# Smart Healthcare

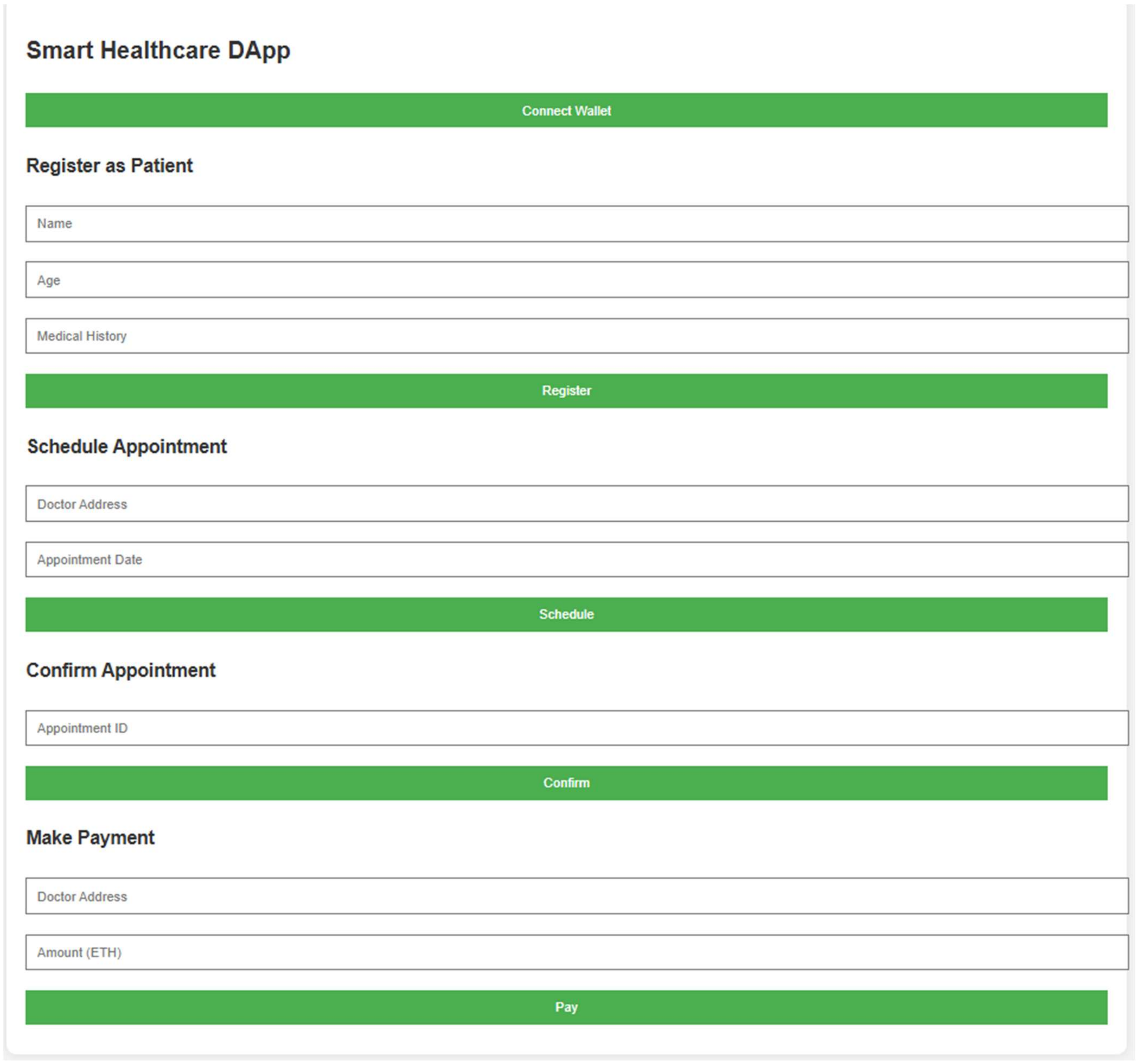
## Smarthealthcare.sol

```solidity
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.19;

contract SmartHealthcare {
    struct Patient {
        string name;
        uint age;
        string medicalHistory;
        bool isRegistered;
    }

    struct Appointment {
        address patient;
        address doctor;
        string date;
        bool isConfirmed;
    }

    struct Payment {
        address patient;
        address doctor;
        uint amount;
        bool isPaid;
    }

    mapping(address => Patient) public patients;
    mapping(uint => Appointment) public appointments;
    mapping(uint => Payment) public payments;

    uint public appointmentCount;
    uint public paymentCount;

    address public feeCollector; // Address to receive the service fee
    uint public feePercentage = 5; // Example fee percentage

    uint public constant REQUIRED_PAYMENT = 100000000000000; // 0.0001 ETH in Wei

    event PatientRegistered(address indexed patient, string name);
    event AppointmentScheduled(uint indexed appointmentId, address indexed patient,
address indexed doctor, string date);
    event AppointmentConfirmed(uint indexed appointmentId, address indexed doctor);
    event PaymentProcessed(uint indexed paymentId, address indexed patient, address
indexed doctor, uint amount);

    modifier onlyRegisteredPatient() {
        require(patients[msg.sender].isRegistered, "Patient not registered");
        _;
    }

    constructor(address _feeCollector) {
        feeCollector = _feeCollector; // Set the fee collector address
    }

    function registerPatient(string memory _name, uint _age, string memory
_medicalHistory) public {
        require(!patients[msg.sender].isRegistered, "Already registered");
        patients[msg.sender] = Patient(_name, _age, _medicalHistory, true);
        emit PatientRegistered(msg.sender, _name);
    }

    function scheduleAppointment(address _doctor, string memory _date) public
onlyRegisteredPatient {
        appointmentCount++;
        appointments[appointmentCount] = Appointment(msg.sender, _doctor, _date,
false);
        emit AppointmentScheduled(appointmentCount, msg.sender, _doctor, _date);
    }

    function confirmAppointment(uint _appointmentId) public {
        require(appointments[_appointmentId].doctor == msg.sender, "Only doctor can
confirm");
        require(!appointments[_appointmentId].isConfirmed, "Appointment already
confirmed");
        appointments[_appointmentId].isConfirmed = true;
        emit AppointmentConfirmed(_appointmentId, msg.sender);
    }

    function makePayment(address _doctor) public payable onlyRegisteredPatient {
        require(msg.value == REQUIRED_PAYMENT, "Payment must be exactly 0.0001 ETH");

        uint fee = (msg.value * feePercentage) / 100; // Calculate the service fee
        uint doctorPayment = msg.value - fee; // The amount the doctor receives

        payable(feeCollector).transfer(fee); // Transfer fee to the fee collector
        payable(_doctor).transfer(doctorPayment); // Transfer payment to the doctor

        paymentCount++;
        payments[paymentCount] = Payment(msg.sender, _doctor, msg.value, true);

        emit PaymentProcessed(paymentCount, msg.sender, _doctor, msg.value);
    }
}
```

**Smarthealthcare.html**

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Smart Healthcare DApp</title>
    <script src="https://cdn.jsdelivr.net/npm/web3@latest/dist/web3.min.js"></script>
    <style>
        body {
            font-family: Arial, sans-serif;
            margin: 20px;
            padding: 20px;
            background-color: #f4f4f4;
        }
        .container {
            background: white;
            padding: 20px;
            border-radius: 10px;
            box-shadow: 0px 0px 10px rgba(0,0,0,0.1);
            width: 50%;
            margin: auto;
        }
        input, button {
            margin: 10px 0;
            padding: 10px;
            width: 100%;
        }
        button {
            background-color: #4CAF50;
            color: white;
            border: none;
            cursor: pointer;
        }
        button:hover {
            background-color: #45a049;
        }
        h2, h3 {
            color: #333;
        }
    </style>
</head>
<body>
    <div class="container">
        <h2>Smart Healthcare DApp</h2>
        <button onclick="connectWallet()">Connect Wallet</button>

        <h3>Register as Patient</h3>
        <input type="text" id="name" placeholder="Name">
        <input type="number" id="age" placeholder="Age">
        <input type="text" id="medicalHistory" placeholder="Medical History">
        <button onclick="registerPatient()">Register</button>

        <h3>Schedule Appointment</h3>
        <input type="text" id="doctorAddress" placeholder="Doctor Address">
        <input type="text" id="appointmentDate" placeholder="Appointment Date">
        <button onclick="scheduleAppointment()">Schedule</button>

        <h3>Confirm Appointment</h3>
        <input type="number" id="appointmentId" placeholder="Appointment ID">
        <button onclick="confirmAppointment()">Confirm</button>

        <h3>Make Payment</h3>
        <input type="text" id="paymentDoctorAddress" placeholder="Doctor Address">
        <input type="number" id="paymentAmount" placeholder="Amount (ETH)">
        <button onclick="makePayment()">Pay</button>
    </div>

    <script>
        let web3;
        let contract;
        const contractAddress = "0x97217C18dE84a8ab5A5764e228778074edCf8BFc";

        // Contract ABI (Replace with your actual contract ABI)
        const contractABI = [
    {
        "inputs": [
            {
                "internalType": "address",
                "name": "_feeCollector",
                "type": "address"
            }
        ],
        "stateMutability": "nonpayable",
        "type": "constructor"
    },
    {
        "anonymous": false,
        "inputs": [
            {
                "indexed": true,
                "internalType": "uint256",
                "name": "appointmentId",
                "type": "uint256"
            },
            {
                "indexed": true,
                "internalType": "address",
                "name": "doctor",
                "type": "address"
            }
        ],
        "name": "AppointmentConfirmed",
        "type": "event"
    },
    {
        "anonymous": false,
        "inputs": [
            {
                "indexed": true,
                "internalType": "uint256",
                "name": "appointmentId",
                "type": "uint256"
            },
            {
                "indexed": true,
                "internalType": "address",
                "name": "patient",
                "type": "address"
            },
            {
                "indexed": true,
                "internalType": "address",
                "name": "doctor",
                "type": "address"
            },
            {
                "indexed": false,
                "internalType": "string",
                "name": "date",
                "type": "string"
            }
        ],
        "name": "AppointmentScheduled",
        "type": "event"
    },
    {
        "inputs": [
            {
                "internalType": "uint256",
                "name": "_appointmentId",
                "type": "uint256"
            }
        ],
        "name": "confirmAppointment",
        "outputs": [],
        "stateMutability": "nonpayable",
        "type": "function"
    },
    {
        "inputs": [
            {
                "internalType": "address",
                "name": "_doctor",
                "type": "address"
            }
        ],
        "name": "makePayment",
        "outputs": [],
        "stateMutability": "payable",
        "type": "function"
    },
    {
        "anonymous": false,
        "inputs": [
            {
                "indexed": true,
                "internalType": "address",
                "name": "patient",
                "type": "address"
            },
            {
                "indexed": false,
                "internalType": "string",
                "name": "name",
                "type": "string"
            }
        ],
        "name": "PatientRegistered",
        "type": "event"
    },
    {
        "anonymous": false,
        "inputs": [
            {
                "indexed": true,
                "internalType": "uint256",
                "name": "paymentId",
                "type": "uint256"
            },
            {
                "indexed": true,
                "internalType": "address",
                "name": "patient",
                "type": "address"
            },
            {
                "indexed": true,
                "internalType": "address",
                "name": "doctor",
                "type": "address"
            },
            {
                "indexed": false,
                "internalType": "uint256",
                "name": "amount",
                "type": "uint256"
            }
        ],
        "name": "PaymentProcessed",
        "type": "event"
    },
    {
        "inputs": [
            {
                "internalType": "string",
                "name": "_name",
                "type": "string"
            },
            {
                "internalType": "uint256",
                "name": "_age",
                "type": "uint256"
            },
            {
                "internalType": "string",
                "name": "_medicalHistory",
                "type": "string"
            }
        ],
        "name": "registerPatient",
        "outputs": [],
        "stateMutability": "nonpayable",
        "type": "function"
    },
    {
        "inputs": [
            {
                "internalType": "address",
                "name": "_doctor",
                "type": "address"
            },
            {
                "internalType": "string",
                "name": "_date",
                "type": "string"
            }
        ],
        "name": "scheduleAppointment",
        "outputs": [],
        "stateMutability": "nonpayable",
        "type": "function"
    },
    {
        "inputs": [],
        "name": "appointmentCount",
        "outputs": [
            {
                "internalType": "uint256",
                "name": "",
                "type": "uint256"
            }
        ],
        "stateMutability": "view",
        "type": "function"
    },
    {
        "inputs": [
            {
                "internalType": "uint256",
                "name": "",
                "type": "uint256"
            }
        ],
        "name": "appointments",
        "outputs": [
            {
                "internalType": "address",
                "name": "patient",
                "type": "address"
            },
            {
                "internalType": "address",
                "name": "doctor",
                "type": "address"
            },
            {
                "internalType": "string",
                "name": "date",
                "type": "string"
            },
            {
                "internalType": "bool",
                "name": "isConfirmed",
                "type": "bool"
            }
        ],
        "stateMutability": "view",
        "type": "function"
    },
    {
        "inputs": [],
        "name": "feeCollector",
        "outputs": [
            {
                "internalType": "address",
                "name": "",
                "type": "address"
            }
        ],
        "stateMutability": "view",
        "type": "function"
    },
    {
        "inputs": [],
        "name": "feePercentage",
        "outputs": [
            {
                "internalType": "uint256",
                "name": "",
                "type": "uint256"
            }
        ],
        "stateMutability": "view",
        "type": "function"
    },
    {
        "inputs": [
            {
                "internalType": "address",
                "name": "",
                "type": "address"
            }
        ],
        "name": "patients",
        "outputs": [
            {
                "internalType": "string",
                "name": "name",
                "type": "string"
            },
            {
                "internalType": "uint256",
                "name": "age",
                "type": "uint256"
            },
            {
                "internalType": "string",
                "name": "medicalHistory",
                "type": "string"
            },
            {
                "internalType": "bool",
                "name": "isRegistered",
                "type": "bool"
            }
        ],
        "stateMutability": "view",
        "type": "function"
    },
    {
        "inputs": [],
        "name": "paymentCount",
        "outputs": [
            {
                "internalType": "uint256",
                "name": "",
                "type": "uint256"
            }
        ],
        "stateMutability": "view",
        "type": "function"
    },
    {
        "inputs": [
            {
                "internalType": "uint256",
                "name": "",
                "type": "uint256"
            }
        ],
        "name": "payments",
        "outputs": [
            {
                "internalType": "address",
                "name": "patient",
                "type": "address"
            },
            {
                "internalType": "address",
                "name": "doctor",
                "type": "address"
            },
            {
                "internalType": "uint256",
                "name": "amount",
                "type": "uint256"
            },
            {
                "internalType": "bool",
                "name": "isPaid",
                "type": "bool"
            }
        ],
        "stateMutability": "view",
        "type": "function"
    },
    {
        "inputs": [],
        "name": "REQUIRED_PAYMENT",
        "outputs": [
            {
                "internalType": "uint256",
                "name": "",
                "type": "uint256"
            }
        ],
        "stateMutability": "view",
        "type": "function"
    }
];

        // Connect to MetaMask
        async function connectWallet() {
            if (window.ethereum) {
                try {
                    web3 = new Web3(window.ethereum);
                    await window.ethereum.request({ method: "eth_requestAccounts" });
                    const accounts = await web3.eth.getAccounts();
                    contract = new web3.eth.Contract(contractABI, contractAddress);
                    alert("Wallet connected: " + accounts[0]);
                    console.log("Connected Account:", accounts[0]);
                } catch (error) {
                    console.error("Connection failed", error);
                    alert("Connection failed: " + error.message);
                }
            } else {
                alert("Please install MetaMask.");
            }
        }

        // Register Patient Function (Placeholder)
        async function registerPatient() {
            alert("Registration functionality to be implemented.");
        }

        // Schedule Appointment Function (Placeholder)
        async function scheduleAppointment() {
            const patient = await web3.eth.getAccounts();
            const doctor = document.getElementById("doctorAddress").value;
            const date = document.getElementById("appointmentDate").value;

            if (!doctor || !date) {
                alert("Please fill in both doctor address and appointment date.");
                return;
            }

            try {
                await contract.methods.scheduleAppointment(patient[0], doctor,
date).send({ from: patient[0] });
                alert("Appointment Scheduled Successfully!");
            } catch (error) {
                console.error("Error scheduling appointment", error);
                alert("Error scheduling appointment: " + error.message);
            }
        }

        // Confirm Appointment Function (Placeholder)
        async function confirmAppointment() {
            alert("Appointment confirmation functionality to be implemented.");
        }
        // Make Payment Function
        async function makePayment() {
            const doctor = document.getElementById("paymentDoctorAddress").value;
            const amountEth = document.getElementById("paymentAmount").value;

            if (!doctor || !amountEth) {
                alert("Please enter doctor address and amount.");
                return;
            }

            const amountInWei = web3.utils.toWei(amountEth, "ether");
            const accounts = await web3.eth.getAccounts();

            console.log("Doctor Address:", doctor);
            console.log("Payment Amount in Wei:", amountInWei);
            console.log("Sender Account:", accounts[0]);

            try {
                await contract.methods.makePayment(doctor).send({
                    from: accounts[0],
                    value: amountInWei
                });

                alert("Payment of " + amountEth + " ETH was successful.");
            } catch (error) {
                console.error("Transaction Failed:", error);
                alert("Payment Failed: " + error.message);
            }
        }
    </script>
</body>
</html>
```

**Output:**



Smart Healthcare DApp

Connect Wallet

Register as Patient

Name

Age

Medical History

Register

Schedule Appointment

Doctor Address

Appointment Date

Schedule

Confirm Appointment

Appointment ID

Confirm

Make Payment

Doctor Address

Amount (ETH)

Pay