Passport

authenticating with passport

Lesson Plan

- What is authentication
- Different forms of authentication
- What is Passport
- How passport works
- passport sessions
- Connecting passport to express
- Custom Strategy
- JWT strategy

Lesson summary:

https://www.nerdeez.com/articles/node/express-passport

Authentication

- if we have a set of users in our app, authentication helps us determine who the user that sent the request
- we have to make sure that it cannot be faked
- we have to decide if we want to persist it on session

different forms of authentication

- username + password in a login page, then keeping the user logged in in the session
- facebook login
- JWT

Passport.js

- used for authenticating users
- can be combined with different forms of authentication (referred to as strategies)
- easily use session if we want
- easily customized if we want our own authentication

How passport works

- install passport
- configure strategy
- attach to any route that you want to use that authentication
  - attaching to route with **passport.authenticate** which create a listener before our listener
  - if authentication passed will call next with req.user populated
  - if auth fails will send 401 optionally redirecting to a failed page
- you can combine multiple strategies
- you can create a strategy of your own
- passport will populate req.user
- req.isAuthenticated can check if the session contains authentication

How passport works

- you can consider passport as doing something like this
  - app.get('/auth', function whatPassportIsDoing(req, res, next) {
    req.user = 'i will populate the user with some data and then call next';
    next(); //this will jump to the next listener and we will have req.user for that listener
    });

Our custom strategy

- Our first strategy will verify that the request contains the header: **hello: world**
- if it does we will grant access
- if not we will redirect to error page
- we will not use session for now
- to load a custom strategy we will also need to install the package **passport-custom**
- we will have 3 routes:
    - /login
    - /restricted
    - /error
    -

Passport Session

- In most strategies you will authenticate once and want to save in the session that the user is authenticated and the user data.
- By default session will keep the user data in session
- todo that we need to tell passport what user data we want to save in the session (usually pk) **serializeUser**
- we would also need from the session to grab the data and deserialize the user (usually take the pk and grab the user from the db) **deserializeUser**
- passport provide a function in the request object **isAuthenticated** that we can use that will return true if the user already authenticated
- We can use the **isAuthenticated** method to build a middleware that checks if the user logged in

JWT Strategy

- for rest servers it’s common to use JWT authentication
- on the login route we will authenticate the user with our custom strategy, if the user is authenticated we will create a jwt token for him
- the client needs to make sure to take that token and add it to the Authorization header with the bearer schema
- we will create a rest api that returns a fixed todo array and we will add our jwt authentication to that api

Summary

In most cases of authentication passport will present us with a really easy solution that we can use instead of creating everything ourselves.

EX.

- in your login page try to integrate passport username password solution.
- Also for your rest api try to use JWT authentication