```python
from asyncio.windows_events import NULL
from flask import Flask, render_template, request, url_for, redirect, session, Response, make_response,send_file, send_from_directory
from datetime import timedelta
import datetime
from werkzeug.utils import secure_filename # fileを保存するのに必要なライブラリ
import flask
from flask import send_from_directory
import os
import psycopg2
import shutil
import urllib.parse

from urllib.parse import quote

# 2007 Office system ファイル形式の MIME タイプをサーバーで登録する
# https://technet.microsoft.com/ja-jp/library/ee309278(v=office.12).aspx
XLSX_MIMETYPE = 'application/vnd.openxmlformats-officedocument.spreadsheetml.sheet'

app = Flask(__name__, static_folder=None)


app.config['MAX_CONTENT_LENGTH'] = 1 * 1024 * 1024 *1024
# 最大アップロード容量 : 1GB
app.secret_key = 'abcdefghijklmn'
app.permanent_session_lifetime = timedelta(minutes=30)
# セッション期限は30分




def get_db_connection():    # データベースの接続設定
    conn = psycopg2.connect(host='localhost',
                            database='test',
                            user='eno',
                            password='pass')
    return conn


def uidSes():   # データベースの接続設定
    if "uid" in session:
        uid=session["uid"]
        return uid
    else:
        return None


#  -----目次--------
#  1.TOPページの処理
#  2.新規登録の処理
#  3.ログインページの処理
#  4.ログアウトの処理
#  5.マイページの処理
#  6.新規登録完了の処理
#  7.1作品の新規登録の処理
#  7.2作品の新規登録の処理2
#  8.作品の編集・登録完了
#  9.検索処理
```

```
59 # 10.成果物の詳細ページ
60 # 11.マイページ編集ページ
61 # 12.1成果物編集ページ
62 # 12.2成果物編集ページ
63 # 13.成果物downloadページ
64 # 14.作成者の詳細ページ
65 # 14.成果物の削除ページ
66
67
68
69 # 1.TOPページの処理----------------------------------------------------------------
   ------------1
70 @app.route('/')
71 def top():
72
73     uid=uidSes()
74     # ログインされていれば、uidを取得
75
76     conn = get_db_connection()
77     cur = conn.cursor()
78     cur.execute('SELECT p_name,rf_date,user_id,p_id FROM product_data ORDER BY
   rf_date DESC LIMIT 10;')
79     c_products = cur.fetchall()
80     #  成果物データの「成果物名、更新日時、名前（ユーザデータ）」をrf_date（更新日時）の降
   順で、10件取得
81     cur.close()
82     conn.close()
83     return render_template('top.html', uid=uid,c_products=c_products)
84     # topページに取得した値を渡す
85
86
87
88 # 2.新規登録の処理----------------------------------------------------------------
   --------------------2
89 @app.route('/add_member/', methods=('GET', 'POST'))
90 def newmember():
91
92     uid=uidSes()
93     # ログインされていれば、uidを取得
94
95     if request.method == 'POST':
96         id=request.form['id']
97         name = request.form['name']
98         born = request.form['born']
99         email = request.form['email']
100        op_email = request.form['op_email']
101        passWord = request.form['passWord']
102        check_pass = request.form['check_pass']
103
104        session.permanent = True
105        session["fid"] = id     # IdをセッションIdとして格納
106
107        if passWord != check_pass:      #  パスワード(passWord)が確認用(check_pass)と
   一致しなかった時の処理
108            messege='パスワードが”確認用”と一致しませんでした。'
109            return render_template('newmember.html',uid=uid,messege=messege)
110
111        conn = get_db_connection()
112        cur = conn.cursor()
113        cur.execute(f"SELECT * FROM user_data WHERE user_id ='{id}'")
```

```
114         result = cur.fetchall()     # IDがテーブルに存在するかチェックする処理
115 
116         if result!=[]:      ##  IDがすでに存在した時の処理 ##
117             messege2='このIDは使用できません。'
118             return render_template('newmember.html',uid=uid,messege2=messege2,
119 
 id=id,name=name,born=born,email=email,op_email=op_email,passWord=passWord,check_pas
s=check_pass)
120 
121         cur.execute("INSERT INTO user_data (name, email, p_email, birth_day,
password,user_id) VALUES "
122                     f"('{name}','{email}', '{op_email}', '{born}',
'{passWord}','{id}')")
123         conn.commit()
124         cur.close()
125         conn.close()
126         return redirect(url_for('newmember_check'))
127 
128     return render_template('newmember.html',uid=uid)
129 
130 
131 
132 # 3.ログインページの処理----------------------------------------------------------------
----------------------------3
133 @app.route('/login/', methods=('GET', 'POST'))
134 def login():
135 
136     uid=uidSes()
137     # ログインされていれば、uidを取得
138 
139     if request.method == 'POST':
140         uid = request.form['id']
141         passWord = request.form['password']
142         #IDとパスワードの受け取り
143 
144         conn = get_db_connection()
145         cur = conn.cursor()
146         cur.execute(f"SELECT id FROM user_data WHERE user_id='{uid}' AND password
='{passWord}' ")
147         result = cur.fetchall()
148         conn.commit()
149         cur.close()
150         conn.close()
151 
152         if result == []:
153             messege='IDまたはパスワードが間違っています。'
154             return render_template('login.html',uid=uid,messege=messege)
155             # IDかパスワードがあっていなかったときの処理
156 
157         session.permanent = True
158         session["uid"] = uid
159         ID=result[0]
160         # print(ID,type(ID)) # result[0]中身と型の確認
161         d_id=ID[0]
162         session["id"]=d_id
163         # print(d_id,type(d_id)) # ID[0]の中身と型の確認
164         # idをセッションuidに格納
165 
166         return redirect(url_for('mypage'))
167 
```

```
168     return render_template('login.html',uid=uid)
169
170
171
172 # 4.ログアウトの処理----------------------------------------------------------------
    -------------------------4
173 @app.route('/logout/')
174 def logout():
175
176     if "uid" in session: #sessionにユーザー情報があったとき
177         session.pop("uid", None)
178         session.pop("id", None)
179         return render_template('logout.html')
180
181     return redirect(url_for('login'))
182     #sessionにユーザー情報がなかったときはloginページに遷移
183
184
185
186 # 5.マイページの処理----------------------------------------------------------------
    -------------------------5
187 @app.route('/mypage/', methods=('GET', 'POST'))
188 def mypage():
189
190     if "uid" in session: #sessionにユーザー情報があったとき
191         uid=session["uid"]
192         conn = get_db_connection()
193         cur = conn.cursor()
194
195         cur.execute("SELECT p_name,rf_date,user_id, p_id FROM product_data"
196         f" WHERE user_id ='{uid}' order by rf_date;")
197         c_products = cur.fetchall()
198         #  成果物データの「成果物名、更新日時、ID（本当は名前にしたい）」をrf_date（更新日
    時）の昇順で、10件取得
199         return render_template('mypage.html',c_products=c_products,uid=uid)
200
201     return redirect(url_for('login')) #sessionにユーザー情報がなかったときはloginページ
    に遷移
202
203
204
205 # 6.新規登録完了の処理----------------------------------------------------------------
    ---------------------------6
206 @app.route('/add_success/', methods=('GET', 'POST'))
207 def newmember_check():
208
209     uid=uidSes()
210     # ログインされていれば、uidを取得
211
212     if "fid" in session:
213         fid=session["fid"]
214         conn = get_db_connection()
215         cur = conn.cursor()
216         cur.execute(f"SELECT user_id,id,name,email,p_email,birth_day,password FROM
    user_data WHERE user_id='{fid}';")
217         u_data = cur.fetchall()
218         cur.close()
219         conn.close()
220         return render_template('newmember-check.html',u_data=u_data,uid=uid)
221
```

```
222     return redirect(url_for('login'))
223
224
225
226 # 7.1作品の新規登録の処理----------------------------------------------------------------
    -------------------------7.1
227 @app.route('/my_prd/', methods=('GET','POST'))
228 def my_prd():
229
230     uid=uidSes()
231     # ログインされていれば、uidを取得
232
233     if "uid" in session:
234         if request.method == 'POST':
235
236             name = request.form['p_name']
237             text = request.form['text']
238             code = request.form['code']
239
240             conn = get_db_connection()
241             cur = conn.cursor()
242             cur.execute(f"SELECT * FROM product_data WHERE p_name='{name}' AND
    user_id='{uid}'")
243             result = cur.fetchall()
244             conn.commit()
245             cur.close()
246             conn.close()
247
248             if result != []:
249                 messege='すでに登録されている名前です。'
250                 return render_template('my-
    prd.html',uid=uid,messege=messege,name=name,text=text,code=code)
251                 # 成果物名が被ったときの処理
252
253             session["name"]=name
254             session["text"]=text
255             session["code"]=code
256             # 次のページまで維持するため、入力内容を一時保存
257
258             return redirect(url_for('my_prd2'))
259
260         return render_template('my-prd.html',uid=uid)
261
262     return redirect(url_for('login'))
263
264
265
266 # 7.2作品の新規登録の処理2---------------------------------------------------------------
    -------------------------7.2
267 @app.route('/my_prd2/', methods=('GET','POST'))
268 def my_prd2():
269
270     uid=uidSes()
271     # ログインされていれば、uidを取得
272
273     if "uid" in session:
274         id=session["id"]
275         name = session["name"]
276         text = session["text"]
277         code = session["code"]
```

```
278         filenames=''
279         if request.method == 'POST':
280 
281             time=str(datetime.datetime.now())   # 更新日時用の時間を取得
282 
283             if 'file' not in flask.request.files:
284                 messege='ファイルが未指定です'
285                 return render_template('my-prd2.html',uid=uid,messege=messege)
286             # ファイルがなかった場合の処理
287             files = request.files.getlist('file')   # ファイルを変数に格納
288 
289             upath=f"./uploads/{uid}"
290             npath=upath+f"/{name}"
291             if not os.path.exists(upath):
292                 os.mkdir(upath)
293             os.mkdir(npath)  # ユーザ、成果物ディレクトリの作成。
294 
295             c=0 # シーケンス処理用の値
296             for file in files:
297                 if file.filename == '':   # ファイル名がなかった時の処理
298                     messege='いずれかのファイルの名前がありません'
299                     return render_template('my-prd2.html',uid=uid,messege=messege)
300 
301                 filename = file.filename
302                 # ファイル名の取り出し
303                 if c==0:
304                     filenames+=filename
305                 else:
306                     filenames+=','+filename
307                 # filenamesにファイルの名前を追加
308                 file.save(os.path.join(npath, str(c))) # ファイル名はシーケンスで保存（日本語バグを防ぐため）
309                 c+=1
310 
311             conn = get_db_connection()
312             cur = conn.cursor()
313             cur.execute("INSERT INTO product_data (p_name, p_detail, code, u_id,rf_date,user_id,f_path,file_name) VALUES "
314             f"('{name}','{text}','{code}', {id},'{time}','{uid}','{npath}','{filenames}')")
315             conn.commit()
316             cur.close()
317             conn.close()
318 
319             session.pop(name,None)
320             session.pop(text,None)
321             session.pop(code,None)
322             # 一時保存していたセッションを削除
323             return redirect(url_for('prd_add_success'))
324 
325         return render_template('my-prd2.html',uid=uid,name=name,text=text,code=code)
326 
327     return redirect(url_for('login'))
328 
329 
330 
331 # 8.作品の編集・登録完了-------------------------------------------------------------------------------.8
332 @app.route('/prd_add_success/', methods=('GET', 'POST'))
333 def prd_add_success():
```

```
334
335     if "uid" in session:
336         uid=session["uid"]
337         conn = get_db_connection()
338         cur = conn.cursor()
339         cur.execute(f"SELECT user_id,id,name,email,p_email,birth_day,password FROM
    user_data WHERE user_id='{id}';")
340         u_data = cur.fetchall()
341         cur.close()
342         conn.close()
343         return render_template('my-prd-ed-notice.html',u_data=u_data,uid=uid)
344
345     return redirect(url_for('login'))
346
347
348
349 # 9.検索処理-------------------------------------------------------------------------
    -----9
350 @app.route('/search/', methods=('GET', 'POST'))
351 def search():
352
353     uid=uidSes()
354     # ログインされていれば、uidを取得
355     word=None
356     if request.method == 'POST':
357         word = request.form["search"]
358
359         conn = get_db_connection()
360         cur = conn.cursor()
361         cur.execute(f"SELECT p_name,rf_date,user_id,p_id FROM product_data WHERE
    p_name LIKE '%{word}%';")
362         result = cur.fetchall()
363         # 成果物名の中間一致で、成果物名と更新日時、ID」を検索
364         cur.close()
365         conn.close()
366         return render_template('search.html', uid=uid,word=word,result=result)
367     return render_template('search.html',word=word,uid=uid)
368
369
370
371 # 10.成果物の詳細ページ-------------------------------------------------------------
    ----------------10
372 @app.route('/prd/<string:pid>/')
373 def prd(pid):
374
375     uid=uidSes()
376     # ログインされていれば、uidを取得
377
378     conn = get_db_connection()
379     cur = conn.cursor()
380     cur.execute(f"SELECT user_id,p_name,p_detail,code,rf_date,p_id FROM product_data
    WHERE p_id={pid};")
381     result = cur.fetchall()
382     # 成果物名の中間一致で、成果物名と更新日時、IDを検索
383     cur.execute(f"SELECT file_name FROM product_data WHERE p_id={pid};")
384     result2 = cur.fetchall()
385     cur.close()
386     conn.close()
387
388     #  DBから得た結果を','で区切られた文字列にし、split()でただのlistにする  #
```

```
    ################################## つまり、無駄な文字（(、]、"など）を省く処理##
    files=result2[0]
    l_files=list(files)
    c=0
    files=''
    for n in l_files:
        if c==0:
            files=n
        else:
            files+=','+n
        c+=1
    s_fname=files.split(',')
    ##################################

    return render_template('prd-page.html', uid=uid,result=result,s_fname=s_fname)



# 11.マイページ編集ページ----------------------------------------------------------------
-----------------------------11
@app.route('/mypage_ed/', methods=('GET', 'POST'))
def mypage_ed():

    uid=uidSes()
    # ログインされていれば、uidを取得

    if "uid" in session:
        conn = get_db_connection()
        cur = conn.cursor()

        ##### ここから、再登録ボタンを押された時の処理 ########################
        if request.method == 'POST':
            id=request.form['id']
            name = request.form['name']
            born = request.form['born']
            email = request.form['email']
            op_email = request.form['op_email']
            passWord = request.form['passWord']
            text = request.form['text']

            u_data=[] # u_dataの初期化

            cur.execute(f"SELECT
user_id,name,email,p_email,birth_day,password,ad_detail FROM user_data WHERE
user_id='{uid}';")
            u_data = cur.fetchall()

            # IDを変更したうえで、変更したIDがテーブルに存在するかチェック
            if not id==uid:
                cur.execute(f"SELECT * FROM user_data WHERE user_id ='{id}'")
                result = cur.fetchall()
                if result!=[]:
                    messege=f"ID'{id}'は使用できません"
                    return
render_template('newmember.html',uid=uid,messege=messege,u_data=u_data,

 id=id,name=name,born=born,email=email,op_email=op_email,passWord=passWord,text=text
)
                #  IDがすでに存在した時の処理
            cur.execute(f"UPDATE user_data SET name='{name}', email='{email}', "
```

```
                        f"p_email='{op_email}', birth_day= '{born}',
password='{passWord}', user_id='{id}', ad_detail='{text}' "
                        f"WHERE user_id='{uid}'")
            conn.commit()
            cur.close()
            conn.close()
            return render_template('mypage-ed-notice.html',uid=uid)
        ####  ここまで再登録ボタン押された時処理 ########################


        uid=session["uid"]
        cur.execute(f"SELECT user_id,name,email,p_email,birth_day,password,ad_detail
FROM user_data WHERE user_id='{uid}';")
        u_data = cur.fetchall()
        cur.close()
        conn.close()
        return render_template('mypage-edit.html',u_data=u_data,uid=uid)

    else:
        return redirect(url_for('login'))



# 12.1成果物編集ページ-------------------------------------------------------------
----------------------------12.1
@app.route('/prd_ed/<string:pid>', methods=('GET', 'POST'))
def prd_ed(pid):

    uid=uidSes()
    # ログインされていれば、uidを取得

    if "uid" in session:
        conn = get_db_connection()
        cur = conn.cursor()
        #  DBの接続
        cur.execute(f"SELECT * FROM product_data WHERE p_id='{pid}';")
        p_data = cur.fetchall()
        conn.commit()
        #  現在登録されているデータの取得

        ##### ここから、次へボタンを押された時の処理 ########################
        if request.method == 'POST':
            name = request.form['p_name']
            text = request.form['text']
            code = request.form['code']

            np_data=[] # p_data（現在まで登録されていたデータを取得していたリスト）の初期
化

            cur.execute(f"SELECT * FROM product_data WHERE p_id={pid} AND
p_name='{name}';")
            result = cur.fetchall()

            if result != []:#  元のデータと作品名が一致しているか確認

                cur.execute(f"SELECT * FROM product_data WHERE user_id='{uid}' AND
p_name='{name}';")
                result = cur.fetchall()

                if result == []:#  同じユーザが元のデータ以外と作品名が一致しているか確認
```

```
                    messege='既に登録されている作品名であるため、登録できません。'
                    return render_template('my-prd-edit.html',uid=uid,messege=messege,name=name,text=text,code=code)
            # IDかパスワードがあっていなかったときの処理

            session["pid"]=pid
            session["name"]=name
            session["text"]=text
            session["code"]=code
            cur.close()
            conn.close()
            #  pidを次のページに渡すため、セッションに格納（めんどくさかった）
            return redirect(url_for('prd_ed2',pid=pid,name=name,text=text,code=code))

        ####  ここまで次へボタン押された時処理 ######################

        return render_template('my-prd-edit.html',p_data=p_data,uid=uid)
        #  現在登録されているデータをページ送る

    return redirect(url_for('login'))



# 12.2成果物編集ページ 2 ------------------------------------------------------------------------------12.2
@app.route('/prd_ed2/', methods=('GET', 'POST'))
def prd_ed2():

    uid=uidSes()
    # ログインされていれば、uidを取得

    if "uid" in session:
        conn = get_db_connection()
        cur = conn.cursor()
        pid=session["pid"]
        name=session["name"]
        text=session["text"]
        code=session["code"]
        #  セッションから値を受け取り、不要なのでセッション削除

        ##### ここから、再登録ボタンを押された時の処理 ##########################################################
        if request.method == 'POST':

            files = request.files.getlist('file')
            # ファイルを変数に格納

            if 'file' not in flask.request.files:
                messege='ファイルが未指定です'
                return render_template('my-prd-edit2.html',uid=uid,messege=messege,name=name,text=text,code=code)
                # ファイルがなかった場合の処理

            cur.execute(f"SELECT f_path FROM product_data WHERE p_id={pid};")
            result = cur.fetchall()
            fpath=result[0]
            print(0,result,type(result),'\n',1,fpath,type(fpath))# 型の確認
            fpath=list(fpath)
            oldpath = f"{fpath[0]}"
```

```
552             # DBに格納されていたパスを取得し、使える型に変換
553 
554             shutil.rmtree(f"{oldpath}")
555             #  ディレクトリごとファイルを削除
556 
557             npath=f"./uploads/{uid}/{name}"
558             os.mkdir(npath)
559             # 成果物ディレクトリの作成。
560 
561             c=0
562             filenames=''
563             for file in files:
564                 if file.filename == '':     # ファイル名がなかった時の処理
565                     messege='いずれかのファイルの名前がありません'
566                     return render_template('my-
prd.html',uid=uid,messege=messege,name=name,text=text,code=code)
567 
568                 filename=file.filename
569                 if c==0:
570                     filenames+=filename
571                 else:
572                     filenames+=','+filename     # filenamesにファイルの名前を追加
573                 file.save(os.path.join(npath, str(c)))    # ファイルの保存
574                 c+=1
575 
576             time=str(datetime.datetime.now())   # 更新日時用の時間を取得
577 
578             conn = get_db_connection()
579             cur = conn.cursor()
580             cur.execute(f"UPDATE product_data SET p_name='{name}',
p_detail='{text}', code='{code}', "
581             f"rf_date='{time}',f_path='{npath}',file_name='{filenames}' WHERE p_id=
{pid}")
582             conn.commit() # データベースのデータを更新
583             cur.close()
584             conn.close()
585 
586             #  セッションいらないので削除
587             session.pop("pid",None)
588             session.pop("name",None)
589             session.pop("text",None)
590             session.pop("code",None)
591             return redirect(url_for('prd_add_success'))
592 
593         ####  ここまで再登録ボタン押された時処理
##############################################################################
594 
595         cur.execute(f"SELECT user_id,p_name,p_detail,code,rf_date,p_id FROM
product_data WHERE p_id={pid};")
596         result = cur.fetchall() # 成果物のデータを取得
597 
598         cur.execute(f"SELECT file_name FROM product_data WHERE p_id={pid};")
599         result2 = cur.fetchall() # ファイルの名前を取得
600 
601         #  DBから得た結果を','で区切られた文字列にし、split()でただのlistにする  #
602         # つまり、無駄な文字（(、]、"など）を省く処理
603         # これらの処理の結果、正しいファイルパスが得られるようになり、ファイルのダウンロー
ドページに遷移することが可能に。
604         files=result2[0]
605         l_files=list(files)
```

```
        c=0
        files=''
        for n in l_files:
            if c==0:
                files=n
            else:
                files+=','+n
            c+=1
        s_fname=files.split(',')

        cur.close()
        conn.close()
        return render_template('my-prd-
edit2.html',result=result,uid=uid,s_fname=s_fname,name=name,text=text,code=code)

    return redirect(url_for('login'))



# 13.成果物downloadページ-----------------------------------------------------------
------------------------------13
@app.route('/download/<string:f_data>', methods=('GET', 'POST'))
def download(f_data):

    fd=f_data.split(',') # fd= ['23', '工程表.xlsx']

    conn = get_db_connection()
    cur = conn.cursor()
    cur.execute(f"SELECT f_path,file_name FROM product_data WHERE p_id={fd[0]};")
    result = cur.fetchall()
    results=result[0]
    results=list(results)
    filepath=results[0] # fpath= ./uploads/yuuu/日本語変換
    filenames=results[1]      # list(tuple) >>> tuple
    fnsp=filenames.split(',')

    c=fnsp.index(fd[1]) # クリックした文字が、DBのファイル名一覧の何番目にあるか探す
    print(c,type(c))
    fpath=filepath+'/'+str(c)

    downloadFileName=fnsp[c]   # 保存時のファイル名を一意の数字にするので、
                                        #その数字をもとに名前のリストから名前を引き出す
    print(1,downloadFileName,type(downloadFileName))
    print(2,filepath,type(filepath))
    print(3,fd,type(fd))

    path2=filepath+'/'+downloadFileName

    if os.path.isfile(fpath):
        os.rename(fpath, path2)

    return send_file(path2, as_attachment = True)

    # 日本語でファイルをダウンロードする別の方法
    # しかし、パスの設定や仕組みがよくわからず、形だけ残しておく
    #response = make_response()
    #wb = open( downloadFileName, "r" )
    #response.data = wb.read()
    #wb.close()
```

```
    #response.headers['Content-Disposition'] = "attachment; filename={}".format( urllib.parse.quote( downloadFileName ))
    #response.mimetype = 'text/csv'
    #os.remove(downloadFileName)
    #return response



# 14.作成者の詳細ページ-----------------------------------------------------------------14
@app.route('/editer/<string:id>/')
def editer(id):

    uid=uidSes()
    # ログインされていれば、uidを取得

    conn = get_db_connection()
    cur = conn.cursor()
    cur.execute("SELECT p_name,rf_date,user_id, p_id FROM product_data"
                f" WHERE user_id ='{id}' order by rf_date;")
    c_products = cur.fetchall()
    # 作品のデータ

    cur.execute(f"SELECT name,p_email,ad_detail,user_id FROM user_data WHERE user_id='{id}';")
    u_data = cur.fetchall()
    # 作成者データ
    cur.close()
    conn.close()

    return render_template('editer-page.html', uid=uid,u_data=u_data,c_products=c_products)



# 15.成果物deleteページ-----------------------------------------------------------------15
@app.route('/delete/<string:pid>', methods=('GET', 'POST'))
def delete(pid):

    pid=str(pid)
    uid=uidSes()
    # ログインされていれば、uidを取得

    if "uid" in session:
        conn = get_db_connection()
        cur = conn.cursor()
        cur.execute(f"SELECT p_name,p_detail,code,rf_date,file_name,p_id,f_path FROM product_data WHERE p_id={pid};")
        p_data = cur.fetchall()
        cur.execute(f"SELECT file_name FROM product_data WHERE p_id={pid};")
        result2 = cur.fetchall()

        files=result2[0]
        l_files=list(files)
        c=0
        files=''
        for n in l_files:
            if c==0:
                files=n
```

```
            else:
                files+=','+n
            c+=1
        s_fname=files.split(',')

        if request.method == 'POST':
            cur.execute(f"DELETE FROM product_data WHERE p_id={pid};")
            conn.commit()
            p_datas=p_data[0]
            p_datas=list(p_datas)
            fpath=p_datas[6]
            print(fpath,type(fpath))#中身チェック
            shutil.rmtree(f"{fpath}")
            #  ディレクトリごとファイルを削除
            mess='削除完了'
            messege='削除が完了しました。'
            return render_template('notice.html',mess=mess,messege=messege,uid=uid)

        return render_template('delete.html',u_data=p_data,s_fname=s_fname,uid=uid)

    else:
        return redirect(url_for('login'))
```