SCA.COM

SCA 웹사이트 제작

안성현

SCA.COM

TABLE OF CONTENTS

CHAPTER 00

프로젝트 포인트 키워드

통합 플랫폼 구축

- 분산된 동아리 정보를 하나로 통합
- 멤버 관리, 자료 공유,
  소통 기능을 제공
- 동아리 회원 위한 통합 서비스

실무 중심 학습

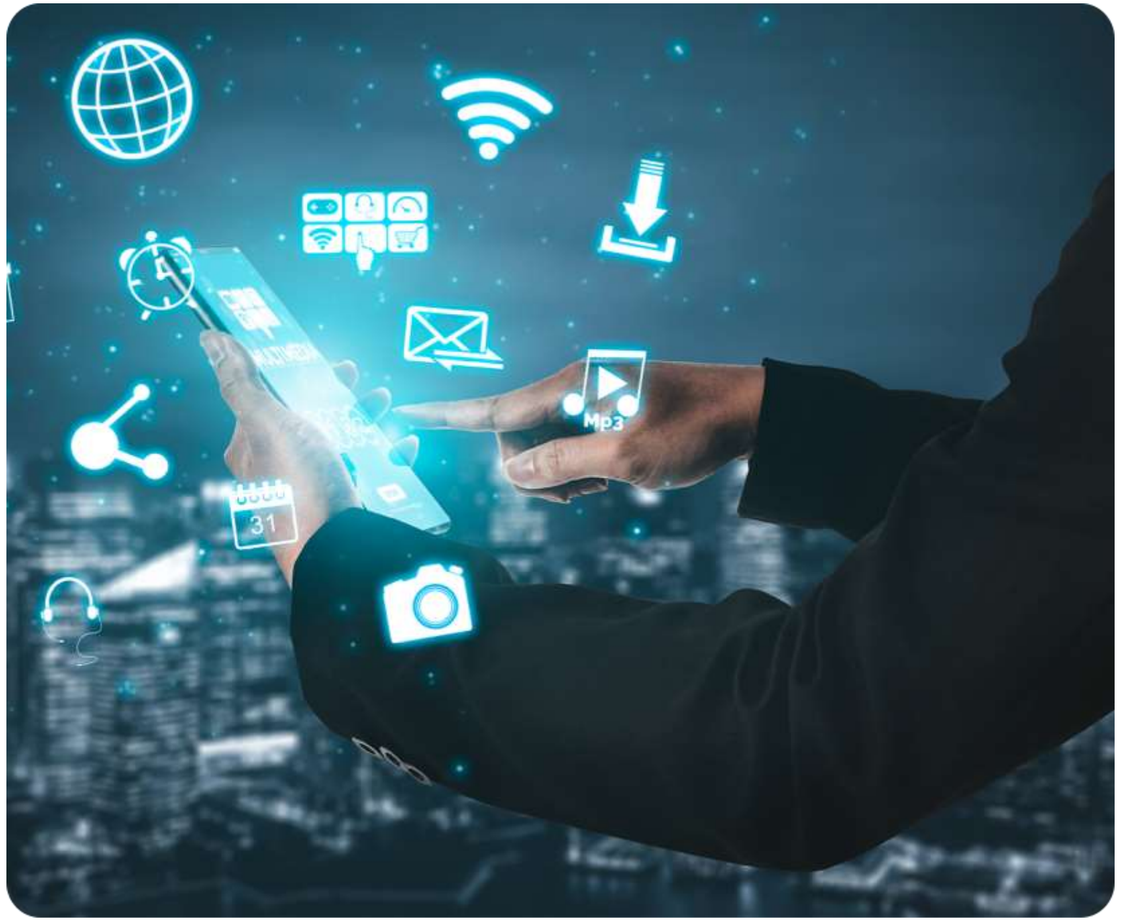
- 5주간의 풀스택 개발 경험
- 서비스 개발 및 운영
- 이론이 아닌 실제 프로젝트를
  통한 역량 향상

도전과 성장

- 실패할 때마다 더 나은
  방법을 찾으려 노력
- 포기하고 싶은 순간들을
  극복하며 끝까지 완주

CHAPTER 01

웹사이트란?



웹? 웹사이트?

Difference

웹: 전 세계 컴퓨터들이 연결된 거대한 네트워크

웹사이트: 웹 위에 만들어진 하나의 정보 공간 또는 페이지 모음

SO

HTML : **방의 구조**

CSS : **벽지, 바닥 색과 방의 크기 등**

JavaScript: **실질적인 기능을 도와주는 전기와 수도의 역할**

#웹? #웹사이트? #HTML,CSS?

CHAPTER 02

프로젝트 개요 + 기술 스택

POINT. 01

프로젝트 기간

5주간 계획, 설계, 개발, 테스트 진행

POINT. 02

개발 목적

- **동아리 활동을 체계적으로 기록**
- **동아리원들과 소통하기 위해**

POINT. 03

주요 기술 스택

Python, Flask, HTML/CSS, SQLite, SQLAlchemy

POINT. 04

보안 관련 스택

Werkzeug: **안전한 비밀번호 해시와 검증 기능 제공**

SCA.COM

웹사이트 주요 기능

프론트 엔드

백엔드

데이터베이스

CHAPTER 00

Jinja2 템플릿이란?

```
1  {% extends "base.html" %}
2  {% block content %}
3    <h2>환영합니다!</h2>
4    <p>세명 컴퓨터고등학교 해킹 전문 전공 동아리 SCA 포탈입니다.</p>
5
6    <hr>
7  {% endblock %}
```

JINJA2 템플릿?

Jinja2는 Flask에서 HTML을 동적으로 출력할 수 있도록 도와주는 템플릿 엔진

왜 필요해?

- 로그인한 사용자 이름 보여주기
- 게시판 글 목록 반복해서 출력하기
- 조건에 따라 버튼 숨기기/보이기

CHAPTER 00

SQLAlchemy + 로그인 설정

```
db = SQLAlchemy(app)
login_manager = LoginManager()
login_manager.init_app(app)
```

SQLALCHEMY?

파이썬에서 데이터베이스를 쉽게 다룰 수 있게 해주는 라이브러리

왜 쓰는가?

- **가독성코드가 직관적이고 깔끔함**
- **간편함SQL 없이도 DB 조작 가능**
- SQL Injection(**해킹**) **방지됨**

CHAPTER 03

웹사이트 주요 기능

STEP.01

게시판,공지사항

- 게시판: 동아리원 글 작성 및 소통 공간
- 공지사항: 관리자가 전달하는 중요 알림 게시판

STEP.02

로그인/회원가입

- SCA 멤버, 전공 선생님들 모두 가능한 회원가입, 로그인
- 로그인 해야 웹사이트 내 주요 기능 사용 가능

STEP.03

파일 업로드/다운로드

- SCA 해킹 분야 별 유용한 파일 업로드/다운로드 가능
- 사용자는 .txt, .pdf, .jpg, .png, .zip 등 특정 형식의 파일만 업로드 가능

STEP.04

프로필 관리

- 사용자는 자기소개 , 연락처 정보를 수정 가능
- 프로필 이미지도 업로드 가능

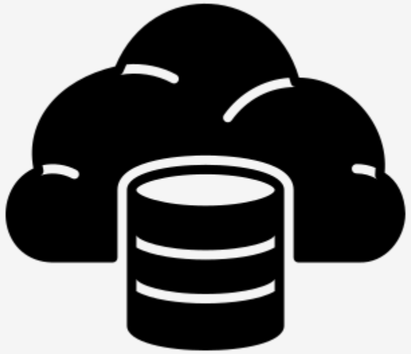
CHAPTER 04

개발 과정 및 문제 해결

01

개발 1주차

- **노션을 통해 계획 수립**
- HTML, CSS, FLASK **강의**
- **전체적인 구조를 생각**



02

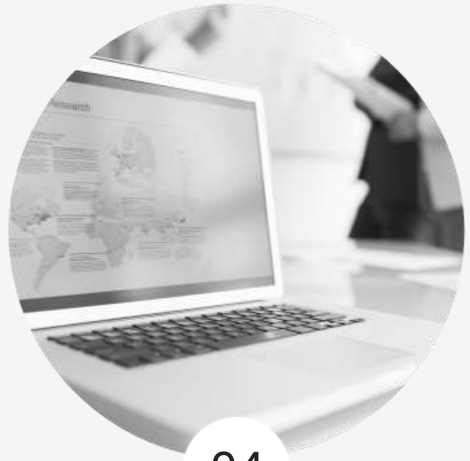
개발 2주차

- **게시판, 회원가입 기능 구현**
- **기본 라우팅 구조 작성**
- **데이터베이스 모델**
  (User, Post **등**) **설계**

03

개발 3주차

- **게시글** CRUD **기능 구현**
  (**작성, 조회, 수정, 삭제**)
- **템플릿 파일 연동**
  (Jinja2, HTML)



04

개발 4주차

- SQL Injection **등 보안 고려**
- **테스트 및 기능 점검**
- **기능 추가**

CHAPTER 04

문제점과 해결 방안

문제점

1. **로그인 후 세션 유지 어려움**
2. **비밀번호 평문 저장 문제**
3. SQL INJECTION **가능성**
4. **글 수정/삭제를 다른 유저가 할 수 있음**

해결 방안

1. FLASK-LOGIN 라이브러리로 세션 관리 자동화
2. WERKZEUG로 해시 암호화 처리
3. SQLALCHEMY ORM 사용으로 쿼리 안전 처리
4. 작성자 확인 후 수정/삭제 권한 제한

CHAPTER 04

문제 해결 방법

문제: 사용자가 로그인해도 페이지 이동 시 세션이 유지되지 않음

```
from flask_login import LoginManager, UserMixin, login_user, login_required, logout_user, current_user
```

```
login_manager = LoginManager()
login_manager.init_app(app)
login_manager.login_view = 'login'
```

```
@login_manager.user_loader
def load_user(user_id):
    return User.query.get(int(user_id))
```

```
@app.route('/board/create', methods=['GET', 'POST'])
@login_required
def post_create():
```

CHAPTER 04

문제 해결 방법

문제: 사용자의 비밀번호가 암호화되지 않고 저장됨

```
from werkzeug.security import generate_password_hash, check_password_hash
from werkzeug.utils import secure_filename
```

```
def set_password(self, password):
    self.password_hash = generate_password_hash(password)

def check_password(self, password):
    return check_password_hash(self.password_hash, password)
```

CHAPTER 04

문제 해결 방법

문제: 사용자의 입력값이 직접 SQL문에 들어가면 공격 가능

```
# 쿼리로 SQL 인젝션 기능 추가
if User.query.filter_by(username=username).first():
    flash('이미 존재하는 사용자 이름입니다.')
    return redirect(url_for('register'))
if User.query.filter_by(email=email).first():
    flash('이미 존재하는 이메일입니다.')
    return redirect(url_for('register'))
```

```
post = Post.query.get_or_404(post_id)
```

```
user = User.query.filter_by(username=username).first()
```

CHAPTER 04

문제 해결 방법

문제: 다른 사용자가 게시글/공지사항 수정, 삭제 가능

```
if post.author != current_user:
    flash('본인 게시글만 수정할 수 있습니다.')
    return redirect(url_for('post_detail', post_id=post.id))
```

```
if notice.author != current_user:
    flash('본인 공지만 수정할 수 있습니다.')
    return redirect(url_for('notice_detail', notice_id=notice_id))
```

CHAPTER 06

프로젝트 성과 및 배운 점

01

SCA 웹사이트 완성

- 목표 했던 것들의 절반이상을 구현 해 만족

02

실제 동아리 활동에 활용 가능

- 배포만 한다면 바로 사용 가능?

03

웹은 무엇인가?

- 웹에 대한 개념 상승
- 웹사이트가 어떻게 구성?

04

웹 보안의 중요성 인지

- 사용자 인증과 권한 관리, 입력 검증의 필요성 체험

CHAPTER 07

향후 계획 및 개선 사항

STEP.01

관리자 전용 페이지 구축

- 게시글, 회원 관리 편리화
- 회원 가입 승인, 권한 변경, 탈퇴 처리 기능 추가
- 부적절한 게시글/댓글 신고 및 삭제 처리

STEP.02

기능 추가

- 인기 게시글 표시 및 좋아요 버튼
- 동아리원들의 자주 묻는 질문 정리

STEP.03

UI/UX 개선

- 모바일 다양한 기기 최적화
- 사용자 친화적 인터페이스 개선

STEP.04

HTTPS 적용 및 인증서 관리

- HTTPS 적용
- 데이터 전송 시 암호화로 중간자 공격 (Man-in-the-Middle) 방지

SCA.COM

마무리