**NODEJS**

1. Tạo project

* Npm init
* Cài module express: npm i express –save

|  |  |
| --- | --- |
| Const express = require(‘express’) | Thêm module express vào project |
| Const app = express(); | Khởi tạo 1 app mới sd module express |
| Const port = 8080 | Định nghĩa tên cổng |
| App.get(‘/’, (req, res) => res.send(‘Hello’)) | + get() có tham số đầu tiên là địa chỉ mà server sẽ nhận req từ client, tham số t2 là 1 func xử lý req, và gửi kq trả về gọi là res  +send(): gửi kq trả về |
| App.listen(port, func() {…}) | Listen() khởi động server. Tham số đầu là port, tham số t2 là callback func sẽ đc gọi khi server khởi động |

1. Get() & set()

* Set(): gắn giá trị cho một biến nào đó

App.set(‘ten\_bien’, ‘gia\_tri”);

* Get(): lấy giá trị của biến nào đó

App.get(‘ten\_bien’);

Vd:

Var express = require(‘express’); //gọi tới module express

Var app = express(); //Khởi tạo đối tượng

App.set(‘title’, ‘abc xyz’);

Var title = app.get(‘title’);

1. Templete engine

Công cụ tạo ra các HTML bằng các đoan mã được tối giản

Res.render(): render ra 1 format HTML mà vẫn ở lại trang hiện tại

1. Query parameters:

Query parameters là một chuỗi truy vấn được client gửi lên server. Server sẽ nhận các thông tin này để xử lý và trả về một kết quả phù hợp với truy vấn được gửi lên.

1. POST

* Dùng để gửi dữ liệu lên client và thêm vào dữ liệu của server
* Với method post, để nhận dl ta use req.body



* Nodemon: - npm i --save-dev nodemon
* "scripts": {
* "start": "nodemon index.js",
* Pug: - npm i pug –save
* Tạo các file với đuôi \_ten\_file.pug
* Trong file root:

+ app.set(‘view engine’, ‘pug’)

+ app.set(‘views’, ‘./views’)

* Req.body trả về 1 object tương tụ như req.query
* Npm i body-parse –save
* Trong file root:

+ const bodyParser = require(‘body-parse’)

+ app.use(express.json()) // for parsing application/json

+ app.use(express.urlencoded({ extended: true }))

* Lowdb
* Npm i lowdb –save
* const low = require('lowdb');
* const FileSync = require('lowdb/adapters/FileSync');
* const adapter = new FileSync('db.json');
* const db = low(adapter);
* Npm i shortid –save : tạo id cho object, có thể dùng or k

const shortid = require('shortid');

1. Res.redirect(‘<url>’): điều hướng người dùng
2. Res.locals : lưu trữ các dl ở trong 1 vòng đời request response, k bị ảnh hưởng từ các req khác
3. Cookie:



- ở 1 req gửi lên server, server có thể trả về cookie hoặc k (trả về 1 cái header gọi là set-cookie và kèm theo giá trị mà mình muốn set)

- khi browser nhận đc res từ server mà có header là set-cookie thì nó sẽ lưu lại bên phía trình duyệt

- từ các lần gửi tiếp theo, nó sẽ gửi tất cả các cookie mà nó đã lưu lại ở trên trình duyệt. server có thể trả về lệnh clear cookie, thì nó sẽ clear đi

\* ứng dụng trong authentication, lưu lại section trong cookie

10. md5

- dùng để mã hóa

11. signed cookie:

- cookie sẽ đc thêm hash ở phía sau gọi là sign nature

- sign nature đảm bảo cho value không bị thay đổi, nếu thay đổi thì sign nature sẽ bị thay đổi theo.

- khi ng dùng thay đổi cookie thì server sẽ phát hiện ra k phải là cookie ban đầu nữa

\* cài đặt: npm i cookie-parser

12. enviroment variables: biến môi trường

Khi ta commit code lên git hay môi trường khác, mà ta k muốn 1 số thông tin bảo mật bị đẩy lên cùng thì ta sẽ dùng enviroment variables. Dùng thư viện dotenv

* Cài đặt:

+ npm i dotenv

+ Ta chỉ cần tạo file .evn sau đó lưu các cặp key-value vào đó. Lưu ý: ta sẽ k commit file này lên git

* Để k commit file .env lên git ta để tên file trong file .gitignore

13. debug:

- để debug thì trong file package.json ta thêm –inspect vào start như sau:

