody>

</table>

</div>

</body>

</html>

**2. Railway Time Table using HTML and CSS**

HTML Code (railway-timetable.html)

html

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>Railway Time Table</title>

<link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.5.2/css/bootstrap.min.css">

</head>

<body>

<div class="container mt-5">

<h2 class="text-center">Railway Time Table</h2>

<table class="table table-bordered">

<thead>

<tr>

<th>Train Number</th>

<th>Train Name</th>

<th>Destination</th>

<th>Departure Time</th>

<th>Arrival Time</th>

</tr>

</thead>

<tbody>

<tr>

<td>12345</td>

<td>Express Train</td>

<td>City A</td>

<td>09:00 AM</td>

<td>01:00 PM</td>

</tr>

<tr>

<td>67890</td>

<td>Local Train</td>

<td>City B</td>

<td>10:30 AM</td>

<td>12:30 PM</td>

</tr>

<tr>

<td>54321</td>

<td>Fast Train</td>

<td>City C</td>

<td>11:00 AM</td>

<td>03:00 PM</td>

</tr>

</tbody>

</table>

</div>

</body>

</html>

**3. Facebook Registration Form with JavaScript Validation**

HTML Code (facebook-registration.html)

html

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>Facebook Registration Form</title>

<link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.5.2/css/bootstrap.min.css">

<script>

function validateForm() {

const email = document.getElementById('email').value;

const password = document.getElementById('password').value;

const name = document.getElementById('name').value;

const phone = document.getElementById('phone').value;

if (name === "" || email === "" || password === "" || phone === "") {

alert("All fields must be filled out");

return false;

}

const emailPattern = /^[^ ]+@[^ ]+\.[a-z]{2,3}$/;

if (!email.match(emailPattern)) {

alert("Please enter a valid email address.");

return false;

}

alert("Registration Successful!");

return true;

}

</script>

</head>

<body>

<div class="container mt-5">

<h2 class="text-center">Facebook Registration</h2>

<form onsubmit="return validateForm()">

<div class="form-group">

<label for="name">Name:</label>

<input type="text" class="form-control" id="name" placeholder="Enter your name" required>

</div>

<div class="form-group">

<label for="email">Email:</label>

<input type="email" class="form-control" id="email" placeholder="Enter your email" required>

</div>

<div class="form-group">

<label for="phone">Phone:</label>

<input type="text" class="form-control" id="phone" placeholder="Enter your phone number" required>

</div>

<div class="form-group">

<label for="password">Password:</label>

<input type="password" class="form-control" id="password" placeholder="Enter your password" required>

</div>

<button type="submit" class="btn btn-primary">Register</button>

</form>

</div>

</body>

</html>

**4. Gmail Registration Form with JavaScript Validation**

HTML Code (gmail-registration.html)

html

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>Gmail Registration Form</title>

<link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.5.2/css/bootstrap.min.css">

<script>

function validateForm() {

const email = document.getElementById('email').value;

const username = document.getElementById('username').value;

const password = document.getElementById('password').value;

const confirmPassword = document.getElementById('confirmPassword').value;

if (username === "" || email === "" || password === "" || confirmPassword === "") {

alert("All fields must be filled out");

return false;

}

const emailPattern = /^[^ ]+@gmail\.com$/;

if (!email.match(emailPattern)) {

alert("Please enter a valid Gmail address.");

return false;

}

if (password !== confirmPassword) {

alert("Passwords do not match.");

return false;

}

alert("Registration Successful!");

return true;

}

</script>

</head>

<body>

<div class="container mt-5">

<h2 class="text-center">Gmail Registration</h2>

<form onsubmit="return validateForm()">

<div class="form-group">

<label for="username">Username:</label>

<input type="text" class="form-control" id="username" placeholder="Enter your username" required>

</div>

<div class="form-group">

<label for="email">Email:</label>

<input type="email" class="form-control" id="email" placeholder="Enter your Gmail" required>

</div>

<div class="form-group">

<label for="password">Password:</label>

<input type="password" class="form-control" id="password" placeholder="Enter your password" required>

</div>

<div class="form-group">

<label for="confirmPassword">Confirm Password:</label>

<input type="password" class="form-control" id="confirmPassword" placeholder="Confirm your password" required>

</div>

<button type="submit" class="btn btn-primary">Register</button>

</form>

</div>

</body>

</html>

**5. 3-Page Website for Online Shopping of Garments**

Structure

index.html: Home page with a product list.

product.html: Product detail page.

cart.html: User's shopping cart.

index.html

html

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>Online Garment Store</title>

<link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.5.2/css/bootstrap.min.css">

</head>

<body>

<div class="container mt-5">

<h2 class="text-center">Online Garment Store</h2>

<div class="row">

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

<div class="card mb-4">

<img class="card-img-top" src="https://via.placeholder.com/150" alt="T-Shirt">

<div class="card-body">

<h5 class="card-title">T-Shirt</h5>

<p class="card-text">$20.00</p>

<a href="product.html" class="btn btn-primary">View Details</a>

</div>

</div>

</div>

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

<div class="card mb-4">

<img class="card-img-top" src="https://via.placeholder.com/150" alt="Jeans">

<div class="card-body">

<h5 class="card-title">Jeans</h5>

<p class="card-text">$30.00</p>

<a href="product.html" class="btn btn-primary">View Details</a>

</div>

</div>

</div>

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

<div class="card mb-4">

<img class="card-img-top" src="https://via.placeholder.com/150" alt="Jacket">

<div class="card-body">

<h5 class="card-title">Jacket</h5>

<p class="card-text">$50.00</p>

<a href="product.html" class="btn btn-primary">View Details</a>

</div>

</div>

</div>

</div>

<a href="cart.html" class="btn btn-success">View Cart</a>

</div>

</body>

</html>

product.html

html

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>Product Details - Garment Store</title>

<link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.5.2/css/bootstrap.min.css">

</head>

<body>

<div class="container mt-5">

<h2 class="text-center">Product Details</h2>

<div class="card mb-4">

<img class="card-img-top" src="https://via.placeholder.com/300" alt="Product Image">

<div class="card-body">

<h5 class="card-title">T-Shirt</h5>

<p class="card-text">This is a comfortable cotton T-shirt available in various sizes.</p>

<p class="card-text"><strong>Price: $20.00</strong></p>

<button class="btn btn-primary">Add to Cart</button>

</div>

</div>

<a href="index.html" class="btn btn-secondary">Back to Home</a>

</div>

</body>

</html>

cart.html

html

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>Shopping Cart - Garment Store</title>

<link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.5.2/css/bootstrap.min.css">

</head>

<body>

<div class="container mt-5">

<h2 class="text-center">Your Shopping Cart</h2>

<table class="table table-bordered">

<thead>

<tr>

<th>Product</th>

<th>Price</th>

<th>Quantity</th>

<th>Total</th>

</tr>

</thead>

<tbody>

<tr>

<td>T-Shirt</td>

<td>$20.00</td>

<td>1</td>

<td>$20.00</td>

</tr>

</tbody>

</table>

<h4 class="text-right">Total Amount: $20.00</h4>

<a href="index.html" class="btn btn-secondary">Continue Shopping</a>

</div>

</body>

</html>

**6. Simple Calculator Using JavaScript**

HTML Code (calculator.html)

html

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>Calculator</title>

<link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.5.2/css/bootstrap.min.css">

<script>

function calculate(operation) {

const num1 = parseFloat(document.getElementById('num1').value);

const num2 = parseFloat(document.getElementById('num2').value);

let result;

switch (operation) {

case 'add':

result = num1 + num2;

break;

case 'subtract':

result = num1 - num2;

break;

case 'multiply':

result = num1 \* num2;

break;

case 'divide':

result = num2 !== 0 ? num1 / num2 : 'Error: Division by zero';

break;

default:

result = 'Unknown operation';

}

document.getElementById('result').innerText = `Result: ${result}`;

}

</script>

</head>

<body>

<div class="container mt-5">

<h2 class="text-center">Simple Calculator</h2>

<div class="form-group">

<input type="number" id="num1" class="form-control" placeholder="Enter first number" required>

</div>

<div class="form-group">

<input type="number" id="num2" class="form-control" placeholder="Enter second number" required>

</div>

<div class="text-center">

<button class="btn btn-primary" onclick="calculate('add')">Add</button>

<button class="btn btn-primary" onclick="calculate('subtract')">Subtract</button>

<button class="btn btn-primary" onclick="calculate('multiply')">Multiply</button>

<button class="btn btn-primary" onclick="calculate('divide')">Divide</button>

</div>

<h3 id="result" class="text-center mt-3"></h3>

</div>

</body>

</html>

**7. HTML File with Linked JavaScript**

HTML Code (index.html)

html

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>JavaScript Example</title>

<link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.5.2/css/bootstrap.min.css">

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

</head>

<body>

<div class="container mt-5">

<h2 class="text-center">JavaScript Alerts and Calculations</h2>

<p>Check your console for average lifetime calculations and time of day messages.</p>

</div>

</body>

</html>

JavaScript Code (script.js)

javascript

// Alert to indicate the script is linked

alert("JavaScript file is connected!");

// Calculate the average number of weeks in a human lifetime (assuming an average of 80 years)

const averageLifetimeYears = 80;

const averageWeeks = averageLifetimeYears \* 52.1775; // 1 year = 52.1775 weeks

console.log(`Average number of weeks in a human lifetime: ${averageWeeks.toFixed(0)}`);

// Function to tell the time of the day

function getTimeOfDay() {

const hours = new Date().getHours();

let timeOfDay;

if (hours < 12) {

timeOfDay = "morning";

} else if (hours < 18) {

timeOfDay = "afternoon";

} else {

timeOfDay = "night";

}

console.log(`Time of day: ${timeOfDay}`);

}

getTimeOfDay();

**8. PHP and MySQL for Complaint Management System**

Database Table Structure

Create database and table:

sql

CREATE DATABASE complaint\_management;

USE complaint\_management;

CREATE TABLE complaints (

id INT AUTO\_INCREMENT PRIMARY KEY,

user\_name VARCHAR(100),

complaint TEXT,

status ENUM('Pending', 'Resolved') DEFAULT 'Pending',

created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP

);

PHP Code (index.php)

php

<?php

$servername = "localhost";

$username = "root"; // Adjust with your username

$password = ""; // Adjust with your password

$dbname = "complaint\_management";

// Create connection

$conn = new mysqli($servername, $username, $password, $dbname);

// Check connection

if ($conn->connect\_error) {

die("Connection failed: " . $conn->connect\_error);

}

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

$user\_name = $\_POST['user\_name'];

$complaint = $\_POST['complaint'];

$sql = "INSERT INTO complaints (user\_name, complaint) VALUES ('$user\_name', '$complaint')";

if ($conn->query($sql) === TRUE) {

echo "New complaint registered successfully.";

} else {

echo "Error: " . $sql . "<br>" . $conn->error;

}

}

$conn->close();

?>

<!DOCTYPE html>

<html>

<head>

<title>Complaint Management</title>

</head>

<body>

<h2>Register a Complaint</h2>

<form method="POST" action="">

<input type="text" name="user\_name" placeholder="Your Name" required><br><br>

<textarea name="complaint" placeholder="Describe your complaint" required></textarea><br><br>

<input type="submit" value="Submit Complaint">

</form>

</body>

</html>

**9. PHP and MySQL for Toll Tax Management System**

Database Table Structure

Create database and table:

sql

CREATE DATABASE toll\_tax\_management;

USE toll\_tax\_management;

CREATE TABLE toll\_records (

id INT AUTO\_INCREMENT PRIMARY KEY,

vehicle\_number VARCHAR(50),

vehicle\_type ENUM('Car', 'Truck', 'Bus', 'Bike'),

toll\_amount DECIMAL(10, 2),

timestamp TIMESTAMP DEFAULT CURRENT\_TIMESTAMP

);

PHP Code (toll.php)

php

<?php

$servername = "localhost";

$username = "root"; // Adjust with your username

$password = ""; // Adjust with your password

$dbname = "toll\_tax\_management";

// Create connection

$conn = new mysqli($servername, $username, $password, $dbname);

// Check connection

if ($conn->connect\_error) {

die("Connection failed: " . $conn->connect\_error);

}

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

$vehicle\_number = $\_POST['vehicle\_number'];

$vehicle\_type = $\_POST['vehicle\_type'];

// Set toll amount based on vehicle type

$toll\_amount = ($vehicle\_type == 'Car') ? 2.00 : (($vehicle\_type == 'Truck') ? 5.00 : (($vehicle\_type == 'Bus') ? 4.00 : 1.00));

$sql = "INSERT INTO toll\_records (vehicle\_number, vehicle\_type, toll\_amount) VALUES ('$vehicle\_number', '$vehicle\_type', '$toll\_amount')";

if ($conn->query($sql) === TRUE) {

echo "Toll record added successfully. Amount payable: $$toll\_amount";

} else {

echo "Error: " . $sql . "<br>" . $conn->error;

}

}

$conn->close();

?>

<!DOCTYPE html>

<html>

<head>

<title>Toll Tax Management</title>

</head>

<body>

<h2>Record Toll Payment</h2>

<form method="POST" action="">

<input type="text" name="vehicle\_number" placeholder="Vehicle Number" required><br><br>

<select name="vehicle\_type" required>

<option value="">Select Vehicle Type</option>

<option value="Car">Car</option>

<option value="Truck">Truck</option>

<option value="Bus">Bus</option>

<option value="Bike">Bike</option>

</select><br><br>

<input type="submit" value="Record Toll">

</form>

</body>

</html>

**10. PHP and MySQL for Pharmacy Management System**

Database Table Structure

Create database and table:

sql

CREATE DATABASE pharmacy\_management;

USE pharmacy\_management;

CREATE TABLE medicines (

id INT AUTO\_INCREMENT PRIMARY KEY,

medicine\_name VARCHAR(100),

quantity INT,

price DECIMAL(10, 2)

);

PHP Code (pharmacy.php)

php

<?php

$servername = "localhost";

$username = "root"; // Adjust with your username

$password = ""; // Adjust with your password

$dbname = "pharmacy\_management";

// Create connection

$conn = new mysqli($servername, $username, $password, $dbname);

// Check connection

if ($conn->connect\_error) {

die("Connection failed: " . $conn->connect\_error);

}

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

$medicine\_name = $\_POST['medicine\_name'];

$quantity = $\_POST['quantity'];

$price = $\_POST['price'];

$sql = "INSERT INTO medicines (medicine\_name, quantity, price) VALUES ('$medicine\_name', '$quantity', '$price')";

if ($conn->query($sql) === TRUE) {

echo "New medicine added successfully.";

} else {

echo "Error: " . $sql . "<br>" . $conn->error;

}

}

$conn->close();

?>

<!DOCTYPE html>

<html>

<head>

<title>Pharmacy Management</title>

</head>

<body>

<h2>Add New Medicine</h2>

<form method="POST" action="">

<input type="text" name="medicine\_name" placeholder="Medicine Name" required><br><br>

<input type="number" name="quantity" placeholder="Quantity" required><br><br>

<input type="number" step="0.01" name="price" placeholder="Price" required><br><br>

<input type="submit" value="Add Medicine">

</form>

</body>

</html>

**11. PHP and MySQL for College Admission Management System**

Database Table Structure

sql

CREATE DATABASE college\_admission;

USE college\_admission;

CREATE TABLE admissions (

id INT AUTO\_INCREMENT PRIMARY KEY,

student\_name VARCHAR(100),

email VARCHAR(100),

course VARCHAR(100),

admission\_date DATE

);

PHP Code (admission.php)

php

<?php

$servername = "localhost";

$username = "root"; // Adjust with your username

$password = ""; // Adjust with your password

$dbname = "college\_admission";

// Create connection

$conn = new mysqli($servername, $username, $password, $dbname);

// Check connection

if ($conn->connect\_error) {

die("Connection failed: " . $conn->connect\_error);

}

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

$student\_name = $\_POST['student\_name'];

$email = $\_POST['email'];

$course = $\_POST['course'];

$admission\_date = date('Y-m-d');

$sql = "INSERT INTO admissions (student\_name, email, course, admission\_date) VALUES ('$student\_name', '$email', '$course', '$admission\_date')";

if ($conn->query($sql) === TRUE) {

echo "Admission record added successfully.";

} else {

echo "Error: " . $sql . "<br>" . $conn->error;

}

}

$conn->close();

?>

<!DOCTYPE html>

<html>

<head>

<title>College Admission Management</title>

</head>

<body>

<h2>College Admission Form</h2>

<form method="POST" action="">

<input type="text" name="student\_name" placeholder="Student Name" required><br><br>

<input type="email" name="email" placeholder="Email" required><br><br>

<input type="text" name="course" placeholder="Course" required><br><br>

<input type="submit" value="Submit Admission">

</form>

</body>

</html>

**12. Simple HTML Forms with Input Elements and a Button**

HTML Code (simple-forms.html)

html

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>Simple HTML Forms</title>

</head>

<body>

<h2>Contact Form</h2>

<form>

<input type="text" name="name" placeholder="Name" required><br><br>

<input type="email" name="email" placeholder="Email" required><br><br>

<input type="tel" name="phone" placeholder="Phone" required><br><br>

<textarea name="message" placeholder="Your Message" required></textarea><br><br>

<button type="submit">Send Message</button>

</form>

</body>

</html>

**13. JavaScript Functions as Event Handlers**

HTML Code with JavaScript (event-handlers.html)

html

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>Event Handlers in JavaScript</title>

<script>

function showAlert() {

alert("Button Clicked!");

}

function changeColor() {

document.getElementById("myText").style.color = "blue";

}

</script>

</head>

<body>

<h2>JavaScript Event Handlers Example</h2>

<p id="myText">This is some text that will change color.</p>

<button onclick="showAlert()">Click Me</button>

<button onclick="changeColor()">Change Text Color</button>

</body>

</html>

**14. Create Fibonacci Series in JavaScript and Display on HTML**

HTML Code (fibonacci.html)

html

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>Fibonacci Series</title>

<script>

function generateFibonacci() {

const n = 10; // number of Fibonacci numbers to generate

let fibSeries = [0, 1];

for (let i = 2; i < n; i++) {

fibSeries[i] = fibSeries[i - 1] + fibSeries[i - 2];

}

document.getElementById("result").innerText = "Fibonacci Series: " + fibSeries.join(", ");

}

</script>

</head>

<body>

<h2>Fibonacci Series Generator</h2>

<button onclick="generateFibonacci()">Generate Fibonacci Series</button>

<p id="result"></p>

</body>

</html>

**15. Sign-Up Page with Validation and Name Display After Login**

HTML Code (signup.html)

html

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>Sign-Up Page</title>

<script>

function validateForm() {

const username = document.getElementById("username").value;

const password = document.getElementById("password").value;

if (username === "" || password === "") {

alert("Both fields are required!");

return false;

}

sessionStorage.setItem("username", username);

document.getElementById("loginDisplay").innerText = `Welcome, ${username}!`;

return false; // Prevent form submission

}

</script>

</head>

<body>

<h2>Sign Up</h2>

<form onsubmit="return validateForm()">

<input type="text" id="username" placeholder="Username" required><br><br>

<input type="password" id="password" placeholder="Password" required><br><br>

<button type="submit">Sign Up</button>

</form>

<h3 id="loginDisplay"></h3>

</body>

</html>

**16. Registration Form with Validation**

HTML Code (registration.html)

html

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>Registration Form</title>

<link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.5.2/css/bootstrap.min.css">

<script>

function validateForm() {

const firstName = document.getElementById("firstName").value;

const lastName = document.getElementById("lastName").value;

const organization = document.getElementById("organization").value;

const hobbies = document.getElementById("hobbies").value;

if (!firstName || !lastName || !organization || !hobbies) {

alert("All fields are required");

return false;

}

// Display the details

document.getElementById("displayDetails").innerText =

`Name: ${firstName} ${lastName}\nOrganization: ${organization}\nHobbies: ${hobbies}`;

return false; // Prevent actual form submission for demo purposes

}

</script>

</head>

<body>

<div class="container mt-5">

<h2 class="text-center">Registration Form</h2>

<form onsubmit="return validateForm()">

<div class="form-group">

<input type="text" id="firstName" class="form-control" placeholder="First Name" required>

</div>

<div class="form-group">

<input type="text" id="lastName" class="form-control" placeholder="Last Name" required>

</div>

<div class="form-group">

<input type="text" id="organization" class="form-control" placeholder="Organization" required>

</div>

<div class="form-group">

<input type="text" id="hobbies" class="form-control" placeholder="Hobbies (comma separated)" required>

</div>

<button type="submit" class="btn btn-primary">Register</button>

</form>

<h3 class="mt-4">Registered Details:</h3>

<pre id="displayDetails"></pre>

</div>

</body>

</html>

**17. Image Gallery Using HTML and Bootstrap**

HTML Code (gallery.html)

html

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>Image Gallery</title>

<link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.5.2/css/bootstrap.min.css">

</head>

<body>

<div class="container mt-5">

<h2 class="text-center">Image Gallery</h2>

<div class="row">

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

<div class="card">

<img src="https://via.placeholder.com/300" class="card-img-top" alt="Image 1">

<div class="card-body">

<p class="card-text">Description for Image 1</p>

</div>

</div>

</div>

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

<div class="card">

<img src="https://via.placeholder.com/300" class="card-img-top" alt="Image 2">

<div class="card-body">

<p class="card-text">Description for Image 2</p>

</div>

</div>

</div>

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

<div class="card">

<img src="https://via.placeholder.com/300" class="card-img-top" alt="Image 3">

<div class="card-body">

<p class="card-text">Description for Image 3</p>

</div>

</div>

</div>

</div>

</div>

</body>

</html>

**18. Page with a Video**

HTML Code (video.html)

html

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>Video Page</title>

</head>

<body>

<div style="text-align: center; margin-top: 50px;">

<h2>Video Presentation</h2>

<video width="640" height="360" controls>

<source src="https://www.w3schools.com/html/mov\_bbb.mp4" type="video/mp4">

Your browser does not support the video tag.

</video>

</div>

</body>

</html>

**19. Notification Popup**

HTML Code (notification-popup.html)

html

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>Notification Popup</title>

<style>

#notification {

display: none;

position: fixed;

top: 20px;

right: 20px;

background-color: #4CAF50;

color: white;

padding: 15px;

border-radius: 4px;

}

</style>

<script>

function showNotification() {

const notification = document.getElementById("notification");

notification.style.display = "block";

setTimeout(() => {

notification.style.display = "none";

}, 3000);

}

</script>

</head>

<body>

<div style="text-align: center; margin-top: 50px;">

<button onclick="showNotification()">Show Notification</button>

<div id="notification">This is a notification message!</div>

</div>

</body>

</html>

**20. Create and Store Cookies**

HTML Code (cookies.html)

html

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>Cookie Management</title>

<script>

function setCookie() {

const username = document.getElementById("username").value;

document.cookie = `username=${username};path=/`; // Set cookie

alert("Cookie has been set!");

}

function getCookie() {

const cookies = document.cookie.split(';');

let username = '';

for (let cookie of cookies) {

const [key, value] = cookie.split('=');

if (key.trim() === 'username') {

username = value;

}

}

alert(`Stored Username: ${username}`);

}

</script>

</head>

<body>

<div style="text-align: center; margin-top: 50px;">

<h2>Manage Cookies</h2>

<input type="text" id="username" placeholder="Enter username">

<button onclick="setCookie()">Set Cookie</button>

<button onclick="getCookie()">Get Cookie</button>

</div>

</body>

</html>

**21. Design User Registration and Login Form Using Vue.js**

Vue.js User Registration/Login Form

html

<template>

<div>

<h2>User Registration</h2>

<form @submit.prevent="registerUser">

<input v-model="username" type="text" placeholder="Username" required />

<input v-model="password" type="password" placeholder="Password" required />

<button type="submit">Register</button>

</form>

<h2>User Login</h2>

<form @submit.prevent="loginUser">

<input v-model="loginUsername" type="text" placeholder="Username" required />

<input v-model="loginPassword" type="password" placeholder="Password" required />

<button type="submit">Login</button>

</form>

</div>

</template>

<script>

export default {

data() {

return {

username: '',

password: '',

loginUsername: '',

loginPassword: ''

};

},

methods: {

registerUser() {

// Registration logic here

alert(`Registered: ${this.username}`);

},

loginUser() {

// Login logic here

alert(`Logged in: ${this.loginUsername}`);

}

}

};

</script>

**22. Design Sign Up Registration and Login Form for Facebook Using Vue.js**

Facebook Login/Registration with Vue.js

You can use the Facebook SDK for authentication.

html

<template>

<div>

<button @click="loginWithFacebook">Login with Facebook</button>

</div>

</template>

<script>

export default {

methods: {

loginWithFacebook() {

window.FB.login((response) => {

if (response.status === 'connected') {

// Handle successful login

alert('Logged in with Facebook');

} else {

alert('Failed to log in');

}

});

},

loadFacebookSDK() {

window.fbAsyncInit = function () {

window.FB.init({

appId: 'YOUR\_APP\_ID',

cookie: true,

xfbml: true,

version: 'v11.0'

});

};

(function (d, s, id) {

var js, fjs = d.getElementsByTagName(s)[0];

if (d.getElementById(id)) return;

js = d.createElement(s); js.id = id;

js.src = 'https://connect.facebook.net/en\_US/sdk.js';

fjs.parentNode.insertBefore(js, fjs);

}(document, 'script', 'facebook-jssdk'));

}

},

mounted() {

this.loadFacebookSDK();

}

};

</script>

**23. Write a Vue.js Program to Create Reusable Grid Components and Use It with External Data**

Reusable Grid Component in Vue.js

html

<template>

<table>

<thead>

<tr>

<th v-for="header in headers" :key="header">{{ header }}</th>

</tr>

</thead>

<tbody>

<tr v-for="item in items" :key="item.id">

<td v-for="header in headers" :key="header">{{ item[header] }}</td>

</tr>

</tbody>

</table>

</template>

<script>

export default {

props: {

items: Array,

headers: Array

}

};

</script>

Using the Grid Component

html

<template>

<div>

<ReusableGrid :items="dataItems" :headers="['id', 'name', 'price']" />

</div>

</template>

<script>

import ReusableGrid from './ReusableGrid.vue';

export default {

components: {

ReusableGrid

},

data() {

return {

dataItems: [

{ id: 1, name: 'Apple', price: '$1' },

{ id: 2, name: 'Banana', price: '$0.5' }

]

};

}

};

</script>

**24. Develop an Ecommerce Platform Ubercart Using PHP**

Basic Structure for Ubercart Installation

You would typically follow these steps for a basic Ubercart installation:

Install Drupal: Download and install the latest version of Drupal.

Install Ubercart Module:

Download the module and enable it in the Drupal admin panel.

Configure Payment and Shipping: Set up payment gateways and shipping options in the Ubercart configuration settings.

Create Products: Use the Products module in Ubercart to create and manage products.

**25. Build a Grocery Store Application Using PHP and MySQL**

Basic PHP Grocery Store Application Structure

Database Structure:

products table (id, name, price, description)

users table (id, username, password)

orders table (id, user\_id, product\_id, quantity)

Sample Code to List Products:

php

<?php

$servername = "localhost";

$username = "username";

$password = "password";

$dbname = "grocery\_store";

$conn = new mysqli($servername, $username, $password, $dbname);

if ($conn->connect\_error) {

die("Connection failed: " . $conn->connect\_error);

}

$sql = "SELECT id, name, price FROM products";

$result = $conn->query($sql);

if ($result->num\_rows > 0) {

while($row = $result->fetch\_assoc()) {

echo "id: " . $row["id"]. " - Name: " . $row["name"]. " - Price: " . $row["price"]. "<br>";

}

} else {

echo "0 results";

}

$conn->close();

?>

**26. Build a Facebook-like Website Using PHP and MySQL**

To build a basic Facebook-like website, you would typically focus on user registration, posting, and a friends system. Below is a simplified version of the essential components.

Basic Structure

Database Structure:

users (id, username, password, email)

posts (id, user\_id, content, created\_at)

friends (id, user\_id, friend\_id)

Sample Code for User Registration:

php

<?php

// registration.php

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

$username = $\_POST['username'];

$password = password\_hash($\_POST['password'], PASSWORD\_DEFAULT);

$email = $\_POST['email'];

// Database connection

$conn = new mysqli('localhost', 'username', 'password', 'social\_network');

if ($conn->connect\_error) {

die("Connection failed: " . $conn->connect\_error);

}

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

if ($conn->query($sql) === TRUE) {

echo "Registration successful!";

} else {

echo "Error: " . $sql . "<br>" . $conn->error;

}

$conn->close();

}

?>

<form method="post">

Username: <input type="text" name="username" required><br>

Password: <input type="password" name="password" required><br>

Email: <input type="email" name="email" required><br>

<input type="submit" value="Register">

</form>

Sample Code for Posting:

php

<?php

// post.php

session\_start();

if ($\_SERVER["REQUEST\_METHOD"] == "POST" && isset($\_SESSION['user\_id'])) {

$content = $\_POST['content'];

$user\_id = $\_SESSION['user\_id'];

// Database connection

$conn = new mysqli('localhost', 'username', 'password', 'social\_network');

$sql = "INSERT INTO posts (user\_id, content, created\_at) VALUES ('$user\_id', '$content', NOW())";

if ($conn->query($sql) === TRUE) {

echo "Post successful!";

} else {

echo "Error: " . $sql . "<br>" . $conn->error;

}

$conn->close();

}

?>

<form method="post">

<textarea name="content" required></textarea><br>

<input type="submit" value="Post">

</form>

**27. Create a Database with User and Book Information and Dynamically Load Books**

Database Structure:

users (id, name, email)

books (id, title, author, user\_id)

Sample Code to Load Books Dynamically:

Database Connection and Loading Books:

php

<?php

// db.php

$servername = "localhost";

$username = "username";

$password = "password";

$dbname = "library";

$conn = new mysqli($servername, $username, $password, $dbname);

if ($conn->connect\_error) {

die("Connection failed: " . $conn->connect\_error);

}

?>

Web Page to Display Books:

php

<?php

// catalog.php

include 'db.php';

$sql = "SELECT books.title, books.author, users.name AS user\_name FROM books JOIN users ON books.user\_id = users.id";

$result = $conn->query($sql);

?>

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<title>Book Catalog</title>

</head>

<body>

<h1>Book Catalog</h1>

<table border="1">

<tr>

<th>Title</th>

<th>Author</th>

<th>Uploaded By</th>

</tr>

<?php

if ($result->num\_rows > 0) {

while($row = $result->fetch\_assoc()) {

echo "<tr>

<td>{$row['title']}</td>

<td>{$row['author']}</td>

<td>{$row['user\_name']}</td>

</tr>";

}

} else {

echo "<tr><td colspan='3'>No books found</td></tr>";

}

$conn->close();

?>

</table>

</body>

</html>

**28. Develop Email Verification Application Using PHP**

Simple Email Verification Script

User Registration with Email Send:

php

<?php

// registration.php

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

$email = $\_POST['email'];

$token = bin2hex(random\_bytes(50)); // Generate a random token

// Database connection

$conn = new mysqli('localhost', 'username', 'password', 'email\_verification');

$sql = "INSERT INTO users (email, token) VALUES ('$email', '$token')";

if ($conn->query($sql) === TRUE) {

$subject = "Email Verification";

$message = "Click the link to verify your email: http://yourdomain.com/verify.php?token=$token";

mail($email, $subject, $message);

echo "Registration successful! Please check your email.";

}

$conn->close();

}

?>

<form method="post">

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

<input type="submit" value="Register">

</form>

Email Verification:

php

<?php

// verify.php

if (isset($\_GET['token'])) {

$token = $\_GET['token'];

// Database connection

$conn = new mysqli('localhost', 'username', 'password', 'email\_verification');

$sql = "UPDATE users SET verified = 1 WHERE token = '$token'";

if ($conn->query($sql) === TRUE) {

echo "Email verified successfully!";

} else {

echo "Invalid token.";

}

$conn->close();

}

?>

**29. Develop Web Application Using PHP**

Basic Web Application Structure

User Authentication: Implement user login and registration.

CRUD Operations: Create, Read, Update, Delete for resource management.

Sample CRUD for a Simple Notes App:

Create: Use the $\_POST method to insert notes into the database.

Read: Fetch notes from the database to display them on the webpage.

Update: Provide a form to update existing notes.

Delete: Implement a button to remove notes.

**30. Design Restaurant Data Entry Form Using Table Layout**

HTML Form Using Table Layout for Restaurant Data Entry

html

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<title>Restaurant Data Entry</title>

<style>

table { border-collapse: collapse; }

td, th { border: 1px solid #ddd; padding: 8px; }

</style>

</head>

<body>

<h1>Restaurant Data Entry Form</h1>

<form action="submit\_restaurant.php" method="post">

<table>

<tr>

<th>Field</th>

<th>Input</th>

</tr>

<tr>

<td>Restaurant Name</td>

<td><input type="text" name="name" required></td>

</tr>

<tr>

<td>Address</td>

<td><input type="text" name="address" required></td>

</tr>

<tr>

<td>Cuisine Type</td>

<td><input type="text" name="cuisine" required></td>

</tr>

<tr>

<td>Contact Number</td>

<td><input type="text" name="contact" required></td>

</tr>

<tr>

<td colspan="2"><input type="submit" value="Submit"></td>

</tr>

</table>

</form>

</body>

</html>

**31. Create a Website to Register a User and Validate User’s Input Using ASP.NET**

**ASP.NET Core User Registration Example**

Create a New ASP.NET Core Project:

Use Visual Studio to create a new ASP.NET Core Web Application.

Choose "Web Application (Model-View-Controller)" template.

Model Class:

Create a User model in Models/User.cs.

csharp

using System.ComponentModel.DataAnnotations;

public class User

{

[Required(ErrorMessage = "Username is required")]

public string Username { get; set; }

[Required(ErrorMessage = "Password is required")]

[DataType(DataType.Password)]

public string Password { get; set; }

[Required(ErrorMessage = "Email is required")]

[EmailAddress(ErrorMessage = "Invalid Email Address")]

public string Email { get; set; }

}

Registration View:

Create a registration view in Views/Account/Register.cshtml.

html

@model User

<form asp-action="Register" method="post">

<div>

<label asp-for="Username"></label>

<input asp-for="Username" />

<span asp-validation-for="Username"></span>

</div>

<div>

<label asp-for="Password"></label>

<input asp-for="Password" type="password" />

<span asp-validation-for="Password"></span>

</div>

<div>

<label asp-for="Email"></label>

<input asp-for="Email" />

<span asp-validation-for="Email"></span>

</div>

<button type="submit">Register</button>

</form>

Account Controller:

Implement the registration action in Controllers/AccountController.cs.

csharp

using Microsoft.AspNetCore.Mvc;

public class AccountController : Controller

{

[HttpGet]

public IActionResult Register()

{

return View();

}

[HttpPost]

public IActionResult Register(User user)

{

if (ModelState.IsValid)

{

// Save user to database (not implemented here)

return RedirectToAction("Index", "Home");

}

return View(user);

}

}

Validation:

Ensure you have client-side validation using jQuery Unobtrusive Validation in \_Layout.cshtml.

html

<script src="https://ajax.aspnetcdn.com/jquery/jquery-3.4.1.min.js"></script>

<script src="https://ajax.aspnetcdn.com/jquery.validation/1.19.2/jquery.validate.min.js"></script>

<script src="https://ajax.aspnetcdn.com/jquery.validation.unobtrusive/3.2.12/jquery.validate.unobtrusive.min.js"></script>

**32. Design UI and Perform CRUD Operations on MySQL Using ASP.NET**

ASP.NET Core with MySQL for CRUD Operations

Install Required Packages:

Use NuGet Package Manager to install:

Pomelo.EntityFrameworkCore.MySql

Microsoft.EntityFrameworkCore.Tools

Database Context:

Create a DataContext class in Data/DataContext.cs.

csharp

using Microsoft.EntityFrameworkCore;

public class DataContext : DbContext

{

public DataContext(DbContextOptions<DataContext> options) : base(options) { }

public DbSet<Product> Products { get; set; }

}

Model Class:

Create a Product model in Models/Product.cs.

csharp

public class Product

{

public int Id { get; set; }

public string Name { get; set; }

public decimal Price { get; set; }

}

Configuration in Startup.cs:

Configure the MySQL database in ConfigureServices method.

csharp

public void ConfigureServices(IServiceCollection services)

{

services.AddDbContext<DataContext>(opt => opt.UseMySql(Configuration.GetConnectionString("DefaultConnection"),

new MySqlServerVersion(new Version(8, 0, 21))));

services.AddControllersWithViews();

}

Product Controller:

Create a ProductsController for CRUD operations.

csharp

using Microsoft.AspNetCore.Mvc;

public class ProductsController : Controller

{

private readonly DataContext \_context;

public ProductsController(DataContext context)

{

\_context = context;

}

public async Task<IActionResult> Index() => View(await \_context.Products.ToListAsync());

[HttpGet]

public IActionResult Create() => View();

[HttpPost]

public async Task<IActionResult> Create(Product product)

{

if (ModelState.IsValid)

{

\_context.Products.Add(product);

await \_context.SaveChangesAsync();

return RedirectToAction(nameof(Index));

}

return View(product);

}

[HttpGet]

public async Task<IActionResult> Edit(int id) => View(await \_context.Products.FindAsync(id));

[HttpPost]

public async Task<IActionResult> Edit(Product product)

{

if (ModelState.IsValid)

{

\_context.Products.Update(product);

await \_context.SaveChangesAsync();

return RedirectToAction(nameof(Index));

}

return View(product);

}

... // Implement Delete and Details similarly

}

Create Views: Create views for Index, Create, Edit using Razor syntax.

Example: Create.cshtml

html

@model Product

<form asp-action="Create" method="post">

<label asp-for="Name"></label>

<input asp-for="Name" required />

<label asp-for="Price"></label>

<input asp-for="Price" required />

<button type="submit">Create</button>

</form>

**33. Install Ruby Environment and Write a Ruby Program to Print Reversed Names**

**Ruby Environment Setup**:

Install Ruby: Install Ruby using RubyInstaller for Windows or use Homebrew for macOS.

IDE/Text Editor: Use any editor like VS Code, Sublime Text, or RubyMine.

Ruby Program to Reverse User's Names:

ruby

# reverse\_names.rb

puts "Enter your first name:"

first\_name = gets.chomp

puts "Enter your last name:"

last\_name = gets.chomp

# Print names in reverse order

puts "#{last\_name} #{first\_name}"

Run the Program:

bash

ruby reverse\_names.rb

**34. Write a Ruby Script to Send an Email to a Specific User**

**Using Ruby's SMTP Library:**

Install the Required Gem: Make sure to have the mail gem. You can install it using:

bash

gem install mail

Ruby Script for Sending Email:

ruby

# send\_email.rb

require 'mail'

options = {

address: "smtp.gmail.com",

port: 587,

user\_name: 'your\_email@gmail.com',

password: 'your\_password',

authentication: 'plain',

enable\_starttls\_auto: true

}

Mail.defaults do

delivery\_method :smtp, options

end

Mail.deliver do

to 'recipient\_email@example.com'

from 'your\_email@gmail.com'

subject 'Test Email'

body 'This is a test email from Ruby!'

end

puts "Email sent successfully!"

Run the Email Script:

bash

ruby send\_email.rb