



# VacciTrack - Hệ thống theo dõi

Phân tích dữ liệu tiêm chủng

Tên: Nguyễn Minh Kha

Lớp: TC-PY4E03

Người hướng dẫn : Lý Quốc Lâm

# Mục Lục

**01** Giới thiệu dự án

**02** Công nghệ

**03** Vai Trò

**04** Phân luồng

**05** Quản Lý

**06** Phát triển

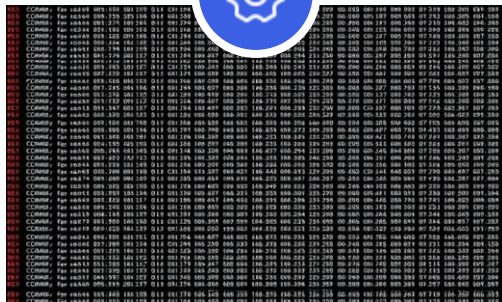


# 01

## Tổng quan dự án VacciTrack



# Giải quyết vấn đề dữ liệu



## Vấn đề dữ liệu thô

Dữ liệu COVID-19 tồn tại dạng CSV khổng lồ, khó đọc và khó so sánh.



## Thách thức phân tích

Hàng nghìn dòng số liệu rời rạc không thể hỗ trợ ra quyết định nhanh chóng.



## Giải pháp VacciTrack

Web App tự động xử lý, làm sạch và trực quan hóa dữ liệu thành Dashboard dễ hiểu.

# Ứng dụng Python Flask

## Xử lý dữ liệu

Python và Pandas xử lý file CSV phức tạp hiệu quả.

## Tư duy Fullstack

Kết hợp Data Engineering, Backend và Frontend tạo hệ thống hoàn chỉnh.

## Micro-framework

Flask nhẹ, triển khai nhanh, phù hợp xây dựng RESTful API.

## Kiến trúc Client-Server

Tách biệt tầng dữ liệu và hiển thị, tối ưu luồng thông tin.

Index	File Size	File Location	Device	Temp	Events
1.1	15.2M	0.0000	0.0000	0.0000	0.0000
1.2	15.2M	0.0000	0.0000	0.0000	0.0000
1.3	15.2M	0.0000	0.0000	0.0000	0.0000
1.4	15.2M	0.0000	0.0000	0.0000	0.0000
1.5	15.2M	0.0000	0.0000	0.0000	0.0000
1.6	15.2M	0.0000	0.0000	0.0000	0.0000
1.7	15.2M	0.0000	0.0000	0.0000	0.0000
1.8	15.2M	0.0000	0.0000	0.0000	0.0000
1.9	15.2M	0.0000	0.0000	0.0000	0.0000
1.10	15.2M	0.0000	0.0000	0.0000	0.0000
1.11	15.2M	0.0000	0.0000	0.0000	0.0000
1.12	15.2M	0.0000	0.0000	0.0000	0.0000
1.13	15.2M	0.0000	0.0000	0.0000	0.0000
1.14	15.2M	0.0000	0.0000	0.0000	0.0000
1.15	15.2M	0.0000	0.0000	0.0000	0.0000
1.16	15.2M	0.0000	0.0000	0.0000	0.0000
1.17	15.2M	0.0000	0.0000	0.0000	0.0000
1.18	15.2M	0.0000	0.0000	0.0000	0.0000
1.19	15.2M	0.0000	0.0000	0.0000	0.0000
1.20	15.2M	0.0000	0.0000	0.0000	0.0000
1.21	15.2M	0.0000	0.0000	0.0000	0.0000
1.22	15.2M	0.0000	0.0000	0.0000	0.0000
1.23	15.2M	0.0000	0.0000	0.0000	0.0000
1.24	15.2M	0.0000	0.0000	0.0000	0.0000
1.25	15.2M	0.0000	0.0000	0.0000	0.0000
1.26	15.2M	0.0000	0.0000	0.0000	0.0000
1.27	15.2M	0.0000	0.0000	0.0000	0.0000
1.28	15.2M	0.0000	0.0000	0.0000	0.0000
1.29	15.2M	0.0000	0.0000	0.0000	0.0000
1.30	15.2M	0.0000	0.0000	0.0000	0.0000
1.31	15.2M	0.0000	0.0000	0.0000	0.0000
1.32	15.2M	0.0000	0.0000	0.0000	0.0000
1.33	15.2M	0.0000	0.0000	0.0000	0.0000
1.34	15.2M	0.0000	0.0000	0.0000	0.0000
1.35	15.2M	0.0000	0.0000	0.0000	0.0000
1.36	15.2M	0.0000	0.0000	0.0000	0.0000
1.37	15.2M	0.0000	0.0000	0.0000	0.0000
1.38	15.2M	0.0000	0.0000	0.0000	0.0000
1.39	15.2M	0.0000	0.0000	0.0000	0.0000
1.40	15.2M	0.0000	0.0000	0.0000	0.0000
1.41	15.2M	0.0000	0.0000	0.0000	0.0000
1.42	15.2M	0.0000	0.0000	0.0000	0.0000
1.43	15.2M	0.0000	0.0000	0.0000	0.0000
1.44	15.2M	0.0000	0.0000	0.0000	0.0000
1.45	15.2M	0.0000	0.0000	0.0000	0.0000
1.46	15.2M	0.0000	0.0000	0.0000	0.0000
1.47	15.2M	0.0000	0.0000	0.0000	0.0000
1.48	15.2M	0.0000	0.0000	0.0000	0.0000
1.49	15.2M	0.0000	0.0000	0.0000	0.0000
1.50	15.2M	0.0000	0.0000	0.0000	0.0000
1.51	15.2M	0.0000	0.0000	0.0000	0.0000
1.52	15.2M	0.0000	0.0000	0.0000	0.0000
1.53	15.2M	0.0000	0.0000	0.0000	0.0000
1.54	15.2M	0.0000	0.0000	0.0000	0.0000
1.55	15.2M	0.0000	0.0000	0.0000	0.0000
1.56	15.2M	0.0000	0.0000	0.0000	0.0000
1.57	15.2M	0.0000	0.0000	0.0000	0.0000
1.58	15.2M	0.0000	0.0000	0.0000	0.0000
1.59	15.2M	0.0000	0.0000	0.0000	0.0000
1.60	15.2M	0.0000	0.0000	0.0000	0.0000
1.61	15.2M	0.0000	0.0000	0.0000	0.0000
1.62	15.2M	0.0000	0.0000	0.0000	0.0000
1.63	15.2M	0.0000	0.0000	0.0000	0.0000
1.64	15.2M	0.0000	0.0000	0.0000	0.0000
1.65	15.2M	0.0000	0.0000	0.0000	0.0000
1.66	15.2M	0.0000	0.0000	0.0000	0.0000
1.67	15.2M	0.0000	0.0000	0.0000	0.0000
1.68	15.2M	0.0000	0.0000	0.0000	0.0000
1.69	15.2M	0.0000	0.0000	0.0000	0.0000
1.70	15.2M	0.0000	0.0000	0.0000	0.0000
1.71	15.2M	0.0000	0.0000	0.0000	0.0000
1.72	15.2M	0.0000	0.0000	0.0000	0.0000
1.73	15.2M	0.0000	0.0000	0.0000	0.0000
1.74	15.2M	0.0000	0.0000	0.0000	0.0000
1.75	15.2M	0.0000	0.0000	0.0000	0.0000
1.76	15.2M	0.0000	0.0000	0.0000	0.0000
1.77	15.2M	0.0000	0.0000	0.0000	0.0000
1.78	15.2M	0.0000	0.0000	0.0000	0.0000
1.79	15.2M	0.0000	0.0000	0.0000	0.0000
1.80	15.2M	0.0000	0.0000	0.0000	0.0000
1.81	15.2M	0.0000	0.0000	0.0000	0.0000
1.82	15.2M	0.0000	0.0000	0.0000	0.0000
1.83	15.2M	0.0000	0.0000	0.0000	0.0000
1.84	15.2M	0.0000	0.0000	0.0000	0.0000
1.85	15.2M	0.0000	0.0000	0.0000	0.0000
1.86	15.2M	0.0000	0.0000	0.0000	0.0000
1.87	15.2M	0.0000	0.0000	0.0000	0.0000
1.88	15.2M	0.0000	0.0000	0.0000	0.0000
1.89	15.2M	0.0000	0.0000	0.0000	0.0000
1.90	15.2M	0.0000	0.0000	0.0000	0.0000
1.91	15.2M	0.0000	0.0000	0.0000	0.0000
1.92	15.2M	0.0000	0.0000	0.0000	0.0000
1.93	15.2M	0.0000	0.0000	0.0000	0.0000
1.94	15.2M	0.0000	0.0000	0.0000	0.0000
1.95	15.2M	0.0000	0.0000	0.0000	0.0000
1.96	15.2M	0.0000	0.0000	0.0000	0.0000
1.97	15.2M	0.0000	0.0000	0.0000	0.0000
1.98	15.2M	0.0000	0.0000	0.0000	0.0000
1.99	15.2M	0.0000	0.0000	0.0000	0.0000
2.00	15.2M	0.0000	0.0000	0.0000	0.0000

# Trực quan hóa dữ liệu

## Biểu đồ động

Chuyển đổi dữ liệu thô thành biểu đồ trực quan, dễ hiểu và tương tác.

## Chart.js

Thư viện JavaScript tạo biểu đồ tương tác đa dạng: đường, tròn, cột.

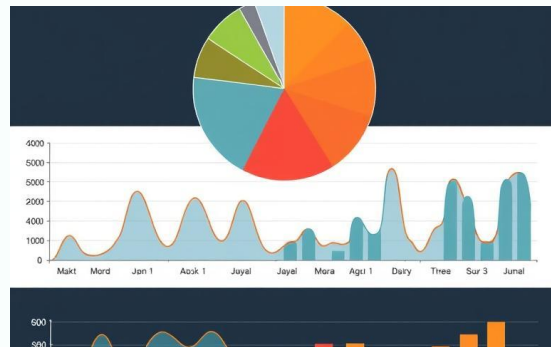


## Fetch API

Kết nối bất đồng bộ giữa frontend và backend, cập nhật dữ liệu linh hoạt.

## Dashboard hiện đại

Giao diện Glassmorphism với Tailwind CSS, hiển thị đồng bộ nhiều loại biểu đồ.



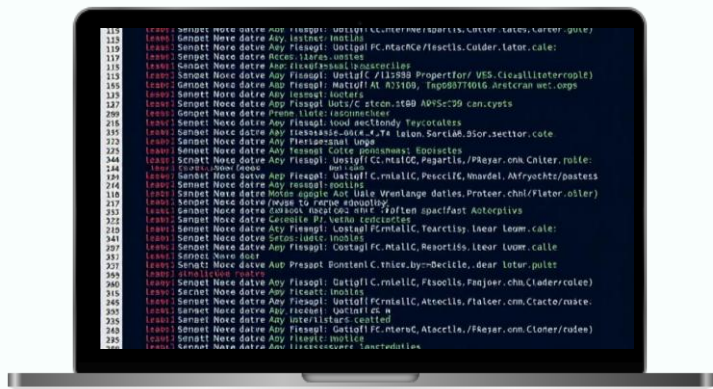


# 02

**Công nghệ sử dụng**

---

# Python và Pandas



# 1

## Xử lý dữ liệu mạnh mẽ

Thư viện tốt nhất xử lý CSV phức tạp, dễ thao tác hàng triệu dòng dữ liệu.

# 2

## Flask framework nhẹ

Triển khai nhanh, phù hợp xây dựng RESTful API cho dự án phân tích dữ liệu.

# 3

## Làm sạch dữ liệu

Xử lý giá trị rỗng, ép kiểu dữ liệu, tạo cấu trúc sẵn sàng cho API.



# Flask Micro-framework



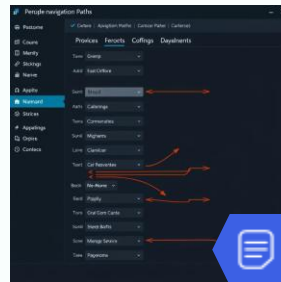
## Micro-framework nhẹ

Triển khai nhanh, phù hợp cho dự án data-driven.



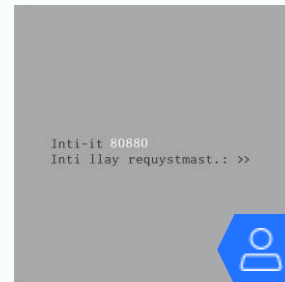
## RESTful API

Xây dựng API hiệu quả cho giao tiếp client-server.



## Routing linh hoạt

Cấu hình đường dẫn điều hướng dễ dàng.



## Web Server

Khởi tạo server tại port 8080, xử lý request.

# HTML5/Tailwind CSS

---



## Giao diện Glassmorphism

Sử dụng Tailwind CSS tạo hiệu ứng kính mờ, tối ưu hiển thị mọi thiết bị.



## Responsive Design

Thiết kế thích ứng tự động với mọi kích thước màn hình, từ điện thoại đến máy tính.



## Chart.js

Thư viện vẽ biểu đồ tương tác, hiển thị dữ liệu trực quan sinh động.

# Chart.js và GenAI



01

## Biểu đồ tương tác

Chart.js tạo đồ thị động, trực quan từ dữ liệu JSON, hỗ trợ nhiều loại biểu đồ.



02

## Hỗ trợ AI

GenAI sinh mã CSS nhanh chóng, tăng tốc phát triển giao diện, giảm lỗi cú pháp.



03

## Tối ưu UX

AI gợi ý cấu trúc MVC, debug JavaScript, cải thiện trải nghiệm người dùng.



# 03

## Vai trò và quy trình



# Xử lý dữ liệu thô

01

## Thách thức dữ liệu

File CSV khổng lồ với hàng nghìn dòng số liệu rời rạc, khó đọc và phân tích.

03

## Quy trình làm sạch

Đọc file an toàn, xử lý lỗi, chuẩn hóa giá trị rỗng và ép kiểu dữ liệu.

02

## Công cụ Pandas

Thư viện mạnh mẽ xử lý file CSV phức tạp, dễ dàng thao tác hàng triệu dòng dữ liệu.

04

## Kết quả đầu ra

Dữ liệu sạch dạng List of Dictionaries, sẵn sàng cho API và hiển thị.

# Xây dựng API Backend

## Khởi tạo Flask

Tạo web server tại port 8080, cấu hình routing và API endpoints.

1

## Xử lý dữ liệu

Sử dụng Pandas làm sạch dữ liệu CSV, chuyển thành JSON.

2

## API Endpoints

Tạo `/api/vaccines` phân phối dữ liệu cho client.

3

## Mô hình MVC

Tách biệt logic xử lý và hiển thị dữ liệu.

4

# Trực quan hóa Frontend



01

## Giao diện Glassmorphism

Sử dụng Tailwind CSS tạo hiệu ứng kính mờ, tối ưu hiển thị mọi thiết bị.

02

## GenAI Support

AI sinh mã CSS nhanh, tăng tốc phát triển 50%, giảm lỗi cú pháp.

03

## Hiển thị dữ liệu

Fetch API gọi dữ liệu, chuyển JSON thành biểu đồ Chart.js đa dạng.



# 04

## Triển khai dự án





# Xử lý Raw Data

Input: File **country\_vaccinations.csv** chứa nhiều nhiều (NaN, null, sai format ngày tháng). dẫn đến sập server

## Hành Động:

Khi bao cho biết file csv sử dụng ; : sep=';'

Đọc file an toàn bằng engine python : (engine='python')

Xử lý dòng khi xuất hiện lỗi: (on\_bad\_lines='skip').

```
ParserError
pandas.errors.ParserError: Error tokenizing data. C error: Expected 4 fields in line 2056, saw 6

Traceback (most recent call last):
  File "C:\Users\devinstor\AppData\Local\Programs\Python\Python311\Lib\site-packages\fastapi.py", line 1406, in _call__
    return self._app._call__(env, start_response)
    ~~~~~^~~~~~
  File "C:\Users\devinstor\AppData\Local\Programs\Python\Python311\Lib\site-packages\fastapi.py", line 1476, in _call__
    response = self._handle_exception(e)
    ~~~~~^~~~~~
  File "C:\Users\devinstor\AppData\Local\Programs\Python\Python311\Lib\site-packages\fastapi.py", line 1477, in _call__
    response = self._full_dispatch_request()
    ~~~~~^~~~~~
  File "C:\Users\devinstor\AppData\Local\Programs\Python\Python311\Lib\site-packages\fastapi.py", line 882, in full_dispatch_request
    rv = self._handle_exception(e)
    ~~~~~^~~~~~
  File "C:\Users\devinstor\AppData\Local\Programs\Python\Python311\Lib\site-packages\fastapi.py", line 883, in full_dispatch_request
    rv = self._dispatch_request()
    ~~~~~^~~~~~
  File "C:\Users\devinstor\AppData\Local\Programs\Python\Python311\Lib\site-packages\fastapi.py", line 883, in dispatch_request
    return self._ensure_sync(self._call_function(*args, **kwargs)) # type: ignore[no-any-return]
    ~~~~~^~~~~~
  File "C:\Users\devinstor\AppData\Local\Programs\Python\Python311\Lib\site-packages\fastapi.py", line 883, in _call_function
    if = self._read_csv()
    ~~~~~^~~~~~
  File "C:\Users\devinstor\AppData\Local\Programs\Python\Python311\Lib\site-packages\pandas\io\parsers.py", line 1026, in read_csv
    return _read_csv_path_or_buffer(buf, **kwargs)
    ~~~~~^~~~~~
  File "C:\Users\devinstor\AppData\Local\Programs\Python\Python311\Lib\site-packages\pandas\io\parsers.py", line 626, in _read
    return parser.read(nrows)
    ~~~~~^~~~~~
  File "C:\Users\devinstor\AppData\Local\Programs\Python\Python311\Lib\site-packages\pandas\io\parsers.py", line 1023, in read
    ) = self._engine.read() # type: ignore[attr-defined]
    ~~~~~^~~~~~
  File "C:\Users\devinstor\AppData\Local\Programs\Python\Python311\Lib\site-packages\pandas\io\parsers.py", line 234, in read
    chunk = self._reader.read_nrows(nrows)
    ~~~~~^~~~~~
  File "pandas\_libs.parsers.py", line 638, in pandas._libs.parsers.TextReader._read_nrows
  File "pandas\_libs.parsers.py", line 600, in pandas._libs.parsers.TextReader._read_nrows
  File "pandas\_libs.parsers.py", line 674, in pandas._libs.parsers.TextReader._tokenize_rows
  File "pandas\_libs.parsers.py", line 682, in pandas._libs.parsers.TextReader._check_column_names
```

# Làm sạch dữ liệu

---

làm sạch dữ liệu(Data Cleaning):

Chuẩn hóa: Xử lý giá trị rỗng bằng `df.where(pd.notnull(df), None)`

Ép kiểu: Chuyển đổi dữ liệu sang `object` hoặc `datetime`.



## Output: Structured Data

Dạng List of Dictionaries, sẵn sàng cho API  
JSON.

# Xây dựng Server

## Xây dựng web server:

khởi tạo flask app tại port **8080**:

## Routing & API:

Routes: Cấu hình đường dẫn **@app.route** để điều hướng (Home, Analytics)

API Endpoint: Tạo **/api/vaccines** để phân phối dữ liệu JSON cho Client.



# Giao diện hiển thị dữ liệu



## Hiển thị dữ liệu (Visualization):

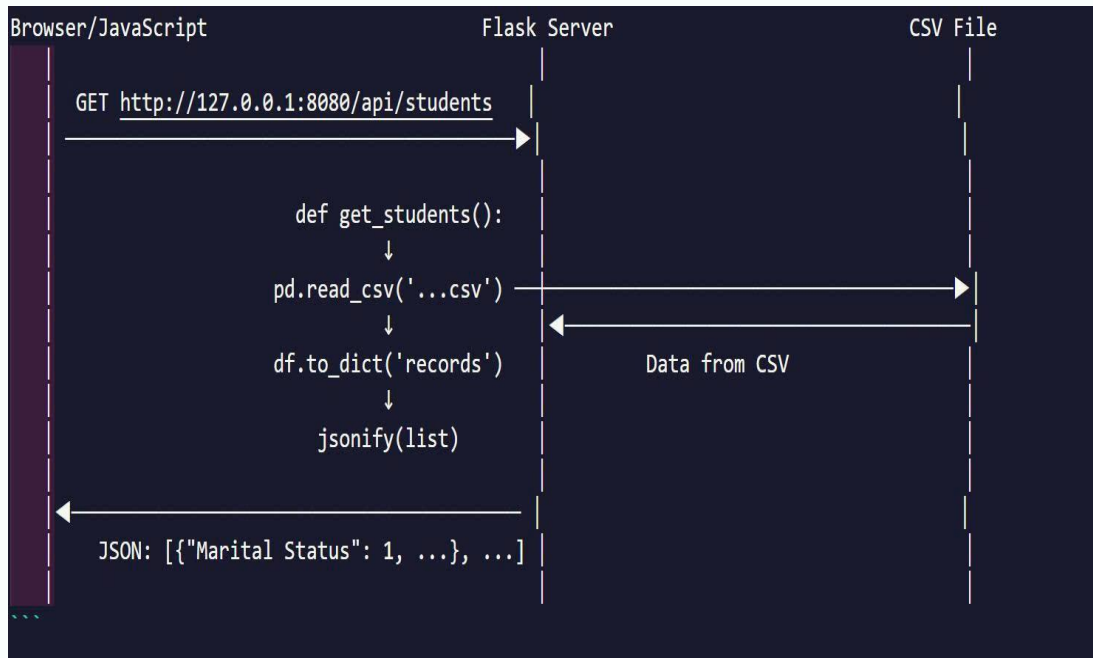
Client-Server :Sử dụng **Fetch API** để gọi dữ liệu từ Backend

Chart.js: Convert dữ liệu JSON thành các biểu đồ tương tác (Line, Doughnut, Polar Area)

# Kiến trúc hệ thống

## Mô hình Client-Server:

1. **Client:** Gửi Request
2. **Flask Server:** Nhận lệnh → Gọi Pandas
3. **Pandas:** Đọc & Làm sạch
4. **Server:** Trả về JSON
5. **Client:** Hiển thị biểu đồ



# Quản lý mã nguồn Git

---

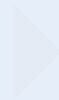
## Quản lý phiên bản

Sử dụng Git theo dõi thay đổi mã nguồn, dễ dàng quay lại phiên bản trước.



## Commit thường xuyên

Áp dụng Feature-based Commits, ghi chú rõ ràng cho từng tính năng mới.



## Đảm bảo an toàn

Bảo vệ mã nguồn, cho phép rollback nhanh chóng khi gặp lỗi.

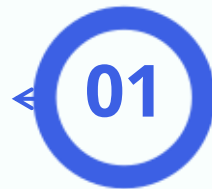


# Triển khai qua Ngrok

---

## Localhost

Chạy thử nghiệm và Debug trên máy cá nhân.



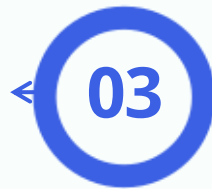
## Ngrok Tunnel

Tạo đường hầm bảo mật để public web server ra Internet.



## Demo từ xa

Trình diễn sản phẩm trực tiếp từ bất kỳ đâu.



The background is a solid blue color. Overlaid on this are several large, overlapping, wavy lines in a slightly lighter shade of blue. These lines create a sense of movement and depth, resembling stylized waves or flowing ribbons. In the center of the image, the word "Thanks !" is written in a bold, white, sans-serif font. The exclamation mark is slightly larger than the letters, emphasizing the sentiment.

**Thanks !**