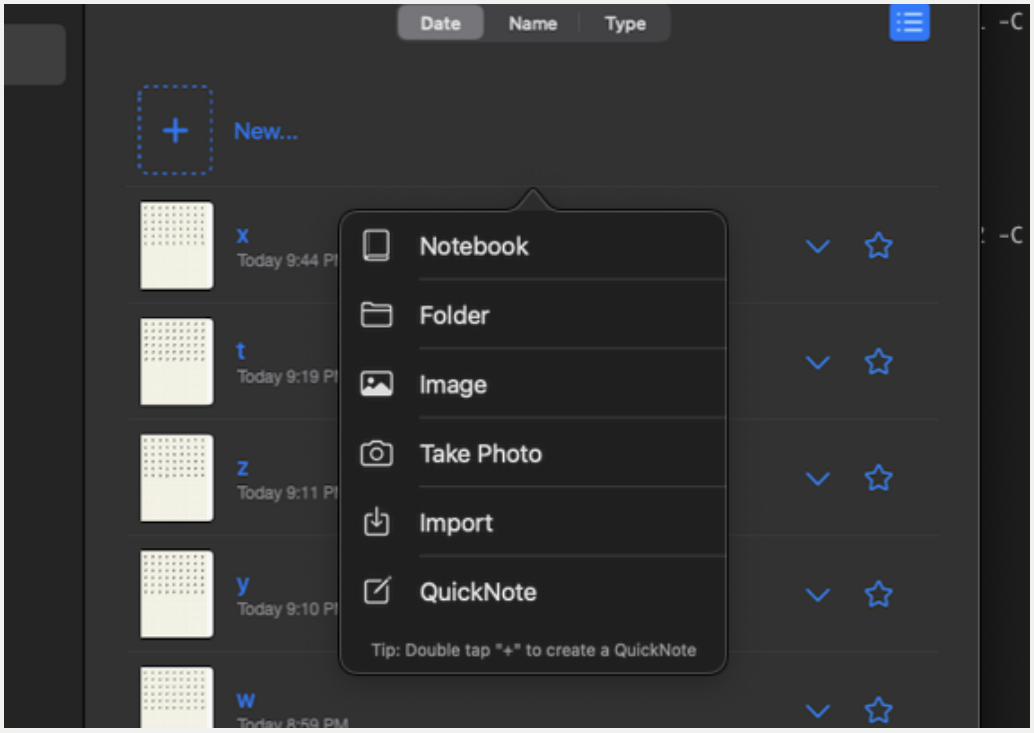
손글씨 알파벳 데이터셋 구축

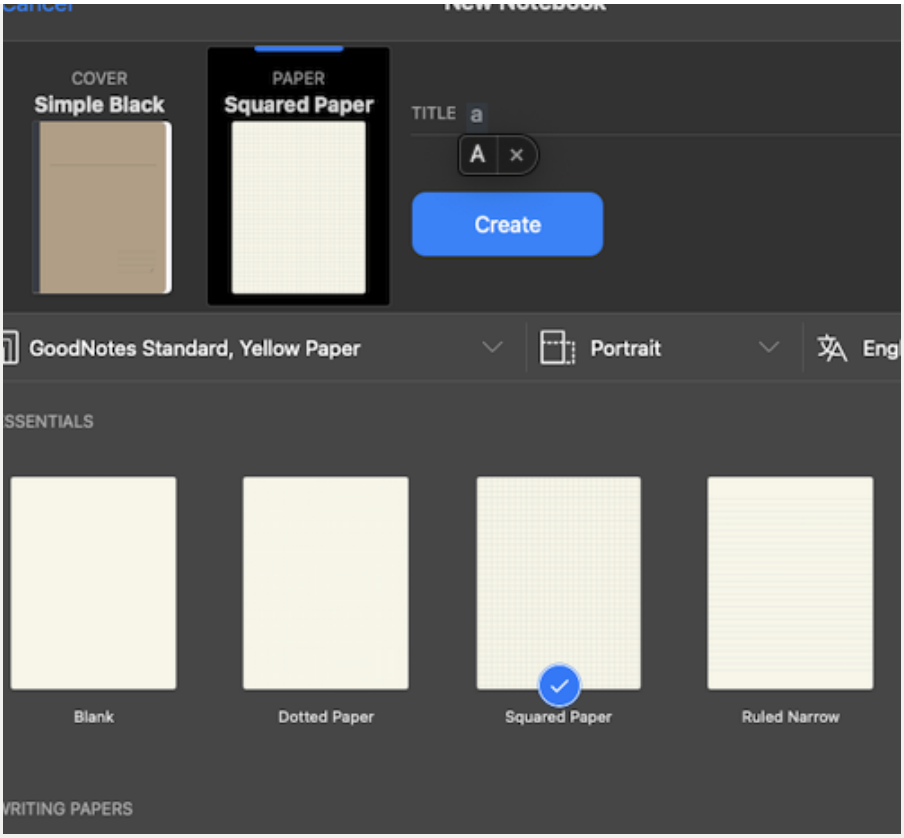
Guide For user of GoodNote

1. 알파벳 적기

- goodnote에는 아래와 같이 grid가 있는 배경을 템플릿으로 사용할 수 있습니다.

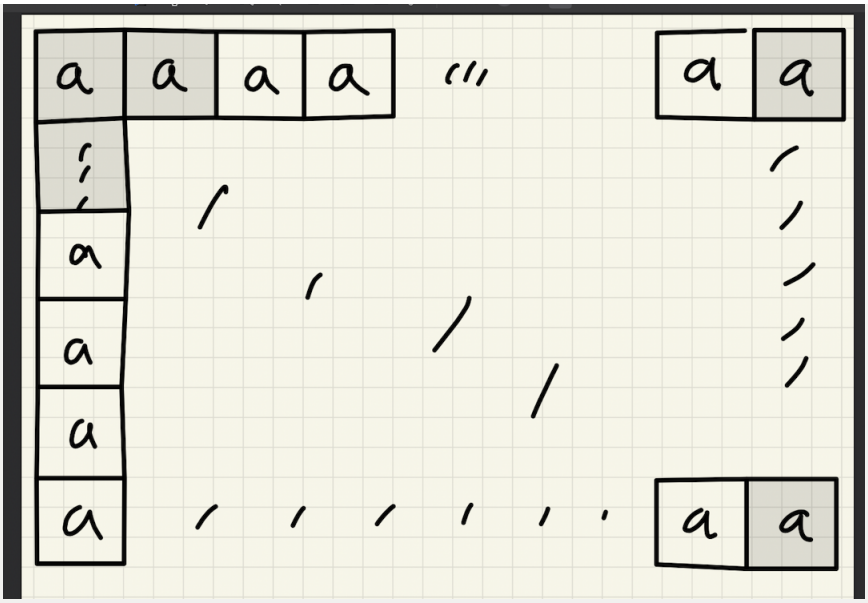


  - 여기서 `Notebook`을 선택하고



- 여기서 `Squared Papaer` 선택, title은 a라고 합니다.
  - 이따 export할 때 이름을 다시 설정해야하므로 중요하지 않습니다.

- 아래와 같이 3칸의 square를 하나의 글자가 들어갈 공간이라고 생각하고
  총 6x9 로 알파벳을 패드로 그립니다.



- 여기서 주의할 점은 아래와 같이 양끝 마진은 없다고 생각하셔야합니다.

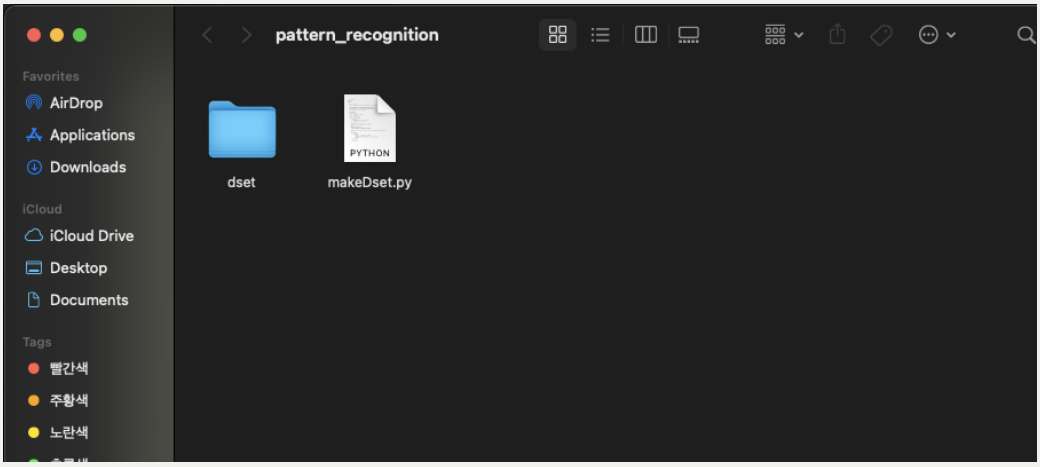
- 완성본은 아래와 같습니다.

- 50개만 적으셔도 좋지만 그러면 빈 문자들이 생길 수 있으니 지우셔야하고
  50개만 만들기 위해 아래 제공해드릴 코드를 수정하셔도 좋습니다.
- 전 귀찮아서 그냥 했습니다.

☝️ 위와 같이 a 부터 z까지 모두 그려 패드에 저장합니다.

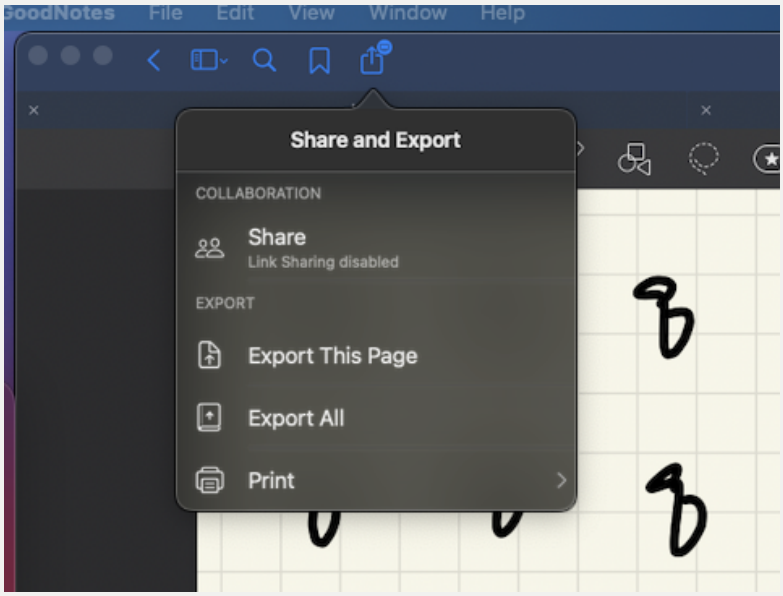
2. export 하여 아래와 같이 directory structure 구성하기

1. **아래와 같이 makeDset.py 와 `dset` 이라는 directory를 만듭니다**

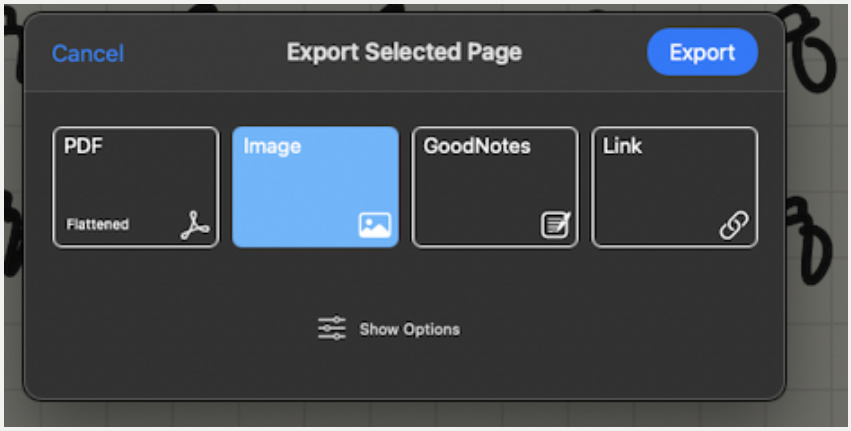


☝️ dset 이름은 꼭 "dset" 이어야합니다.

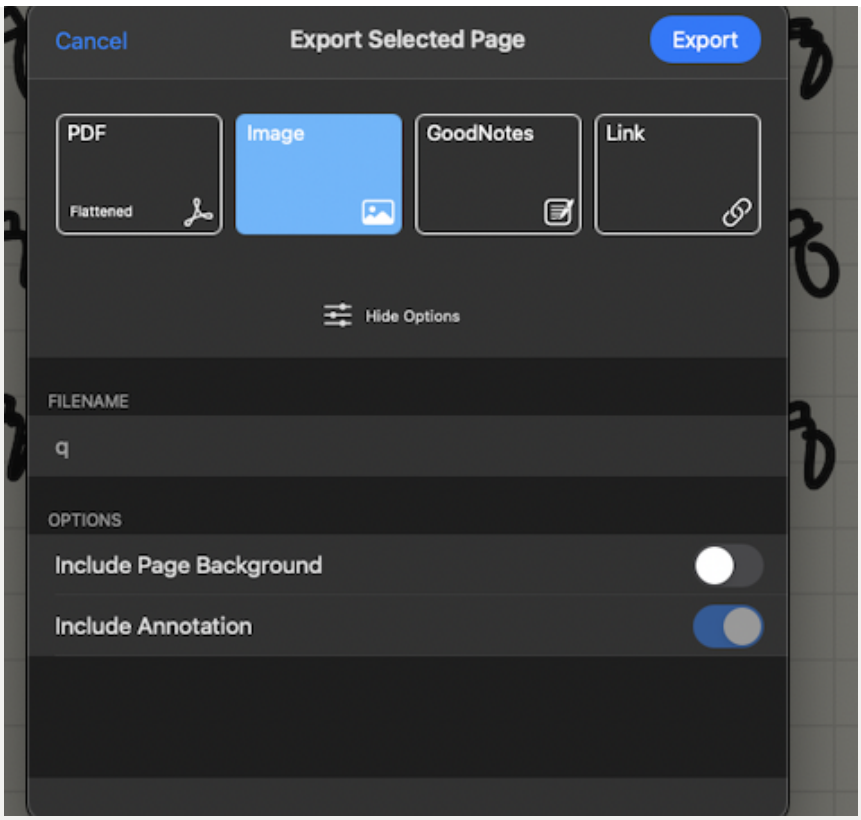
2. **그리고 dset아래에 위에서 작성하였던 page들을 export하여 dset아래에 저장합니다.**
   - export 방법



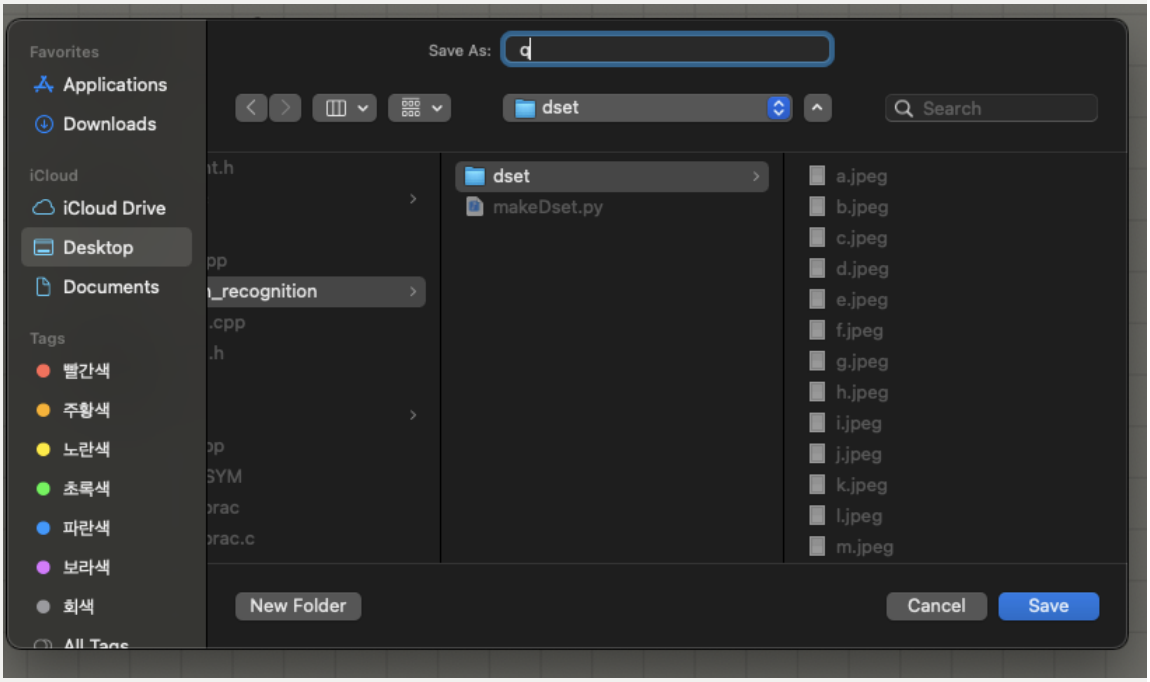
   - 이와 같이 export 버튼을 찾이시고 `Export this Page`를 누릅니다.



   - 맥북 goodnote 기준으로 위와 같이 나오는데 조금 다를 수 있으나 `Show Options` 와 같이 option 셋팅 버튼을 누릅니다.



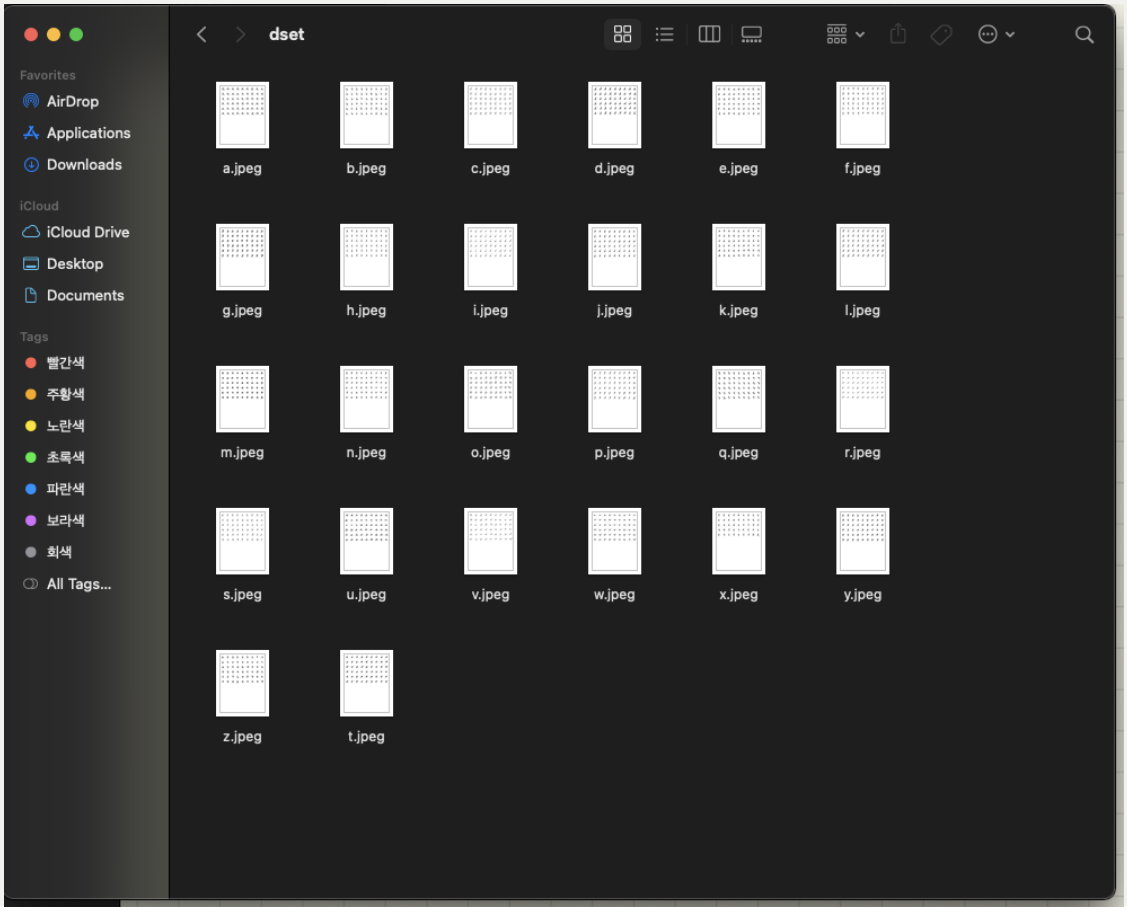
- 위와 같이 `Include Page Background` 를 `off` 해줍니다.
- 그리고 `Export` 버튼을 누르면



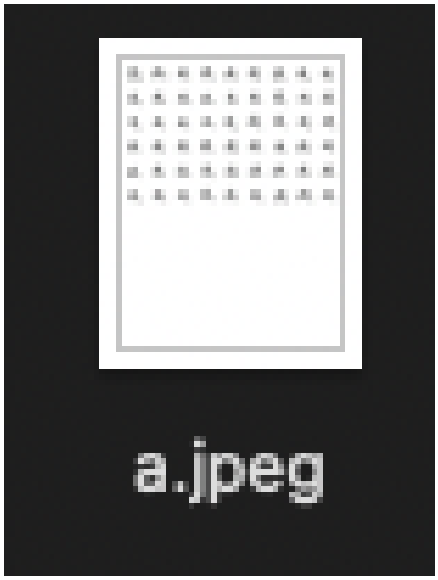
☝️ 주의할점!
꼭 a는 a b는 b로 이름을 정해주셔야합니다.

- 이와 같이 뜨는데 `Save As` 에 **q 와 같이 작성한 알파벳 하나만**을 입력해줍니다. extension은 뒤에 자동으로 붙습니다.

3. **아래와 같이 dset 아래에 a 부터 z까지 파일들이 저장합니다.**



4. 그리고 해당 extension이 어떻게 저장됐는지 확인합니다.



- 위와 같은 경우에는 `jpeg` 입니다.

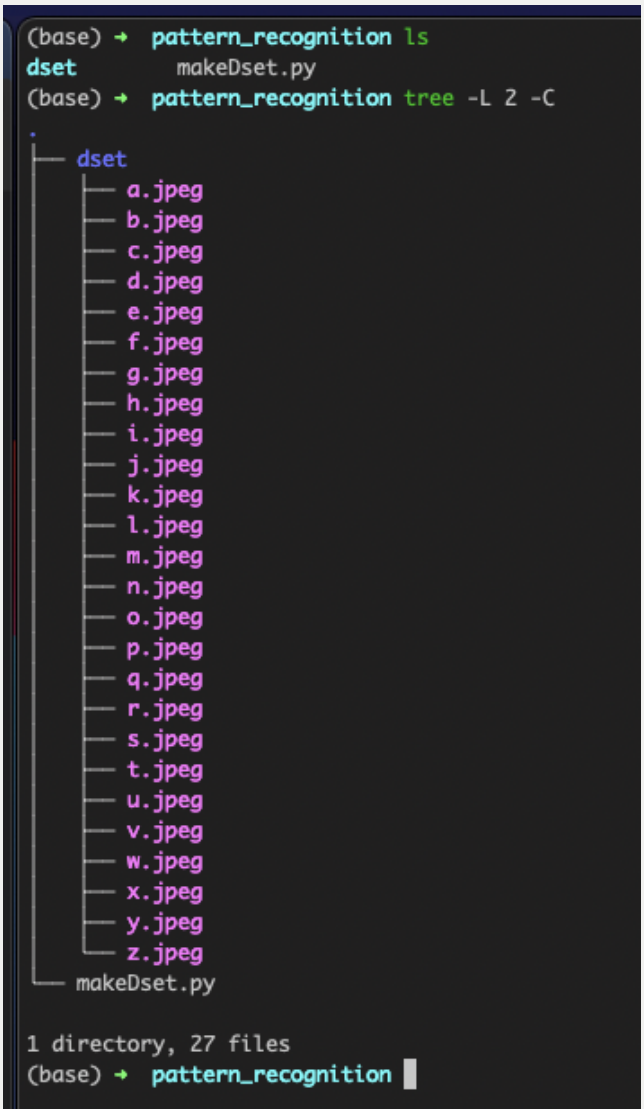
3. makeDset.py 수행하기

1. **터미널을 열어줍니다.**

2. **아래 명령어를 수행해 결과를 비교합니다. (optional)**

```
$ ls
$ tree -L 2 -C
```

☝🏻 tree 명령어는 설치가 안되어있을 수 있으므로 굳이 a부터 z까지 잘 만들었다면 굳이 수행하지 않으셔도 됩니다.



3. **아래와 같이 명령어를 입력하여 주어진 파이썬 코드를 수행합니다.**

```
$ python3 makeDset.py --name=<본인의 영어 이름> --id=<본인의 학번> --ext=<위에서 확인한 확장자>
```

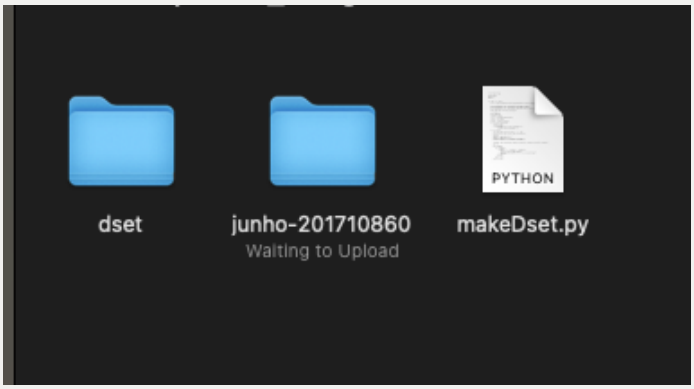
- example

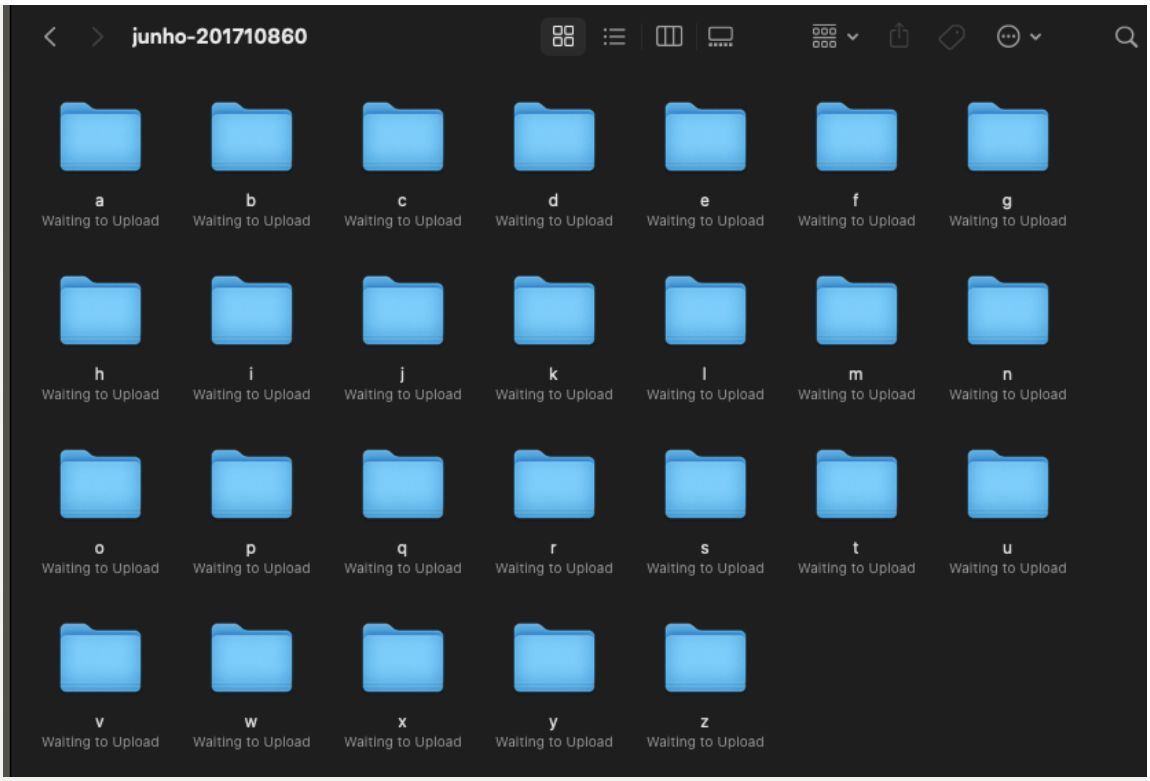
```
$ python3 makeDset.py --name=junho --id=201710860 --ext=jpeg
```

4. **수행결과 확인**

```
(base) ➜  pattern_recognition python3 makeDset.py --name=junho --id=201710860 --ext=jpeg
a Done.
b Done.
c Done.
d Done.
e Done.
f Done.
g Done.
h Done.
i Done.
j Done.
k Done.
l Done.
m Done.
n Done.
o Done.
p Done.
q Done.
r Done.
s Done.
t Done.
u Done.
v Done.
w Done.
x Done.
y Done.
z Done.
(base) ➜  pattern_recognition
```

- 수행 결과 화면이며 아래와 같이 본인의 이름과 학번으로 생성된 directory가 하나 생깁니다.





- 아래와 같이 잘 저장되었는지 확인하고 제출하시면 됩니다.


a
a0-201710860.png
a1-201710860.png
a2-201710860.png
a3-201710860.png
a4-201710860.png
a5-201710860.png
a6-201710860.png
a7-201710860.png
a8-201710860.png
a9-201710860.png
a10-201710860.png
a11-201710860.png
a12-201710860.png
a13-201710860.png
a14-201710860.png
a15-201710860.png
a16-201710860.png
a17-201710860.png
a18-201710860.png
a19-201710860.png
a20-201710860.png
a21-201710860.png
a22-201710860.png
↑ 33 bytes
a23-201710860.png
↑ 34 bytes
a24-201710860.png
↑ 33 bytes
a25-201710860.png
↑ 32 bytes
a26-201710860.png
↑ 33 bytes
a27-201710860.png
↑ 31 bytes
a28-201710860.png
↑ 31 bytes
a29-201710860.png
↑ 32 bytes
a30-201710860.png
↑ 32 bytes
a31-201710860.png
↑ 33 bytes
a32-201710860.png
↑ 31 bytes
a33-201710860.png
↑ 30 bytes
a34-201710860.png
↑ 33 bytes

Code

```
from PIL import Image

import argparse
import PIL
import os


if __name__ == '__main__':

    parser = argparse.ArgumentParser(description='SMU-pattern recognition making dataset')

    parser.add_argument("--name", required=True, help="name in English")
    parser.add_argument("--id", required=True, help="student ID (학번)")
    parser.add_argument("--ext", required=True, help="jpg or jpeg or png etc..")

    FLAGS, FIRE_FLAGS = parser.parse_known_args()

    ext = FLAGS.ext
```

```
    name = FLAGS.name
    studentID = FLAGS.id

    charset = "abcdefghijklmnopqrstuvwxyz"
    charList = list(charset)

    dsetPath = f"{name}-{studentID}"

    if not os.path.isdir(dsetPath):
        os.mkdir(dsetPath)
        for c in charList:
            if not os.path.isdir(os.path.join(dsetPath, c)):
                os.mkdir(os.path.join(dsetPath, c))

    for c in charList:
        img = Image.open(os.path.join("dset",c + "." + ext))

        # empirically set values, do not touch =================
        nGrid = 28
        marginPix = (img.size[0] / 28) / 2
        gridWidth = (img.size[0] / 28)
        fontWidth, fontHegith = 3 * gridWidth, 3 * gridWidth
        # =====================================================

        croppedImg = img.crop((marginPix, marginPix, img.size[0] - marginPix, img.size[1] - marginPix))

        iter = 0
        x, y = 0, 0
        for i in range(6):
            for j in range(9):
                croppedImg\
                    .crop((x, y, x + fontWidth, y + fontHegith))\
                    .resize((28, 28), PIL.Image.LANCZOS)\
                    .save(os.path.join(dsetPath, c, c + f"{iter}-{studentID}.png"))
                x += fontWidth
                iter += 1
            y += fontHegith
            x = 0

        print(f"{c} Done.")
```