**DISTRIBUTED COMPONENTS laboratory**

**MINI PROJECT**

**aim:**

To create a secure application to share the electronic health between doctors using Blockchain.

**PROBLEM DESCRIPTION:**

Electronic Health Records are generated by hospitals every day. These records are now shared as hard copies between doctors. To avoid this situation, electronic health records can be shared using the Blockchain application which will make the records immutable and in turns makes the system secure.

The idea of this project is to develop a decentralized application which enables sharing of Electronic Health Records between multiple doctors and track the flow of Electronic Health Records using Blockchain to improve privacy and security.

**SCOPE:**

* + Hospitals where there are no high technological medical devices
  + Consultancy with senior doctors who are abroad
  + A distributed community among doctors.

**TECHNIQUES USED:**

* + Blockchain – Core technology
  + React-Js – for Front-end
  + Meta mask – to import Blockchain account
  + Ganache – for local Blockchain server
  + Truffle – for local deployment of smart contracts

**CODE:**

pragma solidity ^0.5.0;

pragma experimental ABIEncoderV2;

contract HealthCare {

    struct Doctor{

        string name;

        uint age;

        string field;

        uint myPatientsCount;

        mapping(uint=>Patient) myPatients;

    }

    struct Patient {

        address myDocAddress;

        string name;

        uint age;

        string disease;

        string description;

        string ehrhash;

    }

    uint public doctorsCount = 0;

    mapping(address=>Doctor) public doctors;

    mapping(address=>bool) public added;

    mapping(uint=>address) public index;

    function addDoctors(string memory name, uint age, string memory field) public {

        if(added[msg.sender]!=true) {

            doctors[msg.sender] = Doctor(name,age,field,0);

            added[msg.sender] = true;

            index[++doctorsCount] = msg.sender;

        }

    }

   function getDoctorsCount() public view returns (uint) {

        return doctorsCount;

    }

    function getDoctors(uint pos) public view returns (string memory) {

        return doctors[index[pos]].name;

    }

    function getMyPatientsCount() public view returns (uint) {

        return doctors[msg.sender].myPatientsCount;

    }

    function getMyPatients(uint pos) public view returns (Patient memory) {

        return doctors[msg.sender].myPatients[pos];

    }

    function addPatients(string memory name, uint age, string memory disease,string memory des,string memory Hash) public {

        doctors[msg.sender].myPatients[++doctors[msg.sender].myPatientsCount] = Patient(msg.sender,name,age,disease,des,Hash);

    }

    function sendDetails(string memory name, uint age, string memory disease, string memory des, string memory Hash, uint pos) public {

        address docAddress = index[pos];

        if(added[docAddress]==true) {

            doctors[docAddress].myPatients[++doctors[docAddress].myPatientsCount] = Patient(docAddress, name, age, disease, des, Hash);

        }

    }

    function check() public view returns (bool) {

        if(added[msg.sender]==true){

            return true;

        }

        return false;

    }

}

**CONNECTION WITH BLOCKCHAIN:**

  async loadBlockchainData() {

    const web3 = window.web3;

    const accounts = await web3.eth.getAccounts()

    this.setState({

      account: accounts[0]

    })

    const networkId = await web3.eth.net.getId()

    const networkData = HealthCare.networks[networkId]

    if(networkData){

      const abi = HealthCare.abi

      const address = networkData.address

      const contract = new web3.eth.Contract(abi, address)

      this.setState({

        contract: contract

      })

      console.log(this.state.contract);

    }else{

      window.alert("Smart contract not deployed to detected network")

    }

    console.log(networkId)

**ROUTES FILE:**

import React from 'react';

import  {BrowserRouter, Switch, Route} from 'react-router-dom';

import Home from './core/Home'

import Dashboard from './user/UserDashboard';

import AdminRoute from './auth/AdminRoute'

import PatientDashboard from './user/AdminDashboard';

import Signin from './auth/index'

import CreatePatient from './Patient'

const Routes = ()=>{

    return (

        <BrowserRouter>

        <Switch>

        <Route path="/" exact component={Home}/>

        <Route path="/signin" exact component={Signin}/>

        <Route path="/user/dashboardhere" exact component={Dashboard} />

            <AdminRoute path="/user/dashboard" exact component={AdminRoute} />

            <Route path="/patient/dashboard" exact component={PatientDashboard}/>

            <Route path="/create/patient" exact component={CreatePatient}/>

        </Switch>

        </BrowserRouter>

        );

};

export default Routes;

**IMPLEMENTATION SCREENSHOTS:**

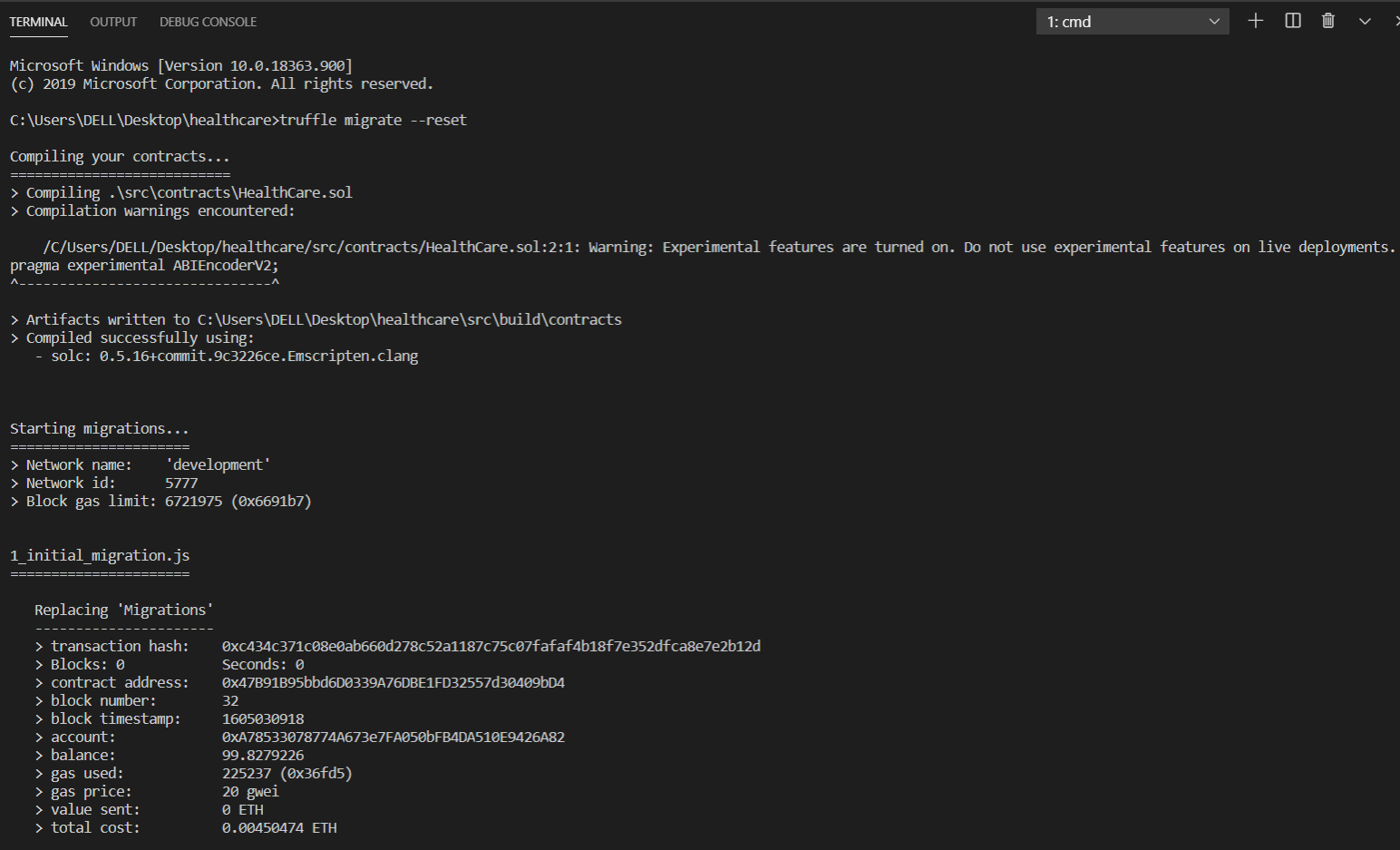
****

Figure 1 Deploying contract

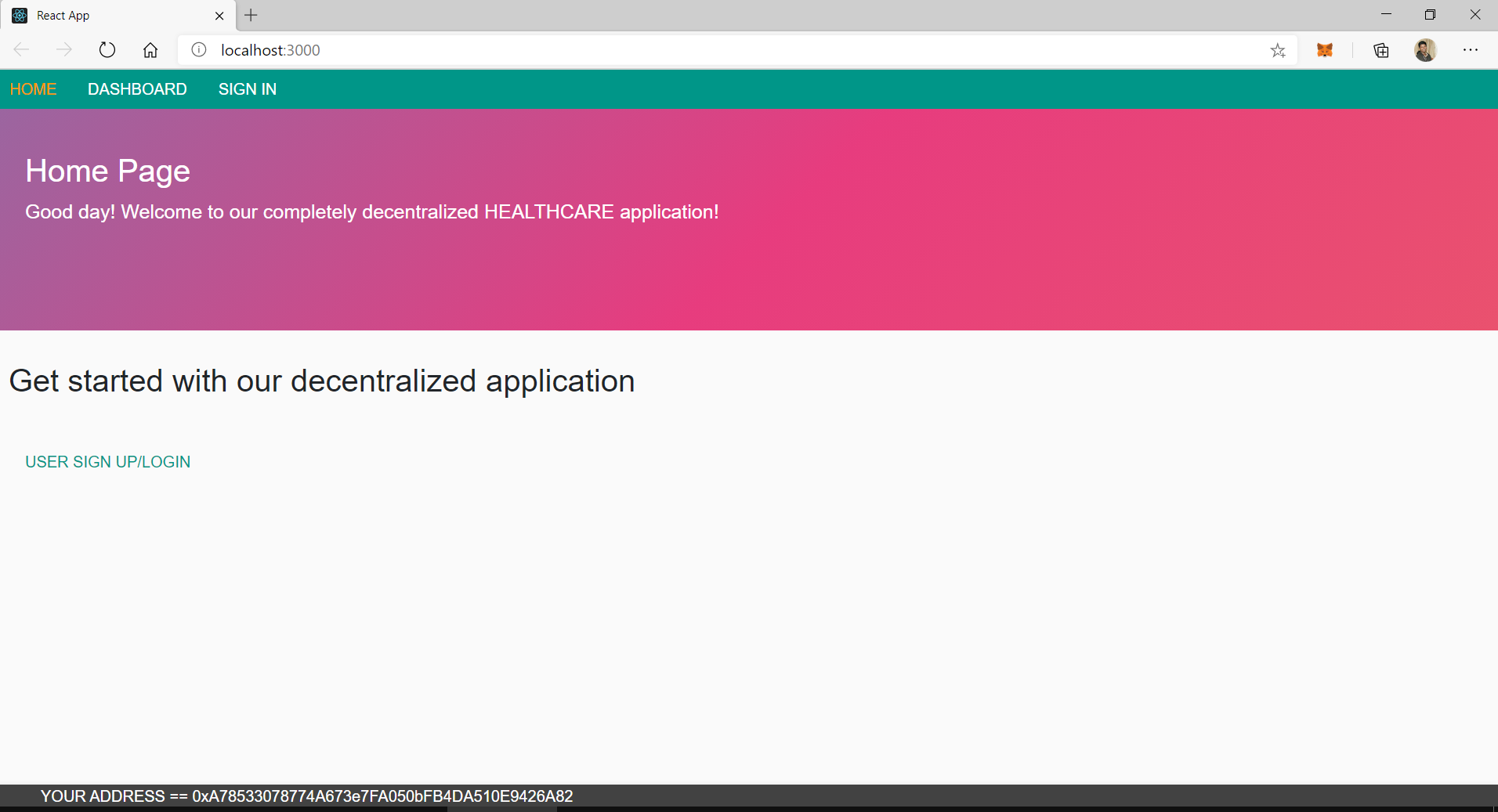


Figure 2 Home page

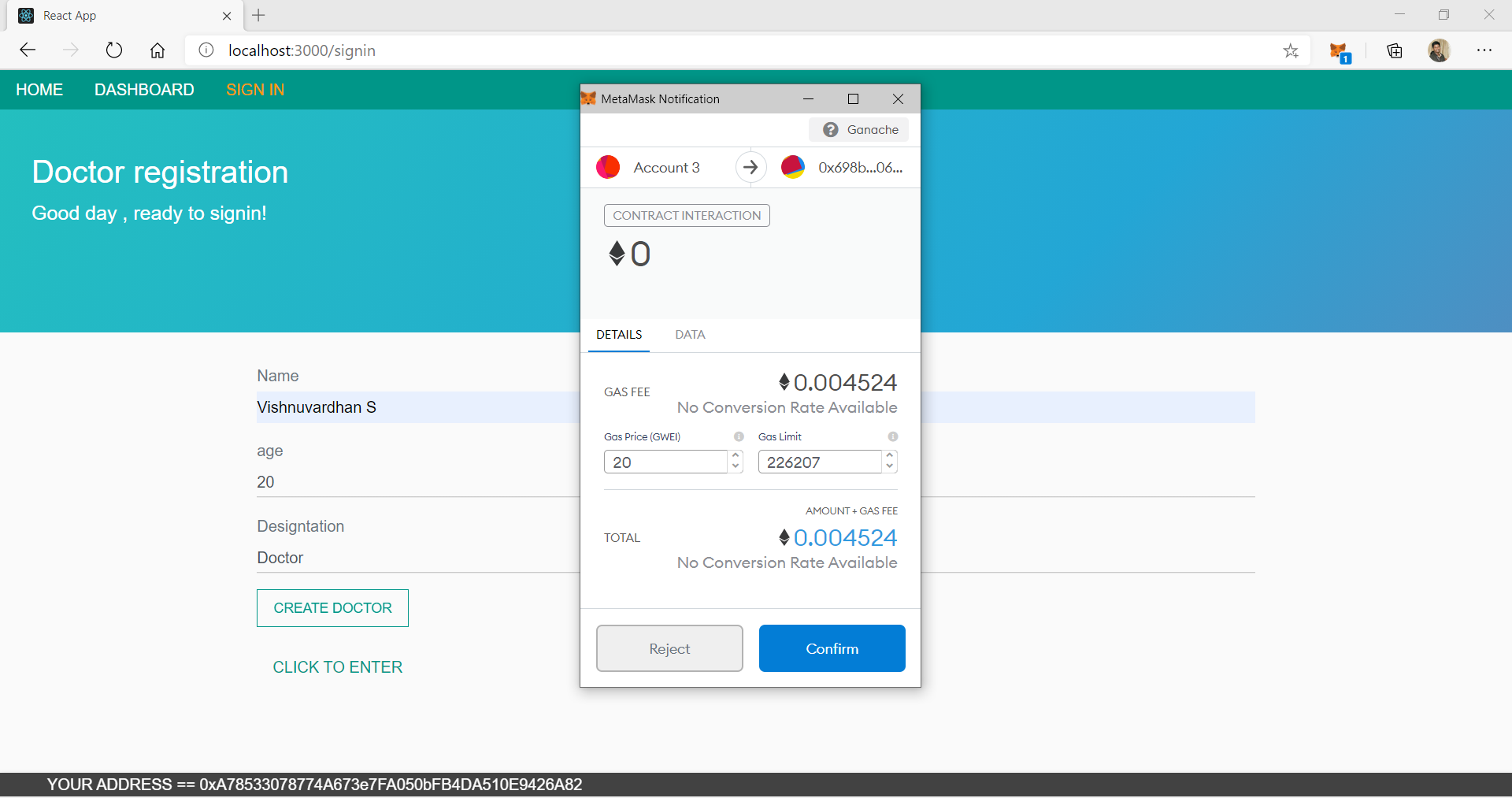


Figure 3 Adding a doctor

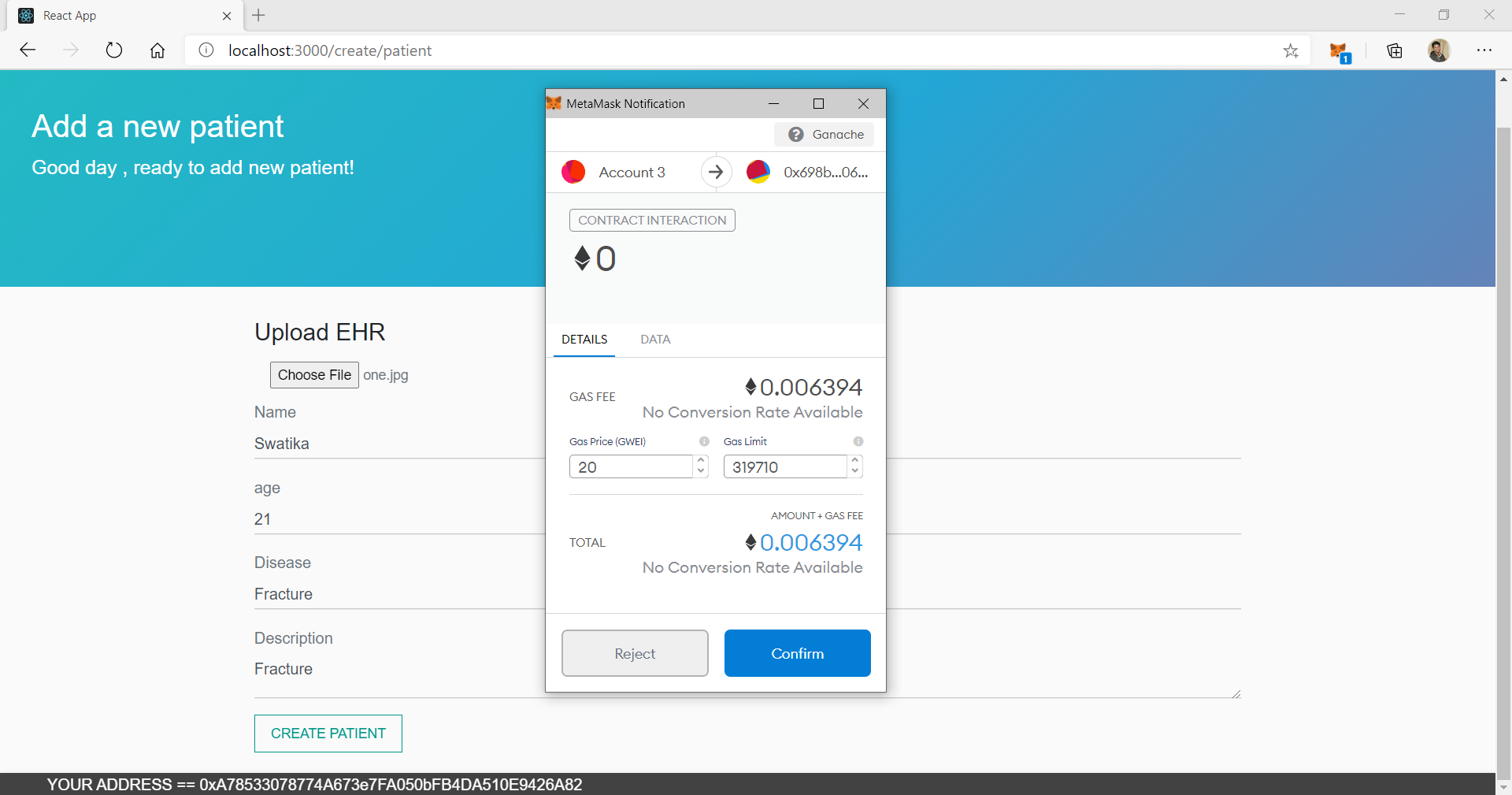


Figure 4 Inserting a patient

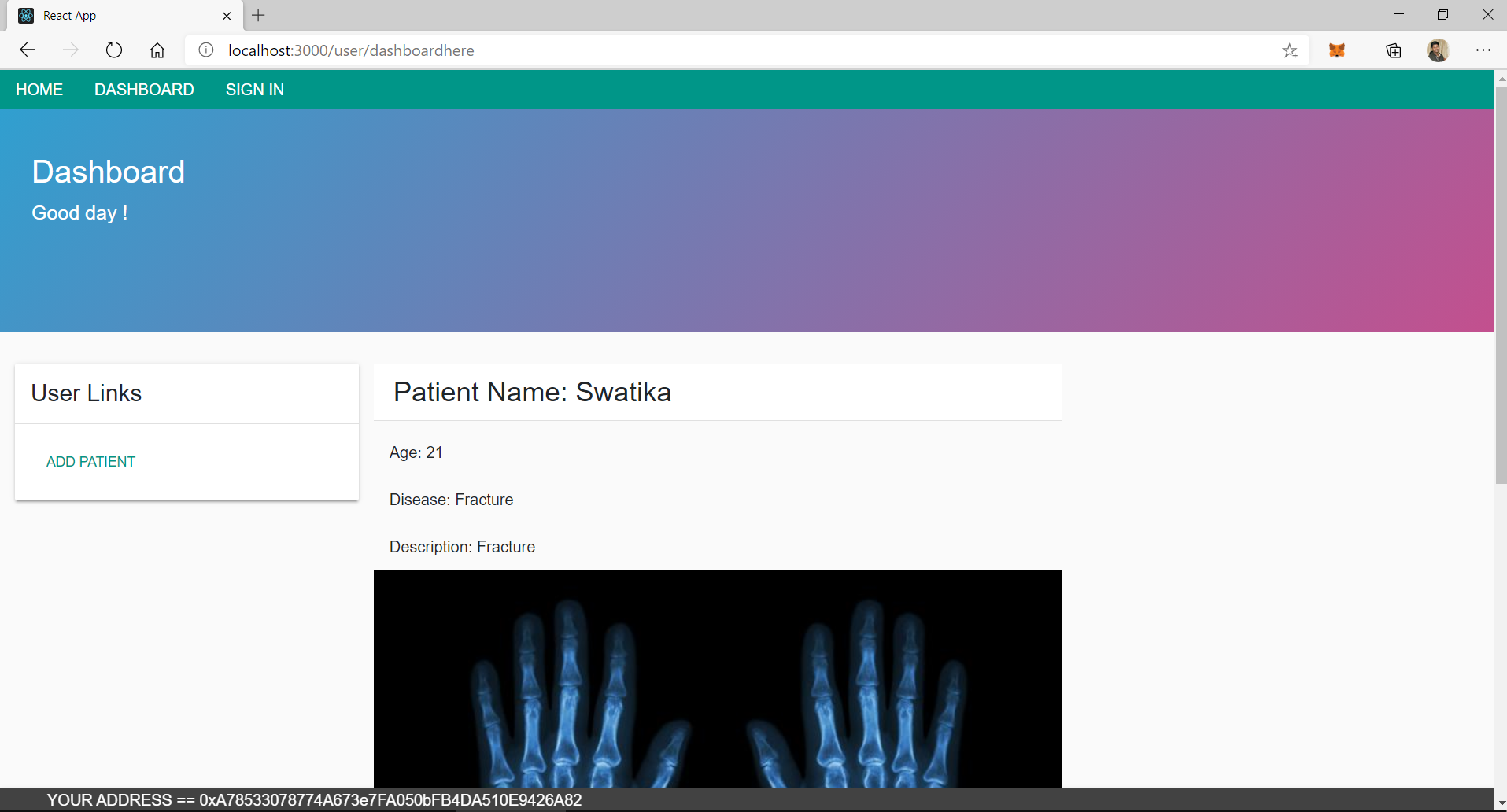


Figure 5 Dashboard of the doctor

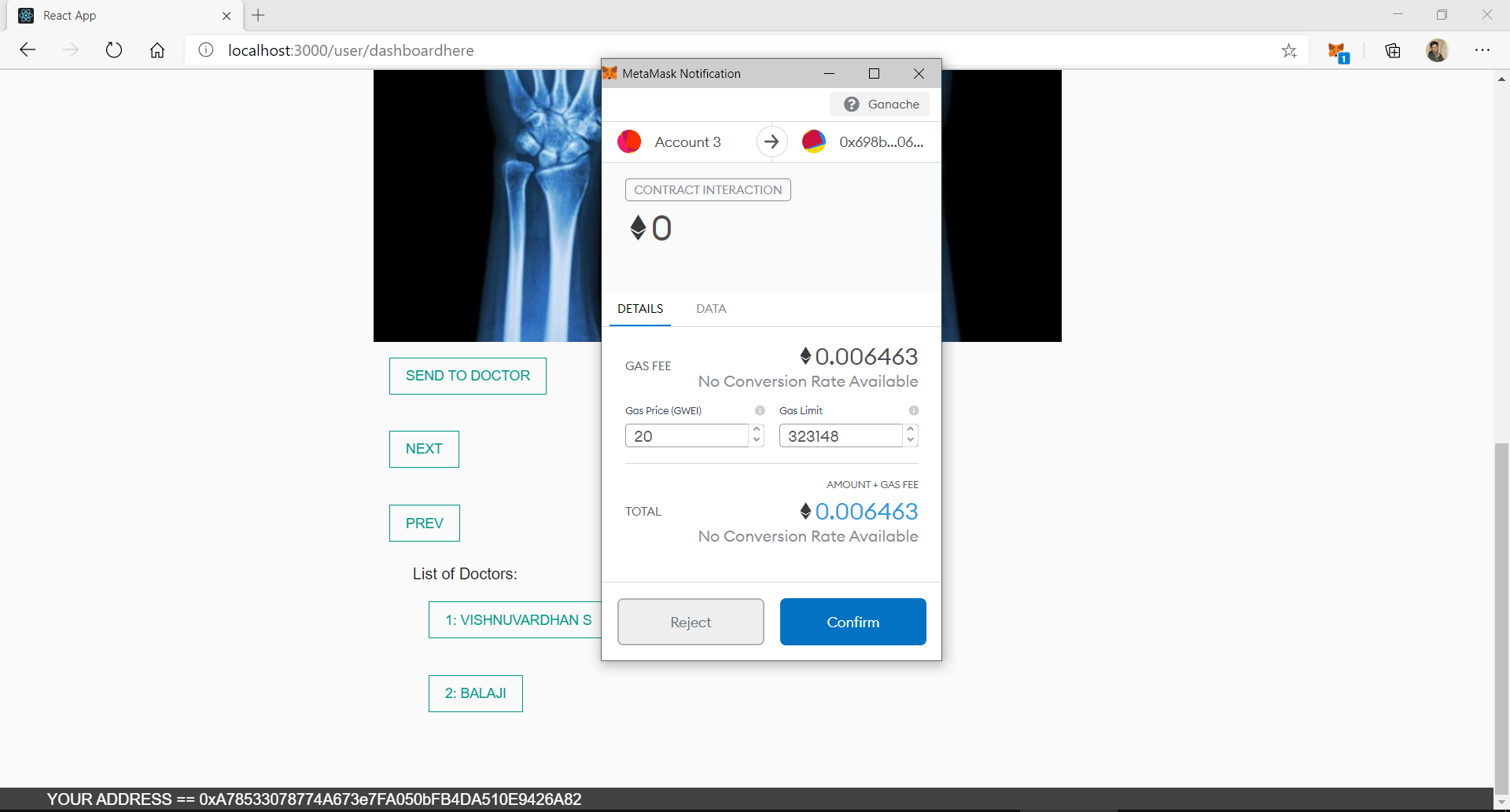


Figure 6 Sending EHR to another doctor

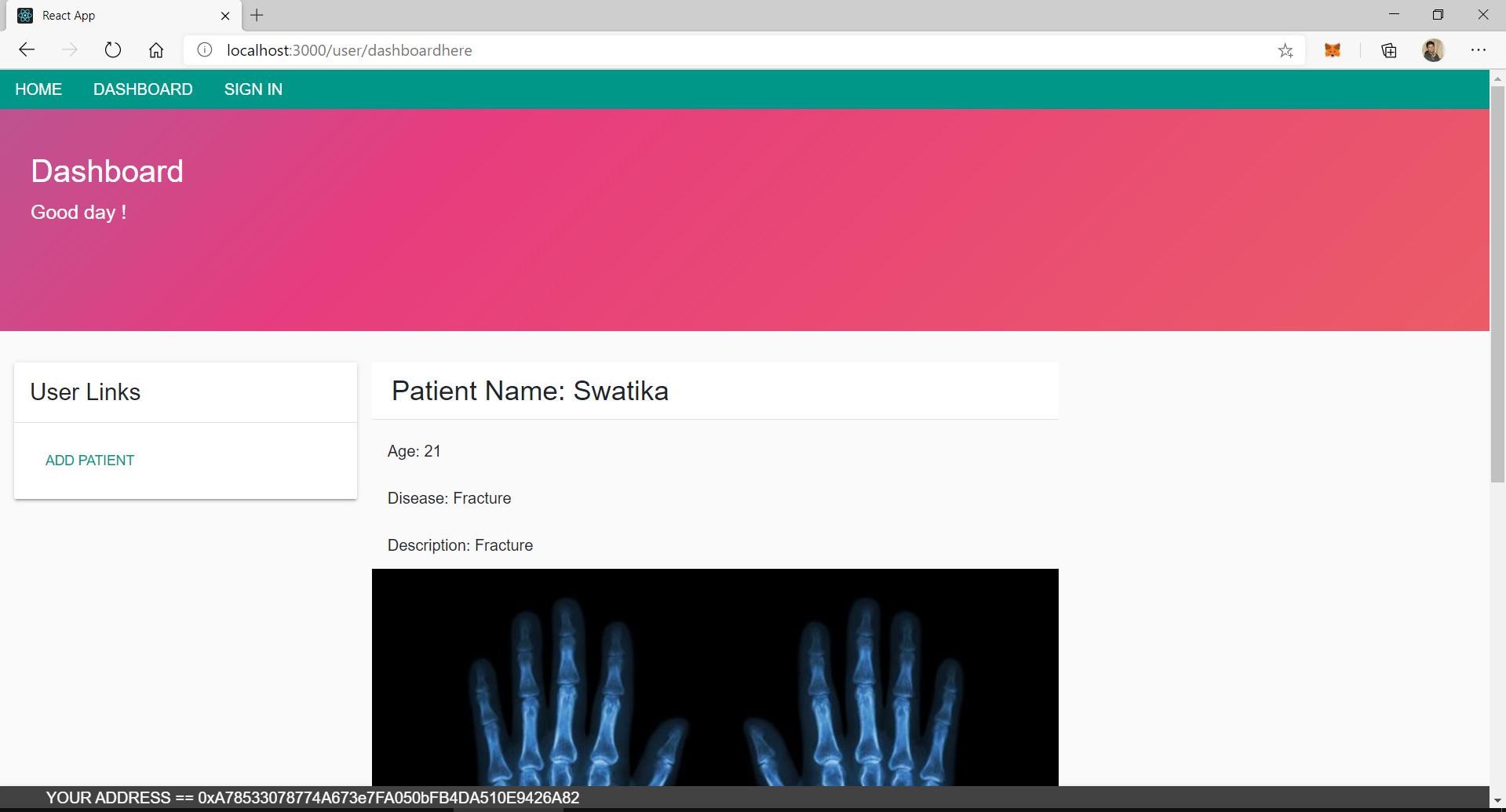


Figure 7 The EHR is displayed in the other doctor's account

**RESULT:**

The application on EHR sharing is implemented successfully using Blockchain.