Nodejs cheatsheet

This cheatsheet summarizes **some** useful methods/properties that can be used in a Nodejs project. Not everything but some of the learned materials during the course have been included here.

Some commands required to initialize, develop, and run a project:

|  |  |
| --- | --- |
| Commands | Description |
| npm init -y | To initialize the project |
| npm install <package-name> | Installs a package from NPM’s own repository |
| npm install --save-dev nodemon | To permanently install nodemon |
| node <nodejs-file-name > | To run the project (It will keep previous state when you make change to any file until you run the command again). Example command for running app.js file: node app.js |
| nodemon < nodejs-file-name > | To run the project (It will behave according to the latest changes in the code once refreshing the page, without rerunning the app.) Example command for running app.js file: nodemon app.js |

Importing rules for most of the modules is the same except for express-validator.

Main syntax: const your\_naming = require(‘module\_name’);

|  |  |
| --- | --- |
| Modules | Importing syntax |
| express | const express = require(‘express’) |
| express-validator | const { body, validationResult } = require("express-validator"); |
| express-session | const session = require(‘express-session’) |

Some middleware code snippets and description:

|  |  |
| --- | --- |
| Code snippet | Description |
| app.use(express.static(‘public’)); | To tell the program to look inside public folder for static files such as css files. |
| app.use(express.urlencoded(  {extended: true} ) | To make the program parse the variables submitted with the form. |
| app.set(‘view engine’, ‘ejs’) | To tell the program to look inside ‘views’ folder for .ejs files. |
| app.use(session({  secret: ‘your-secret-key’,  resave: false,  saveUninitialized: false,  cookie: {maxAge: ‘For how many milliseconds a cookie will be usable as an integer’} })) | To create a session middleware inside the program, so it will work anywhere in the program inside requested parameters |
| app.use(cookieParser()) | After installing cookie-parser module, this middleware will allow you use cookies inside your app. |
| require('dotenv').config() | In order to use environment variables |
| var conn = mysql.createConnection({  host: 'localhost',  user: process.env.USER,  password: process.env.PASSWORD,  database: 'nodeform'  }) | To establish Mysql connection. |
| conn.connect((err) => {  if (err) throw err;  }) | Connecting database |
| app.get(‘/’, (req, res) => {}) | Basic route syntax |
| body('<param\_name>').isLength({min: 8}).withMessage().matches('[0-9]') | Express validator |
| body(<param\_name>).escape().custom((value, {req}) => {  }).withMessage() | Express validator for custom validation |
| conn.query(query\_name, [req.body.param], (err, result) => {}) | Sql query |
| bcrypt.hash(req.body.password, 12).then(function(hashpass){}) | Bcrypt hash |
| bcrypt.compare(password, passwordDb).then(function(result){}) | Bcrypt compare |
| res.cookie('cookie\_name', {username: username}, {maxAge: 30000}) | Set cookie |
| let current\_user = req.cookies.cookie\_name | Read cookie |
| res.clearCookie(cookie\_name) | Clear cookie |
| req.session.isAuth = true | Session |
| req.session.destroy(function (err) {}) | Destroy session |
| <% %> and <%= %> | = when reading variable (ejs) |
| [0-9] | Regex for number |
| [A-Z] | Regex for uppercase |