```
from flask import Flask
from flask import Flask, flash, redirect, render_template, request, session,
abort, json, jsonify, url_for, send_from_directory
from werkzeug.utils import secure_filename
import pymysql
import requests
import os
import sys

app = Flask(__name__)
app.secret_key = "hogehoge"

table_list = []
column_list = []

def db_access(db_name, sql_query):
    conn = pymysql.connect(host='localhost',
                           user='ItsukiNagao',
                           passwd='nagaoitsuki',
                           db='%s' % (db_name),
                           charset='utf8',
                           cursorclass=pymysql.cursors.DictCursor
                           )
    try:
        with conn.cursor() as cursor:
            sql = "%s" % (sql_query)
            cursor.execute(sql)
            result = cursor.fetchall()
    finally:
        conn.close()
    return result

def db_insert(db_name, sql_query):
    conn = pymysql.connect(host='localhost',
                           user='ItsukiNagao',
                           passwd='nagaoitsuki',
                           db='%s' % (db_name),
                           charset='utf8',
                           cursorclass=pymysql.cursors.DictCursor
                           )
    try:
        with conn.cursor() as cursor:
            sql = "%s" % (sql_query)
            cursor.execute(sql)
        conn.commit()
    finally:
        conn.close()
    return "Done!"

def login_check():
    login_info = None
    if not session.get('logged_in'):
        login_info = "ログイン"
    else:
        login_info = session['user_name']
    return login_info

#ホームページ
@app.route('/', methods = ["GET", "POST"])
def index():
    login_info = login_check()
    return render_template("index.html", login_info=login_info)

#Ajax でデータベース選択時のテーブル一覧送信
@app.route('/ajax_db', methods=["GET", "POST"])
def ajax_001():
    selected_db = request.json['select_db']
    table_list.clear()
    for i in db_access(str(selected_db), str("show tables;")):
        print(i["Tables_in_" + str(selected_db)])
        table_list.append(i["Tables_in_" + str(selected_db)])
    json_for_js = []
    for h in table_list:
        json_for_js.append({"tables": h})

    return jsonify(json_for_js)

#Ajax でテーブル選択時のカラム一覧送信
@app.route('/ajax_table', methods=["GET", "POST"])
def ajax_002():
    selected_db = request.json['select_db']
    selected_table = request.json['select_table']
    column_list.clear()
    for i in db_access(str(selected_db), str("show columns from " +
selected_table + ";")):
        print(i["Field"])
        column_list.append(i["Field"])
    json_for_checkbox = []
    for h in column_list:
        json_for_checkbox.append({"columns": h})
    print("/ajax_table の jsonify の値は、"+str(json_for_checkbox))
    return jsonify(json_for_checkbox)

#Ajax でカラム選択時のデータベース結果送信
@app.route('/ajax_column', methods=["GET", "POST"])
def ajax_003():
    selected_db = request.json['select_db']
    selected_table = request.json['select_table']
    selected_columns = request.json['check_001']
    test_search_word = request.json['text_001']
    sql_query = "select "+", ".join(str(e) for e in selected_columns)+" from
"+str(selected_table)+" where "+" or ".join(str(e)+" like
'%"+test_search_word+"%'" for e in selected_columns)+";"
    print("●クエリはこちら→"+"("+sql_query+")")
    db_result = db_access(selected_db, sql_query)
    print("db_result をまるごと表示→"+str(db_result))
    return jsonify(db_result)
```

```python
#会員登録ページ
@app.route('/register', methods = ["GET", "POST"])
def register_form():
    login_info = login_check()
    return render_template("register.html", login_info=login_info)

#登録内容確認ページ
@app.route('/register_check', methods = ["GET", "POST"])
def register_check():
    login_info = login_check()
    mail_str = request.form['mail_str']
    user_str = request.form['user_str']
    passwd_str = request.form['password_str']
    sql_query = "select count(*) from member where user ='%s' or mail =
'%s';"%(user_str, mail_str)
    print("クエリは"+sql_query)
    db_result = db_access(str('final_research'), sql_query)
    print("これが問い合わせ結果→"+str(db_result[0]['count(*)']))
    if db_result[0]['count(*)'] != 0:
        Already = "そのユーザーは既に存在しています"
    else:
        Already = "登録しました！"
        #db に insert
        print("db に insert")
        sql_query = "insert into member (mail, user, password) values('%s',
'%s', '%s');"%(mail_str, user_str, passwd_str)
        db_insert("final_research", sql_query)

    return render_template("register.html", login_info=login_info,
Already=Already, MailAddress=mail_str, UserName=user_str, PassWord=passwd_str)

#ログインページ
@app.route('/login', methods = ["GET", "POST"])
def login_form():
    login_info = login_check()
    return render_template("login.html", login_info=login_info)

@app.route('/re_login', methods = ["GET", "POST"])
def home():
    user_str = request.form['username']
    passwd_str = request.form['password']
    sql_query = "select count(*) from member where user = '%s' and password =
'%s';"%(user_str, passwd_str)
    print(sql_query)
    db_result = db_access(str('final_research'), sql_query)
    print(db_result[0]['count(*)'])

    if db_result[0]["count(*)"] == 1:
        session['logged_in'] = True
        session['user_name'] = user_str+"はログイン中"
    else:
        session['logged_in'] = False
    return login_form()

#検索ページ
@app.route('/search', methods = ["GET", "POST"])
def search_form():
    login_info = login_check()
    db_result = db_access("", "show databases;")
    db_list = []
    for i in db_result:
        db_list.append(i["Database"])
    return render_template("search.html", db_list=db_list,
login_info=login_info)

UPLOAD_FOLDER = './static/uploads'
ALLOWED_EXTENSIONS = set(['pdf', 'docx', 'pptx', 'doc', 'ppt', 'txt'])
app.config['UPLOAD_FOLDER'] = UPLOAD_FOLDER

#共有ページ
@app.route('/upload', methods = ["GET", "POST"])
```

```python
def pptx_upload():
    login_info = login_check()
    try:
        if session['logged_in'] == True:
            #↓アップロード試作
            if request.method == 'POST':
                if 'file' not in request.files:
                    print('ファイルがありません')
                    return redirect(request.url)
                file = request.files['file']
                if file.filename == '':
                    print('ファイルがありません')
                    return redirect(request.url)
                if file and allwed_file(file.filename):
                    filename = secure_filename(file.filename)
                    file.save(os.path.join(app.config['UPLOAD_FOLDER'],
filename))
                return redirect(url_for('uploaded_file', filename=filename))
            #↑ここまで
            return render_template("upload.html", login_info=login_info,
upload_status="ファイルを選択してください")
        else:
            return render_template("login.html", Already="共有するにはログインしてくださ
い", login_info=login_info)
    except:
        #一度もログイン作業を行なっていないとエラーになるからそのときもログインを促す
        return render_template("login.html", Already="共有するにはログインしてください",
login_info=login_info)
    #return render_template("upload.html", login_info=login_info)

@app.route('/uploads/<filename>')
def uploaded_file(filename):
    #return send_from_directory(app.config['UPLOAD_FOLDER'], filename)
    return render_template("upload.html", login_info=login_check(),
upload_status="アップロードしました！")
```

```python
def allwed_file(filename):
    return '.' in filename and filename.rsplit('.', 1)[1].lower() in
ALLOWED_EXTENSIONS

if __name__ == '__main__':
    app.secret_key = os.urandom(12)
    app.run(host='0.0.0.0', port=5500, debug=True)
```