회원 관리 사이트 4

회원 관리 모듈

❖ model_member.py – DB 접속

```
import sqlite3
from flask import request

# db 연결(접속)
def getconn():
    conn = sqlite3.connect('./members.db')
    return conn
```

회원 관리 모듈

❖ 회원 목록

```
# 회원 목록
def select_member():
    conn = getconn()
    cur = conn.cursor()
    sql = "SELECT * FROM member ORDER BY regDate DESC"
    cur.execute(sql)
    rs = cur.fetchall()
    conn.close()
    return rs
```

회원 관리 모듈

❖ 회원 상세 보기

```
# 회원 상세 보기
def select_one(id):
    conn = getconn()
    cur = conn.cursor()
    sql = "SELECT * FROM member WHERE mid = '%s' " % (id)
    cur.execute(sql)
    rs = cur.fetchone()
    conn.close()
    return rs
```

회원 관리 모듈

❖ 회원 가입 – 자동 로그인

```
def join_member():
    id = request.form['mid']
    pwd = request.form['passwd']
    name = request.form['name']
    age = request.form['age']

    conn = getconn()  # db 연결
    cur = conn.cursor()
    sql = "INSERT INTO member (mid, passwd, name, age) VALUES " \
          "( '%s', '%s', '%s', '%s' )" % (id, pwd, name, age)
    cur.execute(sql)
    conn.commit()
```

회원 관리 모듈

❖ 회원 가입 – 자동 로그인

```
# 회원 가입후 자동 로그인
sql = "SELECT * FROM member WHERE mid = '%s' AND passwd = '%s' " % (id, pwd)
cur.execute(sql)
rs = cur.fetchone()
conn.close()
return rs
```

회원 관리 모듈

❖ 로그인

```
# 회원 로그인
def login_member():
    id = request.form['mid']  # 자료 수집
    pwd = request.form['passwd']

    conn = getconn()  # db 연결
    cur = conn.cursor()
    sql = "SELECT * FROM member WHERE mid = '%s' AND passwd = '%s' " % (id, pwd)
    cur.execute(sql)
    rs = cur.fetchone()
    conn.close()
    return rs
```

회원 관리 모듈

❖ 회원 수정

```
# 회원 수정
def update_member(id):
    # 데이터 수집
    id = request.form['mid']
    pwd = request.form['passwd']
    name = request.form['name']
    age = request.form['age']

    conn = getconn()  # db 연결
    cur = conn.cursor()
    sql = "UPDATE member SET passwd = '%s', name = '%s', age = '%s' " \
          "WHERE mid = '%s' " % (pwd, name, age, id)
    cur.execute(sql)
    conn.commit()
    conn.close()
```

회원 관리 모듈

❖ 회원 삭제

```
# 회원 삭제
def delete_member(id):
    conn = getconn()
    cur = conn.cursor()
    sql = "PRAGMA foreign_keys=ON"  #delete on cascade 작동함
    cur.execute(sql)
    sql = "DELETE FROM member WHERE mid = '%s' " % (id)
    cur.execute(sql)
    conn.commit()
    conn.close()
```

게시판 관리 모듈

❖ model_board.py 게시글 목록

```
import sqlite3
from flask import request, session

# db 연결(접속)
def getconn():
    conn = sqlite3.connect('./members.db')
    return conn

# 게시글 목록
def select_board():
    conn = getconn()
    cur = conn.cursor()
    sql = "SELECT * FROM board ORDER BY bno DESC"
    cur.execute(sql)
    rs = cur.fetchall()
    conn.close()
    return rs
```

게시판 관리 모듈

❖ 게시글 쓰기

```
def insert_board():
    title = request.form['title']
    content = request.form['content']
    mid = session.get('userName')  # 외래키 mid에 글쓴이 저장
    hit = 0

    conn = getconn()
    cur = conn.cursor()
    sql = "INSERT INTO board (title, content, hit, mid) " \
          "VALUES ('%s', '%s', %s, '%s')" % (title, content, hit, mid)
    cur.execute(sql)
    conn.commit()
    conn.close()
```

게시판 관리 모듈

❖ 게시글 상세 보기

```
# 게시글 상세 보기
def select_bo_one(bno):
    conn = getconn()
    cur = conn.cursor()
    sql = "SELECT * FROM board WHERE bno = %s " % (bno)
    cur.execute(sql)
    rs = cur.fetchone()

    # 조회수 1증가후 업데이트
    hit = rs[4]  # 조회수 칼럼 값 설정
    hit += 1  # hit 1증가
    sql = "UPDATE board SET hit = %s WHERE bno = %s" % (hit, bno)
    cur.execute(sql)
    conn.commit()
    conn.close()
    return rs
```

게시판 관리 모듈

❖ 게시글 삭제

```
def delete_board(bno):
    conn = getconn()
    cur = conn.cursor()
    sql = "DELETE FROM board WHERE bno = %s " % (bno)
    cur.execute(sql)
    conn.commit()
    conn.close()
```

게시판 관리 모듈

❖ 게시글 수정

```
def update_board(bno):
    title = request.form['title']
    content = request.form['content']
    mid = session.get('userName')  # 외래키 mid에 글쓴이 저장

    conn = getconn()
    cur = conn.cursor()
    sql = "UPDATE board SET title = '%s', content = '%s', mid = '%s' " \
          "WHERE bno = %s " % (title, content, mid, bno)
    cur.execute(sql)
    conn.commit()
    conn.close()
```

app.py – 회원 관리 부분

❖ app.py – 플라스크 객체, 인덱스 페이지

```
from flask import Flask, render_template, request, redirect, url_for, session
from libs.model_member import select_member, select_one, join_member, \
    login_member, update_member, delete_member
from libs.model_board import select_board, insert_board, \
    select_bo_one, delete_board, update_board

# Flask 객체 생성
app = Flask(__name__)

#비밀키(암호키) 설정 - 로그인 세션 발급시 필요함
app.secret_key = "#abcdef!"

# 인덱스 페이지
@app.route('/')
def index():
    return render_template('index.html')
```

app.py – 회원 관리 부분

❖ app.py – 회원 목록, 회원 가입

```python
@app.route('/memberlist/')
def memberlist():   # 회원 목록
    rs = select_member()
    return render_template('memberlist.html', rs=rs)

# 회원 가입
@app.route('/register/', methods = ['GET', 'POST'])
def register():
    if request.method == "POST":
        rs = join_member()
        if rs:
            session['userID'] = rs[0]    # 세션 발급 - mid
            session['userName'] = rs[2]  # 세션 발급 - name
        return redirect(url_for('memberlist'))
    else:
        return render_template('register.html')
```

app.py – 회원 관리 부분

❖ app.py – 회원 상세

```
@app.route('/member_view/<string:id>/')
def member_view(id):  #회원 상세 - mid를 전달받음
    rs = select_one(id)
    return render_template('member_view.html', rs=rs)
```

app.py – 회원 관리 부분

❖ app.py – 로그인, 로그아웃

```python
@app.route('/login/', methods = ['GET', 'POST'])
def login():
    if request.method == "POST":
        rs = login_member()
        if rs:
            session['userID'] = rs[0]  # 세션 발급
            session['userName'] = rs[2]
            return redirect(url_for('index'))
        else:
            error = "아이디나 비밀번호가 일치하지 않습니다."
            return render_template('login.html', error=error)
    else:
        return render_template('login.html')

@app.route('/logout/')
def logout():
    #session.pop("userID")  #세션 삭제
    session.clear()        # 모든 세션 삭제
    return redirect(url_for('index'))
```

회원 관리 모듈

❖ app.py – 회원 수정, 삭제

```python
# 회원 수정
@app.route('/member_edit/<string:id>/', methods = ['GET', 'POST'])
def member_edit(id):
    if request.method == "POST":
        update_member(id)
        return redirect(url_for('member_view', id=id))  #경로 주의
    else:
        rs = select_one(id)
        return render_template('member_edit.html', rs=rs)

# 회원 삭제
@app.route('/member_del/<string:id>/')
def member_del(id):
    delete_member(id)
    return redirect(url_for('memberlist'))
```

app.py – 게시판 관리 부분

❖ app.py – 게시글 목록, 게시글 쓰기

```python
# 게시글 목록
@app.route('/boardlist/')
def boardlist():
    rs = select_board()
    return render_template('boardlist.html', rs=rs)

# 게시글 쓰기
@app.route('/writing/', methods=['GET', 'POST'])
def writing():
    if request.method == "POST":
        insert_board()
        return redirect(url_for('boardlist'))
    else:
        return render_template('writing.html')
```

app.py – 게시판 관리 부분

❖ app.py – 게시글 상세보기, 삭제

```python
# 게시글 상세보기, 조회수 증가
@app.route('/board_view/<int:bno>/')
def board_view(bno):
    rs = select_bo_one(bno)
    return render_template('board_view.html', rs=rs)

# 게시글 삭제
@app.route('/board_del/<int:bno>/')
def board_del(bno):
    delete_board(bno)
    return redirect(url_for('boardlist'))
```

app.py – 게시판 관리 부분

❖ app.py – 게시글 수정

```
# 게시글 수정
@app.route('/board_edit/<int:bno>/', methods=['GET', 'POST'])
def board_edit(bno):
    if request.method == "POST":
        update_board(bno)
        return redirect(url_for('board_view', bno=bno))
    else:
        rs = select_bo_one(bno)
        return render_template('board_edit.html', rs=rs)

app.run(debug=True)
```