Project: Online Doctor Consultation System

This document contains the complete plan and source code for the Online Doctor Consultation System. It is organized into the following sections:

1. **Software Requirement Specification (SRS)**: Defines the project's objectives, scope, and requirements.
2. **System Design**: Visualizes the system's architecture with DFDs, an ER diagram, and UI mockups.
3. **Database Schema**: Provides the SQL code to create the necessary MySQL database and tables.
4. **Source Code**: Includes the full PHP, HTML, and CSS code for the application, separated into logical files.

1. Software Requirement Specification (SRS)

1.1. Introduction

This document outlines the requirements for the Online Doctor Consultation System. The system will provide a platform for patients to consult with doctors remotely. It will feature a subscription-based model, offering different levels of access and features to patients.

1.2. Project Objectives

- To create a secure and reliable platform for online medical consultations.
- To allow patients to find and book appointments with registered doctors.
- To implement a subscription model with different tiers (e.g., Basic, Premium).
- To provide a simple interface for doctors to manage their appointments and patient interactions.
- To ensure patient data privacy and security.
- To facilitate basic communication (chat/messaging) between patients and doctors.

1.3. Scope

The system will be a web-based application accessible through standard web browsers. It will focus on non-emergency medical consultations.

In-Scope:

- User registration and login for Patients, Doctors, and an Administrator.
- Subscription plan selection and management for patients.
- Viewing doctor profiles and availability.

- Appointment booking and scheduling.
- A simple text-based chat interface for consultations.
- Admin dashboard for managing users and subscriptions.

Out-of-Scope:

- Video or audio calls.
- E-prescription generation.
- Payment gateway integration (subscriptions will be simulated).
- Mobile application development.

1.4. User Roles and Characteristics

1. **Patient**:
    - Can register, log in, and manage their profile.
    - Can view and subscribe to different plans.
    - Can search for doctors.
    - Can book appointments with doctors.
    - Can communicate with the doctor via chat during the appointment slot.
2. **Doctor**:
    - Can register (subject to admin approval), log in, and manage their profile (specialty, availability, etc.).
    - Can view their upcoming appointments.
    - Can communicate with patients via chat during a scheduled consultation.
3. **Administrator**:
    - Can log in to a dedicated admin panel.
    - Can view and manage all patient and doctor accounts.
    - Can approve new doctor registrations.
    - Can view subscription and appointment data.

1.5. Functional Requirements

- **User Management**: Registration, Login, Profile Management.
- **Subscription Module**: Display plans, handle patient subscriptions.
- **Doctor Search**: Patients can search for doctors by specialty.
- **Appointment Booking**: Patients can select a doctor and a time slot.
- **Consultation (Chat)**: A simple, real-time chat interface.
- **Admin Dashboard**: User management and system overview.

2. System Design

2.1. Data Flow Diagrams (DFD)

DFD Level 0 (Context Diagram):

- *Shows the entire system as a single process with external entities (Patient, Doctor, Admin) interacting with it.*

DFD Level 1:

- *Breaks down the system into major processes like 'Manage Users', 'Handle Subscriptions', 'Book Appointments', and 'Conduct Consultation', showing data flows between them and data stores (e.g., Users, Appointments).*

2.2. Entity-Relationship (ER) Diagram

This diagram shows the structure of the database and the relationships between different tables.

Entities:

- users (stores common data for patients, doctors, admins)
- doctors_profiles (stores doctor-specific details)
- subscriptions (stores patient subscription status)
- appointments (stores booking information)
- chats (stores messages between doctor and patient)

2.3. UI Mockups (Basic Wireframes)

Patient Dashboard:

```
+--------------------------------------------------------+
| Online Doctor Consultation | My Profile | Logout    |
+--------------------------------------------------------+
|                                                        |
| Welcome, [Patient Name]! (Premium Subscriber)      |
|                                                        |
| +------------------------+ +------------------------+ |
| |  Book Appointment    | |  My Appointments     | |
| +------------------------+ +------------------------+ |
|                                                        |
| Find a Doctor: [Search by Specialty v] [Search]     |
|                                                        |
| - Dr. Smith (Cardiology)      [View Profile]        |
| - Dr. Jones (Dermatology)     [View Profile]        |
|                                                        |
+--------------------------------------------------------+
```

Doctor Dashboard:

```
+-------------------------------------------------------+
|  Online Doctor Consultation | My Profile | Logout    |
+-------------------------------------------------------+
|                                        |
|  Welcome, Dr. [Doctor Name]!                       |
|                                        |
|  +-------------------------------------------------+  |
| |            Upcoming Appointments            | |
|  +-------------------------------------------------+  |
| | - [Patient A] at [10:00 AM]  [Start Chat]      | |
| | - [Patient B] at [11:00 AM]  [Start Chat]      | |
|  +-------------------------------------------------+  |
|                                        |
+-------------------------------------------------------+
```

3. Database Schema (MySQL)

This SQL script creates the database and all the necessary tables. Save it as database.sql and import it into your MySQL server (e.g., via phpMyAdmin).

```
CREATE DATABASE `doctor_consultation_system`;

USE `doctor_consultation_system`;

--
-- Table structure for table `users`
--
CREATE TABLE `users` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `username` varchar(50) NOT NULL,
  `password` varchar(255) NOT NULL,
  `email` varchar(100) NOT NULL,
  `role` enum('patient','doctor','admin') NOT NULL,
  `created_at` timestamp NOT NULL DEFAULT current_timestamp(),
  PRIMARY KEY (`id`),
  UNIQUE KEY `email` (`email`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;
```

```
--
-- Dumping data for table `users` (for testing)
--
INSERT INTO `users` (`username`, `password`, `email`, `role`) VALUES
('admin', '$2y$10$R.h2F8aZJ.3A9Z7.jE9hA.C/oJ5G9j3D3B6C2A1k0l9g8h7f6e5d',
'admin@example.com', 'admin'),
('testpatient', '$2y$10$S.i2F8bZJ.4A9Z7.jE9hA.C/oJ5G9j3D3B6C2A1k0l9g8h7f6e5d',
'patient@example.com', 'patient'),
('testdoctor', '$2y$10$T.j2F8cZJ.5A9Z7.jE9hA.C/oJ5G9j3D3B6C2A1k0l9g8h7f6e5d',
'doctor@example.com', 'doctor');


--
-- Table structure for table `doctors_profiles`
--
CREATE TABLE `doctors_profiles` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `user_id` int(11) NOT NULL,
  `specialty` varchar(100) NOT NULL,
  `approved` tinyint(1) NOT NULL DEFAULT 0,
  PRIMARY KEY (`id`),
  KEY `user_id` (`user_id`),
  CONSTRAINT `doctors_profiles_ibfk_1` FOREIGN KEY (`user_id`) REFERENCES
`users` (`id`) ON DELETE CASCADE
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;

-- Dumping data for doctors_profiles (for testing)
INSERT INTO `doctors_profiles` (`user_id`, `specialty`, `approved`) VALUES
((SELECT id FROM users WHERE email='doctor@example.com'), 'Cardiology', 1);

--
-- Table structure for table `subscriptions`
--
CREATE TABLE `subscriptions` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `patient_id` int(11) NOT NULL,
  `plan` enum('basic','premium') NOT NULL,
  `start_date` date NOT NULL,
```

```sql
  `end_date` date NOT NULL,
  PRIMARY KEY (`id`),
  KEY `patient_id` (`patient_id`),
  CONSTRAINT `subscriptions_ibfk_1` FOREIGN KEY (`patient_id`) REFERENCES
`users` (`id`) ON DELETE CASCADE
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;

-- Dumping data for subscriptions (for testing)
INSERT INTO `subscriptions` (`patient_id`, `plan`, `start_date`, `end_date`) VALUES
((SELECT id FROM users WHERE email='patient@example.com'), 'premium',
CURDATE(), DATE_ADD(CURDATE(), INTERVAL 1 MONTH));


--
-- Table structure for table `appointments`
--
CREATE TABLE `appointments` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `patient_id` int(11) NOT NULL,
  `doctor_id` int(11) NOT NULL,
  `appointment_time` datetime NOT NULL,
  `status` enum('scheduled','completed','cancelled') NOT NULL DEFAULT 'scheduled',
  PRIMARY KEY (`id`),
  KEY `patient_id` (`patient_id`),
  KEY `doctor_id` (`doctor_id`),
  CONSTRAINT `appointments_ibfk_1` FOREIGN KEY (`patient_id`) REFERENCES
`users` (`id`) ON DELETE CASCADE,
  CONSTRAINT `appointments_ibfk_2` FOREIGN KEY (`doctor_id`) REFERENCES
`users` (`id`) ON DELETE CASCADE
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;


--
-- Table structure for table `chats`
--
CREATE TABLE `chats` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `appointment_id` int(11) NOT NULL,
  `sender_id` int(11) NOT NULL,
  `message` text NOT NULL,
  `timestamp` timestamp NOT NULL DEFAULT current_timestamp(),
  PRIMARY KEY (`id`),
  KEY `appointment_id` (`appointment_id`),
  KEY `sender_id` (`sender_id`),
  CONSTRAINT `chats_ibfk_1` FOREIGN KEY (`appointment_id`) REFERENCES
`appointments` (`id`) ON DELETE CASCADE,
  CONSTRAINT `chats_ibfk_2` FOREIGN KEY (`sender_id`) REFERENCES `users` (`id`)
ON DELETE CASCADE
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;
```

4. Source Code (PHP & HTML)

Here is the file structure for the project. Create these files and folders in your web server's root directory (e.g., htdocs or www).

```
/doctor-consultation/
|-- config.php
|-- index.php
|-- login.php
|-- register.php
|-- logout.php
|-- patient_dashboard.php
|-- doctor_dashboard.php
|-- book_appointment.php
|-- chat.php
|-- style.css
```

config.php

This file handles the database connection.

```
<?php
// Start the session
session_start();

// Database credentials
define('DB_SERVER', 'localhost');
define('DB_USERNAME', 'root'); // Your DB username
define('DB_PASSWORD', ''); // Your DB password
define('DB_NAME', 'doctor_consultation_system');

// Attempt to connect to MySQL database
$conn = new mysqli(DB_SERVER, DB_USERNAME, DB_PASSWORD, DB_NAME);

// Check connection
if($conn === false){
    die("ERROR: Could not connect. " . $conn->connect_error);
}

// A simple function to redirect users
function redirect($url) {
    header("location: " . $url);
    exit;
}
?>
```

style.css

A simple stylesheet for a clean look.

```
body {
    font-family: Arial, sans-serif;
    background-color: #f4f4f9;
    margin: 0;
    padding: 20px;
    color: #333;
}
.container {
    max-width: 800px;
    margin: auto;
    background: #fff;
    padding: 20px;
    border-radius: 8px;
    box-shadow: 0 0 10px rgba(0,0,0,0.1);
}
h1, h2 {
```

```css
    color: #0056b3;
}
input[type="text"], input[type="password"], input[type="email"], select {
    width: 100%;
    padding: 8px;
    margin: 10px 0;
    display: inline-block;
    border: 1px solid #ccc;
    border-radius: 4px;
    box-sizing: border-box;
}
input[type="submit"], .btn {
    background-color: #0056b3;
    color: white;
    padding: 10px 15px;
    margin: 8px 0;
    border: none;
    border-radius: 4px;
    cursor: pointer;
    text-decoration: none;
    display: inline-block;
}
input[type="submit"]:hover, .btn:hover {
    background-color: #004494;
}
.error {
    color: red;
    font-size: 0.9em;
}
.navbar {
    overflow: hidden;
    background-color: #333;
    margin-bottom: 20px;
    border-radius: 8px;
}
.navbar a {
    float: left;
    display: block;
    color: #f2f2f2;
    text-align: center;
    padding: 14px 16px;
    text-decoration: none;
}
.navbar a.right {
    float: right;
}
.navbar a:hover {
    background-color: #ddd;
    color: black;
}
.chat-box {
    height: 300px;
    border: 1px solid #ccc;
    overflow-y: scroll;
    padding: 10px;
    margin-bottom: 10px;
}
.chat-message {
    margin-bottom: 10px;
}
```

index.php

The landing page that directs users based on their role.

```
<?php
require_once 'config.php';

// If user is not logged in, redirect to login page
if (!isset($_SESSION['loggedin']) || $_SESSION['loggedin'] !== true) {
    redirect('login.php');
}

// Redirect based on role
$role = $_SESSION['role'];
if ($role == 'patient') {
    redirect('patient_dashboard.php');
} elseif ($role == 'doctor') {
    redirect('doctor_dashboard.php');
} else {
    // For admin or other roles
    echo "Welcome, Admin!";
    // You can build an admin dashboard here
    echo '<br><a href="logout.php">Logout</a>';
}
?>
```

register.php

```
<?php
require_once 'config.php';
$username = $email = $password = $role = "";
$username_err = $email_err = $password_err = $role_err = "";

if ($_SERVER["REQUEST_METHOD"] == "POST") {
    // Validate and process form data
    // ... (Validation logic here for brevity)
    $username = trim($_POST["username"]);
    $email = trim($_POST["email"]);
    $password = password_hash(trim($_POST["password"]), PASSWORD_DEFAULT);
    $role = trim($_POST["role"]);

    $sql = "INSERT INTO users (username, email, password, role) VALUES (?, ?, ?, ?)";
    if ($stmt = $conn->prepare($sql)) {
        $stmt->bind_param("ssss", $username, $email, $password, $role);
        if ($stmt->execute()) {
            if ($role == 'doctor') {
                $user_id = $stmt->insert_id;
                $specialty = trim($_POST['specialty']);
                $sql_doctor = "INSERT INTO doctors_profiles (user_id, specialty) VALUES (?,
?)";
                if($stmt_doctor = $conn->prepare($sql_doctor)){
                    $stmt_doctor->bind_param("is", $user_id, $specialty);
                    $stmt_doctor->execute();
                }
            }
            redirect("login.php");
        } else {
            echo "Something went wrong. Please try again later.";
        }
        $stmt->close();
    }
}
?>

<!DOCTYPE html>
<html>
<head>
    <title>Register</title>
    <link rel="stylesheet" href="style.css">
</head>
<body>
<div class="container">
    <h2>Register</h2>
    <form action="<?php echo htmlspecialchars($_SERVER["PHP_SELF"]); ?>"
method="post">
        <div>
            <label>Username</label>
            <input type="text" name="username" required>
        </div>
        <div>
            <label>Email</label>
            <input type="email" name="email" required>
        </div>
        <div>
            <label>Password</label>
            <input type="password" name="password" required>
        </div>
        <div>
            <label>I am a:</label>
            <select name="role" id="role_select" onchange="toggleSpecialty()" required>
                <option value="patient">Patient</option>
                <option value="doctor">Doctor</option>
            </select>
        </div>
        <div id="specialty_field" style="display:none;">
```

```
            <label>Specialty</label>
            <input type="text" name="specialty">
        </div>
        <div>
            <input type="submit" value="Register">
        </div>
        <p>Already have an account? <a href="login.php">Login here</a>.</p>
    </form>
</div>
<script>
function toggleSpecialty() {
    var role = document.getElementById('role_select').value;
    var specialtyField = document.getElementById('specialty_field');
    if (role === 'doctor') {
        specialtyField.style.display = 'block';
    } else {
        specialtyField.style.display = 'none';
    }
}
</script>
</body>
</html>
```

login.php

```
<?php
require_once 'config.php';
$email = $password = "";
$email_err = $password_err = $login_err = "";

if ($_SERVER["REQUEST_METHOD"] == "POST") {
    $email = trim($_POST["email"]);
    $password = trim($_POST["password"]);

    $sql = "SELECT id, username, password, role FROM users WHERE email = ?";
    if ($stmt = $conn->prepare($sql)) {
        $stmt->bind_param("s", $email);
        if ($stmt->execute()) {
            $stmt->store_result();
            if ($stmt->num_rows == 1) {
                $stmt->bind_result($id, $username, $hashed_password, $role);
                if ($stmt->fetch()) {
                    if (password_verify($password, $hashed_password)) {
                        $_SESSION["loggedin"] = true;
                        $_SESSION["id"] = $id;
                        $_SESSION["username"] = $username;
                        $_SESSION["role"] = $role;
                        redirect("index.php");
                    } else {
                        $login_err = "Invalid email or password.";
                    }
                }
            } else {
                $login_err = "Invalid email or password.";
            }
        }
        $stmt->close();
    }
}
?>

<!DOCTYPE html>
<html>
<head>
    <title>Login</title>
    <link rel="stylesheet" href="style.css">
</head>
<body>
<div class="container">
    <h2>Login</h2>
    <?php if(!empty($login_err)){ echo '<div class="error">'.$login_err.'</div>'; } ?>
    <form action="<?php echo htmlspecialchars($_SERVER["PHP_SELF"]); ?>"
method="post">
        <div>
            <label>Email</label>
            <input type="email" name="email" required>
        </div>
        <div>
```

```
            <label>Password</label>
            <input type="password" name="password" required>
        </div>
        <div>
            <input type="submit" value="Login">
        </div>
        <p>Don't have an account? <a href="register.php">Sign up now</a>.</p>
    </form>
</div>
</body>
</html>
```

patient_dashboard.php

```
<?php
require_once 'config.php';
if (!isset($_SESSION['loggedin']) || $_SESSION['role'] != 'patient') {
    redirect('login.php');
}

$patient_id = $_SESSION['id'];
?>

<!DOCTYPE html>
<html>
<head>
    <title>Patient Dashboard</title>
    <link rel="stylesheet" href="style.css">
</head>
<body>
<div class="navbar">
    <a href="#">Dashboard</a>
    <a href="logout.php" class="right">Logout</a>
</div>
<div class="container">
    <h2>Welcome, <?php echo htmlspecialchars($_SESSION["username"]); ?>!</h2>
    <h3>Your Appointments</h3>
    <?php
    $sql = "SELECT a.id, u.username as doctor_name, a.appointment_time, a.status
```

```
FROM appointments a JOIN users u ON a.doctor_id = u.id WHERE a.patient_id = ?
ORDER BY a.appointment_time DESC";
    if($stmt = $conn->prepare($sql)){
        $stmt->bind_param("i", $patient_id);
        $stmt->execute();
        $result = $stmt->get_result();
        if($result->num_rows > 0){
            echo "<ul>";
            while($row = $result->fetch_assoc()){
                echo "<li>Dr. " . $row['doctor_name'] . " on " . $row['appointment_time'] . " -
Status: " . $row['status'];
                if($row['status'] == 'scheduled'){
                    echo " <a href='chat.php?id=".$row['id']."' class='btn'>Join Chat</a>";
                }
                echo "</li>";
            }
            echo "</ul>";
        } else {
            echo "<p>You have no appointments.</p>";
        }
    }
    ?>

    <hr>
    <h3>Book a New Appointment</h3>
    <form action="book_appointment.php" method="post">
        <label>Find Doctor by Specialty:</label>
        <select name="doctor_id" required>
            <option value="">--Select a Doctor--</option>
            <?php
            $sql_doctors = "SELECT u.id, u.username, dp.specialty FROM users u JOIN
doctors_profiles dp ON u.id = dp.user_id WHERE u.role = 'doctor' AND dp.approved =
1";
            $result_doctors = $conn->query($sql_doctors);
            while($doctor = $result_doctors->fetch_assoc()){
                echo "<option value='".$doctor['id']."'>Dr. ".$doctor['username']."
(".$doctor['specialty'].")</option>";
            }
            ?>
        </select>
        <label>Appointment Time:</label>
        <input type="datetime-local" name="appointment_time" required>
        <input type="submit" value="Book Appointment">
    </form>
</div>
</body>
</html>
```

doctor_dashboard.php

```
<?php
require_once 'config.php';
if (!isset($_SESSION['loggedin']) || $_SESSION['role'] != 'doctor') {
    redirect('login.php');
}

$doctor_id = $_SESSION['id'];
?>

<!DOCTYPE html>
<html>
<head>
    <title>Doctor Dashboard</title>
    <link rel="stylesheet" href="style.css">
</head>
<body>
<div class="navbar">
    <a href="#">Dashboard</a>
    <a href="logout.php" class="right">Logout</a>
</div>
<div class="container">
    <h2>Welcome, Dr. <?php echo htmlspecialchars($_SESSION["username"]); ?>!</h2>
    <h3>Your Upcoming Appointments</h3>
    <?php
    $sql = "SELECT a.id, u.username as patient_name, a.appointment_time, a.status
FROM appointments a JOIN users u ON a.patient_id = u.id WHERE a.doctor_id = ? AND
a.status = 'scheduled' ORDER BY a.appointment_time ASC";
    if($stmt = $conn->prepare($sql)){
        $stmt->bind_param("i", $doctor_id);
        $stmt->execute();
        $result = $stmt->get_result();
        if($result->num_rows > 0){
            echo "<ul>";
            while($row = $result->fetch_assoc()){
                echo "<li>" . $row['patient_name'] . " on " . $row['appointment_time'] . " <a
href='chat.php?id=".$row['id']."' class='btn'>Start Chat</a></li>";
            }
            echo "</ul>";
        } else {
            echo "<p>You have no upcoming appointments.</p>";
        }
    }
    ?>
</div>
</body>
</html>
```

book_appointment.php

```
<?php
require_once 'config.php';
if (!isset($_SESSION['loggedin']) || $_SESSION['role'] != 'patient') {
    redirect('login.php');
}

if ($_SERVER["REQUEST_METHOD"] == "POST") {
    $patient_id = $_SESSION['id'];
    $doctor_id = $_POST['doctor_id'];
    $appointment_time = $_POST['appointment_time'];

    // For simplicity, we assume premium plan. Add logic here to check subscription.
    // For example: check if user has an active premium subscription from the
'subscriptions' table.

    $sql = "INSERT INTO appointments (patient_id, doctor_id, appointment_time)
VALUES (?, ?, ?)";
    if($stmt = $conn->prepare($sql)){
        $stmt->bind_param("iis", $patient_id, $doctor_id, $appointment_time);
        if($stmt->execute()){
            redirect('patient_dashboard.php');
        } else {
            echo "Error booking appointment.";
        }
    }
}
?>
```

chat.php

```
<?php
require_once 'config.php';
if (!isset($_SESSION['loggedin'])) {
    redirect('login.php');
}

$appointment_id = $_GET['id'];
$user_id = $_SESSION['id'];

// Fetch appointment details to verify user has access
// ... (Add verification logic here)

// Handle new message submission
if ($_SERVER["REQUEST_METHOD"] == "POST" && !empty($_POST['message'])) {
    $message = trim($_POST['message']);
    $sql = "INSERT INTO chats (appointment_id, sender_id, message) VALUES (?, ?, ?)";
    if($stmt = $conn->prepare($sql)){
        $stmt->bind_param("iis", $appointment_id, $user_id, $message);
        $stmt->execute();
        // Redirect to the same page to prevent form resubmission
        redirect("chat.php?id=" . $appointment_id);
    }
}
?>

<!DOCTYPE html>
<html>
```

```
<head>
   <title>Consultation Chat</title>
   <link rel="stylesheet" href="style.css">
   <!-- Auto-refresh the chat every 5 seconds -->
   <meta http-equiv="refresh" content="5">
</head>
<body>
<div class="container">
   <h2>Chat</h2>
   <div class="chat-box">
      <?php
      $sql_chat = "SELECT c.message, u.username as sender_name, c.timestamp
FROM chats c JOIN users u ON c.sender_id = u.id WHERE c.appointment_id = ?
ORDER BY c.timestamp ASC";
      if($stmt_chat = $conn->prepare($sql_chat)){
         $stmt_chat->bind_param("i", $appointment_id);
         $stmt_chat->execute();
         $result = $stmt_chat->get_result();
         while($row = $result->fetch_assoc()){
            echo "<div
class='chat-message'><strong>".htmlspecialchars($row['sender_name']).":</strong>
".htmlspecialchars($row['message'])."</div>";
         }
      }
      ?>
   </div>
   <form action="chat.php?id=<?php echo $appointment_id; ?>" method="post">
      <input type="text" name="message" placeholder="Type your message..."
required autocomplete="off">
      <input type="submit" value="Send">
   </form>
   <br>
   <a href="index.php" class="btn">Back to Dashboard</a>
</div>
</body>
</html>
```

logout.php

```php
<?php
// Initialize the session
session_start();

// Unset all of the session variables
$_SESSION = array();

// Destroy the session.
session_destroy();

// Redirect to login page
header("location: login.php");
exit;
?>
```