

Express js

<https://cybersoft.edu.vn/>



Mục Lục

01 Kết quả đạt được

02 Giới thiệu chung

03 Cài đặt

04 Thực hành

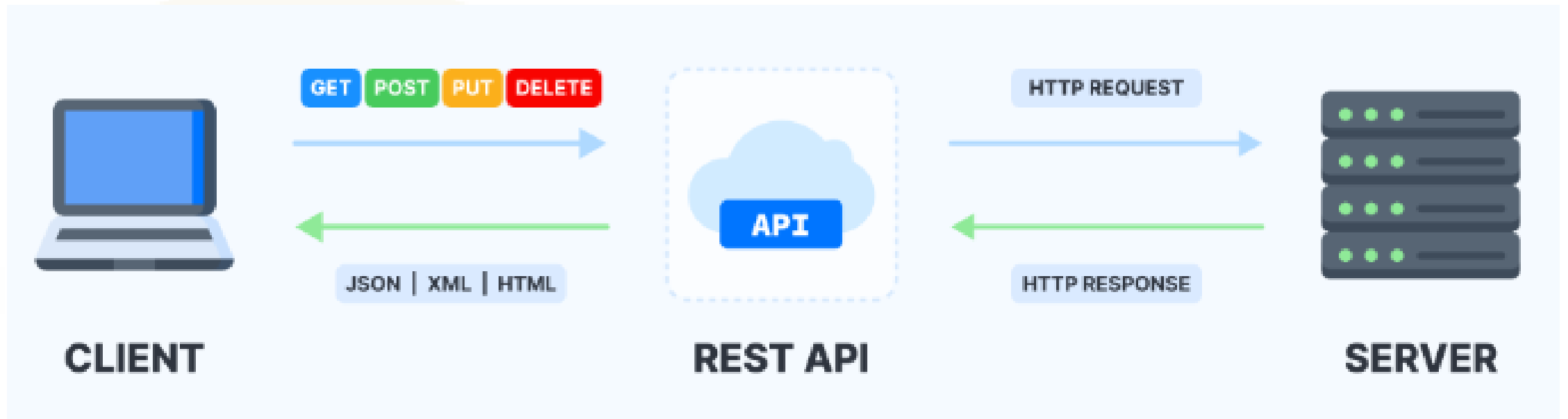
05 Kết nối database

06 Thực hành

07 Structure, env

Kết quả đạt được

Lấy dữ liệu từ database để viết API



Restful API

- **API** (**A**pplication **P**rogramming **I**nterface) là một tập các quy tắc và cơ chế mà theo đó, một ứng dụng hay một thành phần sẽ tương tác với một ứng dụng hay thành phần khác.
- API có thể trả về dữ liệu mà bạn cần cho ứng dụng của mình ở những kiểu dữ liệu khác nhau.
- **REST** (**R**epresentational **S**tate **T**ransfer) là một dạng chuyển đổi cấu trúc dữ liệu, một kiểu kiến trúc để viết API. Nó sử dụng phương thức HTTP đơn giản để tạo cho giao tiếp giữa các máy.
- **RESTful API** là một tiêu chuẩn dùng trong việc thiết kế các API
- Phương thức (method): **GET, POST, PUT, DELETE**

Restful API



Yarn

- Cải tiến từ **npm**
- Công cụ quản lý các gói phần mềm (library)
- Cài đặt: **npm install -g yarn**
- Fix PowerShell:

Set-ExecutionPolicy -Scope CurrentUser -ExecutionPolicy Unrestricted

```
PS C:\> File C:\Users\Muoi\AppData\Roaming\npm\yarn.ps1 cannot be loaded.
```

Express - Postman

- Cài đặt: **yarn add express nodemon**

- **Request:** lấy data từ client (FE)

params, query string

body

headers

- **Response:** trả data về client (FE)

statusCode (200,400,500)

message

data => not send number

```
JS index.js X package.json
JS index.js > ...
1  const express = require('express');
2  const app = express();
3  app.use(express.json())
4
5  app.listen(8080, () => {
6    console.log('server connected')
7  })
8
9  app.get("/api", (req, res) => {
10
11    const { id } = req.query;
12
13    res.status(200).send([id]);
14  })
15
16  app.get("/api/:id", (req, res) => {
17
18    const { id } = req.params;
19    const { hoTen, lopHoc } = req.body;
20
21    res.status(200).send(hoTen);
22  })
```

Thực hành

- **Lấy danh sách phim (GET)**
- **Lấy phim theo id (GET)**
- **Thêm một phim (POST)**
- **Xóa một phim (DELETE)**

Connect database

Cài đặt: **yarn add mysql2**

```
//set up connect mysql
const mysql = require("mysql2");

const conn = mysql.createConnection({
  host: "localhost",
  user: "root",
  password: "1234",
  database: "node"
})

app.get("/api/users", async (req, res) => {

  const sql = "SELECT * FROM users";

  const lstUser = await conn.promise().query(sql);

  res.send(lstUser[0])

})
```

Thực hành

Viết API cho bảng user

GET, POST, PUT, DELETE

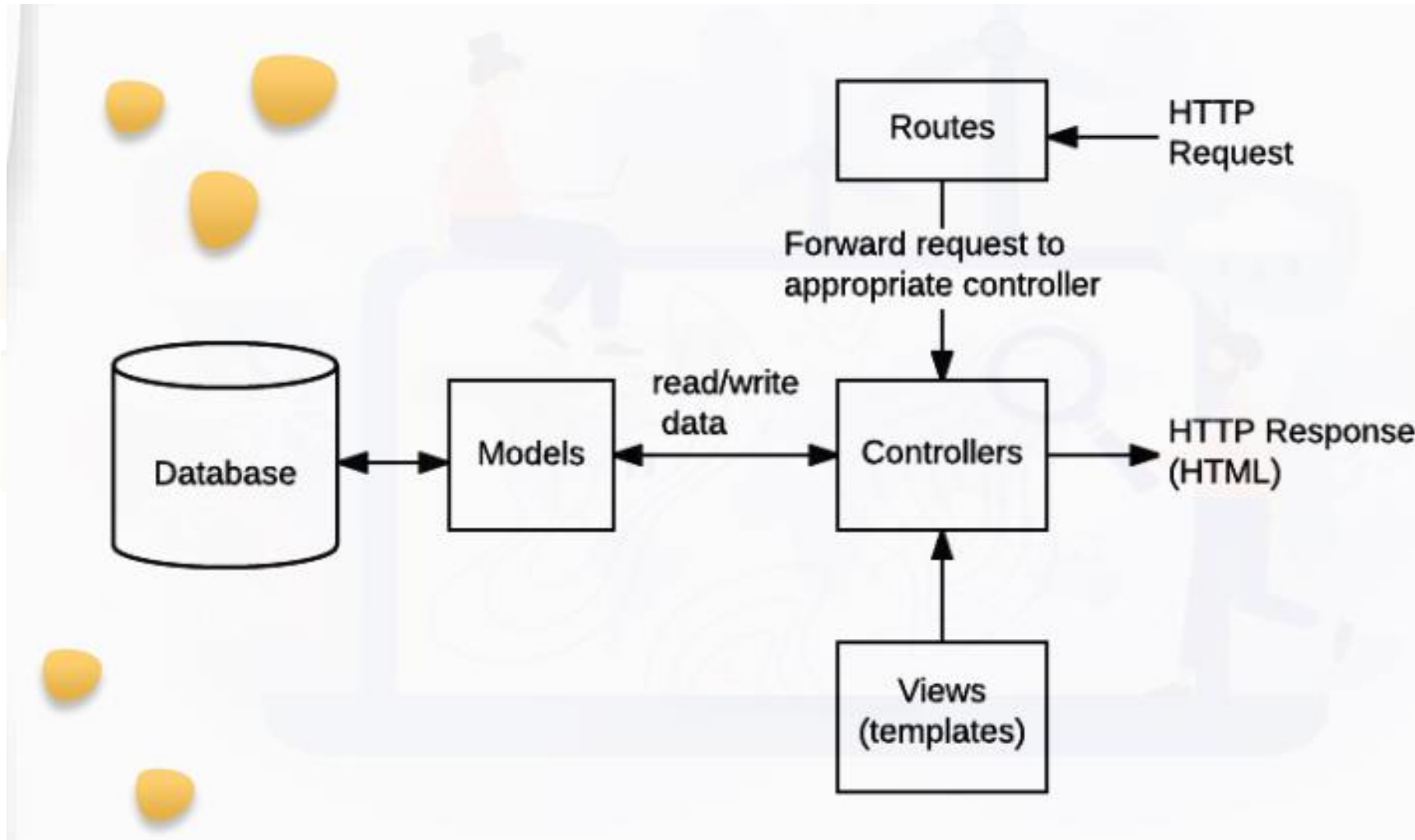
ĐÀO TẠO CHUYÊN GIA LẬP TRÌNH

Tương tác với Front-End

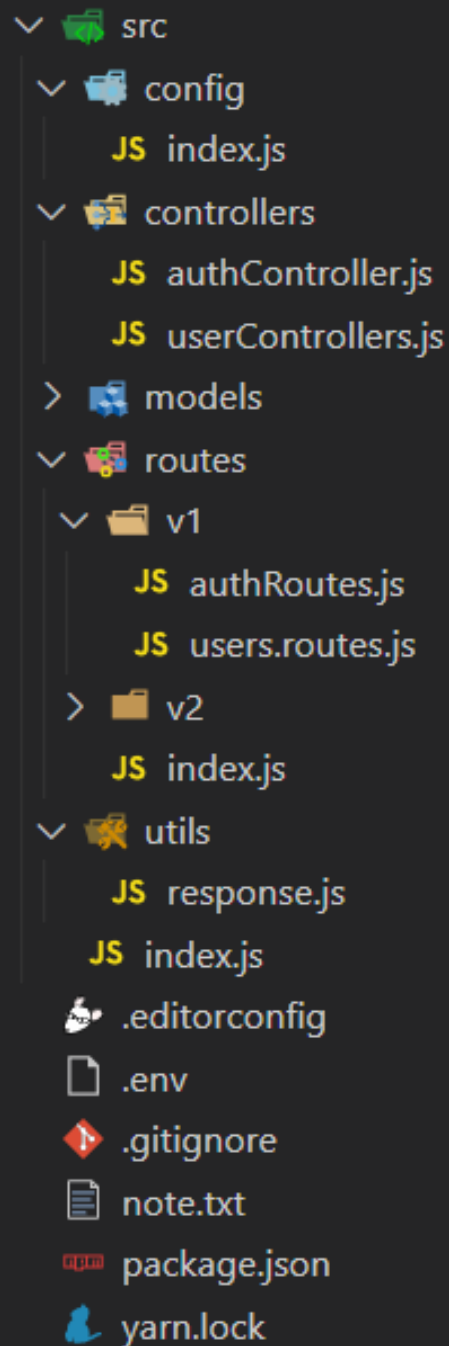
```
app.use(function (req, res, next) {  
  res.header("Access-Control-Allow-Origin", "*");  
  res.header("Access-Control-Allow-Headers", "Origin, X-Requested-With,  
Content-Type, Accept");  
  next();  
});
```

```
const cors = require('cors');  
app.use(cors());
```

Structure



Structure



```
JS index.js ...\routes JS users.routes.js JS user_product.js

src > routes > JS index.js > ...
1  const express = require('express');
2  const userRouter = require('./v1/users.routes');
3
4  const rootRouter = express.Router();
5
6  rootRouter.use("/v1/users", userRouter);
7
8  module.exports = rootRouter;
```

```
JS users.routes.js JS user_product.js JS product_type.js

routes > v1 > JS users.routes.js > ...
const express = require("express");

const userRouter = express.Router();
const userController = require('../../controllers/userControllers');


//thay app = route
userRouter.get("/", userController.getUsers);
userRouter.get("/them", userController.createUser)

module.exports= userRouter;
```

env

Biến môi trường lưu trữ dữ liệu tĩnh ít khi thay đổi

Cài đặt: **yarn add dotenv**



```
1 // dotenv
2 require('dotenv').config();
3 module.exports = {
4   db_alialect: process.env.DB_DIALECT,
5   db_host: process.env.DB_HOST,
6   db_name: process.env.DB_NAME,
7   db_port: process.env.DB_PORT,
8   db_root: process.env.DB_ROOT,
9   db_pass: process.env.DB_PASS,
10 }
```

SOFT
IA LẬP TRÌNH

Response statusCode

```
const successCode = (res, data, message) => {  
  res.status(200).json({  
    message,  
    content: data  
  })  
}  
  
const failCode = (res, data, message) => {  
  res.status(400).json({  
    message,  
    content: data  
  })  
}  
  
const errorCode = (res, message) => {  
  res.status(500).send(message)  
}  
module.exports = {  
  successCode,  
  failCode,  
  errorCode  
}
```