NodeJS

NodeJS = serve para interpretar o código de JavaScript sem precisar ir pro navegador.
Biblioteca = conjunto de códigos que fazem uma tarefa específica.
Package.json = arquivo do node pra listar dependências, informações, versão do programa, autoria e scripts. Como o readme.
NPM = site que disponibiliza várias bibliotecas feitas por outros programadores. Elas são baixadas no arquivo node-modules, e explicitadas nas dependências do package.json.
Módulos = uma parte do código que realiza uma tarefa, que fazem algo específico.
Dependências = módulos que outros programadores criaram e estamos utilizando.
Gitignore = arquivo que lista o que deve ser ignorado no github, pra não sobrecarregar. Como módulos, por exemplo.
CLI = Interface de Linha de Comando. Faz uma ligação entre o terminal e o código.
API = faz a conexão entre o que pedimos e o que o servidor devolve, traduz de um pra outro.
Postman = ambiente para testes de códigos.
Sequelize = possibilita usar o JavaScript para alterar o banco de dados, ao invés do SQL. É um tipo de ORM. Antes de começar a codar, instalar as bibliotecas Express, Body-Parser, NodeMon.
Markdown = linguagem de programação ideal para lugares com vários textos, como blogs.
Chalk = biblioteca pra destaque de textos.
FS = biblioteca pro código acessar arquivos do computador de dentro do código.
Joi e Yup = bibliotecas pra validar formulários.
Mongoose = biblioteca para conexão entre o node e o MongoDB.
Express = biblioteca que gerencia os caminhos entre SQL e Java.
Body-Parser = biblioteca que converte os dados pra json.
NodeMon = biblioteca que atualiza automaticamente, sem precisar derrubar o servidor.
Site = o Regex 101 é um ótimo site pra testar expressões regulares e conferir sua documentação.
MVC = modelo, vista e controlador (intermediário entre os pedidos do usuário e as respostas do código). Padrão para controle onde o modelo é acessado somente pelo controlador, e esse gera a visualização para o usuário final.
node -v = mostra a versão do node, basta digitar no terminal do computador.
node nome_arquivo = roda o arquivo, quando digitado no terminal.
npm install nome_biblioteca = instala uma biblioteca do npm para ser usada no arquivo.
npm install nome_biblioteca@numero_versão --save-exact = instala uma biblioteca do npm com uma versão específica, não deixa atualizar para outras versões.
import nome_biblioteca from "nome_biblioteca"; = importa a biblioteca para ser usada no código, deve estar na primeira linha. Quando for usar uma função da biblioteca, colocar nome_biblioteca.nome_função().
Regex = ou expressão regular. Servem para pedir dados específicos ao código. [] separam elas em classes, e () em grupos. No JavaScript, são usadas dentro de constantes, normalmente em funções. Em qualquer caso, colocar gm; no final.
/palavra/ = mostra todas as ocorrências dessa palavra.
/[caractere 1 caractere 2]/ = mostra todas as ocorrências desses caracteres.

/[^caractere 1 caractere 2]/ = mostra todas as não ocorrências desses caracteres.

/[a-z]/ = mostra todas as ocorrências de letras minúsculas, desde o a até o z.

/[\s]/ = mostra todas as ocorrências de barras de espaço.

/[/ = mostra todas as ocorrências de colchetes abrindo.

/\[[^[\]]*?\]/ = mostra todas as ocorrências de [palavras].

/\(https?:\/\/[^\s?#.].[^\s]*\)/ = mostra todas as ocorrências de (http://palavras).

process.argv = processa os argumentos, os valores passados por CLI para o código. Como um input de CLI.

fs.lstatSync(argumento).isFile(); = mostra se o argumento é um arquivo ou não.

fs.lstatSync(argumento).isDirectory(); = mostra se o argumento é um diretório ou não.

fs.promises.readdir(argumento); = lê um diretório, retornando um array com o nome de todos os arquivos dele.

fetch(url).status = retorna o status do link do navegador. Se for 200 está normal, tudo certo.

...nome_variável = desconstrói a estrutura. Ao invés de vir tudo em arrays, vem em strings.

npm init -y = no terminal, cria um arquivo package.json.

ORM = permite mexer no banco de dados usando linguagem de programação, ao invés de linguagem de banco de dados.

Sequelize = um tipo de ORM.

npm install sequelize sequelize-cli path = instala o sequelize com as propriedades para linhas de comando.

express.use(bodyParser.json()) = chama o bodyparser pra converter os dados em json.

var port = numero_porta = define a porta que será usada, normalmente 3000.

sudo mysqul -u root -p = entra no terminal do MySQL.

show databases; = mostra os bancos de dados do MySQL.

show tabelas; = mostra as tabelas do banco de dados.

create database nome_banco_de_dados; = cria uma base de dados.

npx sequelize-cli model:create --name nome_sequelize --attributes colunal:tipo_dadol,coluna2:tipo_dado2,coluna3:tipo_dado3 = cria uma tabela sequelize com as colunas pedidas e id automático. Sempre colocar os nomes no plural.

npx sequelize-cli db:migrate = cria uma migração, que permite a comunicação entre tabelas e banco de dados.

describe nome_tabela = mostra as propriedades da tabela pedida, como tipo de dado, obrigatoriedade...

insert into nome_tabela (coluna 1, coluna 2, coluna 3) values ("valor 1", "valor 2", "valor 3"); = cria um registro na tabela.

npx sequelize-cli seed:generate --name nome_seed = cria um seed.

npx sequelize-cli db:seed:all = conecta todos os seeds com o banco de dados.

npx sequelize-cli db:migrate:undo = desfaz a última migração feita.

db:migrate:undo --name [data_hora]-create-[nome_tabela].js = desfaz uma migração específica.

npx sequelize db:seed:undo = desfaz o último seed feito.

npx sequelize-cli db:seed:undo --seed nome_seed = desfaz um seed específico.

paranoid: true = não permite deletar algo. O elemento some da tela, mas não é deletado, é como uma ilusão.