**Authorisation @ 17 Jun 2019**

Passport is the equivalent of Devise (rails) and implementing authorisation and security is difficult.

express-session stores username to indicate who is logged in in the session

npm install express-generator

express --view=ejs passport-demo

cd passport-demo folder

npm install

* installs dependencies

npm i mongoose express-session passport passport-local-mongoose

Passport is the authorisation solution. Passport is agnostic to storage methods of passwords and any authenticating methods.

Passport-local-mongoose plugin into passport. Use a local strategy to have username and password stored locally and store it in a mongo database. (i.e. to use facebook authentication then install passport-facebook)

Need to have a model (user) to attach to passport.

Create a folder > models

Inside models> create user.js

const mongoose = require('mongoose')

const passportLocalMongoose = require('passport-local-mongoose') // we don't import passport but the strategy

const UserSchema = new mongoose.Schema({})

const UserModel = mongoose.model('User', UserSchema)

module.exports = {

UserSchema,

UserModel

}

Attach the passport strategy to the user model by adding to user.js

UserSchema.plugin(passportLocalMongoose, {

usernameField: 'email'

})

IN app.js, require in a few extra things

const session = require('express-session')

const passport = require('passport')

const mongoose = require('mongoose')

const {

UserModel,

UserSchema

} = require('./models/user')

after where you created the app in app.js, add the following

passport.use(User.createStrategy())

This creates an instance of the strategy attached to the user model

Passport is agnostic with regards to how it stores the data.

How to store a user and how to retrieve a user (serialize a user)

Serialisation is the process of converting an object into a text object (like JSON.stringify. JSON.parse deserializes back to an object. )

Direclty underneath that, define how the serialisation of the user object

passport.serializeUser(User.serializeUser())

passport.deserializeUser(User.deserializeUser())

to track whether someone is logged, who is logged in for each session 🡪 we use sessions

underneath where you created the app in app.js, add the following

app.use(session({

secret: 'These are not the droids you are looking for'

}))

Underneath the passport serialization of the user object, initialise passport and use the session in app.js

app.use(passport.initialize())

app.use(passport.session())

set up the mongo db connection above the passport serialization. Do this as soon as possible in a project.

mongoose.connect('mongodb://localhost/passport-demo', err => {

// if there was no error, err will be undefined and be falsey

if (err) {

console.log('Error connecting to db:', error)

} else {

console.log('Connected to db')

}

})

Run mongod

Run nodemon

Once you have configured strategy and storage system, just need to use passport in the rest of the code. There needs to be a way register users