

**ĐẠI HỌC CÔNG NGHỆ THÔNG TIN**  
**KHOA KHOA HỌC MÁY TÍNH**



**ĐỒ ÁN: OCR ID CARD**

<b>Thành viên:</b> Trần Hoàng Sơn	: 18521351
Nguyễn Trường Thịnh	: 18521447
Nguyễn Văn Thịnh	: 18521448
Phạm Ngọc Trường	: 18521571

**Lớp:** CS406.M11.KHCL

**GVHD:** TS. Mai Tiến Dũng

## Mục lục

<b>I. Bảng phân công .....</b>	<b>4</b>
<b>II. GIỚI THIỆU VỀ ĐỀ TÀI.....</b>	<b>5</b>
1. Lý do chọn đề tài: .....	5
2. Tổng quan về bài toán.....	5
3. Đối tượng nghiên cứu:.....	6
4. Các thách thức:.....	6
5. Các bước triển khai:.....	7
<b>III. Cơ sở lý thuyết.....</b>	<b>8</b>
1. Các phương pháp tiền xử lý ảnh .....	8
1.1. Blur gaussian .....	8
1.2. Blur bilateral.....	9
1.3. Brightness.....	10
1.3.1. Increase brightness :.....	10
1.3.2. Decrease brightness :.....	12
1.4. Contrast.....	12
1.5. Histogram.....	14
1.6. Corner detector .....	17
1.7. Erosion .....	21
1.8. Dilation .....	22
1.9. Opening .....	22
1.10. Closing.....	22
2. Tổng quan các model được sử dụng .....	23
2.1. EasyOCR.....	23
2.2. VietOCR.....	24
2.3. Pyvi .....	26
3. Giải thuật AI .....	26
<b>IV. Xây dựng mô hình .....</b>	<b>27</b>
1. Tổng quan về pipeline .....	27
2. Client (Front-end).....	28



3. Server (Back-end).....	30
3.1. Preprocess image.....	30
3.2. Core OCR .....	32
3.2.1. Tổng quan về pipeline .....	32
3.2.2. Box detection .....	33
3.2.3. Line detection .....	35
3.2.4. Recognize .....	36
3.2.5. Normalize Text.....	37
3.3. Core AI.....	38
3.3.1. Fit Template .....	38
3.3.2. Fit field V1 .....	39
3.3.3. Fit field V2 .....	40
3.4. Nhận xét về core AI.....	42
4. Các trường hợp ngoại lệ .....	43
5. Kết quả quá trình OCR .....	45
V. Frontend React.....	46
1. Tổng quan .....	46
2. Axios.....	46
VI. Backend Flask.....	47
1. Tổng quan .....	47
2. Flask cors.....	47
3. SSL Flask.....	48
3.1. Certificate .....	48
3.2. Decrypts .....	49
4. API .....	50
5. Server Requirements.....	51
VII. Deploys.....	52
VIII. Demo App .....	53
IX. Phương hướng phát triển .....	54
X. Reference .....	54



## I. Bảng phân công

Thông tin thành viên		Công việc
Tên	MSSV	
Phạm Ngọc Trường (Leader)	18521571	<ul style="list-style-type: none"><li>• Xây dựng kiến trúc hệ thống</li><li>• Core AI</li><li>• Flask API</li></ul>
Trần Hoàng Sơn	18521351	<ul style="list-style-type: none"><li>• Reactjs</li><li>• Deep model</li></ul>
Nguyễn Trường Thịnh	18521447	<ul style="list-style-type: none"><li>• Corner detector</li><li>• Một số phép xử lý ảnh</li></ul>
Nguyễn Văn Thịnh	18521448	<ul style="list-style-type: none"><li>• Reactjs</li><li>• Một số phép xử lý ảnh</li></ul>



## II. GIỚI THIỆU VỀ ĐỀ TÀI

### 1. Lý do chọn đề tài:

- Lý do khách quan:
  - OCR (Optical Character Recognition): là quá trình nhận dạng ký tự quang học từ hình ảnh để chuyển các hình ảnh của chữ viết tay hoặc đánh máy thành văn bản tài liệu cho máy tính có thể đọc và hiểu được.
  - Đây là tiền đề cho nhiều bài toán quan trọng trong việc ứng dụng NLP (Natural Language Processing – Xử lý ngôn ngữ tự nhiên) trong thực tế bên cạnh speed-to-text (chuyển đổi ngôn ngữ tiếng nói thành text)
  - Trong khuôn khổ đề án, nội dung đề tài tập chung chủ yếu vào việc trích xuất dữ liệu text từ các loại giấy tờ định danh người dùng như chứng minh nhân dân, căn cước công dân,... qua đó hỗ trợ quá trình làm thủ tục giấy tờ hành chính pháp lý được nhanh chóng.
- Lý do chủ quan:
  - Ứng dụng công nghệ xử lý ảnh vào bước tiền xử lý trước quá trình OCR, nhằm cải thiện hiệu suất cũng như chất lượng của mô hình.
  - Tìm hiểu cách triển khai phần mềm chạy trong thực tế với môi trường phát triển là web, thiết kế theo dạng là client – server, công nghệ sử dụng ở frontend là Reactjs, công nghệ sử dụng ở backend là Flask API.

### 2. Tổng quan về bài toán

- Các dự án OCR open source hiện này phải kể đến tesseractOCR, easyOCR, paddleOCR,... Về ưu nhược điểm thì mô hình nào cũng có, những mô hình training trên nhiều ngôn ngữ thì thường có một ngôn ngữ là core, là mục tiêu chính của mô hình. Điển hình như ở các loại chữ tượng hình như tiếng Trung, tiếng Hàn, tiếng Nhật, paddleOCR là mô hình có độ chính xác cao nhất. Đối với các ngôn ngữ dựa trên ký tự



aphalbet, không bao gồm dấu thanh thì easyOCR lại là mô hình mạnh hơn cả. Riêng đối với tiếng việt, ngôn ngữ mà ông cha ta sáng tạo ra, những mô hình kể trên thường không đạt được độ chính xác cao.

- Qua đó nhóm chúng em đề xuất việc phối hợp nhiều mô hình khác nhau với core dựa trên detect của easyOCR và recognize của vietOCR (mô hình OCR trên tiếng việt mạnh nhất hiện nay). Ở phần sau nhóm chúng em sẽ trình bày rõ hơn về việc kết hợp này.

### **3. Đối tượng nghiên cứu:**

- Ảnh chụp chứng minh nhân dân, căn cước công dân

### **4. Các thách thức:**

- Đối với dataset:
  - Đối tượng nghiên cứu của đề tài là dữ liệu nhạy cảm vì thế nhóm gặp khó khăn trong quá trình crawl data, lượng dữ liệu ít
  - Đa phần các dữ liệu lấy được từ các nguồn trên internet đều là fake data được sinh ra dựa trên một kiến trúc GAN, hoặc các quá trình xử lý/ cắt ghép ảnh của photoshop
- Đối với tài nguyên phục vụ quá trình nghiên cứu:
  - Do bài toán thuộc nhiều lĩnh vực từ Computer Vision, Deep Learning, NLP và một phần AI. Các bài toán thuộc những lĩnh vực này yêu cầu cấu hình phần cứng cao để tăng tốc độ tính toán. Giảm thiểu thời gian chờ và sai sót do hiện tượng crash trong quá trình tính toán.
- Đối với quá trình triển khai thực tế
  - Vì triển khai theo dạng client – server nên kết nối mạng là vấn đề tối quan trọng.
  - Backend sử dụng nhiều model deep learning nên yêu cầu server có hỗ trợ GPU



## 5. Các bước triển khai:

1. Xây dựng pipeline xử lý
2. Thử nghiệm các model deep
3. Xây dựng các module tiền xử lý ảnh
4. Xử lý logic cho AI
5. Apply model deep có tích hợp AI
6. Xây dựng API qua Flask
7. Xử lý lỗi về HTTP protocol, cũng như các trường hợp exception trong logic
8. Xây dựng app React
9. Responsive và thiết kế animation cho app
10. Connect Flask server với React
11. Thử nghiệm hàng loạt



### III. Cơ sở lý thuyết

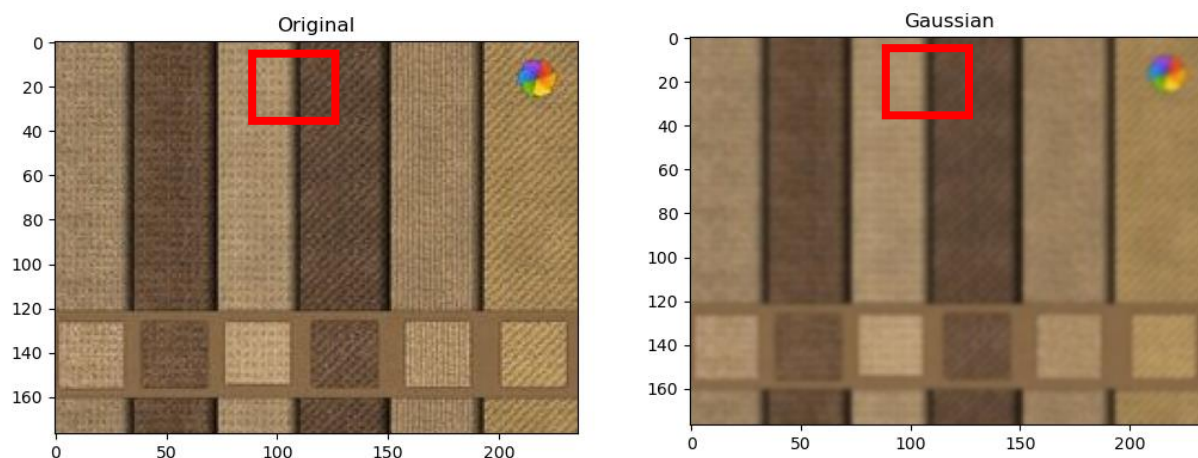
#### 1. Các phương pháp tiền xử lý ảnh

##### 1.1. Blur gaussian

- Blur là quá trình làm mịn ảnh, khử nhiễu. Nguyên tắc chung của quá trình này là nhân ma trận ảnh với một ma trận lọc (kernel)
- Blur gaussian là bộ lọc đơn giản nhất
- Đây là một kernel của gaussian

$$\frac{1}{16} \begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$$

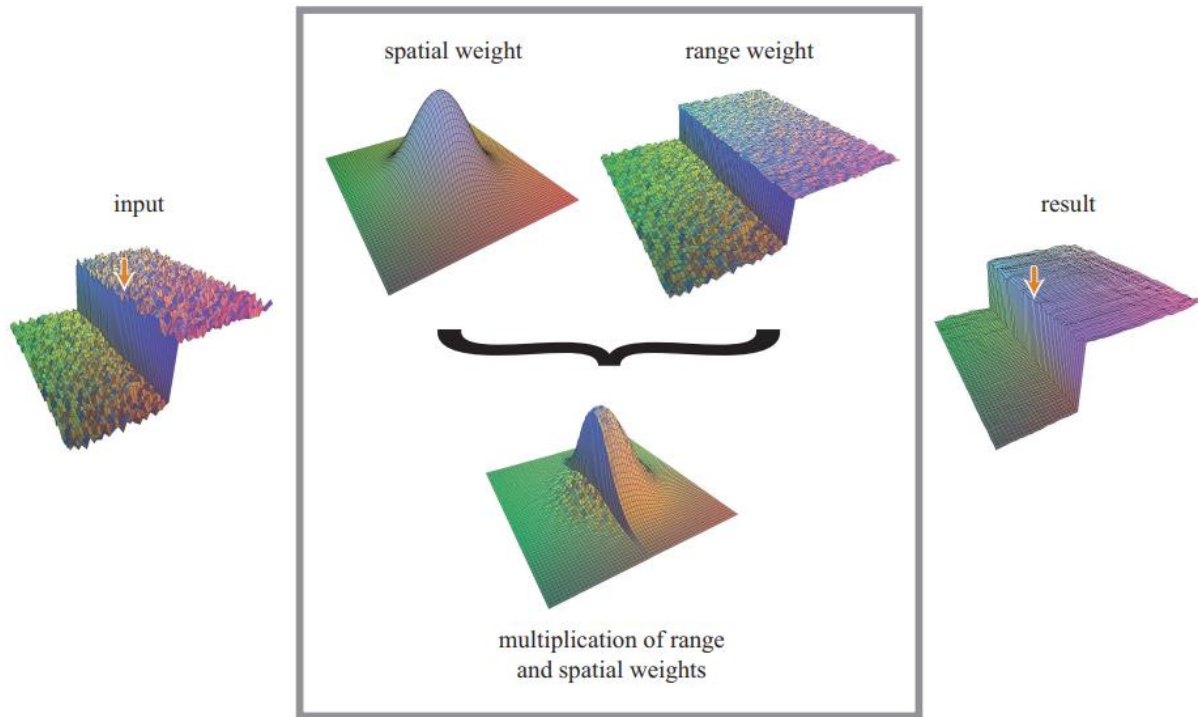
- Kết quả ảnh sau khi trải qua quá trình blur gaussian: các pixel ảnh trong khu vực được khoanh vùng đỏ giảm hiện tượng bị rỗ một các rõ rệt, song phần viền cạnh lại bị làm mờ



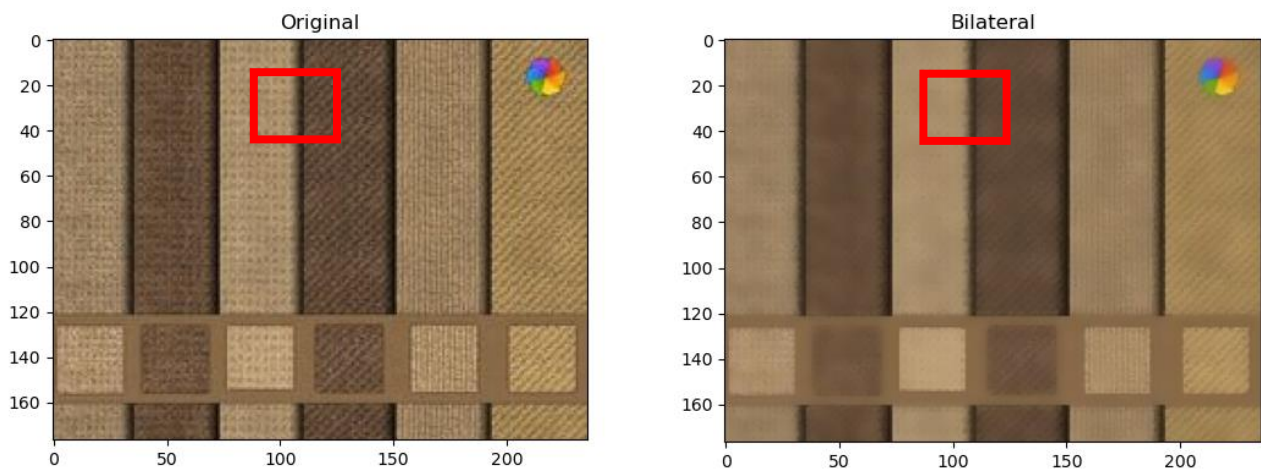


## 1.2. Blur bilateral

- Tương tự Blur gaussian: bilateral cũng sử dụng một ma trận lọc như gaussian.
- So với gaussian, bilateral chỉ tính toán trên các điểm ảnh cùng mức năng lượng, qua đó đảm bảo được độ dốc của biên cạnh sau quá trình tính toán



- Kết quả của blur bilateral: các điểm ảnh nhiễu được làm mờ song vùng biên cạnh vẫn được đảm bảo

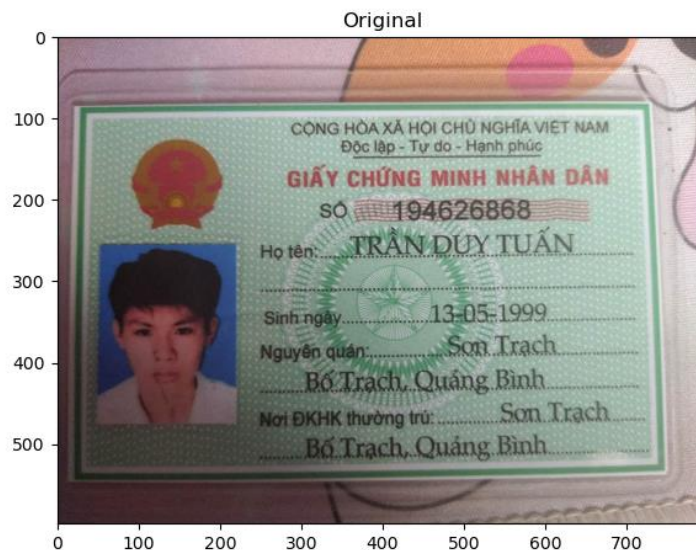


### 1.3. Brightness

- Brightness là quá trình xử lý độ sáng cho hình ảnh. Tùy theo điều kiện ảnh đầu vào mà ta tăng hay giảm độ sáng cho phù hợp.

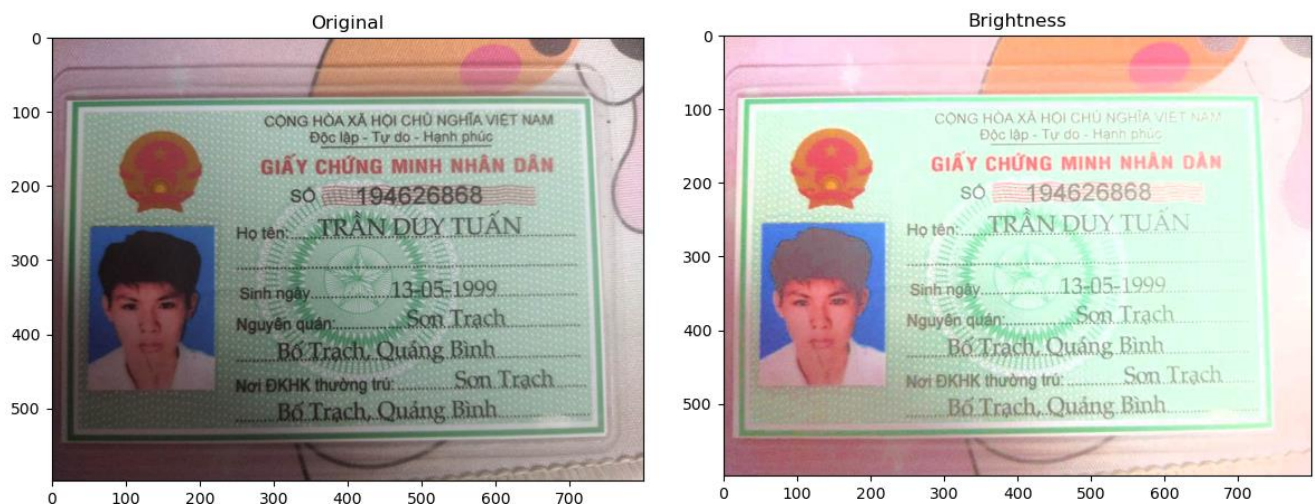
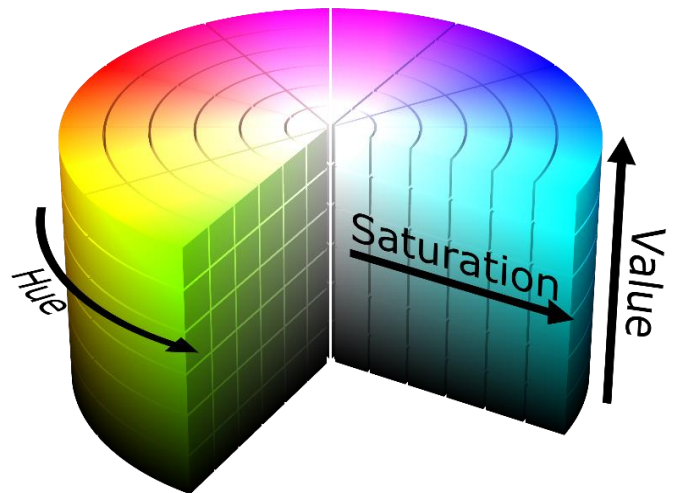
#### 1.3.1. Increase brightness :

- Thực hiện khi ảnh đầu vào quá tối, chụp trong điều kiện thiếu sáng
- Ví dụ về ảnh thiếu sáng :



- Cách thực hiện: để tăng sáng cho ảnh mức xám, đơn giản ta chỉ cần cộng thêm một giá trị cho mọi pixel. 
$$g(x,y) = f(x,y) + \text{const} \quad \text{với } \text{const} > 0$$

- Đối với ảnh màu, ta thực hiện tương tự như ảnh mức xám nhưng trên từng kênh màu riêng biệt.
- Ở đây nhóm chúng em đề xuất đến việc sử dụng kênh màu HSV để bảo toàn sắc thái màu của ảnh đầu vào.
- Trong đó :
  - H (hue) là vùng sắc thái màu
  - S (saturation) là độ bão hòa màu
  - V (value) là độ sáng
- Thông thường với các kênh màu RGB, việc tăng sáng là quá trình công thêm một const cho lần lượt cả 3 kênh R, G, B. Song với việc thực hiện tăng sáng trên kênh HSV, ta chỉ việc tăng giá trị cho kênh V (value) để đạt được hiệu quả như mong muốn.
- Kết quả thực nghiệm: tùy theo điều kiện ảnh đầu vào, mà ta điều chỉnh tham số cho phù hợp.





### 1.3.2. Decrease brightness :

- Thực hiện khi ảnh đầu vào bị chói sáng, nhiều nguồn sáng mạnh.
- **Trong thực tế**, ít khi ta thực hiện thao tác này.
- Cách thực hiện: tương tự với Increase brightness

$$g(x, y) = f(x, y) + \text{const} \quad \text{với } \text{const} < 0$$

- Kết quả thực nghiệm: tùy theo điều kiện ảnh đầu vào, mà ta điều chỉnh tham số cho phù hợp.



### 1.4. Contrast

- Contrast hay còn gọi là độ tương phản, nó thể hiện sự chênh lệch độ sáng giữa các vùng sáng tối khác nhau trong ảnh.
- Cách thực hiện:  $g(x, y) = f(x, y) * \text{contrast}$  với contrast là hệ số tương phản
- Trong thực tế, việc tăng giảm độ tương phản thường đi kèm với việc tăng giảm độ sáng để đạt được hiệu quả tốt nhất, lúc này, cách thực hiện sẽ là:

$$g(x, y) = (f(x, y) * \text{contrast}) + \text{const}$$

- Với contrast và const có xu hướng ngược nhau :

- Khi  $\text{contrast} > 1$  thì  $\text{const} < 0$
- Khi  $\text{contrast} < 1$  thì  $\text{const} > 0$

- Ví dụ khi ta có matrix :  $f(x, y) = \begin{pmatrix} 50 & 90 \\ 50 & 90 \end{pmatrix}$

⇒ Mức độ tương phản hiện tại giữa 2 pixel liên kề là  $90 - 50 = 40$



- Với  $\text{contrast} = 2$  và  $\text{const} = -50$

$$g(x, y) = \begin{pmatrix} 50 * 2 - 50 & 90 * 2 - 50 \\ 50 * 2 - 50 & 90 * 2 - 50 \end{pmatrix} = \begin{pmatrix} 50 & 130 \\ 50 & 130 \end{pmatrix}$$

⇒ Mức độ tương phản giữa 2 pixel liền kề lúc này là  $130 - 50 = 80$

- Kết quả thực nghiệm :
  - Trường hợp  $\text{contrast} > 1$  và  $\text{const} < 0$

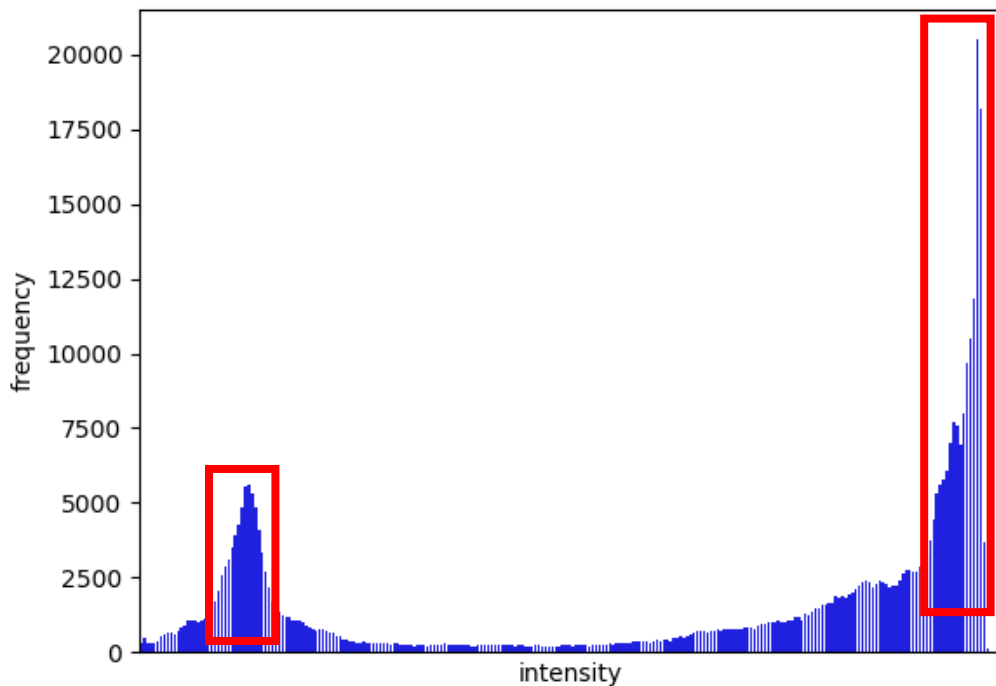


- Trường hợp  $\text{contrast} < 1$  và  $\text{const} > 0$

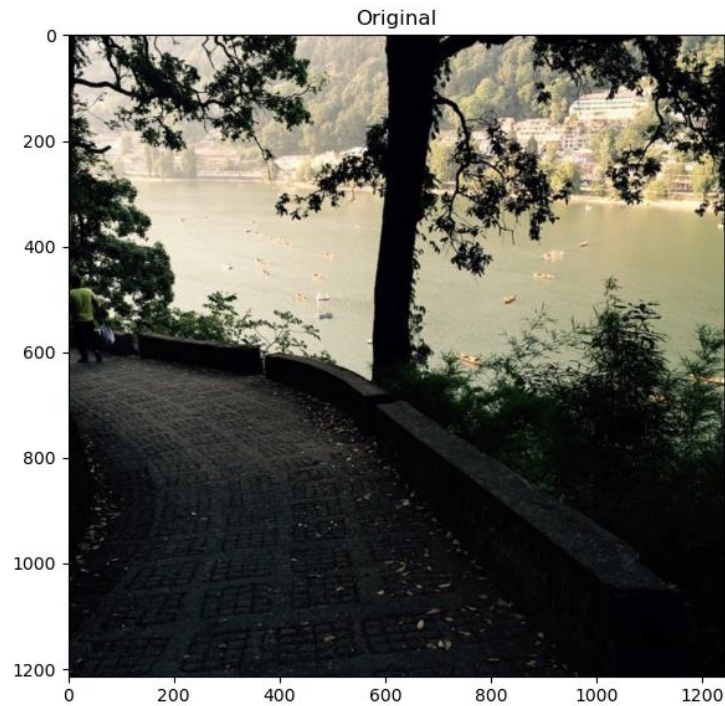


## 1.5. Histogram

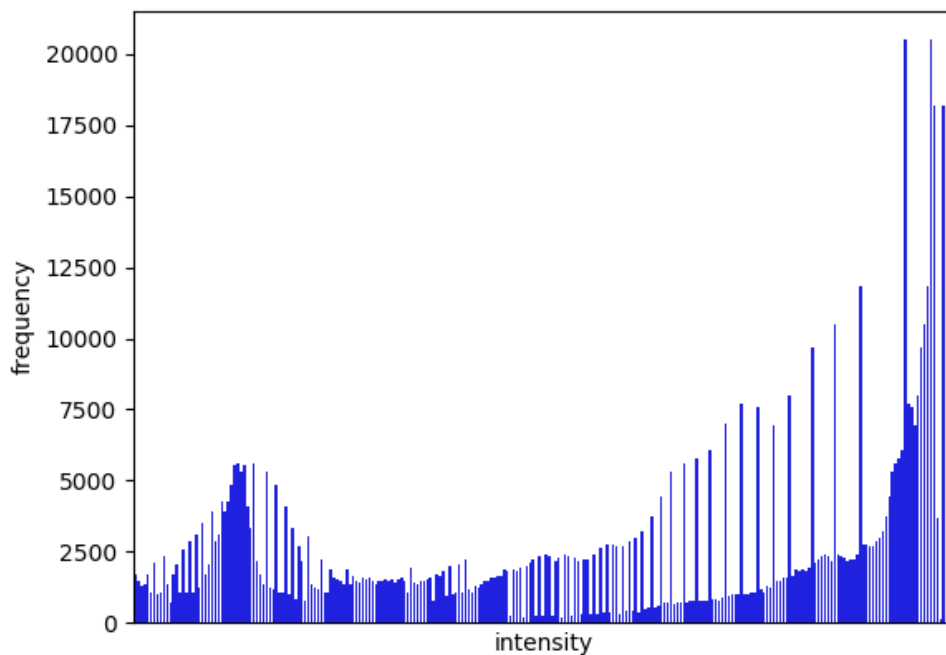
- Histogram là biểu đồ phân bố mật độ dữ liệu, cụ thể ở đây là cường độ của mỗi pixel ảnh (0 đến 255)
- Dựa vào biểu đồ histogram ta có thể nhận định được thiên hướng của bức ảnh, các vùng sáng phân bố như thế nào để điều chỉnh mang lại sự hài hòa cho bức ảnh
- Cân bằng histogram là quá trình phân bố lại cường độ của các pixel để tạo nên sự hài hòa trong bức ảnh
- Ví dụ về một biểu đồ histogram có xu hướng tập trung vào 2 cụm giá trị:



- Còn đây là bức ảnh tương ứng với biểu đồ histogram bên trên: các vùng ảnh bị phân chia rõ rệt, thiếu đi sự hài hòa.

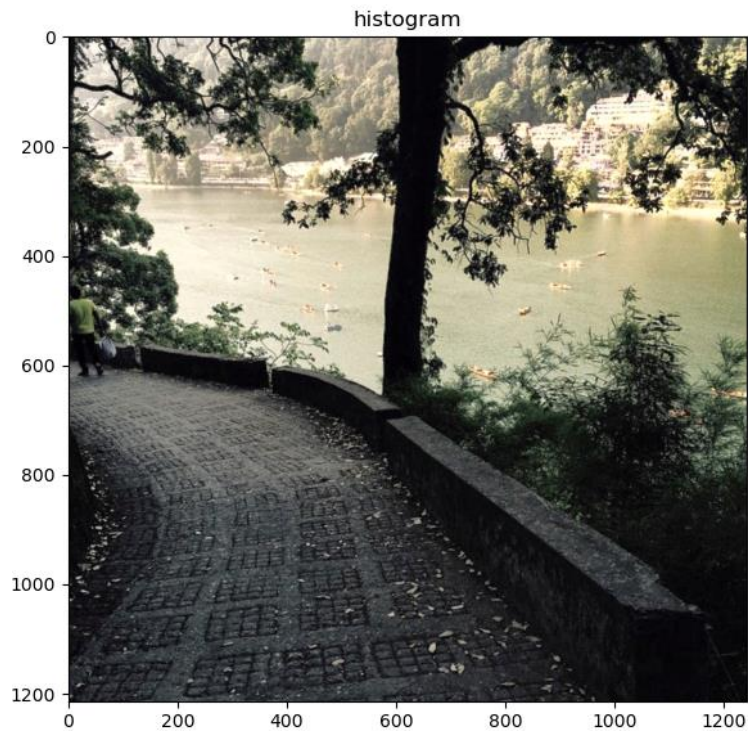


- Biểu đồ histogram sau khi được cân bằng: cường độ của các pixel không còn tập trung như trước, mà được phân bố rải đều khắp các giá trị

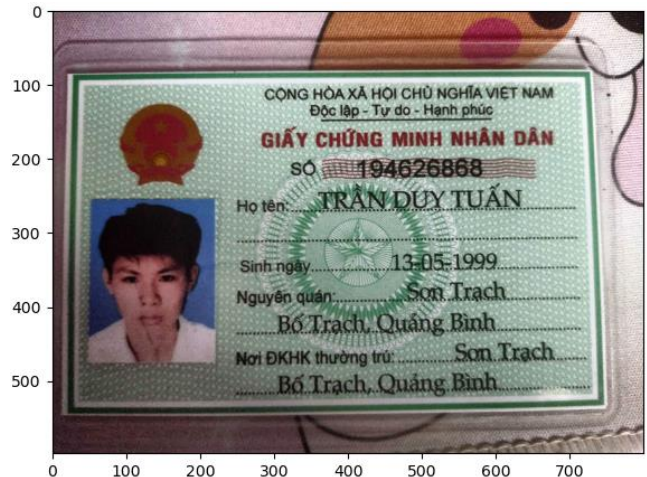




- Kết quả bức ảnh sau khi được cân bằng histogram



- Thực nghiệm thực tế cho thấy, histogram là giải pháp tốt để xử lý các trường hợp ảnh chụp bị chói sáng từ đèn flash của điện thoại
- Ví dụ thực tế khi cân bằng histogram trên ảnh cmnd chụp từ điện thoại.





## 1.6. Corner detector

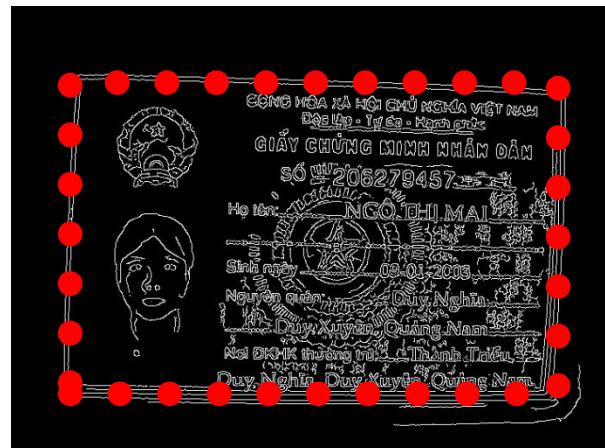
- Là quá trình dò tìm các góc tồn tại trong hình ảnh



- Dựa vào vị trí 4 góc tìm được, mà ta tiến hành các phép biến đổi homography để thu được hình mới chỉ chứa CMND/CCCD.



- Các bước thực hiện:
  - Bước 1: Edge detection.
    - Thông qua các đặc trưng về cạnh, ta tìm được tập hợp các điểm contour (đường viền)



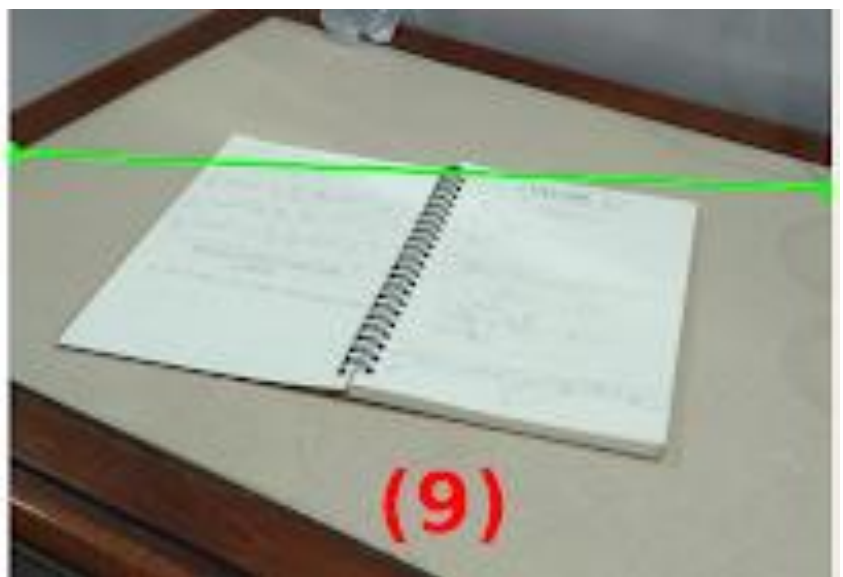
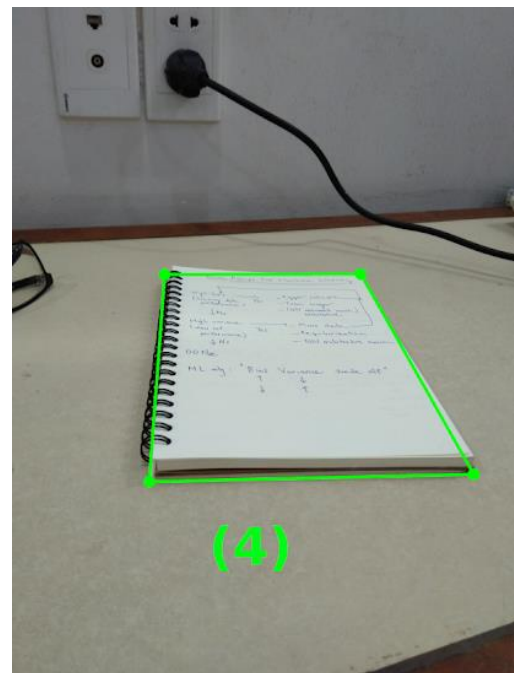
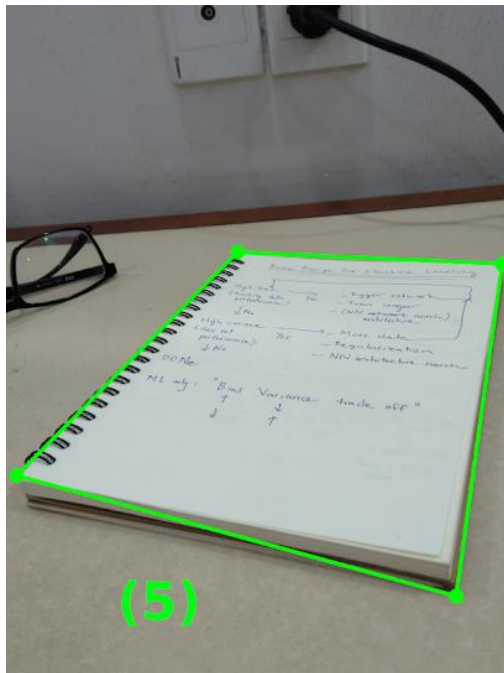
- Bước 2: Rút gọn contour
  - Ta chỉ giữ lại các góc của contour



- Trong trường hợp có nhiều hơn 1 tập hợp contour. Ta chỉ xét đến contour có diện tích lớn nhất và có 4 đỉnh. Lý do ta chỉ xét đến các contour có 4 đỉnh là vì input của bài toán chỉ bao gồm ảnh CMND/CCCD, và các loại giấy tờ này thì có hình chữ nhật và có 4 đỉnh.
- Bước 3 : Sắp xếp lại danh sách đỉnh contour theo vị trí của hệ trục tọa độ
  - Dựa vào danh sách đỉnh contour này, ta tiến hành bounding box và affine transform.



- Dưới đây là một số ví dụ khi chạy thực tế:



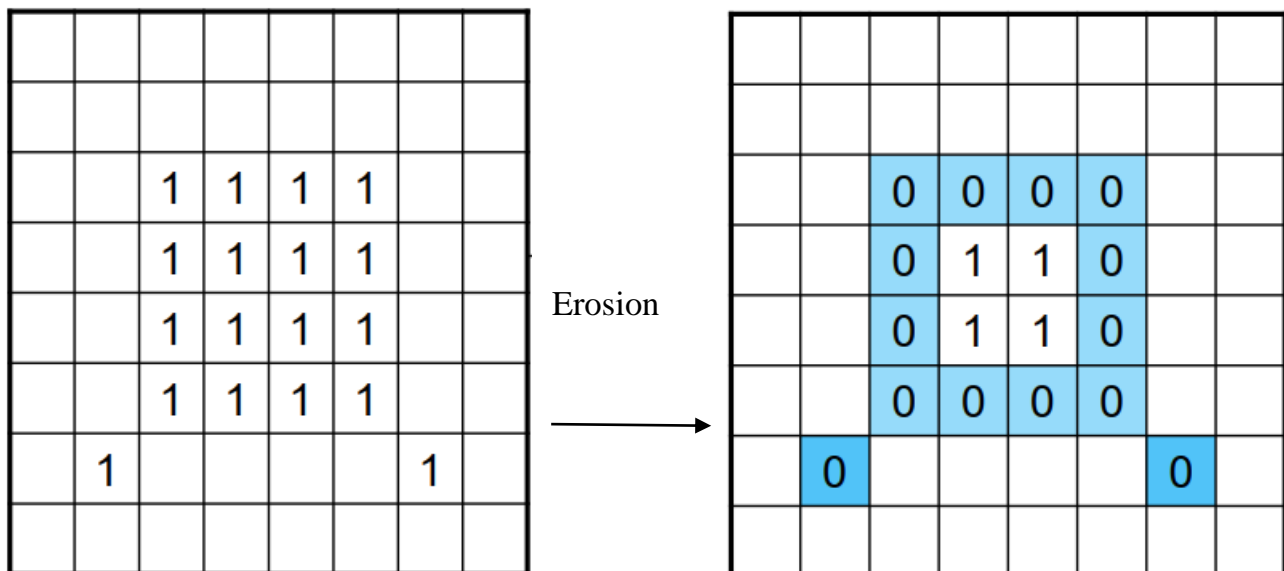
- Nhận xét:
  - Phương pháp này hoạt động chưa thực sự hiệu quả.
  - Thiếu khả năng nội suy khi một góc bị che khuất.
  - Dễ bị nhiễu khi background phức tạp.



## \* Các phép xử lý từ 1.7 trở đi chỉ hoạt động đúng với ảnh nhị phân

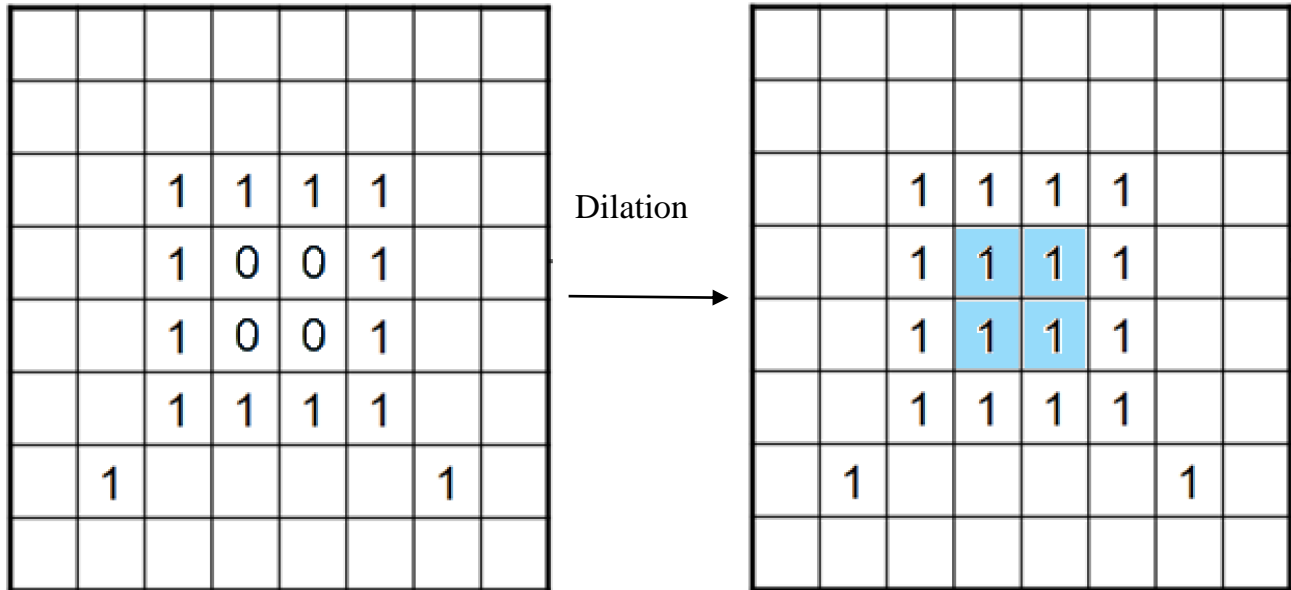
### 1.7. Erosion

- Erosion là quá trình loại bỏ bớt các pixel rời rạc khỏi đối tượng.
- Cách thực hiện: nếu pixel trung tâm là 1, và có ít nhất 1 pixel lân cận là 0 (nhưng không đồng thời bằng 0) thì gán pixel trung tâm bằng 0. Mục đích của việc kiểm tra không đồng thời bằng 0 nhằm mục đích tránh việc xóa nhầm đối tượng
- Ví dụ trên matrix :



## 1.8. Dilation

- Dilation là phép toán ngược với Erosion.
- Cách thực hiện: nếu pixel trung tâm là 0 và có ít nhất 1 pixel lân cận là 1 thì pixel trung tâm được gán bằng 1.
- Ví dụ trên matrix :



## 1.9. Opening

- Là sự kết hợp giữa 1 phép Erosion, sau đó thực hiện tiếp 1 phép Dilation
- Mục đích: khử nhiễu, làm mịn ảnh

## 1.10. Closing

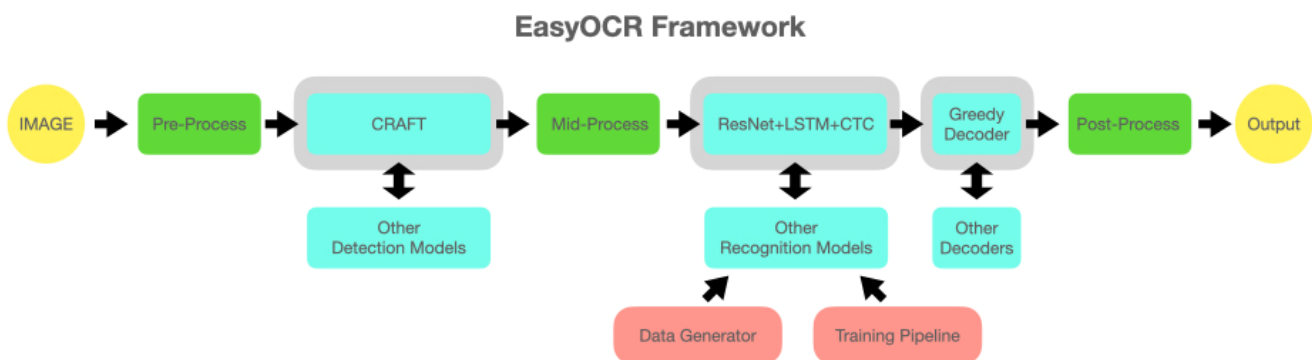
- Là sự kết hợp giữa 1 phép Dilation, sau đó thực hiện tiếp 1 phép Erosion
- Mục đích: lấp đầy khoảng trống, vùng ảnh bị thiếu

## 2. Tổng quan các model được sử dụng

- Trong khuôn khổ nội dung môn học, các model deep dưới đây đều sử dụng pretrain model, không đi sâu vào tìm hiểu kiến trúc neural networks

### 2.1. EasyOCR

- EasyOCR là dự án AI đầu tiên của JaidevAI – tập đoàn công nghệ được thành lập nhằm mục đích chuyên biệt để phát triển công nghệ AI.
- EasyOCR được sinh ra nhằm mục đích giải quyết bài toán OCR đa ngôn ngữ, đa nền tảng, là bước đệm để phát triển trí thông minh nhân tạo. Nó bắt đầu được chạy chính thức từ tháng 5 năm 2021, hỗ trợ trên hơn 80 ngôn ngữ (bao gồm cả tiếng việt).
- Model được cung cấp mã nguồn mở trên github trong link đính kèm trong phần tài liệu tham khảo.
- Các đối tác hiện tại của JaidevAI bao gồm nhiều ông lớn trong lĩnh vực công nghệ như Google, Facebook, Microsoft, AWS, Tencent, Baidu, Alibaba,...
- EasyOCR được xem như là core (nền tảng) cho nhiều hệ thống thông minh phải kể đến như google lens, các loại kính thông minh, hỗ trợ trong tìm kiếm thông tin từ hình ảnh,.....
- Dưới đây là kiến trúc model:



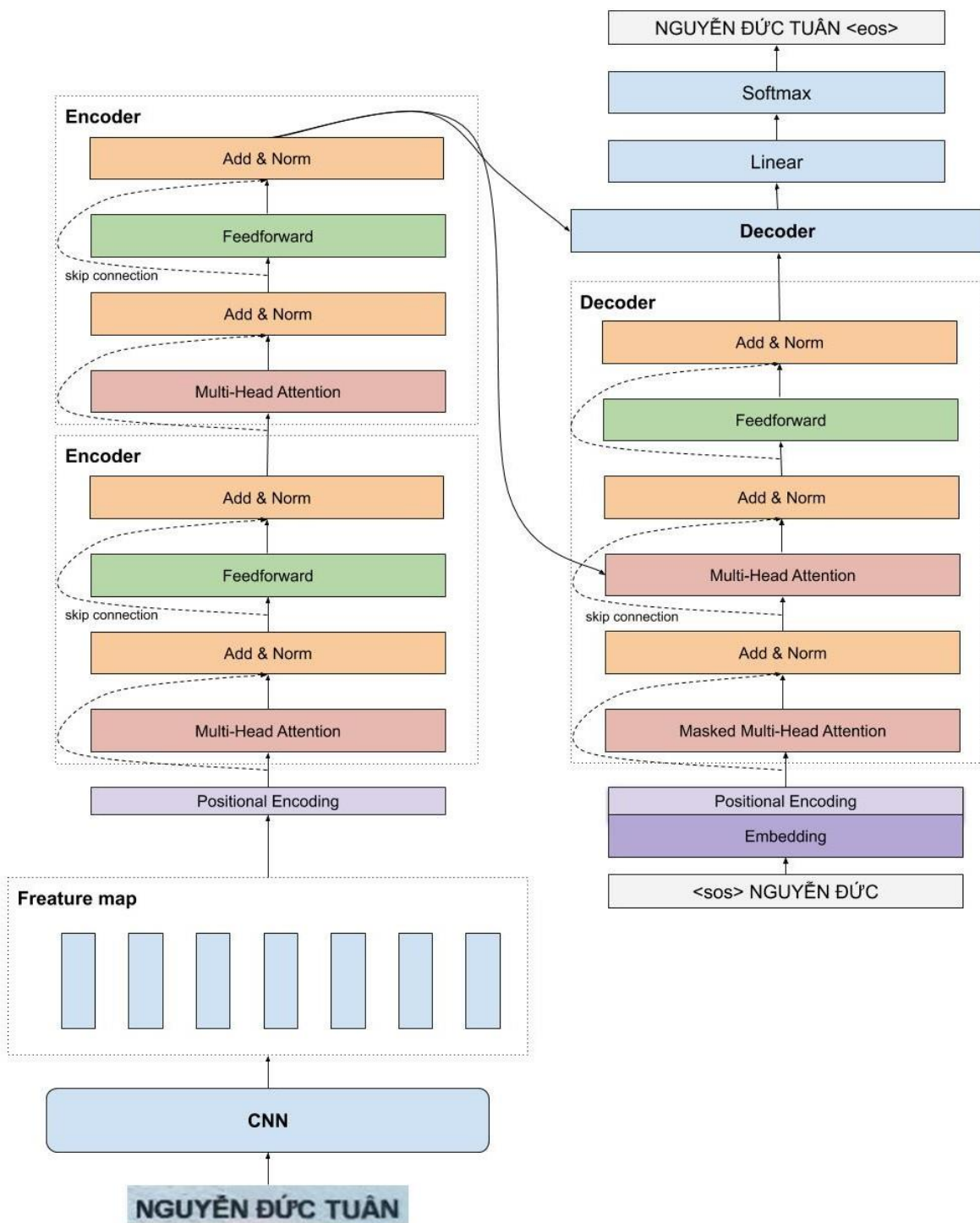
## 2.2. VietOCR

- VietOCR là một mã nguồn mở, phục vụ OCR dành cho tiếng việt, do người việt phát triển. Vì sự chuyên biệt trên, nên nếu so sánh với EasyOCR trong bài toán OCR dành cho tiếng việt, thì VietOCR nhỉnh hơn hoàn toàn.
- Mô hình được sử dụng trong đề án là bản pretrain model dựa trên weight transformer, được training trên tập dữ liệu 10m ảnh, bao gồm chữ in, chữ viết tay, các bản scan thực tế, một phần dữ liệu tự phát sinh => Mô hình có độ tổng quát cao.





- Dưới đây là kiến trúc model của VietOCR:



### 2.3. Pyvi

- Pyvi là một mô hình NLP (Natural Language Processing), là một Vietnamese toolkit được sử dụng trong quá trình preprocessing data cho nhiều bài toán liên quan đến ngôn ngữ tự nhiên.
- Các chức năng chính của Pyvi bao gồm:
  - Tokenization
  - Pos tagging
  - Accents removal
  - Accents adding
- Đây đều là những bài toán căn bản, là nền tảng cho NLP

### 3. Giải thuật AI

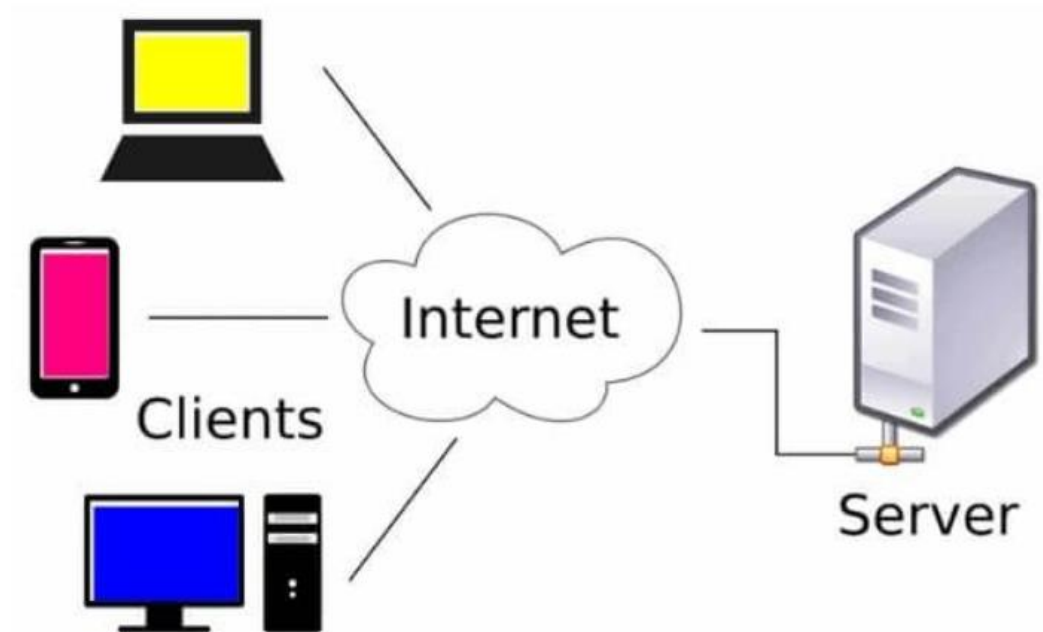
- Trong quá trình tổng quát hóa bài toán, rút trích các trường thông tin đặc trưng từ CMND, CCCD, nhóm chúng em có tìm hiểu về Levenshtein Distance và tích hợp nó vào luồng xử lý.
- Levenshtein Distance là một bộ đo so khớp giữa 2 chuỗi, khoảng cách Levenshtein được định nghĩa là số bước cần thiết để biến chuỗi A thành chuỗi B. Khoảng cách này càng bé, đồng nghĩa với việc độ tương đồng của 2 chuỗi này càng lớn
- Cụ thể về cách sử dụng bộ đo này sẽ được đề cập đến trong phần xây dựng mô hình



## IV. Xây dựng mô hình

### 1. Tổng quan về pipeline

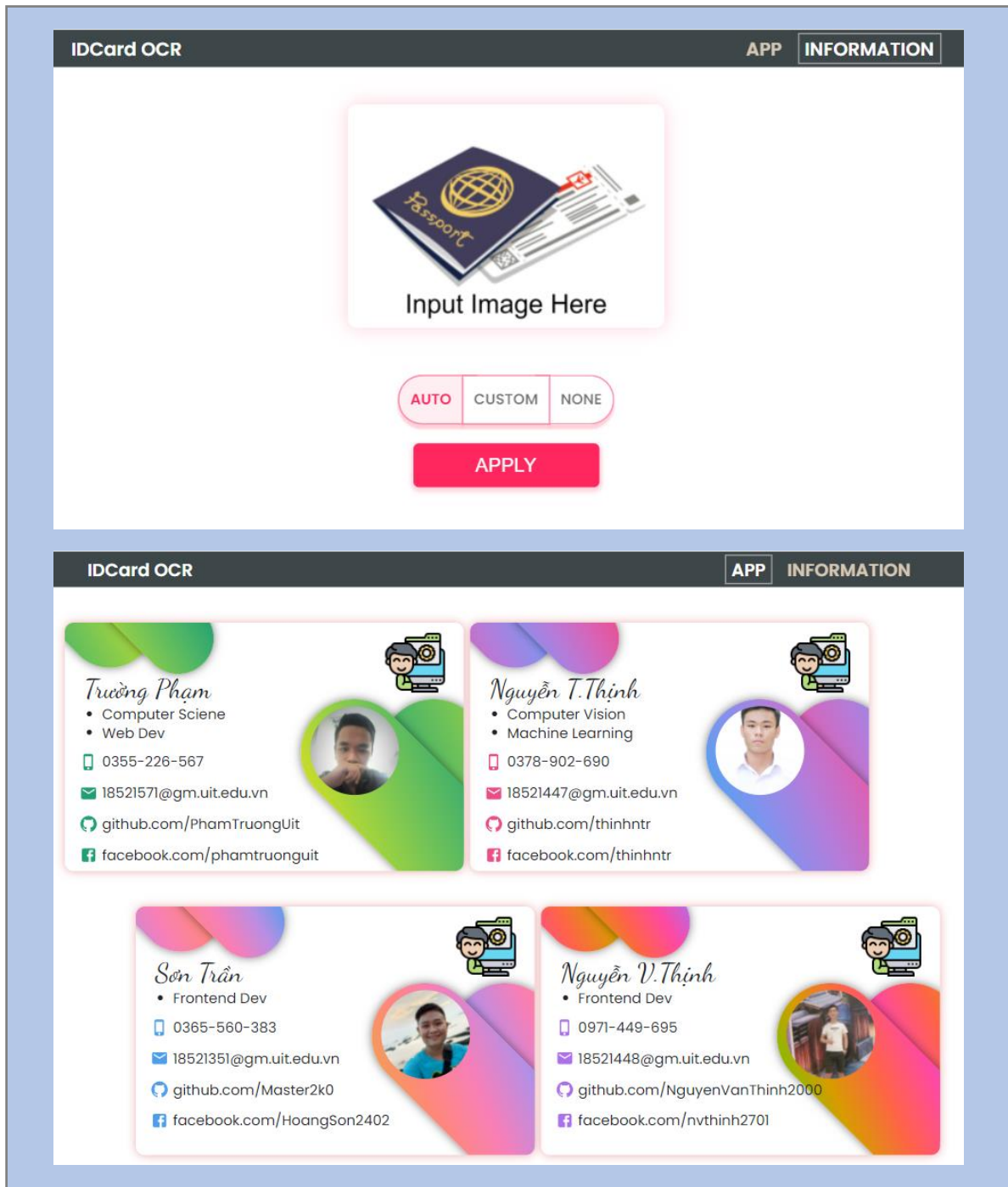
- Kiến trúc được sử dụng trong đồ án này là mô hình client – server
- Client chịu trách nhiệm:
  - Xử lý dữ liệu phía người dùng
  - Hỗ trợ giao diện thao tác trực quan
  - Hỗ trợ đa nền tảng OS, do được triển khai trên web
  - Truyền dữ liệu từ phía người dùng lên server, và hỗ trợ hiển thị kết quả trả về
- Server chịu trách nhiệm:
  - Nhận request từ client
  - Xử lý dữ liệu
  - Gửi response về client sau quá trình xử lý



## 2. Client (Front-end)

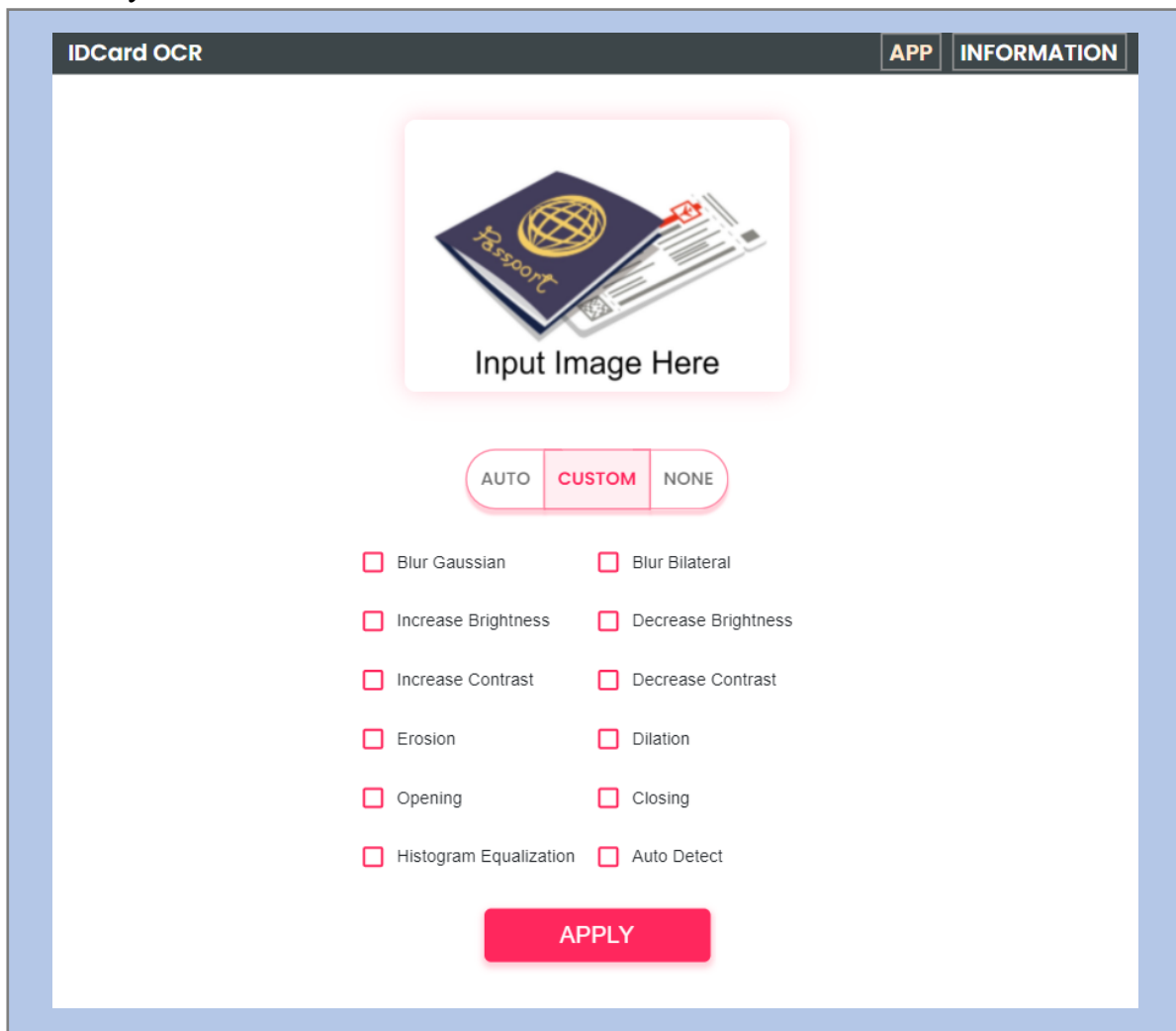
### 2.1. Xây dựng giao diện người dùng

- Dưới đây là giao diện APP
- Ứng dụng đã bao gồm xử lý responsive css trên đa thiết bị, đa nền tảng
- Ngoài ra web còn có nhiều animation nhằm cải thiện trải nghiệm người dùng



## 2.2. Xử lý sự kiện phía user

- Chức năng upload ảnh
- Chức năng tùy chỉnh tiền xử lý:
  - Auto
  - Custom
  - None
- Với chức năng custom, sẽ cho phép người dùng chọn 1 hoặc nhiều trong 12 phép xử lý ảnh



The screenshot displays the 'IDCard OCR' application window. At the top, there are tabs for 'APP' and 'INFORMATION'. The main area features a placeholder for an image with the text 'Input Image Here' and an illustration of a passport and ID card. Below this, there are three buttons: 'AUTO', 'CUSTOM' (which is highlighted in red), and 'NONE'. Under the 'CUSTOM' button, there is a grid of 12 checkboxes for various image processing techniques: 'Blur Gaussian', 'Blur Bilateral', 'Increase Brightness', 'Decrease Brightness', 'Increase Contrast', 'Decrease Contrast', 'Erosion', 'Dilation', 'Opening', 'Closing', 'Histogram Equalization', and 'Auto Detect'. At the bottom of the interface is a large red 'APPLY' button.

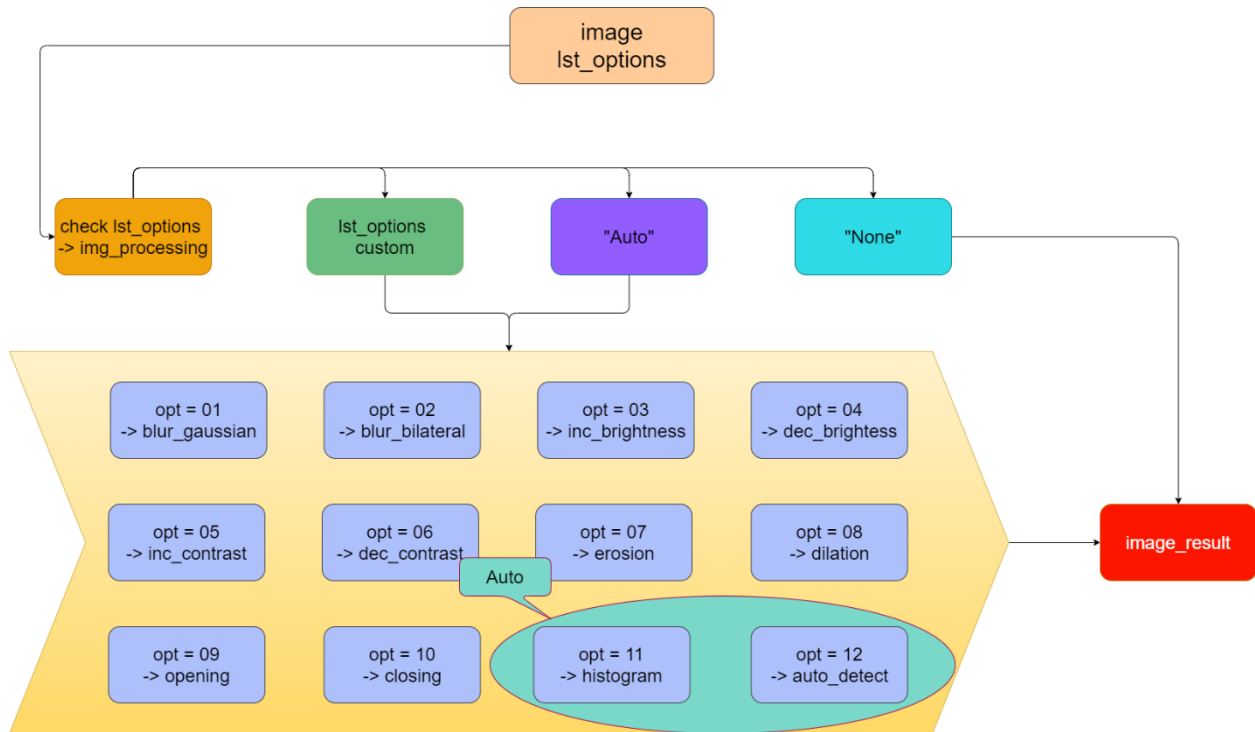
## 2.3. Xử lý sự kiện phía server

- Sau khi nhận được response từ server, tiến hành hiển thị kết quả ra cho người dùng



### 3. Server (Back-end)

#### 3.1. Preprocess image



- Server nhận request bao gồm 1 image định dạng base64 và 1 array lst\_options chứa các tùy chọn do người dùng custom từ phía client
- Quá trình tiền xử lý được chia thành các bước chính:
  - Bước 1 : Convert image base64 về image np.array
  - Bước 2 : Đọc lst\_options để lấy được custom từ phía user, tùy theo custom mà chia luồng xử lý riêng biệt
  - Bước 3 : Tiếp tục quay lại bước 2 cho đến khi lst\_options đã được duyệt qua hết
  - Bước 4 : Truyền image sau xử lý sang các mô hình deep để tiếp tục xử lý.
- Các bước tiền xử lý ảnh được đề cập đến ở chương III mục 1 được viết thành các module riêng biệt, phục vụ cho việc call trong quá trình này.

```
def preprocess(image, lst=[]):
    if lst:
        if lst[0] == 'none':
            None
        elif lst[0] == 'auto':
            image = getattr(libs, "histogram")(image)
            image = getattr(libs, "auto_detect")(image)
        else:
            # image processing
            for attr in lst:
                try:
                    image = getattr(libs, attr)(image)
                except:
                    None
    return image
```

- Với lst\_options == "none" : image sẽ được giữ nguyên
- Với lst\_options == "auto" : image sẽ được xử lý qua histogram và auto\_detect
- Trường hợp lst\_options không chứa 2 tùy chọn trên, ta lần duyệt qua các bước xử lý được đề cập đến trong lst\_options.
- Để tiện cho quá trình giao tiếp giữa client và server, các tùy chọn xử lý ảnh sẽ được mã hóa thành id và được cài đặt trong tệp config.ini.

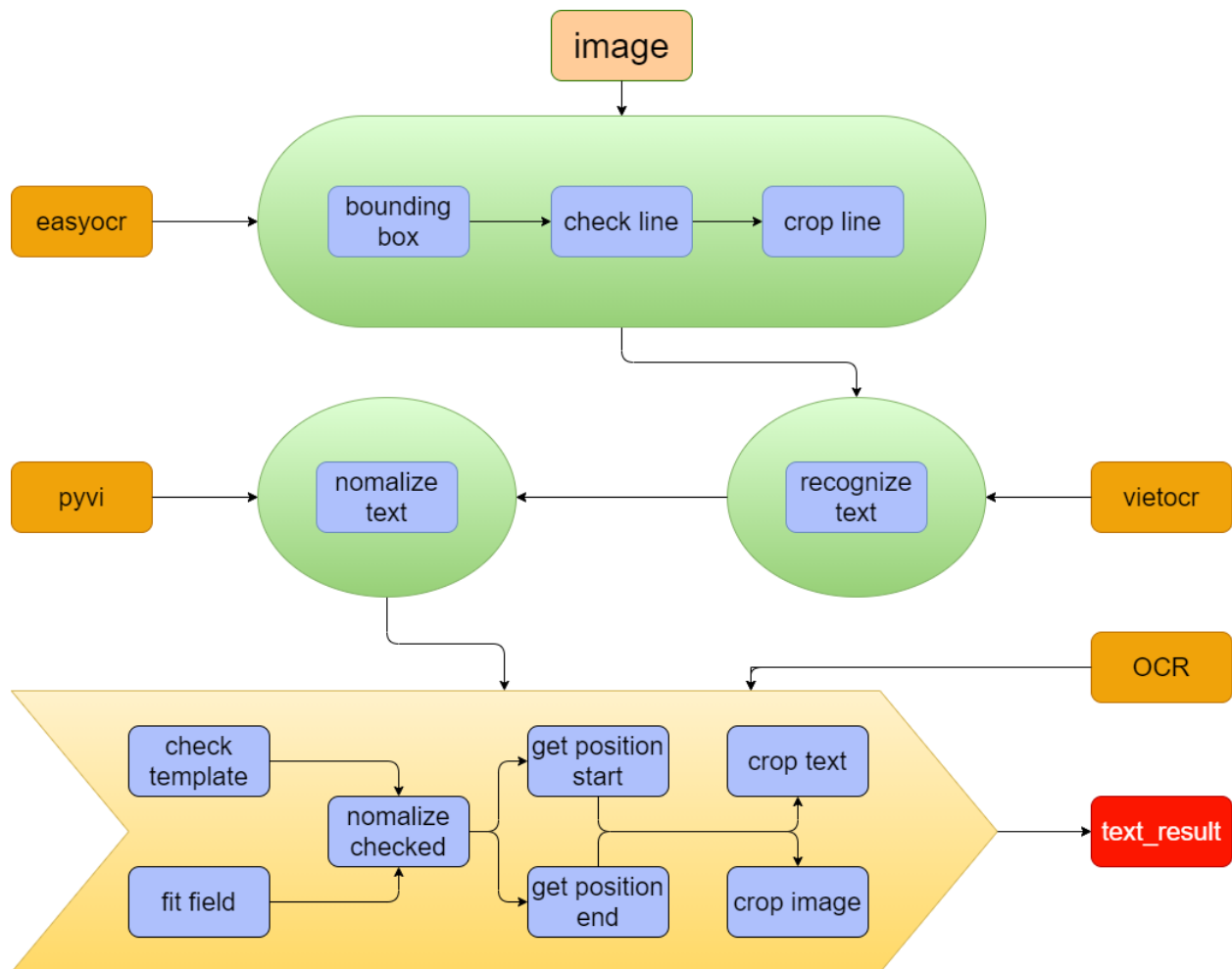
```
[options]
01 = blur_gaussian
02 = blur_bilateral
03 = inc_brightness
04 = dec_brightness
05 = inc_contrast
06 = dec_contrast
07 = Erosion
08 = Dilation
09 = Opening
10 = Closing
11 = histogram
12 = auto_detect
```



## 3.2. Core OCR

### 3.2.1. Tổng quan về pipeline

- OCR là một bài toán lớn, bao gồm nhiều bài toán con, với mỗi bài toán con là một bước, sai sót trong bất kì bước nào cũng gây ảnh hưởng đến các bước tiếp theo.
  - Bước 1: box detection – trích xuất được phân vùng chứa text
  - Bước 2: text recognize – trích xuất văn bản từ hình ảnh
  - Bước 3: xử lý hậu kỳ, bao gồm các quá trình biên dịch NLP để chuẩn hóa lại text
- Trong nội dung đồ án, do nhóm chúng em dùng tích hợp 2 mô hình OCR riêng biệt như có đề cập đến ở bên trên để cải thiện kết quả, nên các bước xử lý được custom lại để phù hợp với bài toán.
- Dưới đây là pipeline của quá trình OCR





### 3.2.2. Box detection

- EasyOCR đảm nhiệm quá trình này

```
def readtext(self, image, decoder = 'greedy', beamWidth= 5, batch_size = 1,\
    workers = 0, allowlist = None, blocklist = None, detail = 1,\
    rotation_info = None, paragraph = False, min_size = 20,\
    contrast_ths = 0.1,adjust_contrast = 0.5, filter_ths = 0.003,\
    text_threshold = 0.7, low_text = 0.4, link_threshold = 0.4,\
    canvas_size = 2560, mag_ratio = 1.,\
    slope_ths = 0.1, ycenter_ths = 0.5, height_ths = 0.5,\
    width_ths = 0.5, y_ths = 0.5, x_ths = 1.0, add_margin = 0.1, output_format='standard', flag = False):
    ...

    Parameters:
    image: file path or numpy-array or a byte stream object
    ...

    img, img_cv_grey = reformat_input(image)

    horizontal_list, free_list = self.detect(img, min_size, text_threshold,\
    low_text, link_threshold,\
    canvas_size, mag_ratio,\
    slope_ths, ycenter_ths,\
    height_ths,width_ths,\
    add_margin, False)
    # get the 1st result from hor & free list as self.detect returns a list of depth 3
    horizontal_list, free_list = horizontal_list[0], free_list[0]
    if flag:
        return horizontal_list
    else:
        result = self.recognize(img_cv_grey, horizontal_list, free_list,\
            decoder, beamWidth, batch_size,\
            workers, allowlist, blocklist, detail, rotation_info,\
            paragraph, contrast_ths, adjust_contrast,\
            filter_ths, y_ths, x_ths, False, output_format)

    return result
```

- Cụ thể sau khi khởi tạo, load weight, trong lúc gọi phương thức readtext của EasyOCR, nhóm chúng em có thêm 1 param là flag nhằm mục đích handle lại quá trình xử lý.
- Phương thức readtext bao gồm 2 quá trình:
  - Quá trình 1: self.detect đảm nhiệm quá trình box detection, và trả về horizontal\_list và free\_list. 2 list này chứa thông tin về vị trí của các box. Trong đó horizontal\_list là các box tốt nhất của quá trình detect
  - Quá trình 2: self.recognize đảm nhiệm quá trình recognize, trích xuất text từ các box được detect ở quá trình 1
- "flag" được thêm vào nhằm mục đích chặn quá trình 2 và trả về các box detection.



- Kết quả của quá trình này ta được



### 3.2.3. Line detection

```
def boxes_line(bounds=[], min=20):
    lst_box = []
    flag = bounds[0]
    mutil_box = False # flag for 1 box in 1 line
    for i in range(len(bounds)):
        if bounds[i][3] - flag[3] < min: # distance for top of 2 line
            mutil_box = True
            position = cluster_lines(flag, bounds[i])
        else:
            if not mutil_box:
                position = flag
            else:
                mutil_box = False
            lst_box.append(position)
            flag = bounds[i]

    #Last value
    if not mutil_box:
        position = flag
    lst_box.append(position)

    #Format list box
    lst_box = unique_list(lst_box)
    return lst_box
```

- Mục đích của quá trình này là xếp các box detect được từ EasyOCR thành từng line một.
- Điều kiện để 2 box cùng một line được nhóm chúng em đề xuất đến là khoảng cách giữa 2 box theo trục Oy < min. Với min ở đây là ngưỡng được set cố định và có giá trị bằng 20px.
- Với những box sau khi tính ngưỡng, vượt qua khỏi 20px, thì sẽ được set thành 1 line mới.
- Kết quả của quá trình này ta được:





### 3.2.4. Recognize

- Sau khi có được các box line, ta tiến hành crop image và đưa image này vào VietOCR để trích xuất ra text.

```
def OCR(lst_lines, detector)->list:  
    lst_text = []  
    for img in lst_lines:  
        try:  
            img = Image.fromarray(img)  
            text = detector.predict(img)  
        except:  
            text = ''  
        lst_text.append(text)  
    return lst_text
```

- Cụ thể với mỗi ảnh, ta sẽ được một text tương ứng, trong trường hợp fail thì text này sẽ được thay thế bằng 1 chuỗi rỗng.



- Kết quả của quá trình:

CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM

➔ CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM

NGUYỄN THỊ THANH TUYỀN

➔ NGUYỄN THỊ THANH TUYỀN

### 3.2.5. Normalize Text

```
def format_text(lst_text):  
    lst_upper = []  
    for text in lst_text:  
        temp = text.upper()  
        temp = viUtils.remove_accents(temp)  
        temp = temp.decode('utf-8')  
        lst_upper.append(temp)  
    return lst_upper
```

- Quá trình chuẩn hóa gồm 2 bước chính:
    - Bước 1: upper toàn bộ text
    - Bước 2: dùng pyvi để remove accents (ViUtils là một module của pyvi)
  - Kết quả của quá trình:
    - Original: "Họ tên: NGUYỄN THỊ THANH TUYỀN"
- ➔ Normalize text: "HO TEN: NGUYEN THI THANH TUYEN"



### 3.3. Core AI

#### 3.3.1. Fit Template

- Các templates này setting trong config.ini.
- Mỗi trường của template tương ứng với các trường data có thể xuất hiện trong input

```
[templates]
00 = ["Họ tên", "Nơi ĐKKH", "Sinh ngày", "Nguyên quán"]
01 = ["tên", "Nơi ĐKKH", "Sinh ngày", "Nguyên quán"]
03 = ["Họ và tên", "Nơi thường trú", "Ngày", "Giới tính", "Quê quán"]
04 = ["tên", "Nơi thường trú", "Ngày", "Giới tính", "Quê quán"]
05 = ["Họ tên", "Nơi ĐK", "Ngày sinh", "Địa chỉ"]
```

- Ở đây chúng ta đặc biệt quan tâm tới field start và field end lần lượt ở vị trí 0 và 1

#### \*\* Các bước cần làm

- **Bước 1:** Bắt được vị trí của field start và field end
- **Bước 2:** Dựa vào vị trí từ bước 1, ta tiến hành tính toán thống kê độ matching giữa template với input.

\*\* Bắt vị trí field start và field end được chia thành 2 phương thức riêng biệt, nhưng dùng chung phần tính toán thống kê độ matching giữa template với input.

```
for i in range(len(lst_count)):
    if lst_count[i] != 0:
        lst_count[i] = round(lst_count[i]/default_count[i], 4)

id = argmax(lst_count)
```

- Trong đó:
  - lst\_count là số field tìm được trong khoảng từ field\_start đến field\_end
  - default\_count là số field vốn dĩ phải có trong khoảng từ field\_start đến field\_end
- Lúc này độ matching giữa input và template sẽ bằng số field tìm được chia cho số field mặc định vốn dĩ phải có.



- Ví dụ:
    - Template 1 có số field = 4, số field tìm thấy là 3 => độ matching =  $3/4 = 75\%$
    - Template 2 có số field = 5, số field tìm thấy là 4 => độ matching =  $4/5 = 80\%$
    - Template 3 có số field = 6, số field tìm thấy là 5 => độ matching =  $5/6 = 83.33\%$
- ⇒ Template số 3 là template phù hợp nhất với input

### 3.3.2. Fit field V1

- Dưới đây core của giải thuật

```
for id_temp in range(len(templates)):
    fields = templates[id_temp]
    # get flag_start and flag_end of template about one obj
    for i in range(index_start, len(temp)):
        if ocr.search_custom(temp[i], [fields[0]]):
            flag_start = i
            break

    for i in range(index_start+1, len(temp)):
        if ocr.search_custom(temp[i], [fields[1]]):
            flag_end = i
            break

    if flag_start >= flag_end:
        continue
```

- Hàm search\_custom ở đây chỉ đơn giản là tìm kiếm chuỗi, kiểm tra sự tồn tại của chuỗi này trong chuỗi khác nên đôi khi quá trình bắt field\_start và field\_end xảy ra lỗi.
- Các lỗi điển hình thường gặp như, vị trí không tồn tại, field\_start xuất hiện sau field\_end....

**\*Lúc này ta cần sự can thiệp của levenshtein distance để giải quyết.**



### 3.3.3. Fit field V2

- Khác với search string, levenshtenin chỉ là phương thức tính toán nên luôn trả về giá trị phù hợp.

```
for j in range(len(lst_text)):
    text_start = lst_text[j][:l_start]
    text_end = lst_text[j][:l_end]

    distance_start = levenshtein(text_start, field_start)
    distance_end = levenshtein(text_end, field_end)

    lst_distance_start.append(distance_start)
    lst_distance_end.append(distance_end)
```

- Ứng với field\_start và field\_end theo từng template, ta sẽ tính toán levenshtein distance của của field\_start với đoạn text tương ứng trong lst\_text có độ dài đúng bằng field\_start. Thực hiện tương tự với field\_end.
- Về levenshtein, phương thức tính toán này được tích hợp sẵn trong module Enchant của python, ta chỉ cần import để sử dụng.
- Dưới đây là một số ví dụ lấy từ geeksforgeeks:





- Ví dụ 1:

```
# import the enchant module
import enchant

# determining the values of the parameters
string1 = "Hello World"
string2 = "Hello d"

# the Levenshtein distance between
# string1 and string2
print(enchant.utils.levenshtein(string1, string2))
```

**Output :**

4

- Ở trường hợp này, levenshtein distance của chuỗi 1 và chuỗi 2 là 4. Tương ứng với đó ta cần thêm 4 ký tự "Worl" để biến chuỗi 1 thành chuỗi 2.
- Ví dụ 2:

```
# import the enchant module
import enchant

# determining the values of the parameters
string1 = "abc"
string2 = "aef"

# the Levenshtein distance between
# string1 and string2
print(enchant.utils.levenshtein(string1, string2))
```

**Output :**

2



- Ở trường hợp này, levenshtein distance là 2, tương ứng với 2 ký tự sai khác 2 ký tự "ef" đáng lẽ ra phải là "bc".
- Do độ dài của chuỗi field khá nhỏ, nên levenshtein distance có khả năng trùng lặp giá trị khá cao.
- Ví dụ:
  - Trong trường hợp field, str1 = "Họ"
  - str2 = "Hoa"
  - str3 = "Ho"
- Lúc này levenshtein distance (str1, str2) = levenshtein distance (str1, str3) = 1
- Do khi tính toán, ta chỉ lấy độ dài chuỗi đối sánh đúng bằng độ dài field và bằng 2, thì str2 = "Ho", ký tự "a" không được đưa vào tính toán.
- Khi đó sự sai khác giữa str1 với str2 và str3 là "ø" thay vì "o"
- Để giải quyết vấn đề này, ta có 2 hướng:
  - Một là sử dụng normalize text để xóa dấu
  - Hai là, sử dụng phương pháp thống kê so khớp template như đã đề cập mục 3.7

### 3.4. Nhận xét về core AI

- So với fit field V1, fit field V2 tốn nhiều thời gian cho tính toán, phức tạp trong tổ chức dữ liệu. Vì thế fit field V2 chưa phải là giải pháp tối ưu trong mọi tình huống.
- Trong trường hợp fit field V1 không trả về kết quả hợp lệ thì mới tiếp tục chạy xuống fit field V2

```
try:
    id, flag_param = fit_fields(lst_text)
    if check_index_fail(flag_param):
        id, flag_param = fit_fields_v2(lst_text)
```

- flag\_param ở đây chính là vị trí của field\_start và field\_end
- Mặc định trong tất cả các trường hợp, luồng xử lý sẽ chạy vào fit field V1 trước, sau đó nếu flag\_param gặp lỗi, thì mới chạy tiếp vào fit field V2



## 4. Các trường hợp ngoại lệ

- Trường hợp 1: Định dạng tệp người dùng gửi lên không phải định dạng ảnh, tiến hành bỏ qua, không xử lý tiếp

```
lst_tail = ['jpg', 'jpeg', 'png']

handlers = [logging.FileHandler('./logs/debug_url.log', 'w', 'utf-8')]
logging.basicConfig(handlers=handlers, level=logging.DEBUG)
time_average = []
for id in range(len(list_files)):
    file = list_files[id]
    name = file.split('.')[0]
    tail = file.split('.')[1]

    if tail not in lst_tail:
        continue
```

- Trường hợp 2: Tên file chứa kí tự đặc biệt, có dấu gây ra lỗi trong quá trình đọc, ta tiến hành đưa pyvi vào để xóa dấu, tránh gây lỗi.

```
# auto rename files
new_name = ViUtils.remove_accents(name).decode("utf-8")
old_file = os.path.join(temp_path, f'{name}.{tail}')
new_file = os.path.join(temp_path, f'{new_name}.{tail}')
os.rename(old_file, new_file)
```

- Trường hợp 3: Hình ảnh người dùng gửi lên không chứa text gây ra lỗi trong quá trình bounding box. Ta gán giá trị text OCR = "None"

```
#None bounding box
try:
    bounds = reader.readtext(image, flag = True)
    box = ocr.bboxes_line(bounds)
except:
    image_crop = image
    text_result = "None"
```



- Trường hợp 4: Trong trường hợp EasyOCR bắt được box nhưng khi đưa box này qua cho VietOCR, quá trình recognize xảy ra lỗi. Ta gán giá trị text OCR = "None"

```
try:
    lst_lines = ocr.crop_lines(image, box)
    lst_text = ocr.OCR(lst_lines, detector)
except:
    lst_text = "None"
```

- Trường hợp 5: tuy fit field V2 đã giải quyết phần lớn vướng mắc còn tồn đọng ở trong fit fields V1, nhưng không có nghĩa là không còn lỗi. Khi người dùng cố tình gửi lên một loại giấy tờ khác, nằm ngoài phạm vi input (CMND/CCCD) thì quá trình fit field sẽ xảy ra lỗi. Lúc này ta sẽ trả về cho người dùng toàn bộ đoạn text OCR được

```
#None get paragraph
try:
    id, flag_param = fit_fields(lst_text)
    if check_index_fail(flag_param):
        id, flag_param = fit_fields_v2(lst_text)

    pos_start = flag_param[0]
    pos_end = flag_param[1]
    # print(pos_start, pos_end, id)

    #crop image
    box_temp = box[pos_start:pos_end+1]
    position = ocr.cluster_paragraph(box_temp)
    image_crop = ocr.crop_lines(image, [position])[0]
    text_result = lst_text[pos_start:pos_end+1]
    raise_exception(type_r=True, path=path)

except:
    image_crop = image
    text_result = lst_text
    raise_exception(type_r=False, path=path)
```



## 5. Kết quả quá trình OCR

- Nếu chương trình chạy ổn, không rơi vào các trường hợp ngoại lệ đã đề cập đến ở bên trên. Kết quả của chương trình sẽ bao gồm 3 phần.

```
if debug:
    if id < 2:
        type_template = "CMND"
    elif id < 4:
        type_template = "CCCD"
    else:
        type_template = "Exception"
    show([image_crop], [f'Type: {type_template}'])
else:
    if save_img:
        return {
            "text": text_result
        }
    else:
        return {
            "img": image_crop,
            "text": text_result
        }
```

- Tùy theo param được set mà giá trị trả về sẽ khác nhau.
  - Khi bật mode debug: giá trị trả về là loại giấy tờ của input vào image sau khi crop được phần thông tin có trong template
  - Trong trường hợp mode debug = False, có bật mode save\_img: giá trị trả về là đoạn text ứng với template
  - Cuối cùng, trong trường hợp mặc định: giá trị trả về bao gồm image đã crop và đoạn text ứng với template. Đây cũng chính là giá trị mà client sẽ nhận được.



## V. Frontend React

### 1. Tổng quan

- React là một thư viện GUI nguồn mở JavaScript với mã nguồn mở miễn phí để xây dựng giao diện người dùng hoặc các thành phần UI. Nó được sử dụng phổ biến trên thế giới bao gồm nhiều tập đoàn lớn về công nghệ như Facebook, Netflix, WhatsApp, eBay, Instagram,...
- Ưu điểm của việc sử dụng React để phát triển front-end web:
  - React là một DOM (Document Object Model) ảo nơi mà các component thực sự tồn tại trên đó, dễ dàng trong vấn đề test giao diện.
  - Dễ dàng tái sử dụng các component từ các ứng dụng khác nhau.
  - Hiệu năng cao đối với các ứng dụng dữ liệu thay đổi liên tục
  - Dễ dàng bảo trì và sửa lỗi
- Nhược điểm của React:
  - Chỉ hỗ trợ view trong mô hình MVC vì thế khó kết hợp với các framework MVC truyền thống
  - React khá nặng so với các framework JavaScript khác
  - Khó tiếp cận đối với người mới học web

### 2. Axios

- Axios là một HTTP client được viết dựa trên Promises được dùng để hỗ trợ cho việc xây dựng các ứng dụng API từ đơn giản đến phức tạp và có thể được sử dụng cả ở trình duyệt hay Node.js
- Việc tạo ra một HTTP request dùng để fetch hay lưu dữ liệu là một nhiệm vụ thường thấy mà một ứng dụng JavaScript phía client cần phải làm khi muốn giao tiếp với phía server.
- Các thư viện bên thứ 3, đặc biệt là jQuery từ xưa đến nay vẫn là một trong những cách phổ biến để giúp cho các browser API tương tác tốt hơn, rõ ràng và rành mạch hơn, xóa đi những khác biệt giữa các browser với nhau.





## VI. Backend Flask

### 1. Tổng quan

- **Flask** là một web frameworks, nó thuộc loại micro-framework được xây dựng bằng ngôn ngữ lập trình Python. Flask cho phép bạn xây dựng các ứng dụng web từ đơn giản tới phức tạp. Nó có thể xây dựng các api nhỏ, ứng dụng web chẳng hạn như các trang web, blog, trang wiki hoặc một website dựa theo thời gian hay thậm chí là một trang web thương mại. Flask cung cấp cho bạn công cụ, các thư viện và các công nghệ hỗ trợ bạn làm những công việc trên.
- **Flask** là một micro-framework. Điều này có nghĩa Flask là một môi trường độc lập, ít sử dụng các thư viện khác bên ngoài.
  - Do vậy, Flask có ưu điểm là nhẹ, có rất ít lỗi do ít bị phụ thuộc cũng như dễ dàng phát hiện và xử lý các lỗi bảo mật.
  - Nhược điểm là đôi khi bạn phải tự thêm các danh sách phụ thuộc bằng việc thêm các plugin. Trong Flask, các phụ thuộc đó là Werkzeug WSGI và Jinja2.

### 2. Flask cors

- **CORS** là một cơ chế cho phép nhiều tài nguyên khác nhau (fonts, Javascript, v.v...) của một trang web có thể được truy vấn từ domain khác với domain của trang đó. CORS là viết tắt của từ Cross-origin resource sharing.
- **CORS** là một cơ chế xác nhận thông qua Header của request. Cụ thể là trên Server sẽ nói với browser về quy định chấp nhận những request từ domain nào và phương thức ra sao (GET, POST, PUT, v.v..)
  - *Access-Control-Allow-Origin*: Những origin mà server cho phép. (ví dụ server loda.his chỉ chấp nhận loda.me request lên)
  - *Access-Control-Allow-Headers*: Những Header được server cho phép. (ví dụ x-authorize, origin, cái này do server bạn quy định)
  - *Access-Control-Allow-Methods*: Những Method được server cho phép (POST, GET, v.v..)



### 3. SSL Flask

- SSL là viết tắt của từ Secure Sockets Layer. SSL là tiêu chuẩn của công nghệ bảo mật, truyền thông mã hoá giữa máy chủ Web server và trình duyệt. Tiêu chuẩn này hoạt động và đảm bảo rằng các dữ liệu truyền tải giữa máy chủ và trình duyệt của người dùng đều riêng tư và toàn vẹn.
- SSL hiện tại cũng là tiêu chuẩn bảo mật cho hàng triệu website trên toàn thế giới, nó bảo vệ dữ liệu truyền đi trên môi trường internet được an toàn.
- SSL đảm bảo rằng tất cả các dữ liệu được truyền giữa các máy chủ web và các trình duyệt được mang tính riêng tư, tách rời.

#### 3.1. Certificate

- **SSL certificate** là "a bit of code" trên webserver, cung cấp bảo mật cho các giao tiếp trực tuyến. Khi web browser liên lạc với website được bảo mật, chứng chỉ SSL sẽ cho phép kết nối được mã hóa. Nó giống như niêm phong một lá thư trong một phong bì trước khi gửi đi.
- **SSL certificates** có độ tin cậy khá cao vì mỗi SSL certificate đều chứa thông tin nhận dạng. Khi bạn yêu cầu chứng chỉ SSL, bên thứ ba sẽ xác minh thông tin của tổ chức và cấp chứng chỉ duy nhất cho bạn với thông tin đó - gọi là quá trình xác thực (authentication process).
- **Certificate Authority (CA)** là tổ chức phát hành các chứng thực các loại chứng thư số cho người dùng, doanh nghiệp, máy chủ (server), mã code, phần mềm. Nhà cung cấp chứng thực số đóng vai trò là bên thứ ba (được cả hai bên tin tưởng) để hỗ trợ cho quá trình trao đổi thông tin an toàn.
- **SSL Certificate có một cặp khóa:** một công khai (public key) và khóa riêng (private key). Các khóa này làm việc cùng nhau để thiết lập một kết nối được mã hóa. SSL Certificate cũng có chứa những thông tin của chứng chỉ / website.
  - **Khóa công khai - Public key:** được share với bất kì ai/client/browser nào có nhu cầu giao tiếp với server.



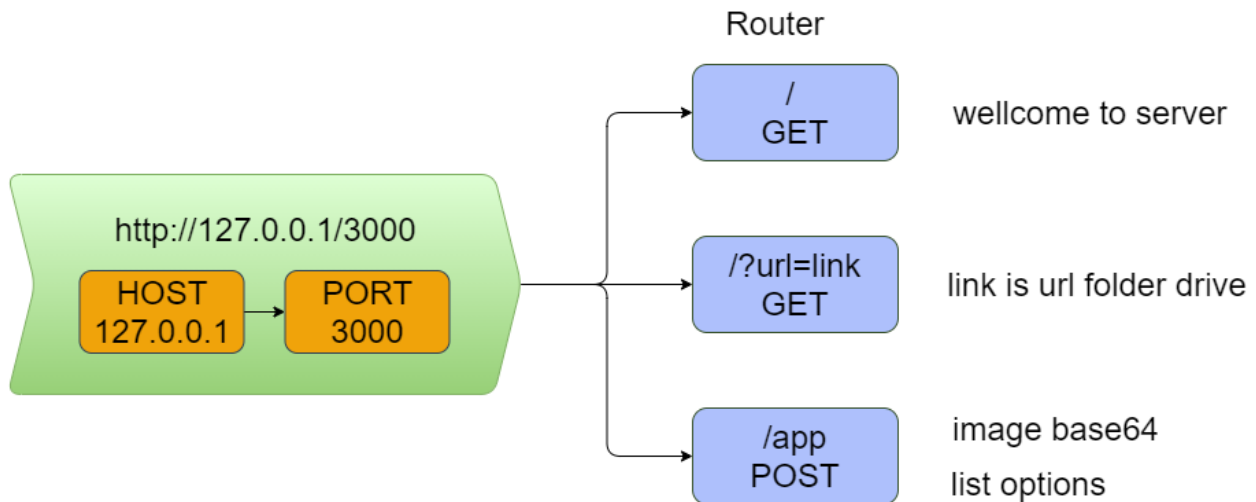
- **Khóa bí mật - Private key:** được server lưu giữ cẩn thận, không được phép tiết lộ ra bên ngoài.
- Cặp khóa này được sinh ra bởi thuật toán dùng để mã hóa và giải mã data. Cụ thể là, data được mã hóa bởi Public key chỉ có thể được giải mã bởi Private key của cặp khóa đó (và ngược lại).

### 3.2. Decrypts

- **Decryption SSL** là một phương pháp mà các kết nối internet được giữ an toàn, cho dù chúng là máy khách với máy khách, máy chủ đến máy chủ hoặc (thường xuyên hơn) máy khách với máy chủ. Điều này ngăn các bên thứ ba trái phép nhìn thấy hoặc thay đổi bất kỳ dữ liệu nào được trao đổi trên internet.
- Khi bạn kết nối với các trang web, một kết nối được mã hóa giữa trình duyệt của bạn và trang web (ứng dụng đám mây) sẽ được thiết lập.
- Mặc dù mã hóa này ngăn những con mắt tò mò xem dữ liệu nhạy cảm của bạn, nhưng điều quan trọng là phải giảm thiểu rủi ro trong lưu lượng truy cập này. Các mối đe dọa nâng cao và phần mềm độc hại thường xuyên được phân phối trong lưu lượng được mã hóa. Đây là lúc **Decryption SSL** xuất hiện. **Decryption SSL** cho phép các tổ chức phá vỡ lưu lượng được mã hóa mở và kiểm tra nội dung của nó. Lưu lượng sau đó được mã hóa lại và gửi trên đường đi. Nhưng việc kiểm tra lưu lượng được mã hóa không phải là điều quá quan trọng và nó yêu cầu một kiến trúc proxy).



## 4. API



API được tích hợp 3 chức năng chính:

- `/` : giao diện chào mừng bạn đến server
- `/?url=link` : request thông qua phương thức get, với body là
  - `url` : link google drive, giúp chạy hàng loạt file từ thư mục google drive
- `/app` : request thông qua phương thức post, với body là
  - `img` : hình ảnh được mã hóa định dạng base64
  - `lst` : chứa các tùy chọn chỉnh sửa ảnh từ phía người dùng

## 5. Server Requirements

- Python==3.8.x
- tensorflow==2.6.1
- keras\_applications==1.0.8
- sklearn-crfsuite==0.3.6
- opencv-python==4.5.4.58
- efficientnet==1.1.1
- torch==1.10.0
- torchvision==0.11.1
- einops==0.3.2
- gdown==4.2.0
- flask==1.1.2
- flask-cors==3.0.10
- python-bidi==0.4.2
- pyyaml==5.4.1
- pyenchant==3.2.2
- matplotlib==3.5.1

\*\* Server nên có **GPU** để cải thiện tốc độ tính toán. Các thiết lập cơ bản được định nghĩa sẵn trong file “config.ini”. Người dùng có thể tái định nghĩa.



## VII. Deploys

- Clone git từ link sau: [PhamTruongUit/IDCard\\_OCR: CS406 \(github.com\)](https://github.com/PhamTruongUit/IDCard_OCR_CS406)
- Cài đặt theo hình bên dưới, hoặc xem thông tin trực tiếp từ file Readme.md

### Setting server:

#### Install

- with python  $\leq 3.8.x$
- using terminal
- create virtualenv

```
cd server
python -m venv myenv
myenv/Scripts/activate
```

- install require

```
python -m pip install -- upgrade pip
pip install -r requirements.txt
```

#### Run

- using terminal
- using virtualenv

```
python main.py
```

- Cấu hình server thêm nếu cần trong file config.ini





## Setting client:

### Install

- with nodejs
- using terminal

```
npm install
```

### Run

- using terminal

```
npm start
```

## VIII. Demo App



## IX. Phương hướng phát triển

- Nâng cao hiệu quả của quá trình tiền xử lý ảnh
- Cải thiện khả năng auto detect, auto rotation
- Cải thiện khả năng OCR trên tiếng việt
- Mở rộng bài toán ra thêm cho nhiều loại giấy tờ
- Phát triển bộ OCR trên chữ viết tay
- Tích hợp vào thêm các công cụ search trực tiếp
- Tích hợp trình phiên dịch, cho phép dịch thuật đa ngôn ngữ từ text OCR

## X. Reference

- [1]. [Jaided AI - Distribute the benefits of AI to the world](#)
- [2]. [JaidedAI/EasyOCR](#)
- [3]. [pbcquoc/vietocr](#)
- [4]. [trungtv/pyvi](#)
- [5]. [Levenshtein distance](#)

