## Install (đã có sẵn cookie-parser trong này)

$ npm install express-session

var session **=** require('express-session')

app.use(session({

    secret: "xHjnM5sTJungtin",

    resave: true,

    saveUninitialized: false

}));

## Cookie Options

app.set('trust proxy', 1) // trust first proxy

/\* Nếu cookie.secure = true thì phải có trust 1st proxy \*/

app.use(

    session({

        genid: function(req) {

            // Function để tạo SESSION\_ID

            return genuuid() // use UUIDs for session IDs

        },

        name: "connect.sid", // tên sessionID in response

        secret: "xHjnM5sTJungtin",

        resave: true,

        saveUninitialized: false,

        /\*

            force uninit session phải saved to store

            - false : impl login session & help racing condition

        \*/

        cookie: {

            path: "/", // Set-Cookie Path : root path of domain

            httpOnly: true, // Chỉ chấp nhập http -> document.cookie => '' ở client(tránh xss)

            secure: false, // chỉ chấp nhận ssl

            maxAge: null, // Set-Cookie: Expires

            domain: "www.jungtin.me", // default = null, vì cookie sẽ applied lên domain hiện tại

            expires: new Date(Date.now() + 3600), // 3600s

            /\*

            Nếu cả 2 expires & maxAge đều set -> last one defined sẽ được sdung

            \*expires không nên set trực tiếp, luôn sử dụng maxAge

            expires dùng như này :

                req.session.cookie.expires = new Date(Date.now() + 3600)

        \*/

            sameSite: true,

            /\* VALUES :

            true : Strict for strict same site enforcement.

            false

            'lax' : Lax for lax same site enforcement.

            'none' : None for an explicit cross-site cookie.

            'strict' : Strict for strict same site enforcement.

        \*/

        },

        unset: 'keep', // 'destroy' : delete session after response,

        // 'keep' : session kept in store

        store: sessionStore // Object chứa data về cách impl session

    })

);

## Enable “cookie.secure” only for production

vì khi dev mình chạy ở localhost không có ssl 🡪 cookie sẽ không được set nếu enable cookie.secure

\*enable cookie.secure thì phải có “trust 1st proxy”

var app **=** express()

var sess **=** {

  secret**:** 'keyboard cat',

  cookie**:** {}

}

**if** (app.get('env') **===** 'production') {

  app.set('trust proxy', 1) *// trust first proxy*

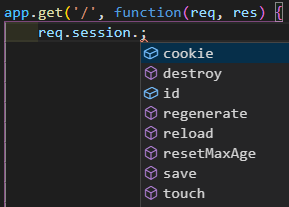
  sess.cookie.secure **=** true *// serve secure cookies*

}

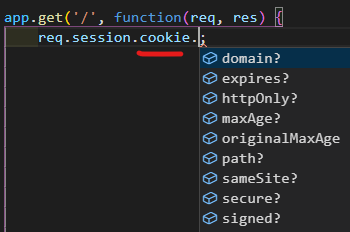
app.use(session(sess))

## In-Built APIs

Fields



req.session.id : id của session trả về cho response



Methods

req.session.\_\_ :

1. req.session.regenerate(err => {})
   * sau khi gọi thì req.session sẽ được tạo mới
2. req.session.destroy(err => {})
3. req.session.reload(err => {}) : reload session từ store
4. req.session.save(err => {}) : save vào store
5. req.session.touch(err => {}) : update maxAge

## Code Template

app.get("/", function (req, res) {

    if (req.session.views) {

        req.session.views++;

        res.setHeader("Content-Type", "text/html");

        res.write("<p>views: " + req.session.views + "</p>");

        res.write("<p>expires in: " + req.session.cookie.maxAge / 1000 + "s</p>");

        res.end();

    } else {

        req.session.views = 1;

        res.end("welcome to the session demo. refresh!");

    }

});

## Session Store Implementation

Sequelize Implementation

$ npm install connect-session-sequelize

// load dependencies

var express = require("express");

var Sequelize = require("sequelize");

var session = require("express-session");

// initalize sequelize with session store

var SequelizeStore = require("connect-session-sequelize")(session.Store);

var sequelize = new Sequelize("database", "username", "password", {

    dialect: "sqlite",

    storage: "./session.sqlite",

});

var app = express();

var myStore = new SequelizeStore({

    db: sequelize,

    checkExpirationInterval: 15 \* 60 \* 1000, // The interval at which to cleanup expired sessions in milliseconds.

    expiration: 24 \* 60 \* 60 \* 1000, // The maximum age (in milliseconds) of a valid session.

});

app.use(

    session({

        secret: "keyboard cat",

        store: myStore,

        resave: false,

        proxy: true,

    })

);

myStore.sync();

1 số methods của store

* store.all((err, sessions) => {…}) : get all session as an array
* store.destroy(sid, (err, sessions) => {…}) : destroy session with id
* store.clear((err) => {…}) : clear all session from store
* store.length((err, len) => {…}) : count of all session in store
* store.get(sid, (err, sessions) => {…}) : null if not found
* store.set(sid, session, (err) => {…})
* store.touch(sid, session, (err) => {…}) : auto idle session

## Authentication

Session Data sẽ không được lưu vào cookie mà chỉ có SESSION\_ID mà thôi

Session Data lưu ở Server