server\controllers\controller.js

```
const bcrypt = require('bcrypt')
const fs = require('fs')
const Auth = require('../models/auth')
const Profile = require('../models/profile')
const Train = require('../models/train')
const City = require('../models/city')
const Log = require('../models/logs')

exports.postSignup = async( req,res ) => {

    if(req.session.auth)
    {
        res.redirect('/')
    }
    else
    {
        const { email , password , cpassword } = req.body

   if ( email == '' || password == '' || cpassword == '')
    {
        req.session.message = {
            type : 'warning',
            message : 'WARNING !',
            desc : 'input fields cannot be empty !'
        }
        res.redirect('/signup')
    }
    else
    {
     if( password != cpassword)
     {
        req.session.message = {
            type : 'warning',
            message : 'WARNING !',
            desc : 'Password did not match !'
        }
        res.redirect('/signup')
     }
     else
     {
        const isEmail = await Auth.findOne({email:email})

        if( isEmail )
        {
            req.session.message = {
                type : 'info',
                message : 'INFORMATION !',
                desc : 'User already exist !\nPlease Sign In'
            }
            res.redirect('/signin')
        }
        else
```

```
        {
            const hashpass = await bcrypt.hashSync(password,12)

        const regEmail = /@railway.com/i
        const isAdmin = regEmail.test(email)

        if( isAdmin )
        {
            const newUser = new Auth({
                email:email,
                password:hashpass,
                role : 'admin'
            })
            newUser.save().catch( err => {
                console.log(err)
            }).then(
                () => {
                    req.session.message = {
                        type : 'success',
                        message : 'Success !',
                        desc : 'Sign up successfull ! Now Please Sign in'
                    }
                    res.redirect('/signin')
                }
            )
        }
        else
        {
            const newUser = new Auth({
                email:email,
                password:hashpass
            })
            newUser.save().catch( err => {
                console.log(err)
            }).then(
                () => {
                    req.session.message = {
                        type : 'success',
                        message : 'Success !',
                        desc : 'Sign up successfull ! Now Please Sign in'
                    }
                    res.redirect('/signin')
                }
            )
        }
        }


    }
}
    }

}

exports.postSignin = async( req,res ) => {
```

```
        if(req.session.auth)
        {
            res.redirect('/')
        }
        else
        {
            const { email , password } = req.body

        if ( email == '' || password == '')
        {
            req.session.message = {
                type : 'warning',
                message : 'warning !',
                desc : 'input fields cannot be empty'
            }
        }
        else
        {
            const isEmail = await Auth.findOne({email:email})

            if ( !isEmail)
            {
                req.session.message = {
                    type : 'info',
                    message : 'Info !',
                    desc : 'Email does not exist ! Please signup first'
                }
                res.redirect('/signup')
            }
            else
            {
                const isMatch = await bcrypt.compareSync(password,isEmail.password)

                if( isMatch )
                {
                    const isProfile = await Profile.findOne({_id:isEmail._id})

                    if(isProfile)
                    {
                        req.session.auth = {
                            _id : isEmail._id,
                            role : isEmail.role,
                            email : isEmail.email
                        }
                        req.session.message = {
                            type : 'success',
                            message : 'Logged In !',
                            desc : 'logged in successfully !'
                        }
                        res.redirect('/')
                    }
                    else
                    {
                        req.session.auth = {
                        _id : isEmail._id,
                        role : isEmail.role,
```

```
                    email : isEmail.email
                    }
                    req.session.message = {
                        type : 'success',
                        message : 'Logged In !',
                        desc : 'logged in successfully !'
                    }
                res.redirect('/profile')


                }

            }
            else
            {
                req.session.message = {
                    type : 'warning',
                    message : 'warning !',
                    desc : 'Password entered was wrong'
                }
                res.redirect('/signin')
            }
        }
    }
    }

}

exports.postProfile = async( req,res ) => {
    if(!req.session.auth)
    {
        res.redirect('/signin')
    }
    else
    {
        const { _id ,role,email ,username , gender , profession} = req.body
    const image = req.file.filename

        const newProfile = new Profile({
            _id : _id,
            role : role,
            email : email,
            username : username,
            gender : gender,
            profession : profession,
            image : image
        })

        newProfile.save().then(
            req.session.message = {
                type : 'success',
                message : 'success !',
                desc : 'profile updated successfully'
            }
            )
```

```
        res.redirect('/')
        }
    }

    exports.postupdateprofile = async( req,res) => {
        if(!req.session.auth)
        {
            res.redirect('/signin')
        }
        else
        {
            const id = req.params.id
        const { username,gender,profession} = req.body
        let new_image = ''

        if(req.file)
        {
            new_image = req.file.filename
            try {
                fs.unlinkSync('uploads/'+req.body.old_image)
            } catch (error) {
                console.log(error)
            }
        }
        else
        {
            new_image = req.body.old_image
        }

        Profile.findByIdAndUpdate(id,{
            username : username,
            gender : gender,
            profession : profession,
            image : new_image
        },(err,result)=>{
            if(err)
            {
                console.log(err)
            }
            else{
                req.session.message = {
                    type : 'success',
                    message : 'success !',
                    desc : 'profile updated successfully'
                }
                res.redirect('/')
            }
        })
        }


    }

    exports.postaddtrain = async( req,res )=> {
        if(!req.session.auth)
        {
```

```
            res.redirect('/signin')
        }
        else
        {
            const { trainname , source , destination , starttrain , endtrain ,ticket, traintime } =
    req.body

        if(!req.body)
        {
            req.session.message = ({
                type:'warning',
                message : 'WARNING !',
                desc:'Input Fields cannot be Empty !'
            })
        }
        else
        {
            if( source == destination)
            {
                req.session.message = ({
                    type:'warning',
                    message : 'WARNING !',
                    desc:'Source and destination cannot be same !'
                })
                res.redirect('/addtrain')
            }
            else
            {
                const newTrain = new Train({
                    name : trainname,
                    source : source,
                    destination : destination,
                    startDate : starttrain,
                    endDate : endtrain,
                    time : traintime,
                    ticket:ticket
                })

                newTrain.save().then(
                    req.session.message = ({
                        type:'success',
                        message : 'Success !',
                        desc:'train added successfully'
                    })
                )
                res.redirect('/train')
            }

        }
        }
    }

    exports.postupdatetrain = async( req,res) => {
        if(!req.session.auth)
        {
            res.redirect('/signin')
```

```
    }
    else
    {
        const id = req.params.id
    const { trainname , source , destination , starttrain , endtrain , traintime , ticket} =
req.body

    const date = new Date

    const updatedOn = date.toLocaleString('en-US',{
    day:'2-digit',
    month : 'short',
    hour : '2-digit',
    minute : '2-digit'
})


    Train.findByIdAndUpdate(id,{
            name : trainname,
            source : source,
            destination : destination,
            startDate : starttrain,
            endDate : endtrain,
            time : traintime,
            ticket:ticket,
            updatedOn : updatedOn
    },(err,result)=>{
        if(err)
        {
            console.log(err)
        }
        else{
            req.session.message = {
                type : 'warning',
                message : 'success !',
                desc : 'Train Details updated successfully'
            }
            res.redirect('/train')
        }
    })
    }


}

exports.deletetrain = async( req,res ) => {
    if(!req.session.auth)
    {
        res.redirect('/signin')
    }
    else
    {
        const id = req.params.id
    Train.findByIdAndDelete(id).then(

        req.session.message = ({
```

```
            type:'danger',
            message : 'Success !',
            desc:'train deleted successfully'
        })

    )
    res.redirect('/train')
    }
}

exports.postaddcity = async( req,res ) => {
    if(!req.session.auth)
    {
        res.redirect('/signin')
    }
    else
    {
        const cityname = req.body.cityname

    if(cityname == '')
    {
        req.session.message = {
            type : 'info',
            message : 'INFO !',
            desc : 'Input fields cannot be empty ! Please fill all input fields'
        }
        res.redirect('/city')
    }
    else
    {
        const isCity = await City.findOne({name:cityname})

        if(isCity)
        {
            req.session.message = {
                type : 'warning',
                message : 'WARNING !',
                desc : 'City Already Exist !'
            }
            res.redirect('/city')
        }
        else
        {
            const newCity = new City({
                name : cityname
            })

            newCity.save().then(
                req.session.message = {
                    type : 'success',
                    message : 'SUCCESS !',
                    desc : 'City Added Successfully !'
                }
            )

            res.redirect('/city')
```

```
        }
    }
    }
}

exports.deletecity = async( req,res ) => {
    if(!req.session.auth)
    {
        res.redirect('/signin')
    }
    else
    {
        const id = req.params.id
    City.findByIdAndDelete(id).then(

        req.session.message = ({
            type:'danger',
            message : 'Success !',
            desc:'city deleted successfully'
        })

    )
    res.redirect('/city')
    }
}

exports.getLogout = async( req,res ) => {
        req.session.destroy()
        res.redirect('/')
}

exports.home = async( req,res ) => {
    const { source , destination , date ,searchedby ,role } = req.body

    if( !req.session.auth)
    {
        req.session.message = {
            type : 'info',
            message : 'INFO !',
            desc : 'Please Do Sign In or Sign Up First'
        }
        res.redirect('/signin')
    }
    else
    {
        if(source == '' || destination == '')
    {
        req.session.message = {
            type : 'warning',
            message : 'WARNING !',
            desc : 'Input Fields Cannot be Empty !'
        }
        res.redirect('/')
    }
    else if( source == destination )
    {
```

```
            req.session.message = {
                type : 'warning',
                message : 'WARNING !',
                desc : 'Source and Destination Address cannot be same !'
            }
            res.redirect('/')
        }
        else
        {
                const istrainfound = await Train.find(
                    {
                        source:source,
                        destination:destination,
                        $or:[{startDate:{$lte:date}},{endDate:{$gte:date}}]
                    })

                if(istrainfound != 0)
                {
                    const lsource = source.toLowerCase()
                    const ldestination = destination.toLowerCase()
                    const lsearchedby = searchedby.toLowerCase()

                    const newlogs = new Log({
                        source : lsource,
                        destination : ldestination,
                        result : 'yes',
                        searchdate : date,
                        searchedBy : lsearchedby,
                        role : role
                    })

                    newlogs.save().then(
                        console.log('log saved successfully')
                    )
                    req.session.train = istrainfound
                    res.redirect('/')

                }
                else
                {
                    const lsource = source.toLowerCase()
                    const ldestination = destination.toLowerCase()
                    const lsearchedby = searchedby.toLowerCase()

                    const newlogs = new Log({
                        source : lsource,
                        destination : ldestination,
                        result : 'no',
                        searchdate : date,
                        searchedBy : lsearchedby,
                        role : role
                    })

                    newlogs.save().then(
                        console.log('log saved successfully')
                    )
```

```
                req.session.train = 0
                res.redirect('/')
            }

    }
    }



}

exports.deletelog = async( req,res ) => {
    if(!req.session.auth)
    {
        res.redirect('/signin')
    }
    else
    {
        const id = req.params.id
        Log.findByIdAndDelete(id).then(

        req.session.message = ({
            type:'success',
            message : 'Success !',
            desc:'Log deleted successfully'
        })

    )
        res.redirect('/userlogs')
    }

}
```