


Team project portfolio

OPICK SHOP

A faint, light green watermark logo is centered in the background. It features a stylized circular emblem with a shield-like shape inside, and the word "opick" written in a lowercase, sans-serif font below it.

Team : 5Ping Mall

Production : 성득명 빙승현 지수정 김태연

Production date : 25.09.22 ~ 25.10.14

Categories

Introduction

^

- Select a Website

Project Plan

v

Implementation

v

Evaluation

v

Finish



Opick shop

- 직접 제작한 본인의 작품을 거래하는 온라인 쇼핑몰
- 심플하지만 함축적으로 필요한 정보들을 담고 있는 레이아웃 형태로 간단히 표현
- 아이디어스, 숨고, 텀블벅 등의 여러 거래 사이트 참고

Categories

Introduction ▾

Project Plan ▲

- Persona

- Prototype

Implementation ▾

Evaluation ▾

Finish



이름 : 윤소연
직업 : 조향사
취미 : 향수 제작

요구 사항

- 본인의 작품 판매 및 다른 작가의 작품 구매를 위한 사이트
- 기업보단 개인 작품 판매의 비중이 큰 사이트 요청



이름 : 이승연
직업 : 도자기 공방 원데이클래스 운영
취미 : 도자기 공예

요구 사항

- 본인의 작품 판매가 쉽게 가능한 사이트
- 카테고리기가 명확히 나눠져있는 사이트 요청

Categories

Introduction

v

Project Plan

^

- Persona

- Prototype

Implementation

v

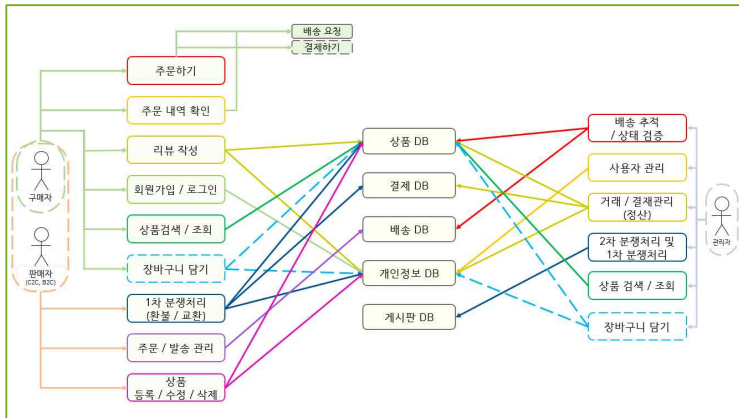
Evaluation

v

Finish

1차 구상 (Diagram)

- UseCase



Categories

Introduction

v

Project Plan

^

- Persona

- Prototype

Implementation

v

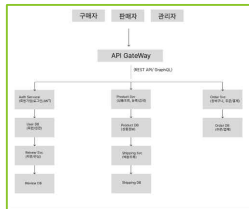
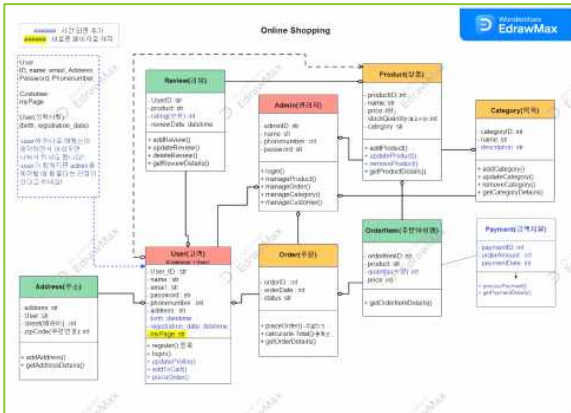
Evaluation

v

Finish

1차 구상 (Diagram)

- Class



Categories

Introduction

Project Plan

- Persona

- Prototype

Implementation

Evaluation

Finish

1차 구상 (Diagram)

- other

| DB 구상 리스트 | | | | | | | | |
|-----------|---------|----------|-------------|----------|----------|-------------|----------|----------|
| User | | | Product | | | Comment | | |
| 필수명 | 데이터 타입 | nullable | 필수명 | 데이터 타입 | nullable | 필수명 | 데이터 타입 | nullable |
| id | int | 자정 X | id | int | 자정 X | id | int | 자정 X |
| userid | str | F | title | str | F | content | str | F |
| password | str | F | content | str | F | create_date | datetime | F |
| username | str | F | create_date | datetime | F | modify_date | datetime | T |
| email | str | F | modify_date | datetime | T | | | |
| phone | str | F | | | | | | |
| address | str | F | | | | | | |
| admin | boolean | F | | | | | | |

필수 기능:

1. 회원가입/로그인
2. 상품검색/조회 (기본)
3. 장바구니 담기 (결제 전)
4. 주문하기 (배송 핵심)
5. 주문 내역 확인 (불만방지)
6. 배송조회/관리 (배송DB관리/주문이후 핵심)
7. 결제처리 (거래 성립)
8. 관리자-상품등록/삭제/조회 (상품DB관리/쇼핑몰 운영)
-거래내역 관리(정산)

구매자(User): 회원가입/로그인 → 상품검색 → 장바구니(?) → 주문 → 결제 → 배송 조회 → 리뷰작성
관리자(Admin): 회원/상품/정산 관리, 분쟁 처리

판매자: 상품 등록/수정/삭제, 주문관리, 리뷰 확인
관리자에 판매자 기능까지 포함시킬건지, 그

선택 기능:

1. 리뷰작성
2. 관심상품 담기 (편의기능)
3. 1차/2차 분쟁처리 (환불/교환)
4. 관리자-사용자 관리 (규모 커질때 필요)
5. 상품 등록 특성/검사(심사)
6. 배송주최 세무 일임/처리 상세 기능
(기본 배송조회만 있으면 됨-출생)

쇼핑몰

개인 / 관리자

개인 로그인 회원가입 마이페이지 상품등록 댓글
관리자 회원관리

<필수>

기능 구현할 명세서
로그인 회원가입 연동 기능
검색
채팅기능(q&a)

카테고리(상품위주)

-상품 등록 (일차순)
신규/인기 이름, 가격, 추천순 조회(?)순 구현 가능한다면
글자 수 사진 크기 제한

-메인페이지 물건((슬라이더))

고객지원 클릭 시 게시판 이동
질문 → 답변

자주 묻는 질문 FAQ

예상질문 답변 써주기(게시판 최상단) 디바사용x
공지사항(?)

Categories

Introduction v

Project Plan ^

- Persona

- Prototype

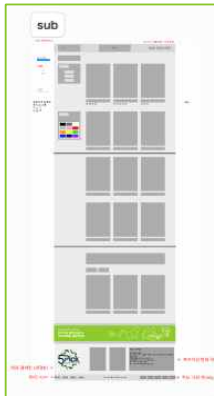
Implementation v

Evaluation v

Finish

2차 기본 코드 작성 및 main, sub 와이어프레임 제작

- Figma, PyCharm, Flask, SQLite, vscode(html, css, javascript) 사용



Categories

Introduction

v

Project Plan

^

- Persona

- Prototype

Implementation

v

Evaluation

v

Finish

3차 로고 디자인 및 sign in, sign up, my page 와이어프레임 제작



최종



Categories

Introduction

v

Project Plan

v

Implementation

^

- Main page

- Sign in / Sign up

- Sub page

Evaluation

v

Finish



Header - 로고, 로그인/회원가입/고객지원(-> 로그아웃/마이페이지/고객지원), 카테고리

Slider - 메인 슬라이드

Section - 이벤트, 카테고리 각각의 상품, 쿠폰팩 등의 기능과 자주 묻는 질문

Footer - 페이지 설명과 회사 소개

Categories

- Introduction
- Project Plan
- Implementation
- [- Main page](#)
- Sign in / Sign up
- Sub page

Evaluation v

Finish

마이페이지

- ✓ 로그인 후 오른쪽 상단의 '마이페이지' 클릭 시 본인의 개인정보 변경과 상품 등록, 댓글 기록 조회가 가능하도록 기능 구현.
- ✓ 상품 등록 버튼을 누르면 상품 등록 페이지로 넘어가도록 설정.

[상품등록 페이지](#)

- ✓ 제목, 할인율, 가격, 카테고리 선택, 이미지 업로드 기능 구현.
- ✓ 상품 등록을 하면 '마이페이지' -> '상품 등록 기록 조회' 칸 추가.

Categories

Introduction

Project Plan

Implementation

- Main page

- Sign in / Sign up

- Sub page

Evaluation

Finish

Slider / Section

- ✓ 페이지 처음과 중간 부분에 이벤트/쿠폰픽 슬라이드.
- ✓ 추석 특가 이벤트 슬라이드로 할인상품 정리.
- ✓ 카테고리 상품 슬라이드가 자동으로 한 열씩 넘어가도록 설정하고 각각의 상품도 2-3장씩 자동 슬라이드 되도록 구현.

```

< Slider를 사용하기 >
var helpers = new helpers("helpers", {
  loop: true,
  slidesToShow: 1,
  grid: {
    rows: 2,
  },
  autoplay: {
    delay: 2000,
    disableInteraction: false,
  },
  responsive: {
    navigation: {
      next: "swiper-button-next",
      prev: "swiper-button-prev",
    },
  },
});

const helpersItem = new helpers("helpersItem", {
  loop: true,
  autoplay: {
    delay: 2000,
    disableInteraction: false,
  },
  responsive: {
    navigation: {
      next: "swiper-button-next",
      prev: "swiper-button-prev",
    },
  },
});

```

```

< Slider를 사용하기 >
var helpers = new helpers("helpers", {
  loop: true,
  autoplay: {
    delay: 2000,
    disableInteraction: false,
  },
  responsive: {
    navigation: {
      next: "swiper-button-next",
      prev: "swiper-button-prev",
    },
  },
});

const helpersItem = new helpers("helpersItem", {
  loop: true,
  autoplay: {
    delay: 2000,
    disableInteraction: false,
  },
  responsive: {
    navigation: {
      next: "swiper-button-next",
      prev: "swiper-button-prev",
    },
  },
});

```

```

< Slider를 사용하기 >
var helpers = new helpers("helpers", {
  loop: true,
  slidesToShow: 1,
  grid: {
    rows: 2,
  },
  autoplay: {
    delay: 2000,
    disableInteraction: false,
  },
  responsive: {
    navigation: {
      next: "swiper-button-next",
      prev: "swiper-button-prev",
    },
  },
});

const helpersItem = new helpers("helpersItem", {
  loop: true,
  autoplay: {
    delay: 2000,
    disableInteraction: false,
  },
  responsive: {
    navigation: {
      next: "swiper-button-next",
      prev: "swiper-button-prev",
    },
  },
});

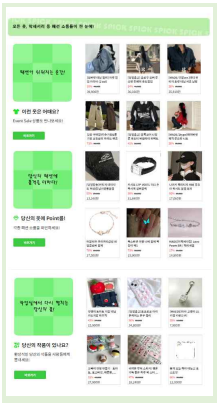
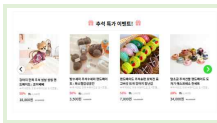
```

```

< Slider를 사용하기 >
var helpers = new helpers("helpers", {
  loop: true,
  slidesToShow: 1,
  grid: {
    rows: 2,
  },
  autoplay: {
    delay: 2000,
    disableInteraction: false,
  },
  responsive: {
    navigation: {
      next: "swiper-button-next",
      prev: "swiper-button-prev",
    },
  },
});

const helpersItem = new helpers("helpersItem", {
  loop: true,
  autoplay: {
    delay: 2000,
    disableInteraction: false,
  },
  responsive: {
    navigation: {
      next: "swiper-button-next",
      prev: "swiper-button-prev",
    },
  },
});

```



Categories

[로그인](#) [회원가입](#) [고객지원](#)

[illegible]

✓ 기본적인 개인정보와 아이디 비밀번호를 입력하면
회원가입이 진행되는 기능을 구현.

✓ 또한 비회원 상태로 상품 클릭 시 회원가입 창으로 이동되도록 설정.

```

class UserCreateForm(FlaskForm):
    def __init__(self, *args, **kwargs):
        self.__password_validation = PasswordValidation()

    user_id = StringField([_('아이디(아이디)'), validators.[DataRequired('아이디를 입력해주세요')], Length(min=3, max=25)])
    username = StringField([_('이름'), validators.[DataRequired('이름을 입력해주세요')], Length(min=1, max=25)])
    password1 = PasswordField([_('비밀번호'), validators.[DataRequired('비밀번호를 입력해주세요')], EqualTo([_('비밀번호')], message='비밀번호가 일치하지 않습니다.')])
    password2 = PasswordField([_('비밀번호를 다시 입력하세요'), validators.[DataRequired('비밀번호를 다시 입력해주세요')], EqualTo([_('비밀번호')], message='비밀번호가 일치하지 않습니다.')])
    phone = StringField([_('전화번호'), validators.[DataRequired('전화번호를 입력해주세요')], Length(min=10, max=15)])
    email = EmailField([_('이메일'), validators.[DataRequired('이메일을 입력해주세요')], Email()])
    address = StringField([_('주소'), validators.[DataRequired('주소를 입력해주세요')]]) # 국가, 지방, 도로, 우편, 집안, 부속
    image = FileField([_('프로필 사진'), validators.[FileAllowed(['jpg', 'png', 'gif'])]])

```

[illegible]

```
def login_required(view):
    @functools.wraps(view)
    def wrapped_view(*args, **kwargs):
        if g.user is None:
            next = request.url if request.method == 'GET' else ''
            return redirect(url_for(endpoint='auth.login', next=next))
        return view(*args, **kwargs)
    return wrapped_view
```

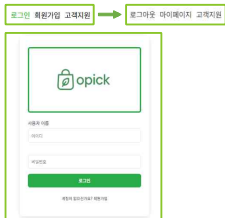
```
dbp.route('/signup', methods=['GET', 'POST']) &lt;@200902110> +1
def signup():
    form = UserCreateForm()
    if request.method == 'POST' and form.validate_on_submit():
        user = User.query.filter_by(username=form.username.data).first()
        if not user:
            user = User(username=form.username.data,
                        password=generate_password_hash(form.password1.data),
                        email=form.email.data,
                        phone=form.phone.data,
                        address=form.address.data)
            db.session.add(user)
            db.session.commit()
            return redirect(url_for('main.index'))
        else:
            flash('이미 존재하는 사용자입니다.')
    return render_template('template_name_or_hic', auth/signup.html', form=form)
```

Categories

- Introduction
- Project Plan
- Implementation
 - Main page
 - Sign in / Sign up
 - Sub page

Evaluation v

Finish



- ✓ 회원가입 진행 후 로그인을 할 경우 '로그인/회원가입'에서 '로그아웃/마이페이지'로 옵션 변경 기능 구현.
- ✓ 로그아웃 시에 모든 session 삭제 후 메인화면으로 이동.
- ✓ 로그인 페이지에서 계정이 없을 경우, 하단의 '회원가입'을 선택해 회원가입 페이지로 이동.

```

class UserLoginForm(FlaskForm):
    """User login form"""

    username = StringField(lable='사용자 이름', validators=[DataRequired('아이디를 입력해주세요.'), Length(min=3, max=25)])
    password = PasswordField(lable='', validators=[DataRequired('비밀번호를 입력해주세요.')])

# 로그인 폼을 검증하고 유효하지 않다면 flash 메시지로 전달한다
def validate(self):
    if self.username.data == '' or self.password.data == '':
        self.errors.append("이름과 비밀번호를 입력하십시오.")
        return False
    user = User.query.filter_by(username=self.username.data).first()
    if not user:
        self.errors.append("해당 아이디의 유저가 없습니다.")
        return False
    if not check_password_hash(user.password, self.password.data):
        self.errors.append("비밀번호가 일치하지 않습니다.")
        return False
    return True

@login_required
@app.route('/logout')
def logout():
    session.clear()
    return redirect(url_for('main.index'))

# 로그인 폼을 렌더링하는 함수
def render_login_form():
    form = UserLoginForm()
    if request.method == 'POST' and form.validate_on_submit():
        error = None
        user = User.query.filter_by(username=form.username.data).first()
        if not user:
            error = "해당 아이디의 유저가 없습니다."
        elif not check_password_hash(user.password, form.password.data):
            error = "비밀번호가 일치하지 않습니다."
        if error is None:
            session['user_id'] = user.id
            next_ = request.args.get('next', '')
            if _next:
                return redirect(next)
            else:
                return redirect(url_for('main.index'))
    flash(error)
    return render_template("templates/auth/login.html", form=form) & GET 방식도 로그인 요청시 html 렌더링 시켜주도록

```

Categories

Introduction ▾

Project Plan ▾

Implementation ▲

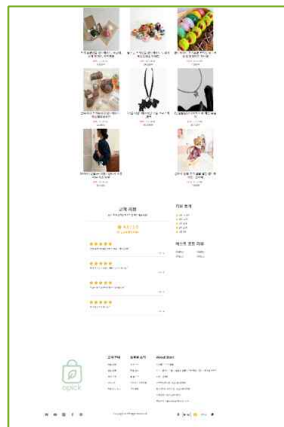
- Main page

- Sign in / Sign up

- Sub page

Evaluation ▾

Finish



Header - 로고, 로그인/회원가입/고객지원 (-> 로그아웃/마이페이지/고객지원), 카테고리

Section - Filter, Size, 상품, 고객 리뷰, 리뷰 통계, 베스트 포토 리뷰

Footer - 페이지 설명과 회사 소개

Categories

Introduction



Project Plan



Implementation



- Main page

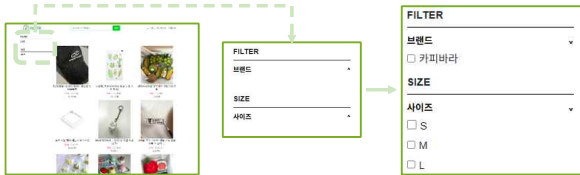
- Sign in / Sign up

- Sub page

Evaluation



Finish



```

<!-- Filter Section -->
<div class="filter">
  <div class="filter-section">
    <div class="filter-title">
      <h3>FILTER</h3>
    </div>
    <div class="filter-list">
      <div class="filter-item">
        <div class="filter-label">
          <h4>브랜드</h4>
        </div>
        <div class="filter-options">
          <div class="filter-option">
            <input type="checkbox"/> 카피바라
          </div>
        </div>
      </div>
      <div class="filter-item">
        <div class="filter-label">
          <h4>사이즈</h4>
        </div>
        <div class="filter-options">
          <div class="filter-option">
            <input type="checkbox"/> S
          </div>
          <div class="filter-option">
            <input type="checkbox"/> M
          </div>
          <div class="filter-option">
            <input type="checkbox"/> L
          </div>
        </div>
      </div>
    </div>
  </div>
</div>

```

```

<!-- Filter Section -->
<div class="filter">
  <div class="filter-section">
    <div class="filter-title">
      <h3>FILTER</h3>
    </div>
    <div class="filter-list">
      <div class="filter-item">
        <div class="filter-label">
          <h4>브랜드</h4>
        </div>
        <div class="filter-options">
          <div class="filter-option">
            <input type="checkbox"/> 카피바라
          </div>
        </div>
      </div>
      <div class="filter-item">
        <div class="filter-label">
          <h4>사이즈</h4>
        </div>
        <div class="filter-options">
          <div class="filter-option">
            <input type="checkbox"/> S
          </div>
          <div class="filter-option">
            <input type="checkbox"/> M
          </div>
          <div class="filter-option">
            <input type="checkbox"/> L
          </div>
        </div>
      </div>
    </div>
  </div>
</div>

```

```

document.querySelectorAll(".filter-toggle").forEach(button => {
  button.addEventListener("click", () => {
    const section = button.parentElement;
    section.classList.toggle("active");
  });
});

```

```

document.querySelectorAll(".size-toggle").forEach(button => {
  button.addEventListener("click", () => {
    const section = button.parentElement;
    section.classList.toggle("active");
  });
});

```

✓ 브랜드와 사이즈 필터를 이용해 클릭 시 클릭한 필터에 해당되는 품목들만 나오도록 기능 구현.

Categories

Introduction



Project Plan



Implementation



- Main page

- Sign in / Sign up

- Sub page

Evaluation



Finish

```


{% if products %}
  {% for p in products %}
  <div class="product-card-wrapper">
    <a href="{% url_for('items_detail', product_id=p.id) %}">
    <div class="product-card" data-color="{% if p.color or 'default' %}">
      {% if p.images %}
      <!-- 이미지가 있으면 -->
      
      {% else %}
      <!-- 이미지가 없으면 기본 이미지 -->
      
      {% endif %}
      <h3>{{ p.title }}</h3>
      <div class="detail">
        <span>{{ p.discount }}%</span>
        <p>{{ '{:,.format( p.price )}}원</p>
      </div>
      <p class="discount">{{ '{:,.format( p.price * (1 - p.discount / 100))round(0)|int )}}원</p>
      </div>
    </div>
  {% endif %}
  {% else %}
  <p style="text-align: center;">등록된 상품이 없습니다.</p>
  {% endif %}
</div>


```

- ✓ 마이페이지에서 등록된 상품이 최신순으로 나열되도록 설정.
- ✓ 같은 카테고리가 아닌 전체 품목이 나오도록 디자인.
- ✓ 작성한 리뷰가 sub page 하단에 나오도록 front로 제작.
- ✓ 품목 이미지가 없으면 기본 이미지가 나오도록 설정



Categories

Introduction ▾

Project Plan ▾

Implementation ▾

Evaluation ▲

- 성득명, 빙승현

- 지수정, 김태연

Finish



성득명

적어달라
맑은고딕 14 라고 한다 ((태연이가))



빙승현

Categories

Introduction ▾

Project Plan ▾

Implementation ▾

Evaluation ▲

- 성득명, 빙승현

- 지수정, 김태연

Finish



지수정

팀 프로젝트인 만큼 정해진 역할을 잘 수행했어야 했는데 많이 미흡하고 이해 못 하는 부분이 많아서 만족할 만한 성과가 나오지 않은 것 같아서 아쉬운 것 같았다.

프론트와 백을 같이 작업해보았는데 이해도가 떨어지다 보니 백을 만드는 과정에서 어려움이 많이 찾아왔지만, 열심히 배우고 물어가며 작성을 해보았다. 하지만, 연결 과정에서 겹치고 오류나는 상황이 발생하여 관련 지식의 이해도와 팀 소통, 협업이 중요한 것 같다는 생각을 하게 되었다.

이번에는 다소 아쉬운 점이 많았지만, 점차 시간이 갈 수록 지금보다 더 발전하고 더 좋은 작업물이 나올 수 있도록 노력할 것이다.



김태연

팀 프로젝트를 진행하면서 기본적으로 잡혀있는 틀이 없었기 때문인지 처음 구상부터 어려움을 많이 느꼈지만, 각자 잘 하는 분야를 내세우니 생기는 각종 문제들을 해결해 나갈 수 있었다.

개인적으로는 백보다 프론트에 집중했었어서 백에 그다지 큰 비중은 없지만 프론트를 두고 봤을 때 조금 더 디테일을 잡을 수 있었을 텐데 하며 아쉬움이 많이 남는 프로젝트였다. 그럼에도 그만큼 적은 시간에 최대한의 디테일을 잡을 수 있는 방법 또한 배울 수 있었기에 다음 프로젝트를 진행하기 전 조금 더 성장함을 느꼈다.

이번 프로젝트에서도 저번 프로젝트와 다를 바 없이 열심히 참여했고, 문제점을 받아들이며 해결해나가는 마음가짐도 이어졌다. 다음 프로젝트까지 초심 잃지 않고 노력해서 더욱 발전할 것이라 확신한다.

Categories

[Introduction](#) ▾[Project Plan](#) ▾[Implementation](#) ▾[Evaluation](#) ▲[Finish](#)

좋은 아이디어를 가지고 있으신가요?

5PICK과 함께 마음 속
프로젝트를 실현하세요



Thank you!

[Main page link](#)[Sub page link](#)

Opick shop - 5ping mall