## LabO1 - Kiến Thức Cơ Bản Về Client-Server

Môn học: Phát Triển Ứng Dụng Web Nâng Cao

Nhóm 3 | Ngày: 12/08/2025



#### CẤU TRÚC NHÓM

## Giới Thiệu Đội Ngũ

1

Nguyễn Văn Anh Đức

Frontend & Giao Diện

2

Phan Ngọc Vỹ

Backend & Logic

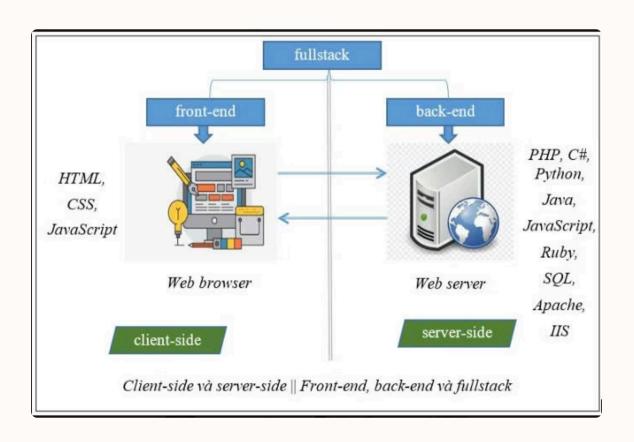
3

Nguyễn Hồ Đức Huy

Kiểm Thử & Báo Cáo

### TỔNG QUAN

## Kiến Trúc Hệ Thống Client-Server



#### Các thành phần cốt lõi:

- Frontend: Giao diện web tương tác
- Backend: Server Express.js với API endpoints
- Client HTTP Tùy Chỉnh: Phát triển bằng Node.js
- Giám Sát Mạng: Theo dõi hiệu suất thời gian thực
- **Tích Hợp Bên Ngoài:** GitHub API, JSONPlaceholder

#### PHÍA SERVER

## Triển Khai HTTP Server

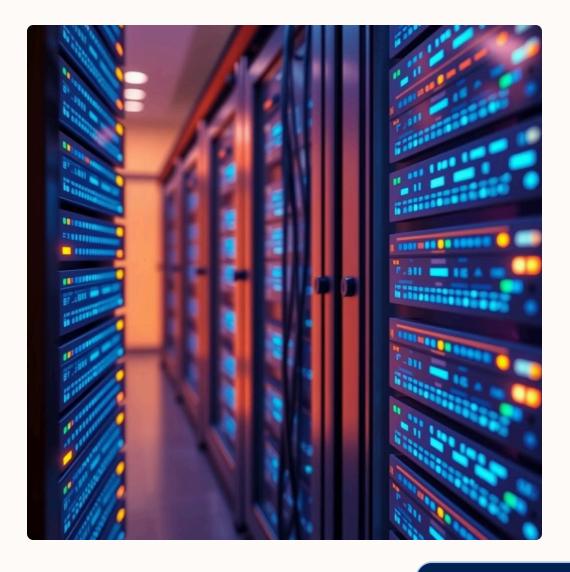
```
// Express.js setup
const app = express();
app.use(express.static('public'));

// Custom headers middleware
app.use((req, res, next) => {
    res.header('X-Powered-By', 'Lab01-Server');
    res.header('X-Server-Time', new Date().tolSOString());
    next();
});

// API endpoints
app.get('/api/server-info', (req, res) => {
    res.json({ /* server information */ });
});
```

#### Các tính năng nổi bật:

- Phục vụ file tĩnh
- RESTful API endpoints
- Custom HTTP headers
- Xử lý lỗi (404, 500)
- Middleware ghi log



### Triển Khai HTTP Client

```
class HTTPClient {
    async request(options) {
        return new Promise((resolve, reject) => {
            const client = options.url.startsWith('https:') ? https :
            http;
            const req = client.request(options, (res) => {
                 // Handle response
            });
            req.on('error', reject);
            req.end(options.data);
            });
        }
}
```

#### Các tính năng chính:

- Sử dụng Pure Node.js
- Hỗ trợ HTTP/HTTPS
- Phương thức GET, POST
- API dựa trên Promise
- Xử lý lỗi hiệu quả

```
Q netwol request: Copiesd
< sparthlietwori/sise Netwoy)</pre>
  penedo()
  leannaine /leadl(fl);
   cartessstip fonck (cordystiny)
   lactpate (estwas(foratov))
   landles: Inal froal untecting)
     ponetant :-aratiasl)
      j flich leaa opryace;
     dorettice countngr ferice:/onawatesfi)-(I-rfonii):
     dartatiry chetoly: Tentle:/cuctertowets-(arfilial)
water toprost/ablle any: fil:fendinge;
   fate Tetall assagt )
     ctefer weepil:
      rastangt);
wastilfice /for({on(ets/focloscqurests-/artical)}
       warisine: reneticol practiar-destica(Ol))
partiller iah (latch sastrica;
cisvicalle corcionel);
       wertilel -arctitip:
   recongeted sie fromettine);
```

HIỆU SUẤT MẠNG

Giám Sát Hiệu Năng

#### TRỰC QUAN HÓA

# Demo Úng Dụng

Các ảnh chụp màn hình thể hiện giao diện và chức năng của ứng dụng Client-Server.



### THỬ THÁCH VÀ KẾT QUẢ

### Những Bài Học Quan Trọng

#### Hiểu Sâu HTTP Protocol

Nắm vững cơ chế hoạt động của HTTP ở tầng thấp nhất.

#### Thiết Kế Client-Server Hiệu Quả

Xây dựng kiến trúc mạnh mẽ và phân tán.

### Lập Trình Bất Đồng Bộ

Thực hành với Promise và async/await.

#### Phân Tích Hiệu Suất Mạng

Sử dụng công cụ giám sát để tối ưu hóa.

#### Xử Lý Lỗi Toàn Diện

Nâng cao độ ổn định và tin cậy của ứng dụng.

#### PHÁT TRIỂN TIẾP THEO

# Các Cải Tiến Tương Lai



#### HTTPS & Chứng Chỉ

Tăng cường bảo mật kết nối.



#### WebSocket

Giao tiếp thời gian thực.



#### JWT Authentication

Xác thực người dùng an toàn.



#### Dashboard Phân Tích

Hiển thị số liệu chi tiết.



#### Tối Ưu Hiệu Năng

Nâng cao tốc độ xử lý.

### TỔNG KẾT

### Thành Tựu & Hướng Đi

#### Thành Tựu Đạt Được

- Hoàn thiện Client-Server
- Đạt mọi yêu cầu đề ra
- Triển khai tính năng nâng cao
- Tài liệu hóa chi tiết

### Insight Quan Trọng

- Sức mạnh của HTTP
- Tầm quan trọng xử lý lỗi
- Lợi ích giám sát hiệu suất
- Thực hành phát triển web hiện đại