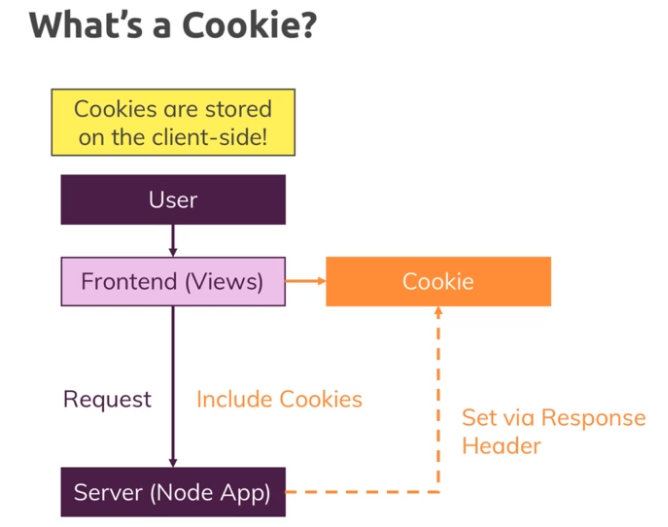
* **Sessions & Cookies**

So by that we can save our data in user side



Suppose the user logins from frontend or from our view so the request goes to server and the server can send back the cookie which can be stored somewhere in frontend so by that **user do not have to write the whole login id again**.

So in this we are not using authentication but instead we focus on the on the cookies and the session

But first lets create the login view page and the functionality of node in our current project

So first we create the route of authentication in our routes file as getLogin imported

const express = require('express');

// calling it from controller

const authController = require('../controllers/auth');

const router = express.Router();

router.get('/login', authController**.getLogin**);

// exporting so that can be used in app

module.exports = router;

after that we have to create authentication controller as we have to maintain a flow

**exports.getLogin** = (req, res, next) => {

  res.render('auth/login', {

    path: '/login',

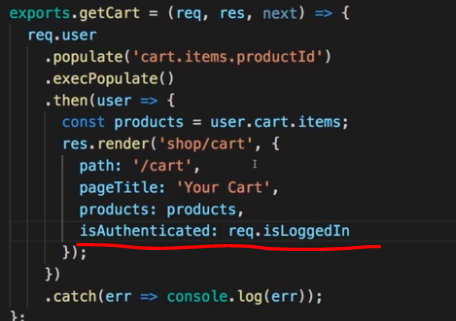
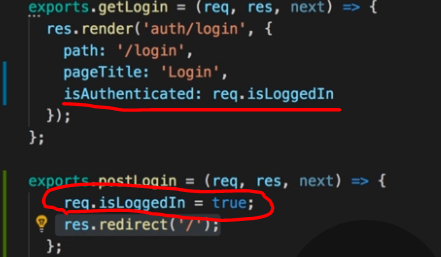
    pageTitle: 'Login'

  });

};

So we can use the cookies for the authentication purpose suppose if we want to enable the cart and product page for the user only when the user is logged in so ho do we do that means there is no means to add a product until a user is logged in.

So what we can do is we save our authentication object is true while we logged in and we can use that object by saving them in our req.isLoggedIn = true and can use it on our cart and product get and post controller so to display at the time of logged in only but this will not going to run because as soon as you save it in the request whenever you going to render the req.data will going to be lost so there is no use for doing this instead we use cookies, here saving the true by req in is auth so if it is true it will run but it not going to be



So what we have to do is we have to use cookies so cookies can make it globally excessable so we can do it by saving it in res header with passing the header name and their key value pair that we want

exports.postLogin = (req, res, next) => {

  res.setHeader(**'Set-Cookie', 'loggedIn=true'**);

  res.redirect('/');

};

So here it in the res header that the set-cookies will contains the key loggedIn and their value is true

exports.getLogin = (req, res, next) => {

  const isLoggedIn = req

    .get('Cookie') // get coolies

    .split(';')[1] // the key value pair is in the first index

    .trim()

    .split('=')[1]; // value is in the 1 index

  res.render('auth/login', {

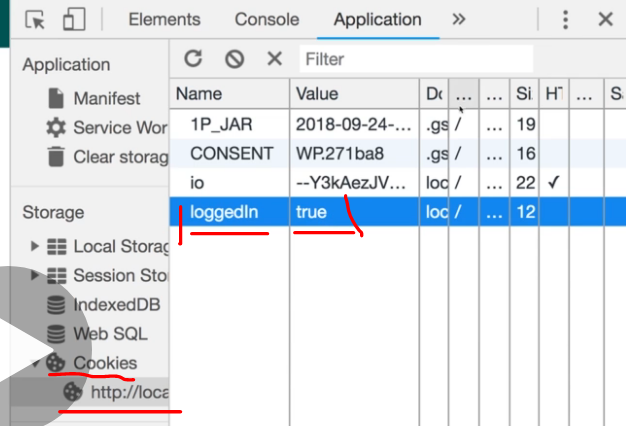
    path: '/login',

    pageTitle: 'Login',

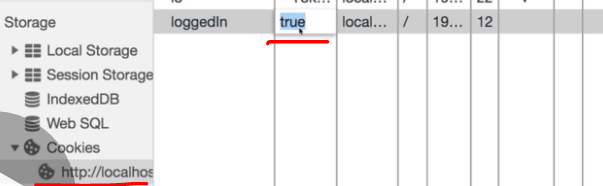
    isAuthenticated: isLoggedIn

  });

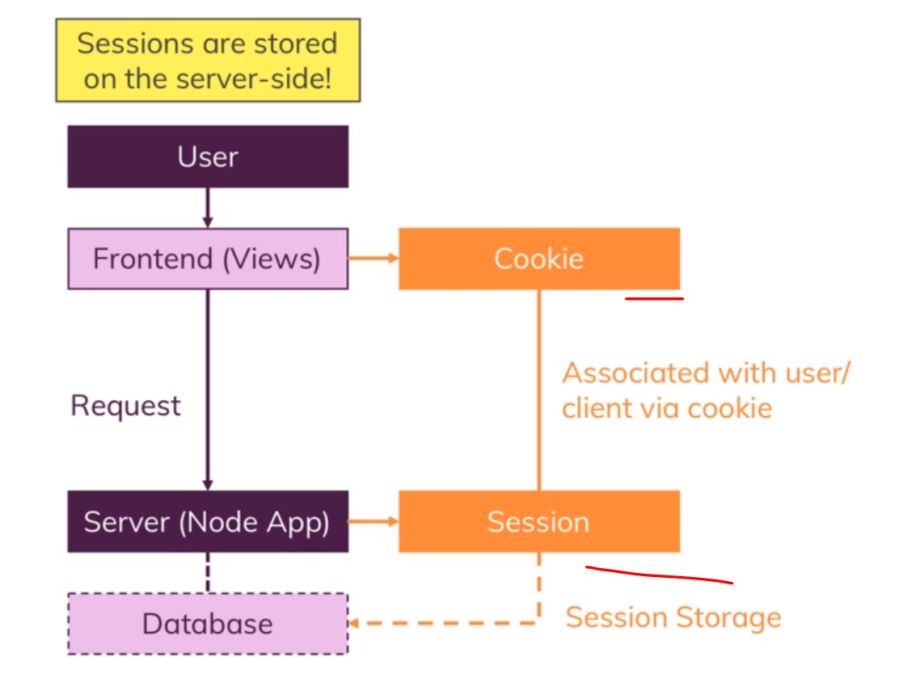
};



So if we can do all things with cookies then why do we need session then and the answer is the cookies can be added by the user from cookies which generally we do not want



but cookies is widely used in tracking the device , making expiration of time such as payment gateway open and expiration time, to run wthe page in secure https only

Session in node, So session is stored in the backend while cookies in the front end

So our idea is instead of storing the data in key value pair in cookies now what we can do is we can save the data in session and their id of the of data inside cookies so we can call the data from session

So first we have to install session with express in our terminal



After that that we have to requires the session in our main app and then use it in app

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

app.use(session({ secret: 'my secret', resave: false, saveUninitialized: false}));

so we create secret and resave : so not get saved untill required

so now we if we do create a session request of object isLoggedIn = true so by that we can use it as res.isLoggedIn for authentication

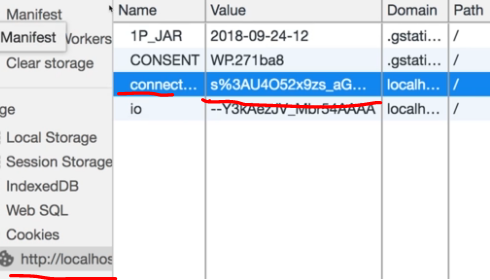
exports.postLogin = (req, res, next) => {

**req.session.isLoggedIn= true**; // now it creates an isloggedIn object

  res.redirect('/');

};

So it will create a new cookie of having the id of that object from which we can call our session



So now if we do  console.log(req.session.isLoggedIn); it will give the result

But as we know the session is stored in memory so if the data is huge then it will slow down the memory and that is not we want so to use the session data efficiently we use database

For that we are using the **connect-mongodb-session** which will store the data in mongodb

* Detail linked <https://www.npmjs.com/package/connect-mongodb-session> so need to install 

Express/ session detailed git main => <https://github.com/expressjs/session>

So now what we can do is instead of storing the data in memory session we now can require our mongodb session as we install and then we create our store by which we can get the place of storing our data and then placing the session

// call our mongodb session

const MongoDBStore = require('connect-mongodb-session')(session);

// saving the session in that uri an at that collection name

const store = new MongoDBStore({

  uri: MONGODB\_URI,

  collection: 'sessions'

});

app.use(

  session({

    secret: 'my secret',

    resave: false,

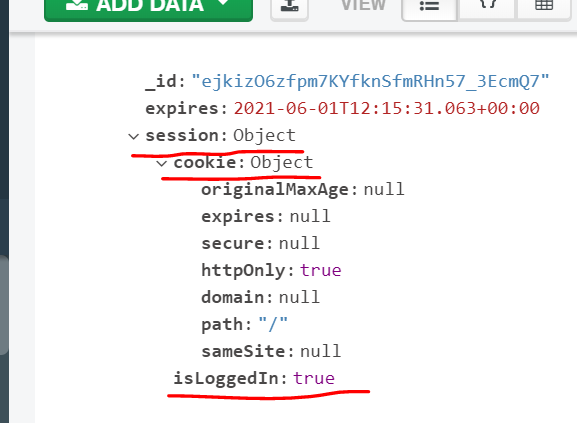
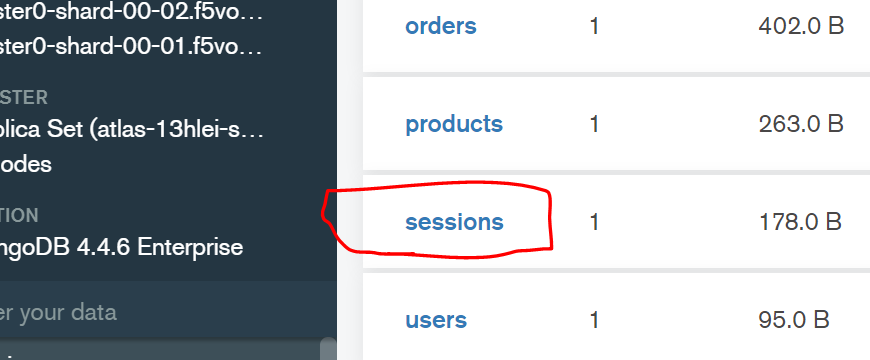
    saveUninitialized: false,

    store: store // passing the place to store session

// here cookies will created automatically

  })

);



So we can add session in our data base

So now we want to create alogout button so we create it in our view page and lest do the action of (/logout) which will go to the auth rote because we create a logout auth there in which we call a logout function from the auth controller in which we are destroying our data an the redirecting in the page

So first we create a logout option button in our page where we going to put inside it form and the action will be (/logout) so it will find the rout with it and is in the auth rout which will call the postLogout Method of the auth controller

// it will remove the users session data from the database

exports.postLogout = (req, res, next) => {

  req.session.destroy((err) => {

    console.log(err);

    res.redirect('/');

  });

};

So the data will be destroyed in our mongodb database

And now we will handle our view folder the includes at navigation and also add to cart page here we set when do want to show what to the user suppose the login button will not shown if the user is login and the logout button will not show if user is logout