Report for hospital backend using node.js

***Index.js page***

Import dependence

* Code for import modules in index.js page

var express=require('express');

var app=express();

const mysql=require("mysql");

const nodemailer = require('nodemailer');

const session = require('express-session');

Create connection to database (mysql)

* Code for connecting
* const connection=mysql.createConnection({
* host:'localhost',
* user:'root',
* password:'',
* database:'hospital'
* });
* connection.connect((error)=>{
* if(error){
* console.error(" error database conntecting ");
* return;
* }
* console.log("connected"+connection.threadId);
* });

Create session to store datas for temporary

* Code for creating session
* app.use(session({
* secret: 'c3312a015664ea80c6e023345559fc687c384c8c1112df52cf0fa7f673953fa8',          // Secret key used to sign the session ID cookie
* resave: false,                     // Do not save session if unmodified
* saveUninitialized: false,          // Do not create a session until something is stored
* // cookie: { maxAge: 120000 }          // Session expiration time
* }));

For registration with otp verification with Email

* Code for registration
* app.post('/register',(req,res)=>{
* const { name,phone,Username,password,email} = req.body;
* connection.query('SELECT \* FROM signin where Username=?',[Username],(err,results)=>{
* if(results.length===0){
* const OTP = Math.floor(100000 + Math.random() \* 900000);
* console.log(OTP);
* var transport = nodemailer.createTransport({
* service: 'gmail',
* auth:{
* user: 'make13058@gmail.com',
* pass: 'keaj awno ztep vcmu'
* }
* });
* message = {
* from: "make13058@gmail.com",
* to: email,
* subject: "OTP of your Registration",
* text: `Dear ${name},Your OTP is ${OTP}`
* }
* transport.sendMail(message, function(err, info) {
* if (err) {
* res.json({message:"Not able to send OTP"});
* }
* else {
* req.session.user = { name: name, phone:phone,Username:Username,password:password,email:email};
* connection.query('INSERT INTO otp(OTP,Email)  VALUES (?,?)',[OTP,email],(error, results) => {
* if (error) {
* console.error("Error posting");
* }});
* console.log("send otp")
* res.json({message:"Check your Email for OTP"});
* }
* });
* }
* else{
* res.send('The username is already Exist')
* }
* })
* });

Api for otp validation

app.post('/otp',(req,res)=>

{

    const {otp} = req.body;

    // const otps=req.session.user.OTP

    const name=req.session.user.name

    const phone=req.session.user.phone

    const Username=req.session.user.Username

    const password=req.session.user.password

    const email=req.session.user.email

    const otps=connection.query('select OTP from otp WHERE Email=?',[email],(error,results)=>{

    const Otp=results[0].OTP;

    console.log(Otp)

    if (otp==Otp){

        console.log(Otp)

        connection.query('INSERT INTO signin(Name,Phone,Username,Email,Password) VALUES (?,?,?,?,?)', [name,phone,Username,email,password], (error, results) => {

            if (error) {

              console.error("Error posting");

            //   return res.status(500).send({ error: "Error posting data" });

            }

            connection.query('DELETE FROM otp WHERE Email=?',[email])

            res.send(`Welcome ${name},happy to join us`);

          });

    }

});});

Api for login

app.post('/login',(req,res)=>{

    if(req.session.login){

        res.send("you are already login")

    }

    else{

    const {user,pass}=req.body

    connection.query('SELECT \* FROM signin WHERE  Username=? AND Password=?',[user,pass],(err,results)=>{

       if(err){

        res.send("Username or Password is wrong,Try again");

       }

       else{

       if(results.length === 0){

           console.log('wrong');

           res.send("Username or Password is wrong,Try again");

       }

       console.log(results)

       req.session.login={user:user,pass:pass}

       res.send("Login successfully")

      }

    });

}

});

Api for logout

app.post('/logout', (req, res) => {

    req.session.destroy(err => {

      if (err) {

        return res.status(500).send('Failed to log out');

      }

      res.send('Logged out successfully');

    });

  });

Api for Booking

app.post('/booking', (req, res) => {

  const { name, phone, Doctor,Age,Place,gender } = req.body;

  connection.query('SELECT Phone FROM bookings Where Phone=?',[phone],(err,results)=>{

    if (results.length!==0){

      res.send('Your already Appointed')

      console.log(results.length!==0)

      console.log('Your already Appointed')

    }

    else{

  // console.log(results)

     connection.query('SELECT MAX(Token) as Token FROM bookings', (err, results) => {

      if (err) {

          console.error("Error retrieving max token:", err);

          return res.status(500).send('Internal Server Error');

      }

      let Token = 100; // Default token if no bookings exist

      if (results[0].Token) {

          Token = results[0].Token + 1; // Increment token if there are bookings

      }

      console.log(`Assigned Token: ${Token}`);

      connection.query(

          'INSERT INTO bookings (Name, Phone, DoctorName,Age,Place,gender, Token) VALUES (?,?,?,?,?,?,?)',

          [name, phone, Doctor,Age,Place,gender ,Token],

          (err, results) => {

              if (err) {

                  console.error("Booking failed:", err);

                  return res.status(500).send('Booking failed');

              }

              res.send(`Your booking is confirmed. Your token number is: ${Token}`);

          }

      );

  });

}

});

});

Api for deleting booking

app.delete ('/deletebooking/:id',(req,res)=>{

  console.log(req.params)

  const {id}=req.params;

   console.log(id)

  connection.query('DELETE FROM bookings WHERE id=?',[id],(err,results)=>{

    if(err){

      console.log('error fetching')

      res.send('error mysql')

    }

    else{

      if(results.length===0){

        res.send("booking not able to delete")

      }

      else{

        res.send("delete successfully")

      }

    }

  });

});

Connecting other pages to index page

* Admin page
* const admin=require('./admin.js');
* app.use('/admin',admin);
* Reception page
* const reception=require('./reception.js');
* app.use('/reception',reception);
* Doctor page

* const doctor=require('./doctor.js');
* app.use('/doctor',doctor);

doctor.js page

Import dependence

const express=require('express');

const doctor=express.Router();

const mysql=require('mysql');

Exports doctor page into index page

module.exports=doctor

modules for generating pdf

const fs = require('fs');

const puppeteer = require('puppeteer');

const path = require('path');

Api for display booking

doctor.get('/booking',(req,res)=>{

    if(!req.session.login){

        res.send('session not available')

    }

     const {user}=req.session.login.name

     connection.query('Select \*from bookings Where DoctorName=?',[user],(err,results)=>{

        if(err){

            console.log('Error fetching')

        }

        else{

            res.send(results)

        }

     })

});

Api for delete booking

doctor.delete ('/deletebooking/:id',(req,res)=>{

    console.log(req.params)

    const {id}=req.params;

     console.log(id)

    connection.query('DELETE FROM bookings WHERE id=?',[id],(err,results)=>{

      if(err){

        console.log('error fetching')

        res.send('error mysql')

      }

      else{

        if(results.length===0){

          res.send("booking not able to delete")

        }

        else{

          res.send("delete successfully")

        }

      }

    });

  });

Api for print or download Prescription page

 reception.get('/peric2/:id', async (req, res) => {

    const { id } = req.params;

    console.log(id);

    connection.query('SELECT \* FROM bookings WHERE id = ?', [id], async (err, results) => {

        if (err) {

            console.error('Error with MySQL:', err);

            return res.status(500).send('Database error.');

        }

        if (results.length === 0) {

            console.warn('No booking found', results);

            return res.status(404).send('No booking found.');

        }

        const { id: ids, Name: name, Age, Place: place, DoctorName: doctorName, gender } = results[0];

        console.log(name, Age, place, gender, doctorName);

        try {

            const pdfPath = path.join(\_\_dirname, 'report',  `prescription${ids}.pdf`);

            // Generate PDF

            async function generatePrescription() {

                // Load HTML template

                let html = fs.readFileSync(path.join(\_\_dirname, 'template.html'), 'utf8');

                const absoluteLogoPath = path.join(\_\_dirname, 'logo', 'logo.jpg');

                const encodedLogoPath = `file://${absoluteLogoPath.replace(/\\/g, '/')}`;

                // Replace placeholders with actual data

                const date = new Date().toLocaleDateString();

                html = html

                    .replace('{{patientName}}', name)

                    .replace('{{Age}}', Age)

                    .replace('{{date}}', date)

                    .replace('{{place}}', place)

                    .replace('{{gender}}', gender)

                    .replace('{{doctor}}', doctorName)

                    .replace('{{doctors}}', doctorName)

                    .replace('{{logo}}',encodedLogoPath)

                // Generate PDF with Puppeteer

                const browser = await puppeteer.launch();

                const page = await browser.newPage();

                await page.setContent(html);

                await page.pdf({ path: pdfPath, format: 'A4' });

                await browser.close();

                console.log('Prescription PDF generated:', pdfPath);

                console.log('Absolute logo path:', absoluteLogoPath);

                console.log('Absolute logo path:',encodedLogoPath );

            }

            // Generate the PDF

            await generatePrescription();

            // Send the response

            res.json({

                message: 'Prescription generated successfully',

                downloadUrl: `/doctor/prescriptions/prescription${ids}.pdf`, // Dynamic URL

                printUrl: `/doctor/prescriptions/prescription${ids}.pdf`,

            });

        } catch (error) {

            console.error('Error generating prescription:', error);

            res.status(500).send('Error generating the prescription.');

        }

    });

  });

Reception.js page

Import dependence

var express=require('express')

const reception=express.Router();

const mysql=require('mysql');

reception.use(express.json());

reception.use(express.urlencoded({ extended: true }));

modules for generating pdf

const fs = require('fs');

const puppeteer = require('puppeteer');

const path = require('path');

Exports doctor page into index page

module.exports=reception

connection to database (mysql)

const connection=mysql.createConnection({

    host:'localhost',

    user:'root',

    password:'',

    database:'hospital'

});

connection.connect((error)=>{

    if(error){

        console.log("recep");

        console.error(" error database conntecting ");

        return;

    }

    console.log("connected"+connection.threadId); });

Api for booking

reception.post('/booking', (req, res) => {

    const { name, phone, Doctor,Age,Place,gender } = req.body;

    connection.query('SELECT Phone FROM bookings Where Phone=?',[phone],(err,results)=>{

      if (results.length!==0){

        res.send('Your already Appointed')

        console.log(results.length!==0)

        console.log('Your already Appointed')

      }

      else{

    // console.log(results)

       connection.query('SELECT MAX(Token) as Token FROM bookings', (err, results) => {

        if (err) {

            console.error("Error retrieving max token:", err);

            return res.status(500).send('Internal Server Error');

        }

        let Token = 100; // Default token if no bookings exist

        if (results[0].Token) {

            Token = results[0].Token + 1; // Increment token if there are bookings

        }

        console.log(`Assigned Token: ${Token}`);

        connection.query(

            'INSERT INTO bookings (Name, Phone, DoctorName,Age,Place,gender, Token) VALUES (?,?,?,?,?,?,?)',

            [name, phone, Doctor,Age,Place,gender ,Token],

            (err, results) => {

                if (err) {

                    console.error("Booking failed:", err);

                    return res.status(500).send('Booking failed');

                }

                res.send(`Your booking is confirmed. Your token number is: ${Token}`);

            }

        );

    });

  }

  });

  });

Api for delete booking

  reception.delete ('/deletebooking/:id',(req,res)=>{

    console.log(req.params)

    const {id}=req.params;

     console.log(id)

    connection.query('DELETE FROM bookings WHERE id=?',[id],(err,results)=>{

      if(err){

        console.log('error fetching')

        res.send('error mysql')

      }

      else{

        if(results.length===0){

          res.send("booking not able to delete")

        }

        else{

          res.send("delete successfully")

        }

      }

    });

  });

Api for display doctors

reception.get('/displaydoc',(req,res)=>{

    connection.query('select \*from doctor',(err,results)=>{

        if(err){

            res.send("error displaying")

        }

        res.send(results);

    })

});

Api for print or download prescription

reception.get('/peric2/:id', async (req, res) => {

  const { id } = req.params;

  console.log(id);

  connection.query('SELECT \* FROM bookings WHERE id = ?', [id], async (err, results) => {

      if (err) {

          console.error('Error with MySQL:', err);

          return res.status(500).send('Database error.');

      }

      if (results.length === 0) {

          console.warn('No booking found', results);

          return res.status(404).send('No booking found.');

      }

      const { id: ids, Name: name, Age, Place: place, DoctorName: doctorName, gender } = results[0];

      console.log(name, Age, place, gender, doctorName);

      try {

          const pdfPath = path.join(\_\_dirname, 'report',  `prescription${ids}.pdf`);

          // Generate PDF

          async function generatePrescription() {

              // Load HTML template

              let html = fs.readFileSync(path.join(\_\_dirname, 'template.html'), 'utf8');

              const absoluteLogoPath = path.join(\_\_dirname, 'logo', 'logo.jpg');

              const encodedLogoPath = `file://${absoluteLogoPath.replace(/\\/g, '/')}`;

              // Replace placeholders with actual data

              const date = new Date().toLocaleDateString();

              html = html

                  .replace('{{patientName}}', name)

                  .replace('{{Age}}', Age)

                  .replace('{{date}}', date)

                  .replace('{{place}}', place)

                  .replace('{{gender}}', gender)

                  .replace('{{doctor}}', doctorName)

                  .replace('{{doctors}}', doctorName)

                  .replace('{{logo}}',encodedLogoPath)

              // Generate PDF with Puppeteer

              const browser = await puppeteer.launch();

              const page = await browser.newPage();

              await page.setContent(html);

              await page.pdf({ path: pdfPath, format: 'A4' });

              await browser.close();

              console.log('Prescription PDF generated:', pdfPath);

              console.log('Absolute logo path:', absoluteLogoPath);

              console.log('Absolute logo path:',encodedLogoPath );

          }

          // Generate the PDF

          await generatePrescription();

          // Send the response

          res.json({

              message: 'Prescription generated successfully',

              downloadUrl: `/reception/prescriptions/prescription${ids}.pdf`, // Dynamic URL

              printUrl: `/reception/prescriptions/prescription${ids}.pdf`,

          });

      } catch (error) {

          console.error('Error generating prescription:', error);

          res.status(500).send('Error generating the prescription.');

      }

  });

});

Admin.js Page

Import dependence

var express=require('express');

const mysql=require('mysql');

var admin=express.Router();

admin.use(express.json());

admin.use(express.urlencoded({ extended: true }));

Exports doctor page into index page

*module.exports=admin*

connection to database

const connection=mysql.createConnection({

    host:'localhost',

    user:'root',

    password:'',

    database:'hospital'

});

connection.connect((error)=>{

    if(error){

        console.error(" error database conntecting ");

        return;

    }

    console.log("connected"+connection.threadId);

});

Api for Doctor registration

admin.post('/Doctorreg',(req,res)=>

{

    const {name,Phone,Email,Catagory,Qualification,Address,Dateofbirth,Username,password}=req.body;

    console.log(name,Phone,Email,Catagory,Qualification,Address,Dateofbirth);

    connection.query('select \* from signin WHERE Username=?',[Username],(err,results)=>{

        if(results.length===0){

            connection.query('INSERT INTO signin(Name,Phone,Email,Password,Username) VAlUES(?,?,?,?,?) ',[name,Phone,Email,password,Username])

            connection.query('INSERT INTO doctor(Name,Phone,Email,Catagory,Qualification,Address,Dateofbirth,Username,Password) VALUES(?,?,?,?,?,?,?,?,?)',[name,Phone,Email,Catagory,Qualification,Address,Dateofbirth,Username,password],(err,results)=>{

                if(err){

                  res.send("error registration")

                }

                res.send(`register success :${results}`)

              });

        }

        else{

            res.send('The username is already register')

        }

    })

});

Api for display all doctors

admin.get('/displaydoc',(req,res)=>{

    connection.query('select \*from doctor',(err,results)=>{

        if(err){

            res.send("error displaying")

        }

        res.send(results);

    })

});

Api for display all booking

admin.get('/displaybooking',(req,res)=>{

    connection.query('select \*from bookings',(err,results)=>{

        if(err){

            console.log('error fetching');

        }

        else{

        if(results.length==0){

           console.log(results.length==0)

           res.send('No Booking found')

        }

        else{

        console.log(results.length)

        res.send(results);

        }

    }

    });

});

Api foe update datas of doctors

admin.put('/updatedata/:id',(req,res)=>{

    const {name,Phone,Email,Catagory,Qualification,Address,Dateofbirth}=req.body;

    const userid=req.params.id

    connection.query('UPDATE doctor SET Name=? ,Phone=?,Email=?,Catagory=?,Qualification=?,Address=?,Dateofbirth=?) WHERE id=?',[name,Phone,Email,Catagory,Qualification,Address,Dateofbirth,userid],(err,results)=>{

        if(err){

            res.send('error mysql')

        }

        if(results){

            res.send('update successfully ')

        }

    })

});

Api for delete doctors

admin.delete('/delete/:id',(req,res)=>{

    const userid=req.params.id

    connection.query('DELETE doctor WHERE id=?',[userid],(err,results)=>{

        if(err){

            res.send('error')

        }

        if(results.length===0){

            res.send('No doctor found')

        }

        else{

        res.send('Delete doctor successfully')

        }

    })

});

Template.html

Add an html template for prescriptions(template.html)

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

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

    <title>Doctor's Prescription</title>

    <style>

        body {

            font-family: Arial, sans-serif;

            margin: 20px;

            line-height: 1.6;

        }

        .header {

            display: flex;

            justify-content: space-between;

            align-items: center;

            border-bottom: 2px solid #000;

            padding-bottom: 10px;

            margin-bottom: 20px;

        }

        .header .doctor-info {

            font-size: 14px;

        }

        .header .clinic-info {

            text-align: right;

            font-size: 14px;

        }

        .header .logo {

            text-align: center;

        }

        .content {

            margin-top: 20px;

        }

        .content .section {

            margin-bottom: 20px;

        }

        .content .section h3 {

            margin-bottom: 10px;

            text-decoration: underline;

            font-size: 16px;

        }

        .prescription-table {

            width: 100%;

            border-collapse: collapse;

        }

        .prescription-table th, .prescription-table td {

            border: 1px solid #000;

            padding: 8px;

            text-align: left;

            font-size: 12px;

        }

        .prescription-table th {

            background-color: #f2f2f2;

        }

        .advice {

            font-size: 12px;

            margin-top: 20px;

        }

        .footer {

            margin-top: 50px;

            text-align: right;

            font-size: 12px;

        }

    </style>

</head>

<body>

    <!-- Header Section -->

    <div class="header">

        <div class="doctor-info">

            <strong>Dr. {{doctor}}</strong><br>

            MBBS <br>

        </div>

        <div class="logo">

            <img src="{{logo}}" alt="" style="height: 50px;">

        </div>

        <div class="clinic-info">

            <strong>Landmark Polyclinic</strong><br>

            12/4, Paud Road, Kothrud, Pune - 411023<br>

            Ph: 02025226425, Timing: 09:00 AM - 02:00 PM,<br>

            05:00 PM - 09:00 PM | Closed: Monday

        </div>

    </div>

    <!-- Patient and Diagnosis Details -->

    <div class="content">

        <div class="section">

            <strong>Name:</strong> {{patientName}}<br>

            <strong>Address:</strong> {{place}}<br>

            <strong>Gender:</strong> {{gender}}<br>

            <strong>Age:</strong> {{Age}}<br>

            <strong>Date:</strong> {{date}}<br>

            <!-- <strong>Referred By:</strong> Dr. Avidip<br> -->

        </div>

        <!-- Footer -->

        <div class="footer">

            <!-- <strong>Next Visit:</strong> 13-08-2019<br> -->

            <p>\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_</p>

            <p>Dr.{{doctors}}</p>

            <p>MBBS</p>

        </div>

    </div>

</body>

</html>