**Production Architecture Overview**

**Frontend**

|  |  |
| --- | --- |
| รายการ | รายละเอียด |
| Framework | React (with Vite + TypeScript + Tailwind CSS) |
| Build Tool | Vite |
| Routing | React Router (ถ้าจำเป็นภายหลัง) |
| Image Source | Mock API (e.g., placehold.co) |
| Image Layout | react-masonry-css |
| Deployment | Static Hosting (เช่น Vercel, Netlify, Cloudflare Pages, หรือ S3 + CloudFront) |

**Backend API (optional สำหรับ mock data)**

|  |  |
| --- | --- |
| รายการ | รายละเอียด |
| API Framework | Express.js หรือ Node.js |
| Host | Render, Railway, หรือ VPS (เช่น DigitalOcean) |
| Database (ถ้ามี) | PostgreSQL หรือ MongoDB Atlas |
| Image Data | ในที่นี้ mock จาก frontend หรือ backend memory |

**หมายเหตุ: ถ้าใช้ mock data และ static image → ไม่มี backend ก็ deploy แบบ Pure Static SPA ได้เลย**

**Server Specifications**

|  |  |
| --- | --- |
| รายการ | คำอธิบาย |
| OS | Linux (Ubuntu 22.04 LTS) |
| CPU | 1 vCPU (ขั้นต่ำ) |
| RAM | 1 GB – 2 GB |
| Disk | 5GB SSD |
| Hosting | [Vercel](https://vercel.com), [Netlify](https://netlify.com), หรือ [Cloudflare Pages](https://pages.cloudflare.com/) *(แนะนำ Vercel สำหรับ React)* |

**CI/CD (Build & Deploy)**

|  |  |
| --- | --- |
| รายการ | เทคโนโลยี |
| Git Platform | GitHub / GitLab |
| CI/CD Pipeline | Built-in ของ Vercel / Netlify |
| Branch Mapping | main → Production, dev → Preview |
| Auto Build & Deploy | เมื่อ push ไปยัง main จะ build และ deploy อัตโนมัติ |

**วิธี Deploy แบบ Step-by-Step (ถ้าใช้ Vercel)**

1. **Push Code ไป GitHub**
2. **เข้า** [**vercel.com**](https://vercel.com)
3. **Login ด้วย GitHub แล้วเลือก repo**
4. **เลือก framework เป็น Vite**
5. **ตั้ง Environment**
6. **คลิก Deploy**

**Conclusion**

* **Vite + React + Tailwind = SPA**
* **Hosting = Static (เช่น Vercel)**
* **ถ้ามี Backend = Node.js + Express + DB**
* **ระบบรองรับ Filter หลายแท็ก, Load เพิ่มอัตโนมัติ, รองรับ Responsive Layout**