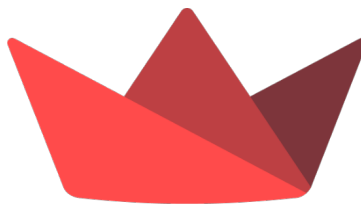


AI VIET NAM – AI COURSE 2025

Project: Building and Deploying Apps using Streamlit

Quoc-Thai Nguyen và Quang-Vinh Dinh

I. Giới thiệu



Streamlit

Hình 1: Streamlit

Streamlit là một thư viện mã nguồn mở của Python, được phát triển nhằm hỗ trợ việc xây dựng các ứng dụng web tương tác phục vụ cho khoa học dữ liệu và machine learning một cách nhanh chóng và thuận tiện. Framework này cho phép các lập trình viên chuyển đổi mã Python thông thường thành ứng dụng web chỉ với vài dòng lệnh đơn giản.

Với Streamlit, bạn có thể dễ dàng tạo dashboard, trực quan hóa dữ liệu qua các biểu đồ, cũng như triển khai các mô hình học máy, tất cả đều không yêu cầu kiến thức nền tảng về lập trình web.

I.1. Lợi ích của Streamlit

Streamlit là một lựa chọn phù hợp cho các ứng dụng web đơn giản, đặc biệt trong các dự án nhỏ hoặc dùng để trình diễn mô hình. Một số lợi ích chính gồm:

- Phù hợp với các dự án quy mô nhỏ, ứng dụng nội bộ hoặc demo nhanh.
- Không yêu cầu cấu hình bảo mật phức tạp.
- Thiết kế theo hướng một trang duy nhất, không cần liên kết nhiều trang.
- Giao diện đơn giản, dễ sử dụng, ít cần tùy chỉnh về layout.
- Có thể tích hợp với Flask, Django hoặc các công cụ frontend khác khi cần mở rộng.

I.2. Một số hàm thông dụng trong Streamlit

Các thành phần giao diện cơ bản trong Streamlit:

Các thành phần giao diện trong Streamlit là các yếu tố trực quan và tương tác, được sử dụng để xây dựng ứng dụng web. Mỗi thành phần cung cấp một chức năng cụ thể và có thể được tùy chỉnh linh hoạt để phù hợp với yêu cầu của ứng dụng. Dưới đây là một số thành phần cơ bản thường được sử dụng:

- `st.title("...")`: Dùng để tạo tiêu đề chính cho ứng dụng.
- `st.file_uploader("...", type=["..."])`: Tạo công cụ tải tệp lên ứng dụng. Tham số `type` cho phép giới hạn định dạng tệp được chấp nhận, ví dụ `["csv", "txt", "pdf"]`.
- `st.write("...")`: Dùng để hiển thị văn bản, bảng dữ liệu, biểu đồ hoặc các loại dữ liệu khác.
- `st.sidebar`: Tạo thanh điều hướng bên trái, giúp tổ chức các widget và điều khiển hợp lý hơn.
- `st.button("...")`: Tạo nút bấm cho phép thực thi hành động khi người dùng nhấn vào.
- `st.plotly_chart()`: Tạo biểu đồ tương tác sử dụng thư viện Plotly, hỗ trợ zoom, pan và các hành động tương tác khác.
- `st.line_chart()`, `st.bar_chart()`, `st.area_chart()`: Các hàm tích hợp sẵn để trực quan hóa dữ liệu dưới dạng biểu đồ đường, cột và vùng, thường sử dụng với `DataFrame` hoặc mảng `NumPy`.
- `st.map()`: Hiển thị dữ liệu địa lý trên bản đồ tương tác, áp dụng cho các tập dữ liệu có tọa độ kinh độ và vĩ độ.

Ngoài các thành phần trên, bạn có thể tham khảo thêm các widget và tính năng khác trong tài liệu chính thức của Streamlit để xây dựng các ứng dụng phong phú và tương tác hơn.

Mục lục

I.	Giới thiệu	1
I.1.	Lợi ích của Streamlit	2
I.2.	Một số hàm thông dụng trong Streamlit	2
II.	Thực hành Streamlit cơ bản	4
II.1.	Giới thiệu ứng dụng	4
II.2.	Pipeline tổng quan	4
II.3.	Cài đặt môi trường và các thư viện cần thiết	4
II.4.	Cài đặt thư viện cần thiết	5
II.5.	Thiết Kế Kiến Trúc Ứng Dụng	5
III.	Xây dựng ứng dụng tính hàm toán học giai thừa	12
III.1.	Phân tích, thiết kế ứng dụng	12
III.2.	Xây dựng ứng dụng	13
III.3.	Xây dựng tính năng phân quyền và xác thực người dùng	16
IV.	Câu hỏi trắc nghiệm	19
	Phụ lục	21

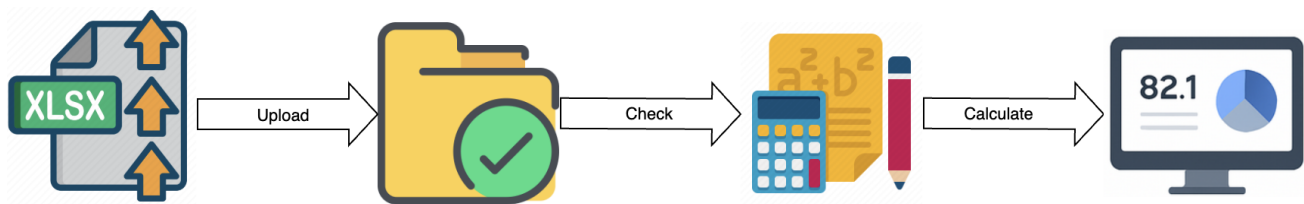
II. Thực hành Streamlit cơ bản

II.1. Giới thiệu ứng dụng

Trong phần thực hành này, ta sẽ xây dựng một ứng dụng hỗ trợ phân tích dữ liệu điểm số học sinh từ các file Excel. Các chức năng chính bao gồm: tính điểm trung bình, phân loại điểm theo từng khoảng, và trực quan hóa kết quả dưới dạng biểu đồ.

II.2. Pipeline tổng quan

Pipeline tổng quan của chương trình như sau: Người dùng upload file, hệ thống tự động tính toán và trả về kết quả phân tích cùng với biểu đồ trực quan.



Hình 2: Pipeline minh họa quá trình xử lý điểm số

II.3. Cài đặt môi trường và các thư viện cần thiết

Để xây dựng ứng dụng Streamlit một cách dễ dàng và tránh xung đột thư viện, ta nên tạo một môi trường ảo Python mới. Dưới đây là hai cách phổ biến để thực hiện: dùng `conda` hoặc `venv`.

Cách 1: Dùng `conda` (nếu bạn dùng Anaconda hoặc Miniconda)

1. Tạo môi trường mới:

```
conda create -n streamlit_app -y
```

- `-n streamlit_app`: chỉ định tên môi trường mới là `streamlit_app`.
- `-y`: tự động xác nhận (yes) tất cả các bước cài đặt, không cần hỏi lại người dùng.

2. Kích hoạt môi trường:

```
conda activate streamlit_app
```

Cách 2: Dùng `venv` (nếu không dùng Anaconda)

1. Tạo môi trường ảo:

```
python3 -m venv streamlit_env
```

2. Kích hoạt môi trường:

```
source streamlit_env/bin/activate
```

II.4. Cài đặt thư viện cần thiết

Sau khi kích hoạt môi trường, cài đặt các thư viện sau:

```
pip install streamlit pandas matplotlib openpyxl
```

- **streamlit**: streamlit xây dựng giao diện web tương tác cho ứng dụng Python.
- **pandas**: xử lý dữ liệu dạng bảng (DataFrame).
- **matplotlib**: vẽ biểu đồ, trực quan hóa dữ liệu.
- **openpyxl**: hỗ trợ đọc và ghi file Excel định dạng **.xlsx**.

II.5. Thiết Kế Kiến Trúc Ứng Dụng

1. Import các thư viện cần thiết

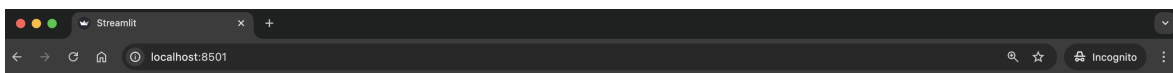
```
1 import matplotlib.pyplot as plt
2 import pandas as pd
3 import streamlit as st
```

- **streamlit**: xây dựng giao diện web.
- **pandas**: đọc và xử lý dữ liệu từ file Excel.
- **matplotlib.pyplot**: vẽ biểu đồ (ở đây là biểu đồ tròn).

2. Tạo tiêu đề ứng dụng

```
st.title("Phân tích dữ liệu điểm số học sinh")
```

Sử dụng `st.title()` để tạo tiêu đề lớn trên giao diện web. Người dùng sẽ thấy dòng tiêu đề nổi bật khi mở ứng dụng.



Deploy

Phân tích dữ liệu điểm số học sinh

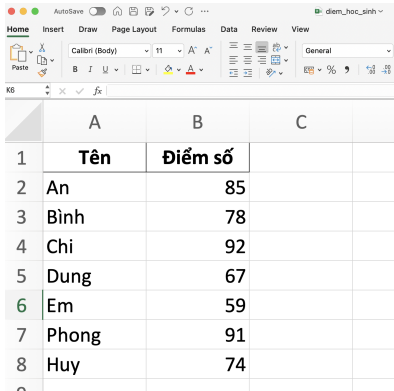
Hình 3: Hiển thị tiêu đề

3. Người dùng tải file lên

```
1 uploaded_file = st.file_uploader("Chọn file Excel (có cột 'Điểm số')", type=["xlsx"])
```

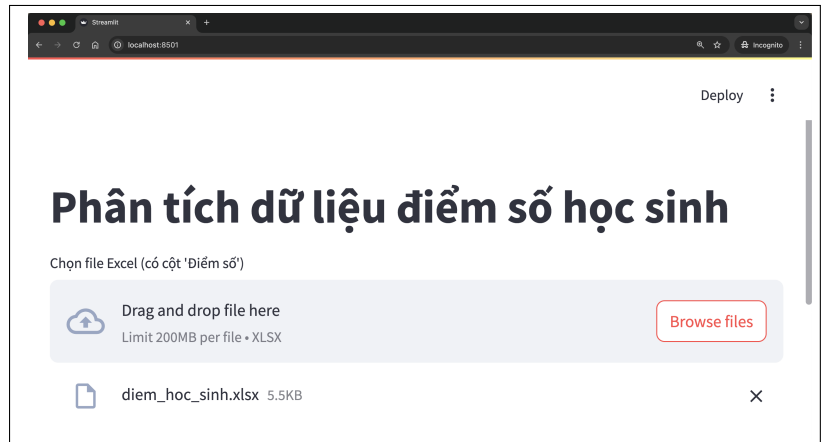
`st.file_uploader()` tạo nút chọn file để người dùng tải file Excel từ máy tính.

Tải lên file .xlsx chứa cột Điểm số như hình: **Lưu ý:** File phải đúng định dạng .xlsx và chứa cột tên "Điểm số".



	A	B	C
1	Tên	Điểm số	
2	An	85	
3	Bình	78	
4	Chi	92	
5	Dung	67	
6	Em	59	
7	Phong	91	
8	Huy	74	

(a) Chuẩn bị File Excel như hình



(b) Giao diện chương trình sau khi tải file Excel lên

Hình 4: Upload file Excel lên giao diện

Kiểm tra nếu người dùng đã upload file chưa. Nếu người dùng đã tải file lên, ứng dụng sẽ tiếp tục xử lý các bước tiếp theo.

4. Viết các hàm xử lý dữ liệu

a. Hàm tính điểm trung bình

```
1 def calculate_average(scores):
2     return sum(scores) / len(scores)
```

Hàm cộng tổng tất cả điểm rồi chia cho số lượng học sinh.

b. Hàm phân loại điểm số

```
1 def percentage_distribution(scores):
2     bins = {"90-100": 0, "80-89": 0, "70-79": 0, "60-69": 0, "<60": 0}
3     for score in scores:
4         if score >= 90:
5             bins["90-100"] += 1
6         elif score >= 80:
7             bins["80-89"] += 1
8         elif score >= 70:
9             bins["70-79"] += 1
```

```

10     elif score >= 60:
11         bins["60-69"] += 1
12     else:
13         bins["<60"] += 1
14     return bins

```

Hàm chia điểm số thành 5 nhóm và đếm số lượng học sinh trong mỗi nhóm.

5. Đọc file Excel và hiển thị trung bình

```

1  # Đọc file
2  df = pd.read_excel(uploaded_file)
3
4  # Xử lý danh sách điểm số
5  scores = df["Điểm số"].dropna().astype(float).tolist()
6
7  # Hiển thị các chỉ số
8  st.write("Tổng số học sinh:", len(scores))
9  st.write("Điểm trung bình:", round(calculate_average(scores), 2))

```

6. Vẽ biểu đồ

Ta chuyển kết quả phân loại về dạng danh sách như sau:

```

1  labels = list(dist.keys())
2  values = list(dist.values())

```

Trong đó:

- `labels`: Danh sách tên của các nhóm điểm số (ví dụ: "90-100", "80-89", ...).
- `values`: Danh sách số lượng học sinh tương ứng với từng nhóm điểm số.

Vẽ biểu đồ tròn

```

1  fig, ax = plt.subplots(figsize=(3, 3))
2  ax.pie(values, labels=labels, autopct="%1.1f%%", startangle=90)
3  ax.axis("equal")
4  st.pyplot(fig)

```

- `plt.subplots(figsize=(1, 1))`: Tạo biểu đồ với kích thước trung bình (1 inch × 1 inch).
- `ax.pie(values, labels=labels, autopct="%1.1f%", startangle=90)`: Vẽ biểu đồ tròn với:
 - `values`: Giá trị tương ứng với từng phần của biểu đồ.

- `labels`: Tên nhóm tương ứng với từng phần của biểu đồ.
- `autopct="%1.1f%":` `autopct` là viết tắt của "automatic percentage", dùng để tự động hiển thị tỷ lệ phần trăm trên từng phần của biểu đồ. Định dạng "%1.1f%" chỉ định rằng phần trăm sẽ được hiển thị dưới dạng số thực với một chữ số sau dấu thập phân.
- `ax.axis("equal")`: Đảm bảo biểu đồ cân đối (tỉ lệ trục X và Y bằng nhau).
- `st.pyplot(fig)`: Hiển thị biểu đồ trên ứng dụng Streamlit.

Hiển thị biểu đồ với độ phân giải cao

Cách 1: Vẽ trực tiếp bằng `st.pyplot(fig)`

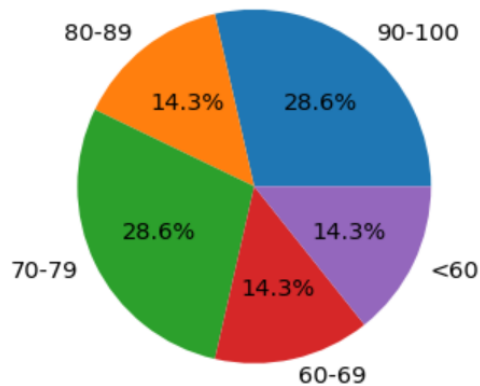
```
1 st.pyplot(fig, use_container_width=False)
2 st.markdown("Biểu đồ phân bố điểm số.")
```

Cách này độ nét của hình ảnh phụ thuộc vào thông số `figsize` và độ phân giải mặc định. Điều này có thể dẫn đến hiện tượng hình ảnh bị mờ khi hiển thị trên giao diện Streamlit, như minh hoạ tại Hình 5a.

Cách 2: Lưu biểu đồ thành ảnh rồi hiển thị.

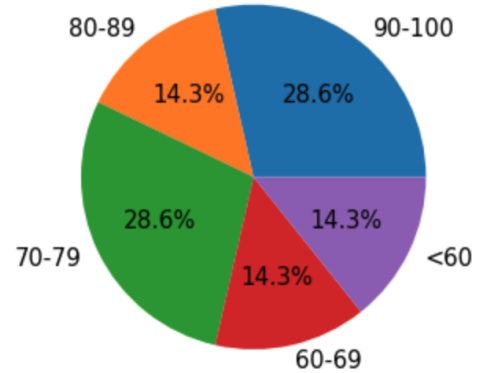
Phương pháp này lưu biểu đồ dưới định dạng PNG với độ phân giải cao (`dpi=300` trước khi hiển thị bằng `st.image`). Điều này giúp đảm bảo hình ảnh rõ nét, đồng thời cho phép kiểm soát chính xác kích thước khi hiển thị (ví dụ, thiết lập `width=250px`). Do đó, cách 2 được khuyến khích sử dụng trong các ứng dụng yêu cầu chất lượng trình bày cao, như minh hoạ tại Hình 5b.

```
1 import io
2 from PIL import Image
3
4 buf = io.BytesIO()
5 fig.savefig(buf, format="png", dpi=300) # Lưu với dpi cao để ảnh sắc nét
6 buf.seek(0)
7
8 st.markdown("Biểu đồ phân bố điểm số.")
9 img = Image.open(buf)
10 st.image(img, width=250) # Hiển thị ảnh với kích thước cố định
```

Biểu đồ phân bố điểm số.

(a) Cách 1



Biểu đồ phân bố điểm số.

(b) Cách 2 (khuyến khích sử dụng)

Hình 5: So sánh hình ảnh trước và sau xử lý.

Thêm ghi chú nhỏ dưới biểu đồ

```
1 st.markdown("Biểu đồ phân bố điểm số.")
```

Đưa biểu đồ vào giữa giao diện:

Để biểu đồ hiển thị ở vị trí trung tâm thay vì căn trái mặc định, ta có thể sử dụng chức năng chia cột trong Streamlit.

Cụ thể, ta tạo ba cột với tỷ lệ `[1, 2, 1]`, trong đó cột giữa có độ rộng lớn hơn để chứa biểu đồ. Sau đó, sử dụng `with col2:` để đặt biểu đồ vào cột giữa, đảm bảo nội dung được căn giữa giao diện.

```
1 col1, col2, col3 = st.columns([1, 2, 1]) # Tạo ba cột, cột giữa rộng hơn
2 with col2:
3     st.image(img, width=300) # Hiển thị biểu đồ ở cột giữa
4     st.markdown("Biểu đồ phân bố điểm số.")
```

Hoàn thiện ứng dụng:

Sau khi hoàn thành các bước trên, ta thu được một tập tin `app.py` hoàn chỉnh, sẵn sàng để chạy trực tiếp trên nền tảng Streamlit.

```
1 import matplotlib.pyplot as plt
2 import pandas as pd
```

```

3 import streamlit as st
4 import io
5 from PIL import Image
6
7
8 # Tiêu đề ứng dụng
9 st.title("Phân tích dữ liệu điểm số học sinh")
10
11 # Upload file
12 uploaded_file = st.file_uploader("Chọn file Excel (có cột \"Điểm số\")", type=["xlsx"])
13
14 # Hàm tính điểm trung bình
15 def calculate_average(scores):
16     return sum(scores) / len(scores)
17
18 # Hàm phân loại điểm số
19 def percentage_distribution(scores):
20     bins = {"90-100": 0, "80-89": 0, "70-79": 0, "60-69": 0, "<60": 0}
21     for score in scores:
22         if score >= 90:
23             bins["90-100"] += 1
24         elif score >= 80:
25             bins["80-89"] += 1
26         elif score >= 70:
27             bins["70-79"] += 1
28         elif score >= 60:
29             bins["60-69"] += 1
30         else:
31             bins["<60"] += 1
32     return bins
33
34 # Khi có file
35 if uploaded_file:
36     df = pd.read_excel(uploaded_file)
37     scores = df["Điểm số"].dropna().astype(float).tolist()
38
39     if scores:
40         st.write("Tổng số học sinh:", len(scores), "Điểm trung bình:", round(
41             calculate_average(scores), 2))
42
43         # Phân loại điểm
44         dist = percentage_distribution(scores)
45         labels = list(dist.keys())
46         values = list(dist.values())
47
48         fig, ax = plt.subplots(figsize=(1, 1))
49         ax.pie(
50             values,
51             labels=labels,
52             autopct="%1.1f%%",
53             textprops={"fontsize": 3.5},
54         )
55         ax.axis("equal")
56         plt.tight_layout(pad=0.1)

```

```

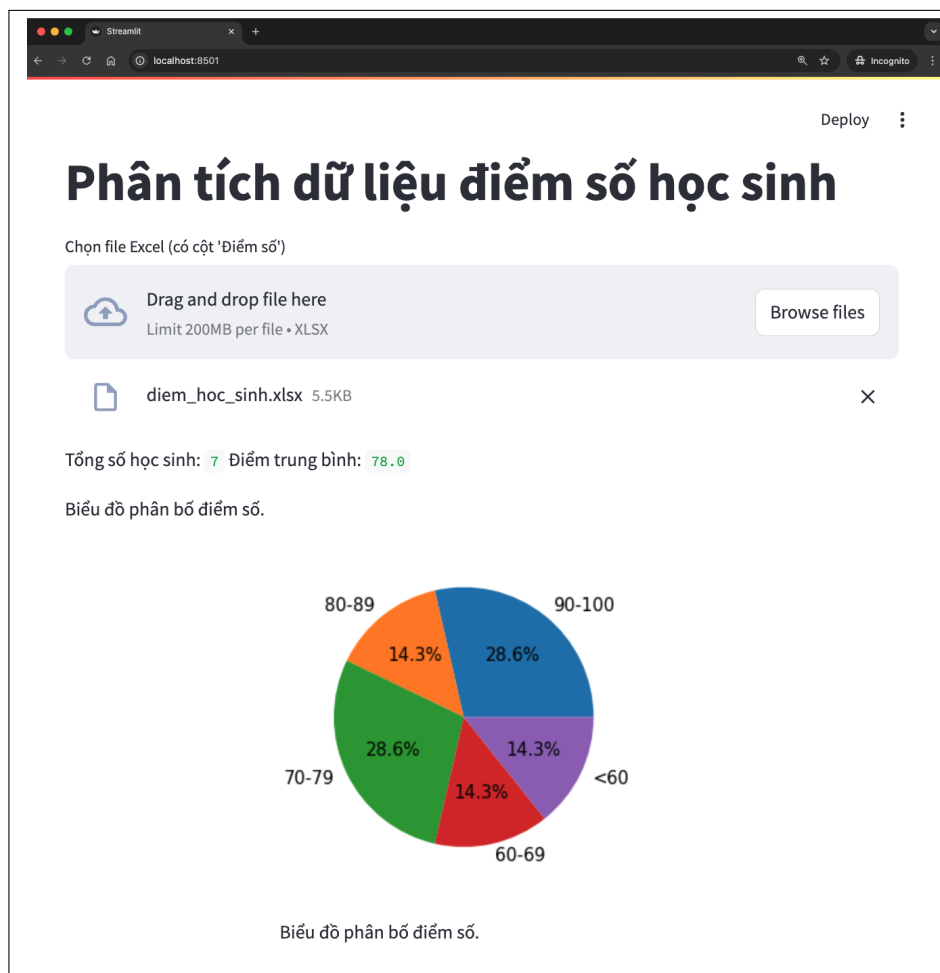
56     buf = io.BytesIO()
57     fig.savefig(buf, format="png", dpi=300)
58     buf.seek(0)
59     st.markdown("Biểu đồ phân bố điểm số.")
60     img = Image.open(buf)
61
62     col1, col2, col3 = st.columns([1, 2, 1]) # Tạo ba cột, cột giữa rộng hơn
63     with col2:
64         st.image(img, width=300) # Hiển thị biểu đồ ở cột giữa
65         st.markdown("Biểu đồ phân bố điểm số.")

```

Sau khi viết xong code (ví dụ lưu thành file `app.py`), bạn chạy ứng dụng bằng

```
streamlit run app.py
```

Sau khi hoàn thành các bước xây dựng, giao diện người dùng (*UI*) của ứng dụng sẽ hiển thị như sau:



Hình 6: Giao diện ứng dụng Streamlit sau khi triển khai.

III. Xây dựng ứng dụng tính hàm toán học giai thừa

Trong phần này, chúng ta sẽ cùng nhau phân tích, thiết kế và triển khai ứng dụng tính giai thừa với streamlit.

III.1. Phân tích, thiết kế ứng dụng

Đầu tiên, cần phải hiểu rõ bài toán và yêu cầu của ứng dụng mà chúng ta sẽ xây dựng. Bước này tuy đơn giản nhưng rất quan trọng, càng mô tả chi tiết, sản phẩm chúng ta tạo ra sẽ càng tốt.

Chẳng hạn, mô tả bài toán có thể như sau: Chúng ta sẽ xây dựng một ứng dụng web để tính giai thừa của một số tự nhiên dành cho học sinh. Ứng dụng này sẽ có giao diện đơn giản, cho phép người dùng nhập số cần tính giai thừa. Sau đó, người dùng sẽ nhấn vào một nút tính toán và ứng dụng sẽ hiển thị kết quả là giai thừa của số tự nhiên mà người dùng đã nhập.

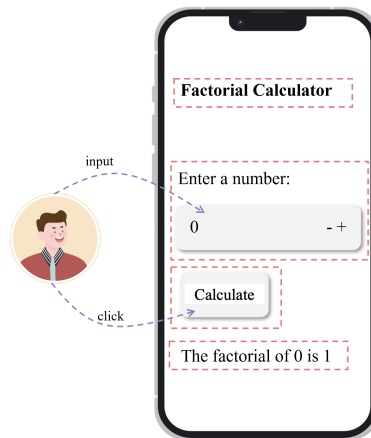
III.1.1. Phân tích yêu cầu bài toán

Với mô tả trên, chúng ta sẽ tiến hành phân tích yêu cầu bài toán này:

- Yêu cầu chức năng:
 - Nhập số tự nhiên: Người dùng có thể nhập vào một số tự nhiên để tính giai thừa.
 - Nút tính toán: Một nút bấm để kích hoạt chức năng tính giai thừa.
 - Hiển thị kết quả: Ứng dụng sẽ hiển thị kết quả giai thừa của số tự nhiên mà người dùng nhập vào.
- Yêu cầu phi chức năng:
 - Giao diện đơn giản và thân thiện: Ứng dụng cần có giao diện đơn giản để học sinh có thể dễ dàng sử dụng.
 - Hiệu suất: Ứng dụng phải tính toán và hiển thị kết quả một cách nhanh chóng.
 - Độ chính xác: Kết quả tính toán phải chính xác tuyệt đối.

III.1.2. Thiết kế ứng dụng

- Giao diện người dùng (UI):
 - Hiển thị tên ứng dụng
 - Nhập số: Một ô nhập văn bản (`st.number_input`) để người dùng nhập số tự nhiên.
 - Nút tính toán: Một nút bấm (`st.button`) để bắt đầu tính toán giai thừa.
 - Hiển thị kết quả: Một vùng để hiển thị kết quả tính toán (`st.write`).
- Luồng hoạt động



Hình 7: Thiết kế giao diện ứng dụng

- Người dùng nhập số tự nhiên vào ô nhập văn bản.
- Người dùng nhấn nút tính toán.
- Ứng dụng tính giai thừa của số đã nhập.
- Ứng dụng hiển thị kết quả lên màn hình.

III.2. Xây dựng ứng dụng

Đầu tiên, chúng ta sẽ tạo một dự án với cấu trúc như sau:

```
1 factorial-app
2 |---- app.py
3 |
4 |---- factorial.py
5 |
6 |---- requirements.txt
```

factorial.py là file chúng ta sẽ viết hàm tính giai thừa, file này là 1 module trong dự án.

```
1 # Factorial function
2 def fact(n):
3     if n == 0 or n == 1:
4         return 1
5     else:
6         return n * fact(n-1)
```

Trong chương trình trên, chúng ta tạo một hàm tên là `fact` với tham số `n`. Hàm này sẽ thực hiện tính giai thừa của số `n` với phương pháp đệ quy.

app.py là file chúng ta sẽ viết giao diện ứng dụng và xử lý tính toán tương tác với người dùng.

```
1 import streamlit as st
2 from factorial import fact
3
4 def main():
5     st.title("Factorial Calculator")
6     number = st.number_input("Enter a number:",
7                               min_value=0,
8                               max_value= 900)
9     if st.button("Calculate"):
10         result = fact(number)
11         st.write(f"The factorial of {number} is {result}")
12
13 if __name__ == "__main__":
14     main()
```

Trong chương trình trên, đầu tiên chúng ta khai báo các thư viện và module. Ở đây chúng ta sử dụng thư viện streamlit và module factorial chúng ta xây dựng phía trên.

Tiếp theo chúng ta xây dựng giao diện ứng dụng, ta sử dụng `st.title` để đặt tiêu đề cho ứng dụng. Sau đó sử dụng `st.number_input` tạo một ô nhập số với nhãn "Enter a number:", giá trị tối thiểu là 0 và giá trị tối đa là 900, vì với thuật toán tính giai thừa ở trên không tính được giai thừa của số lớn hơn 900, bạn có thể thay đổi thuật toán để tính được giai thừa cho các số lớn hơn.

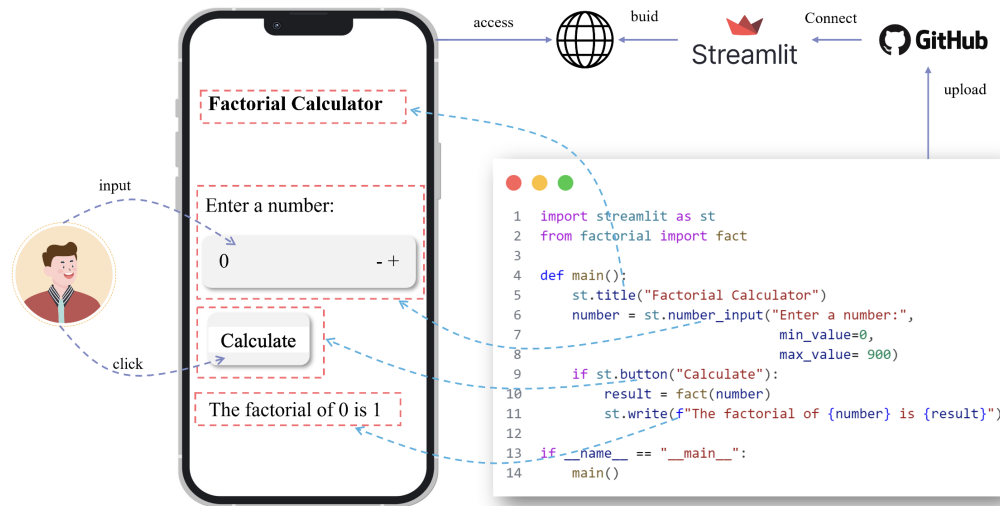
Cuối cùng chúng ta tạo một button caculate, khi sự kiện click vào button này là True thì sẽ thực hiện tính giai thừa cho số người dùng nhập vào và hiển thị kết quả ra màn hình với hàm `st.write()`.

requirement.txt là file chứa các thư viện mà chúng ta cần cài đặt cho dự án.

Đến đây, ứng dụng có thể hoạt động trên máy của chúng ta. Để mở ứng dụng này chúng ta kích hoạt môi trường streamlit_env đã tạo và sử dụng lệnh:

```
streamlit run app.py
```

III.2.1. Triển khai ứng dụng



Hình 8: Triển khai ứng dụng

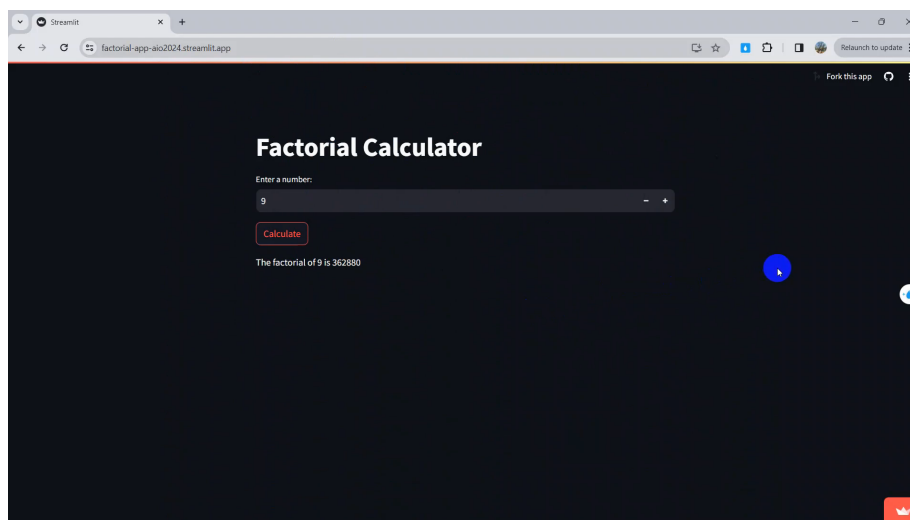
Trong phần này, chúng ta sẽ triển khai ứng dụng trên nền tảng github và streamlit cloud. Đầu tiên chúng ta truy cập vào github và đăng nhập, sau đó chúng ta tạo một repository mới có tên trùng với tên thư mục dự án là factorial-app.

Tiếp theo ta click vào upload an existing file để tải lên toàn bộ source code.

Sau khi upload thành công, chúng ta truy cập vào streamlit và tiến hành đăng nhập.

Tiếp theo chúng ta click vào create new app và thiết lập các thông tin để triển khai ứng dụng này.

Cuối cùng chúng ta click vào deploy và ứng dụng của chúng ta sẽ được triển khai thành công. Ứng dụng có thể truy cập bởi bất kỳ thiết bị nào có kết nối internet, bạn nhấn nút share và copy đường dẫn để chia sẻ ứng dụng của mình đến với bạn bè, người dùng của mình.

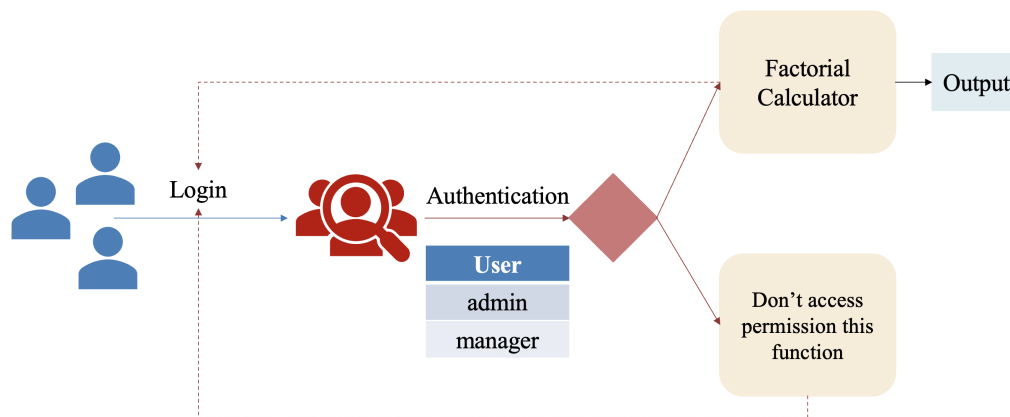


Hình 9: Ứng dụng sau khi triển khai thành công

III.3. Xây dựng tính năng phân quyền và xác thực người dùng

Bên cạnh các tính năng tên, chúng ta có thể xây dựng chức năng đăng nhập, phân quyền:

- Những người có quyền thuộc các tên có trong file user.txt sẽ có quyền truy cập tính năng tính giai thừa
- Những người không thuộc danh sách user thì không được phép truy cập



Hình 10: Xác thực và phân quyền người dùng

Đoạn code bổ sung tính năng sẽ thành:

```

1 import streamlit as st
2 from factorial import fact
3 import os
4
5 def load_users():
6     """Đọc danh sách user từ file user.txt"""
7     try:
8         if os.path.exists("user.txt"):
9             with open("user.txt", "r", encoding="utf-8") as f:
10                 users = [line.strip() for line in f.readlines() if line.strip()]
11                 return users
12         else:
13             st.error("File user.txt không tồn tại!")
14             return []
15     except Exception as e:
16         st.error(f"Lỗi khi đọc file user.txt: {e}")
17         return []
18
19 def login_page():
20     """Trang đăng nhập"""
21     st.title("Đăng nhập")
22
23     # Input username
  
```



```

24     username = st.text_input("Nhập tên người dùng:")
25
26     if st.button("Đăng nhập"):
27         if username:
28             users = load_users()
29             if username in users:
30                 st.session_state.logged_in = True
31                 st.session_state.username = username
32                 st.rerun()
33             else:
34                 # Nếu user không hợp lệ, hiển thị trang chào hỏi
35                 st.session_state.show_greeting = True
36                 st.session_state.username = username
37                 st.rerun()
38         else:
39             st.warning("Vui lòng nhập tên người dùng!")
40
41 def factorial_calculator():
42     """Trang tính giai thừa"""
43     st.title("Factorial Calculator")
44
45     # Hiển thị thông tin user đã đăng nhập
46     st.write(f"Xin chào, {st.session_state.username}!")
47
48     # Nút đăng xuất
49     if st.button("Đăng xuất"):
50         st.session_state.logged_in = False
51         st.session_state.username = ""
52         st.rerun()
53
54     st.divider()
55
56     # Chức năng tính giai thừa
57     number = st.number_input("Nhập vào một số:",
58                             min_value=0,
59                             max_value=900)
60
61     if st.button("Tính giai thừa"):
62         result = fact(number)
63         st.write(f"Giai thừa của {number} là {result}")
64
65 def greeting_page():
66     """Trang chào hỏi cho user không hợp lệ"""
67     st.title("Xin chào!")
68     st.write(f"Xin chào {st.session_state.username}!")
69     st.write("Bạn không có quyền truy cập vào chức năng tính giai thừa.")
70
71     if st.button("Quay lại đăng nhập"):
72         st.session_state.show_greeting = False
73         st.session_state.username = ""
74         st.rerun()
75
76 def main():
77     # Khởi tạo session state

```

```
78     if 'logged_in' not in st.session_state:
79         st.session_state.logged_in = False
80     if 'username' not in st.session_state:
81         st.session_state.username = ""
82     if 'show_greeting' not in st.session_state:
83         st.session_state.show_greeting = False
84
85     # Điều hướng trang dựa trên trạng thái đăng nhập
86     if st.session_state.logged_in:
87         factorial_calculator()
88     elif st.session_state.show_greeting:
89         greeting_page()
90     else:
91         login_page()
92
93 if __name__ == "__main__":
94     main()
```

Giao diện mới sẽ hiển thị như sau sau khi chạy lại app:

Đăng nhập

Nhập tên người dùng:

Đăng nhập

Factorial Calculator

Xin chào, admin!

Đăng xuất

Nhập vào một số:

0

Tính giai thừa

Giai thừa của 0 là 1

Hình 11: Giao diện đăng nhập và tính năng

IV. Câu hỏi trắc nghiệm

1. Hàm nào sau đây được sử dụng để hiển thị chuỗi văn bản trong streamlit.
 - (a) `st.text(...)`
 - (b) `st.image(...)`
 - (c) `st.selectbox(...)`
 - (d) `st.slider(...)`
2. Hàm nào sau đây trong streamlit sử dụng để người dùng nhập văn bản trên giao diện.
 - (a) `st.text(...)`
 - (b) `st.multibox(...)`
 - (c) `st.audio(...)`
 - (d) `st.text_input(...)`
3. Khai báo nào sau đây là sai.
 - (a) `st.image(image_path, caption="A cat", width=100, channels="RGB")`
 - (b) `st.image(image_path, caption="A cat", width=None, channels="BGR")`
 - (c) `st.image(image_path, caption="A cat", width="RGB", channels="BGR")`
 - (d) `st.image(image_path, caption="A cat", width=None, channels="RGB")`
4. Hàm nào sau đây cho phép người dùng tải lên nhiều file.
 - (a) `uploaded_files = st.file_uploader("Choose files", accept_multiple_files=True)`
 - (b) `uploaded_files = st.multifile_uploader("Choose files", accept_multiple_files=True)`
 - (c) `uploaded_files = st.file_uploader("Choose files")`
 - (d) `uploaded_files = st.multifile_uploader("Choose files")`
5. Hàm nào sau đây không được sử dụng để hiển thị code trong streamlit?
 - (a) `st.code(...)`
 - (b) `st.echo(...)`
 - (c) `st.markdown(...)`
 - (d) `st.slider(...)`
6. Dung lượng mặc định tối đa mỗi file được tải lên trong streamlit là bao nhiêu?
 - (a) 100 MB
 - (b) 200 MB
 - (c) 300 MB
 - (d) 400 MB

7. Khi sử dụng `st.sidebar`, điều gì sẽ xảy ra?
- (a) Hiển thị một bảng dữ liệu ở giữa trang
 - (b) Thêm thông báo lỗi trong ứng dụng
 - (c) Hiển thị biểu đồ dạng cột
 - (d) Hiển thị widget trong thanh bên trái (sidebar)
8. Lệnh nào được dùng để chạy một ứng dụng Streamlit từ dòng lệnh?
- (a) `python app.py`
 - (b) `runstreamlit app.py`
 - (c) `streamlit run app.py`
 - (d) `launch streamlit app.py`
9. Nếu bạn muốn hiển thị một biểu đồ Matplotlib hoặc Plotly trong Streamlit, bạn sẽ sử dụng hàm nào?
- (a) `st.image()`
 - (b) `st.dataframe()`
 - (c) `st.pyplot()` hoặc `st.plotly_chart()`
 - (d) `st.audio()`
10. Để tạo một thanh trượt (slider) trong Streamlit, bạn sử dụng hàm nào?
- (a) `st.text_area()`
 - (b) `st.checkbox()`
 - (c) `st.slider()`
 - (d) `st.selectbox()`

Phụ lục

1. **Hint:** Các file code gợi ý có thể được tải [tại đây](#).
2. **Github:** Mã nguồn github để deploy có thể tham khảo thêm [tại đây](#)
3. **Solution:** Các file code cài đặt hoàn chỉnh và phần trả lời nội dung trắc nghiệm có thể được tải [tại đây](#) (**Lưu ý:** Sáng thứ 3 khi hết deadline phần project, ad mới copy các nội dung bài giải nêu trên vào đường dẫn).
4. **Rubric:**

Streamlit - Rubric		
Câu	Kiến Thức	Đánh Giá
1	<ul style="list-style-type: none"> - Làm quen với thư viện streamlit - Hiểu rõ về thiết kế thành phần cơ bản 	<ul style="list-style-type: none"> - Ứng dụng phân tích điểm số học sinh - Ứng dụng tính giai thừa
2	<ul style="list-style-type: none"> - Sử dụng tính năng phân quyền đơn giản - Triển khai ứng dụng trên Cloud 	<ul style="list-style-type: none"> - Xây dựng giao diện với streamlit có tính năng đăng nhập