

인테리어 추천 사이트 분석

5TS (5 tea spoon)



집, 家

빅데이터 자바 개발자

조원 : 고예린, 김승환, 김태훈, 김선종



목차

1. 프로젝트 개요

1. 프로젝트 주제 선정
2. 개발 환경

2. 프로젝트 단계

1. 구성 설계
2. 화면 설계
3. 데이터 수집
4. 데이터 가공
5. 데이터 분석 및 웹 구현



1. 프로젝트 개요 프로젝트 주제 선정

“개인적인 공간이었던 자신의 집 공간들을 자신의 인테리어와
다른 사람의 인테리어를 비교 혹은 공유하는 플랫폼 개발

데이터 분석

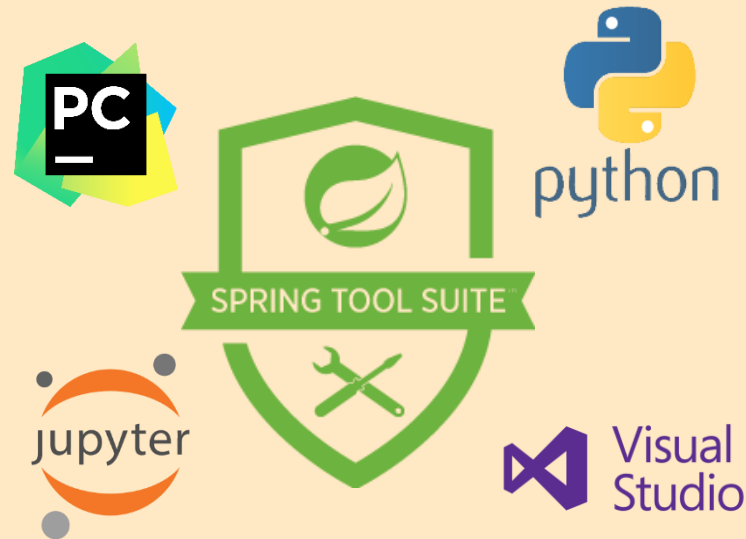


웹 구현

인테리어 추천 사이트 분석



1. 프로젝트 개요 개발 환경





2. 프로젝트 단계 구성 설계



블로그

글 (업로드, 사진, 글, 댓글, 좋아요)

블로그 랭킹

분석: 글 읽은 사람의 회원정보를 바탕으로 블로그 추천

비슷한 성향의 사람의 블로그(인테리어)를 매칭



가구추천

네이버, 다음 카페, 인테리어 사이트 등 내에서 긍정적이고 핫한 인테리어를 찾아주기

(네이버 리뷰를 분석(tf-idf)해서 머신러닝으로 분류 후 카페 리뷰를

분석해서 랭킹을 직접 매김)

여러 카페내의 랭킹을 모두 매김



가구매칭

가구를 입력하면 어울리는 인테리어를 매칭

(인테리어 글에 올라와있는 사진들의 색을 추출해서 추천)

차트에 있는 색 클릭 시 해당 색의 인테리어 매칭 .



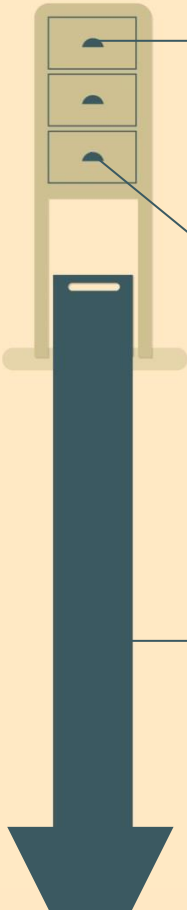
회원관리 / 판매점

인테리어 판매점 검색

판매점의 실제 매장 위치 표시

(회원가입, 로그인, 회원정보 수정, 로그아웃, 메인화면)

프로젝트 과정 (기간/ 총 20일)



● 프로젝트 수행 준비 (10일)

아이디어 구상
화면설계서 구상
WorkFlow 구상
ERD 설계 구상
기능설명서 구상
Database 구상

● 전체 기능 분석 및 설계 (5일)

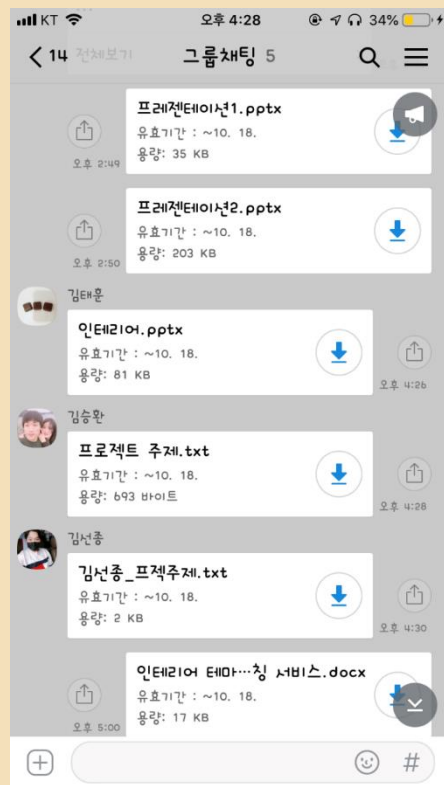
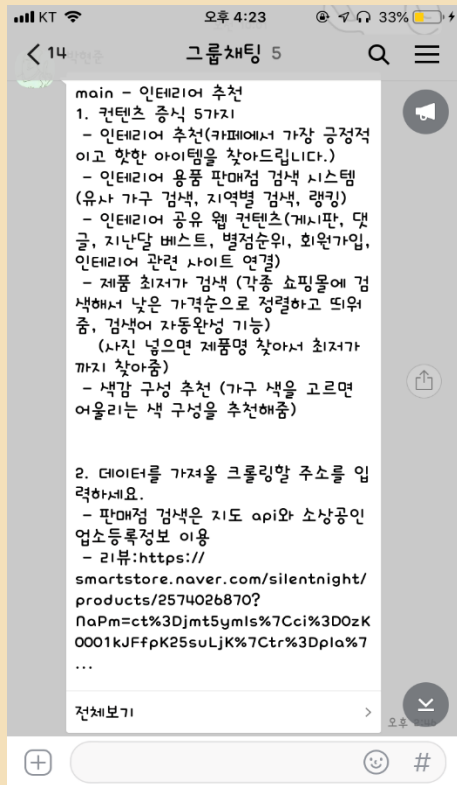
화면 설계서 작성
WorkFlow 작성
ERD 설계 작성
기능 설명서 작성
분석/설계 확인 및 변경, 보완
Database 설계

● 개발 (11일)

블로그 : 김태훈
가구추천 : 김승환
가구매칭 : 김선종
회원관리 / 판매점 / css : 고예린

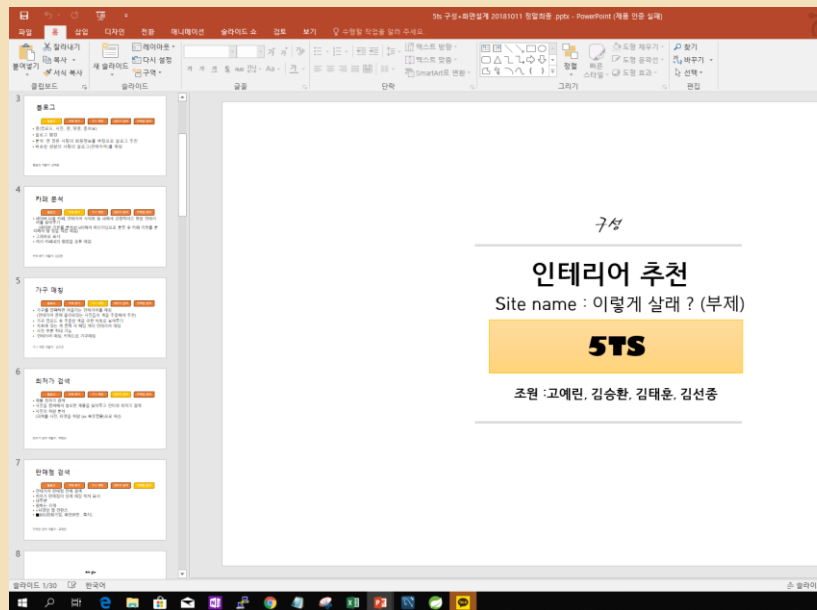
프로젝트 과정 (기간/ 총 20일)

프로젝트 수행 준비 (10일)



프로젝트 과정 (기간/ 총 20일)

전체 기능 분석 및 설계 (5일)



1.4	개발 기술 연구 기간	정재
1.4.1	블로그	김태훈
1.4.2	카페 분석	김승환
1.4.3	가구 매칭	김선준
1.4.5	판매글 검색	고예린
1.5	일 개발	
1.5.1	블로그 글 쓰기, 수정, 삭제, 사진 올리기 리뷰 좋아요, 블로그 전체 페이지 구성	김태훈
1.5.2	카페 분석 카피 리스트 등록 세부사항 댓글 작성/수정/삭제	김승환
1.5.4	회원 관리 회원가입 / 로그인, 회원가입, 회원가입 완료 화면 회원 정보 수정, 비밀번호 수정	고예린
1.5.5	가구 매칭 사진 업로드 인테리에 추천결과 색종이를 걸고 시작함	김선준
1.5.5	가구 매칭 시작화면 그래픽을 클릭 후 해당 색 인테리에 결과 보여주기	김선준
1.5.7	지도 가구 판매글 검색	고예린
1.6	크롤링	
1.6.1	네이버리뷰 수집 네이버 쇼핑 크롤링(리뷰, 가격)	김승환
1.6.2	카페리뷰 수집 카페 크롤링(리뷰, 이미지)	김승환
1.6.3	인테리어 수집 인테리어 사진들 저장	김선준
1.7	분석	
1.7.1	회원정보 분석 (성별, 연령대), 시작함	김태훈
1.7.2	블로그 추천 (같은 스타일 순, 인기순(좋아요순))	김태훈
1.7.3	행위 블로그 동침 (좋아요순)	김태훈
1.7.4	리뷰 분석 (정확도 분석 시작함)	김태훈
1.7.5	분석 feature - 네이버 쇼핑 리뷰 텍스트 label - 네이버 쇼핑 리뷰 평점 predict feature - 카페 별 가구 리뷰 텍스트 predict result - 카페 결과 및 점 (1.5점 정도 제외) (1.7.6 모델 이용)	김승환
1.7.6	행위 카페 리뷰에 대한 평점 예측하고 행진 부여 (행진+포인트+좋아요)	김승환
1.7.8	이미지 분석 이미지 전체 색 추출 -> 추출한 결과 중 가장 많은 비율을 차지하는 색을 분류해주는 모델 만들기 (feature - RGB, label - 색상)	김선준
1.7.9	가구 매칭 인테리어의 색을 분류 가구 업로드 시 색을 분류하여, 분류된 인테리어 색과 매칭 (현재 조원 해보고 만족할만한 결과가 안 나오면, PCA, NMF, Tensorflow, PyTorch)	김선준
1.9	css	
1.9.3	모든 페이지	고예린



인테리어 추천 사이트 화면



홈

블로그

가구추천

가구매칭



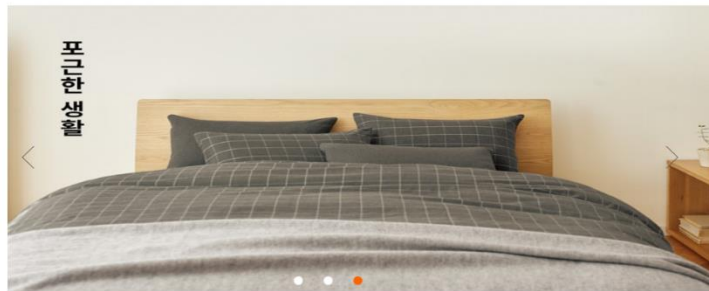
블로그 랭킹

more



블로그 랭킹

more



가구 추천 · 사이트 목록

more

BENS

CHERISH

ARIA Change Your Lifestyle FURNITURE

Ramerit

매장 정보





인테리어 추천 사이트 화면

가구 추천 . 사이트 목록



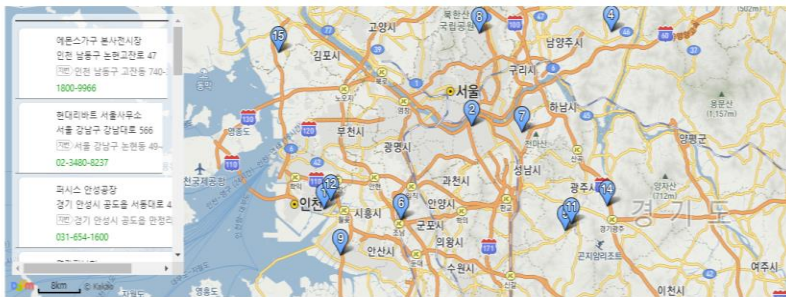
ARIA Change Your Lifestyle
FURNITURE

Ramerit

more



매장 정보



LOGIN !

ID

아이디를 입력하세요

PASSWORD

비밀번호를 입력하세요

☐ 입력을 기억합니다

Log In

Cancel



FIVE TEA SPOON , 5TS



FIVE TEA SPOON , 5TS

고객센터: 김선홍, 김순환, 김학문

집가 (집에 가자) 이메일: interior_5ts@naver.com

Copyright © 2018 by Five Tea Spoon. All Rights Reserved.

본 페이지에 사용된 이미지와 사진, 설명등을 무단 도용시 강력해신당법에 의거 형사처벌 등을 당하게됩니다.



인테리어 추천 사이트 화면

✓ JOIN US !

PROFILE upload



ID

helloHouse

PASSWORD

..

NAME

joinHouse

BIRTHDAY

19950122

SEX

☐ 남자 ☒ 여자

INTERIORSTYLE

모던
모던
복유합
빈티지
내추럴
프로방스&로맨틱
클래식&엔틱

Cancel



Modify user!

PROFILE upload



ID

hello

PASSWORD

INTERIOR STYLE

모던

Modify

Cancel



hello님 안녕하세요!

LOGOUT

modify



FIVE TEA SPOON , 5TS




FIVE TEA SPOON , 5TS



인테리어 추천 사이트 화면

[홈](#)[블로그](#)[가구추천](#)[가구매칭](#)[글쓰기](#)

블로그 랭킹 



블로그 소개

블로그 소개



22 님의 예쁜 블로그입니다.

[눌러가기](#)

조회수 6



smho11 님의 예쁜 블로그입니다.

[눌러가기](#)

조회수 6



FIVE TEA SPOON, 5TS

고예린, 김선영, 김승람, 김태훈

집가 (집에 가) 이메일 : interior_5ts@naver.com

Copyright © 2018 by Five Tea Spoon. All Rights Reserved.

본 페이지에 사용된 이미지와 사진, 설명등을 무단 도용시 정보통신망법에 의거 형사처벌 등을 알려드립니다.



인테리어 추천 사이트 화면

홈

블로그

가구추천

가구매칭

글쓰기



22.님의 블로그입니다
'모던한 스타일을 좋아해요'

작성한 글



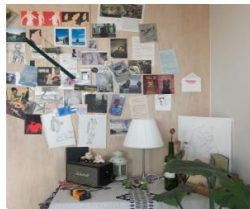
안녕친구
작성일 2018-11-01 11:07:28.0



나는야 두번째 댄싱킹 후니
작성일 2018-10-31 14:49:58.0



나는야 댄싱킹 후니
작성일 2018-10-31 14:46:55.0



캠성
작성일 2018-10-31 14:52:32.0

비슷한 취향의 블로그 추천



볼러가기
나는야 댄싱킹 후니
2018-10-31 14:46:55.0



볼러가기
나는야 두번째 댄싱킹 후니
2018-10-31 14:49:58.0



인테리어 추천 사이트 화면

BLOG IMAGE UPLOAD

TITLE

글쓰기!

INTERIOR STYLE

모던 ▼

좋은 글만 써주세요!!

WRITE!

Cancel



인테리어 추천 사이트 화면



홈

블로그

가구추천

가구매칭

글쓰기



smho11님의 블로그

조회수 7

작성시간 2018-10-31 14:52:32.0

좋아요 1

남, 여 중 누가 더 좋아할까?

남자

나이별 분석



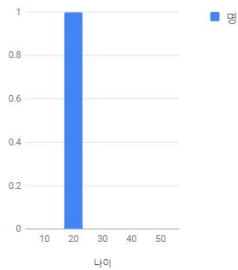
캠성

캠성

스타일 복유업

댓글을 입력하세요

나이별 분석



33

앗.. 저도 캠성스타일 !!!

2018-11-01 16:43:25.0



FIVE TEA SPOON, 5TS

그예민, 김선동, 김승환, 김태훈

김가 (집에 가라) 이메일: interior_5ts@naver.com

Copyright © 2018 by Five Tea Spoon. All Rights Reserved.



인테리어 추천 사이트 화면

홈

블로그

가구추천

가구매칭

카페 랭킹 BEST 5



bens

조회수 : 0

평점 : 4.95

좋아요 : 0



LOFT 3단서랍장 (TS107A)

조회수 : 1

평점 : 5.00

좋아요 : 0

[침대]



키라 침탁

조회수 : 1

평점 : 5.00

좋아요 : 0

[침탁]



바이런 수퍼킹 침대

조회수 : 1

평점 : 5.00

좋아요 : 0

[수퍼킹 침대]



cherish

조회수 : 0

평점 : 4.89

좋아요 : 2



라포레 3인 소파 패브릭 라이트그레이

조회수 : 0

평점 : 5.00

좋아요 : 0

[소파]



포켓 뉴스마트 모션베드 SS 침대 세트

조회수 : 0

평점 : 5.00

좋아요 : 0

[침탁]



브루노 4인 소파 베리오누빅 블루

조회수 : 0

평점 : 5.00

좋아요 : 0

[소파]





인테리어 추천 사이트 화면

ARIA Change Your Lifestyle FURNITURE

aria

조회수 : 0

평점 : 4.90

좋아요 : 0



KW-Q Bed Queen Panel Bed

조회수 : 0

평점 : 5.00

좋아요 : 0

[관심대]



B3604-54H-HB Queen Panel Bed

조회수 : 0

평점 : 5.00

좋아요 : 0

[관심대]



Gerda-1150-Beige Super Single Bed w/ Wood Slats

조회수 : 1

평점 : 5.00

좋아요 : 0

[슈퍼/싱글침대]



카페이름 :bens

바이런 수퍼킹 침대

조회수 : 2

평점 : 5.00

상품 보러가기

좋아요 3



파일 선택 선택된 파일 없음

평점

1

댓글을 입력하세요

COMMENT



인테리어 추천 사이트 화면

[홈](#)[블로그](#)[가구추천](#)[가구매칭](#)[홈](#)[블로그](#)[가구추천](#)[가구매칭](#)

COLOR Matching !

사진을 업로드하면 인테리어 컬러 매칭이 시작됩니다.



Matching START !



FIVE TEA SPOON , 5TS

고예린, 김선중, 김승환, 김태훈

집,가 (집에 가) 이메일 : interior_5ts@naver.com

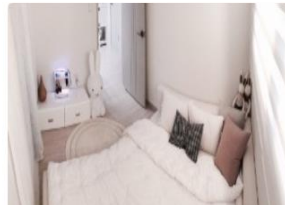
가구 매칭



다른 색으로 보기

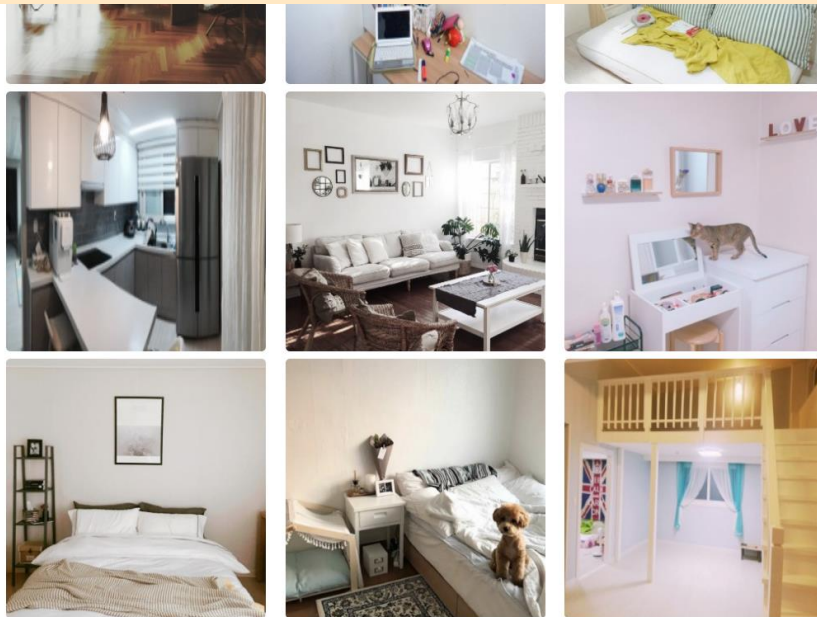


매칭 리스트





인테리어 추천 사이트 화면



[더보기](#)



가구 매칭



가구 매칭 개발 단계 목차

INDEX

1. 개발 동기
2. 개발 설계
3. 데이터 분석 가설
4. 데이터 가공 및 처리 내용
5. 데이터 수집
6. 데이터 정제
7. 머신러닝 활용
8. PyCham 활용
9. WEB 개발



가구 매칭

5TS

쪽지

My blog

Login / join

블로그

카페 분석

가구 매칭

최저가 검색

판매점 검색

Step 1

Upload image file

Upload Image File

File format

Your Image should be in a png and jpeg

Make Subscription



가구 매칭 개발 설계

개발자: 김선중

가구 사진을 업로드 시,
사진에서 색을 추출하여 그 사진의 색들을 기준으로
인테리어를 보여주는 방식

첫 개발 때의 UI





가구 매칭 데이터 분석 가설

개발자: 김선중

인테리어 이미지에서 추출한 R, G, B(*이하 feature로 정의) 값들 중 가장 많은 비율을 차지하는 feature 값과 업로드 된 이미지의 feature 값이 같은 범위에 있으면 어울리는 이미지일 것이다.



5it [00:00, 192.82it/s]

```
0.05522135416666667 [63.96510257 60.79320915 61.46922896]
0.28557291666666667 [211.36664976 211.81476188 211.83444747]
0.20225260416666666 [146.33341917 145.40842078 144.47453808]
0.3387890625 [189.20792079 189.64325971 189.86515613]
0.1181640625 [248.47096419 249.01289256 248.66909091]
```

best_percent,best_color : 0.3387890625,[189.20792079 189.64325971 189.86515613]

```
def centroid_histogram(cit):
    # 한 사진에서 색들의 비율이 얼마나 차지하는지
    numLabels = np.arange(0, len(np.unique(cit.labels_)) + 1)
    (hist, _) = np.histogram(cit.labels_, bins=numLabels)

    # 히스토그램 정규화시키기
    hist = hist.astype("float")
    hist /= hist.sum()

    return hist

def plot_colors(hist, centroids):
    # 이미지에서 각 색이 차지하는 비율에 따라 바 그래프 그리기
    bar = np.zeros((50, 300, 3), dtype="uint8")
    startX = 0

    # 각 색에 비율과 색을 분류하여 그래프를 그리기
    for (percent, color) in zip(hist, centroids):
        endX = startX + (percent * 300)
        cv2.rectangle(bar, (int(startX), 0), (int(endX), 50),
                       color.astype("uint8").tolist(), -1)

        startX = endX

    return bar
```




가구 매칭 데이터 가공 및 처리 내용

개발자: 김선중

이미지 데이터에서 feature 값을 가장 많은 비율부터 5가지로 분류 추출한다.
5가지 중 가장 많은 비율의 R, G, B를 feature 값으로 사용한다.



5it [00:00, 208.88it/s]

```
0.127578125 [40.52903374 23.93229286 41.11483437]
0.084072265625 [125.24651972 128.76009281 121.17192575]
0.314208984375 [221.28474186 101.12644143 68.94349299]
0.211591796875 [185.04006092 56.78432639 42.40379379]
0.262548828125 [227.83754513 185.86006178 165.20614835]
```

```
best_percent,best_color : 0.314208984375,[221.28474186 101.12644143 68.94349299]
```

```
def image_color_cluster(image_path, k = 5):
    # 위 두 함수를 사용하여 이미지에서 클러스터 후 그래프를 그려 시각화
    image = cv2.imread(image_path)
    image = cv2.cvtColor(image, cv2.COLOR_BGR2RGB)
    image = image.reshape((image.shape[0] * image.shape[1], 3))

    clt = KMeans(n_clusters = k)
    clt.fit(image)

    hist = centroid_histogram(clt)
    bar = plot_colors(hist, clt.cluster_centers_)

    plt.figure()
    plt.axis("off")
    plt.imshow(bar)
    plt.show()
```

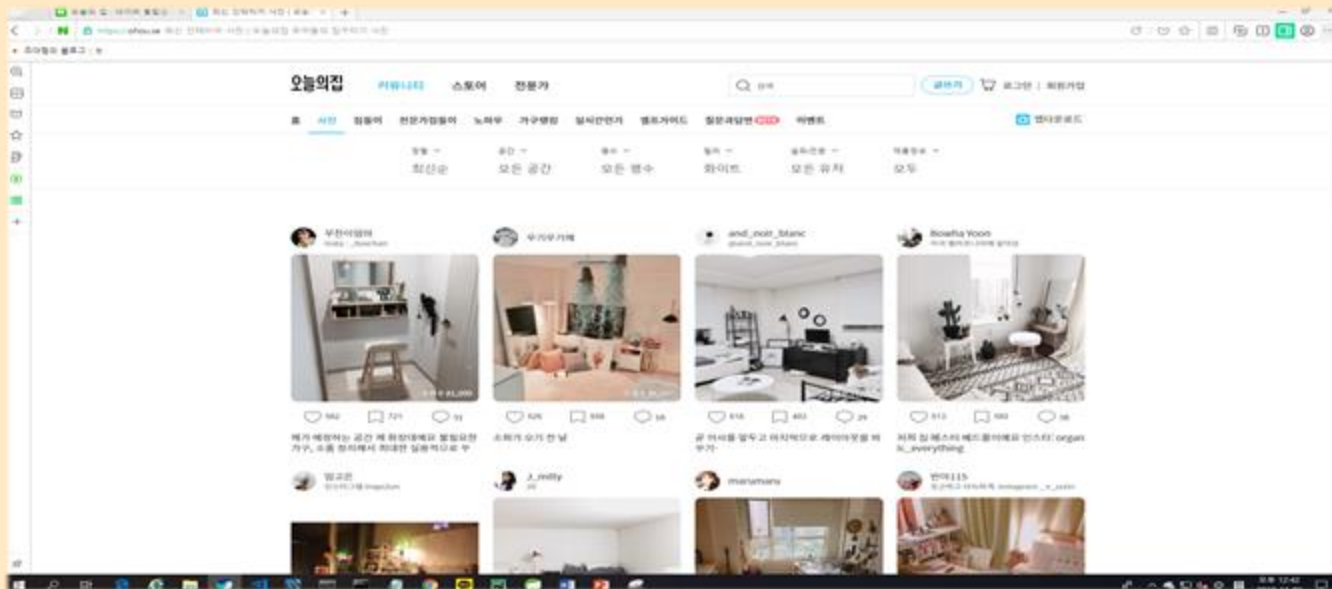


가구 매칭 데이터 수집

개발자: 김선종

오늘의 집에서 색상 별로 데이터를 수집한다.

크롤링한 사이트





가구 매칭 데이터 수집 크롤러

개발자: 김선종

```
# final crawler

driver = webdriver.Chrome('./driver/chromedriver')
color_name = ['black', 'blue', 'green', 'grey', 'mint', 'pink', 'red', 'white']

img_url_dict = {}
# db connection
conn = pymysql.connect(host='localhost',
                        user='root',
                        password='1111',
                        db='db_5ts')

for color in tqdm(color_name):
    try :
        print("-----() color 크롤링 중-----".format(color))
        driver.get("https://ohou.se/contents/card_collections?color={}".format(color))
        html = driver.page_source
        soup = BeautifulSoup(html, 'html.parser')

        # Get scroll height
        last_height = driver.execute_script("return document.body.scrollHeight")

        for i in tqdm(range(1,40)):
            # Scroll down to bottom
            driver.execute_script("window.scrollTo(0, document.body.scrollHeight);")
            # Wait to load page
            time.sleep(3)
            # Calculate new scroll height and compare with last scroll height
            new_height = driver.execute_script("return document.body.scrollHeight")

            if new_height == last_height:
                print("Finish at {} times Scrolls".format(i))
                break
            last_height = new_height
            print("Scroll Down until {} times".format(i))
        print("-----() color 외 {} 번째 다른 스크롤에서 이미지 크롤링 시작-----".format(color,i))
```

크롤링 시 메모리의 과부하를 막기 위해 DB
에 데이터를 저장하면서 크롤링을 한다.

크롤링할 사이트가 무한 스크롤로 구현되어
있기 때문에 스크롤을 자동으로
내릴 수 있도록 구현함



가구 매칭 데이터 수집 크롤러

개발자: 김선중

```
imageshref = driver.find_elements_by_class_name('card_img')
img_url_list = []
for images in todom(imageshref):

    imgs = images.find_elements_by_class_name('horizon') # 이미지 태그
    img_url = images.find_element_by_tag_name('a').get_attribute('href')

    for img in (imgs):
        try:
            with conn.cursor() as cursor:
                nowdatetime = str(datetime.datetime.now())[2:-3].replace(" ", "").replace(":", "").replace(".", "").replace("-", "")
                img_src = img.get_attribute("src") # 이미지 경로
                img_name = nowdatetime[4:] + "{}".format(color) + ".jpg" # 이미지 이름 중복 제거
                img_url_list.append(img_url)
                img_url_dict[ "{}".format(color) ] = img_url_list
                dirpath = "img_test/{}".format(color)

                if not os.path.isdir(dirpath):
                    os.mkdir(dirpath)
                urllib.request.urlretrieve(img_src, dirpath + img_name )

                sql = 'INSERT INTO tb_interior (interior_img, interior_url) VALUES (xs, xs)'
                cursor.execute(sql, (img_name, img_url))
                conn.commit()
            except Exception as e:
                print("예외 발생-I in insert forGrammar", e)
        except Exception as e:
            print("예외 발생-I in crawling", e)
        finally:

            print("{} 크롤링 끝".format(color))

conn.close()
print("conn close--")
driver.close()
print("driver close-I")
```

크롤링 시 메모리의 과부하를 막기 위해
이미지를 저장함과 동시에 이미지의 이름과
이미지의 url을 DB에 저장함



가구 매칭 데이터 정제

개발자: 김선중

```
rgb.loc[:,['R','G','B','image_name','image_urls']].head(10)
```

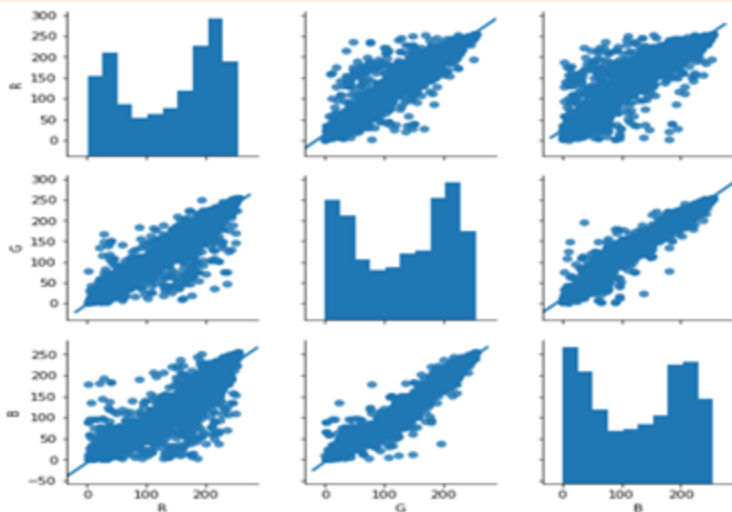
	R	G	B	image_name	image_urls
0	27.327479	22.161508	14.674769	20165256920black.jpg	https://ohou.se/cards/21837/detail?effect_id=0...
1	254.810895	254.350100	253.486960	20165257086black.jpg	https://ohou.se/cards/26836/detail?effect_id=0...
2	24.852416	24.114132	17.940292	20165257207black.jpg	https://ohou.se/cards/21713/detail?effect_id=0...
3	38.335223	33.787675	29.737551	20165257339black.jpg	https://ohou.se/cards/89675/detail?effect_id=0...
4	15.478685	12.228517	8.937726	20165257462black.jpg	https://ohou.se/cards/76759/detail?effect_id=0...
5	36.135206	24.508418	15.959199	20165257586black.jpg	https://ohou.se/cards/41746/detail?effect_id=0...
6	109.589726	101.434342	95.687392	20165257705black.jpg	https://ohou.se/cards/55718/detail?effect_id=0...
7	83.368186	65.423841	48.934083	20165257828black.jpg	https://ohou.se/cards/103202/detail?effect_id=0...
8	61.864885	47.777240	42.377105	20165257939black.jpg	https://ohou.se/cards/76914/detail?effect_id=0...
9	27.533012	22.421161	14.847435	20165258080black.jpg	https://ohou.se/cards/33316/detail?effect_id=0...

```
feature = rgb[['R','G','B']]
feature.head()
```

	R	G	B
0	27.327479	22.161508	14.674769
1	254.810895	254.350100	253.486960
2	24.852416	24.114132	17.940292
3	38.335223	33.787675	29.737551
4	15.478685	12.228517	8.937726

이미지에서 뽑아낸 R,G,B 값들을
feature로 사용한다.

```
import seaborn as sns
# 서로간에 상관관계가 뚜렷함.
sns.pairplot(rgb16clustfinal, vars=['R','G','B'], kind='reg')
```



상관 관계 그래프가 우상향 방향의 그래프를 그리므로
feature의 서로간에 상관관계가 뚜렷함을 볼 수 있음.



가구 매칭 데이터 정제

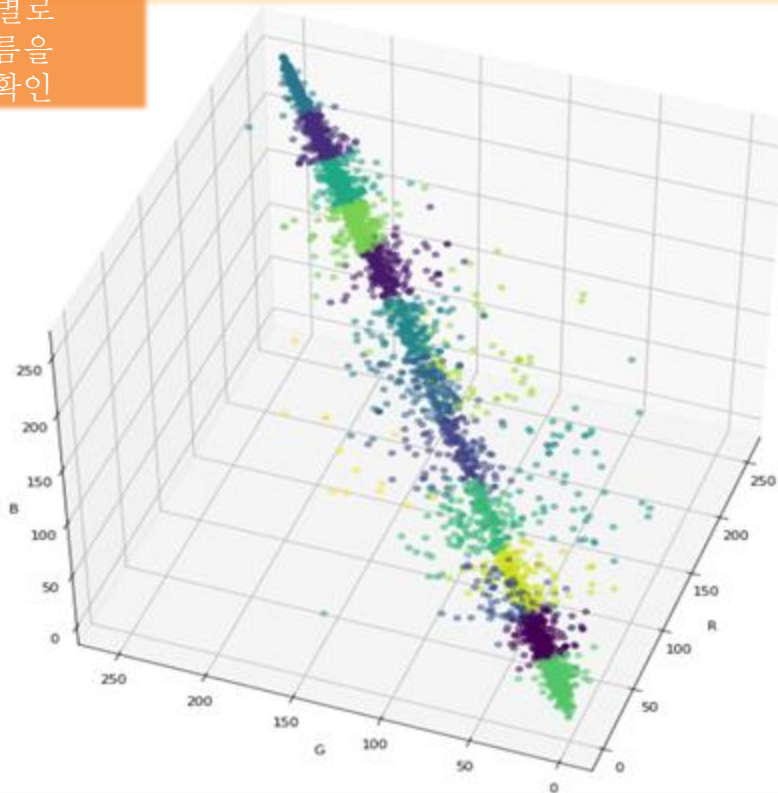
개발자 : 김선종

```
from sklearn.cluster import KMeans
# KMeans로 R,G,B 값을 군집화 시킨다.
kmodel = KMeans(n_clusters=16, algorithm='auto', random_state=1)
kmodel.fit(feature)
predict216 = pd.DataFrame(kmodel.predict(feature))
predict216.columns=['predict216']

# feature 값에 따라 예측 분류된 값을 합친다.
r2 = pd.concat([feature, predict216], axis=1)

fig = plt.figure(figsize=(10,10))
ax = Axes3D(fig, rect=[0, 0, .95, 1], elev=48, azim=200)
ax.scatter(r2['R'], r2['G'], r2['B'], c=r2['predict216'], alpha=0.5)
ax.set_xlabel('R')
ax.set_ylabel('G')
ax.set_zlabel('B')
plt.show()
```

R,G,B 값들 별로
분포도가 다를 것을
시각화하여 확인

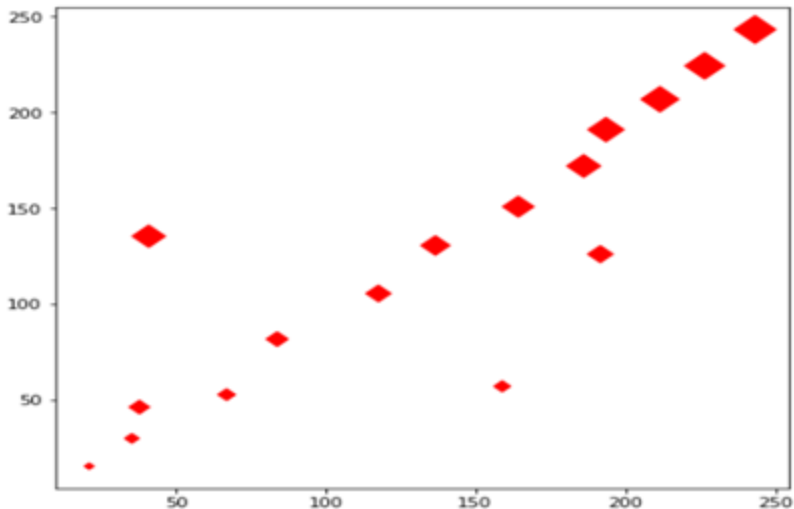




가구 매칭 데이터 정제

개발자: 김선종

```
centers = pd.DataFrame(kmodel.cluster_centers_, columns=['R', 'G', 'B'])
center_x = centers['R']
center_y = centers['G']
center_z = centers['B']
plt.figure(figsize=(7,7))
plt.scatter(center_x, center_y, center_z, marker='D', c='r')
plt.show()
```



KMeans로 label들의 clustering의 center 값을 확인해서 군집이 겹치는 부분이 없는 것을 시각화하여 확인



가구 매칭 머신러닝 활용

개발자: 김선중

```
# 그리드 서치
from sklearn.model_selection import cross_val_score
from sklearn.svm import SVC
from sklearn.model_selection import train_test_split

X_trainval, X_test, y_trainval_label, y_test_label = train_test_split(rgb2_data, rgb2_label, random_state=0)

X_train, X_valid, y_train_label, y_valid_label = train_test_split(X_trainval, y_trainval_label, random_state=1)
best_score = 0

for gamma in [0.001, 0.01, 0.1, 1, 10, 100]:
    for C in [0.001, 0.01, 0.1, 1, 10, 100]:
        svm1abel2 = SVC(gamma=gamma, C=C)

        scores = cross_val_score(svm1abel2, X_trainval, y_trainval_label, cv=5)

        score = np.mean(scores)

        if score > best_score:
            best_score = score
            best_parameters = {"C": C, "gamma": gamma}

svm1abel2 = SVC(**best_parameters).fit(X_trainval, y_trainval_label)
```

```
from sklearn.externals import joblib
joblib.dump(svmModel, 'svmModel.pkl')

['svmModel.pkl']
```

PyCham에서 모델을 활용하기 위하여 모델을 저장한다.

업로드 될 이미지의 feature 값이 새로운 데이터이기 때문에 지도학습된 SVM 모델을 통하여 예측할 수 있도록 한다.

그리드 서치를 활용하여
적절한 매개변수를 선정하고 모델로 활용
k-means 군집으로 분류된 예측 값을 label로 사용하고 다시 SVM 머신러닝을 사용하여 지도학습을 시킨다.



가구 매칭 PyCham 활용

개발자: 김선종

```
@app.route('/bestColor', methods=['GET', 'POST'])
def get_bestColor():

    image = request.args.get('image_name')
    image = urllib.parse.unquote(image, 'utf-8')
    print(image)

    time.sleep(1)

    path = './data/furniture/'
    print("image: {}".format(path + image))

    image = cv2.imread(path + image)
    image = cv2.cvtColor(image, cv2.COLOR_BGR2RGB)
    image = image.reshape((image.shape[0] * image.shape[1], 3)) # height, width 통합

    clt = KMeans(n_clusters=5)
    clt.fit(image)
    numLabels = np.arange(0, len(np.unique(clt.labels_)) + 1)
    (hist, _) = np.histogram(clt.labels_, bins=numLabels)

    hist = hist.astype("float")
    hist /= hist.sum()
    hist
```

```
best_percent = 0

for (percent, color) in zip(hist, clt.cluster_centers_):
    print(percent, color)
    if percent > best_percent:
        best_percent = percent
        if best_percent == percent:
            best_color = color

predict_number = svmLabel2.predict(np.atleast_2d(best_color))

for number in predict_number:
    number

print("number:", number)

return str(number)
```

업로드 되는 이미지 데이터를 모델로 분석하고
label을 받을 수 있도록 PyCham을 활용하였음.



가구 매칭 WEB 개발

개발자: 김선종

```
//파일 업로드
@RequestMapping(value = "/Fileupload", method = RequestMethod.POST)
public String upload(MultipartHttpServletRequest request, Model model) {
    System.out.println("request : " + request);
    try {
        Logger.info("Fileupload 실행");

        MultipartFile mf = request.getFile("file");
        String path = "D:\\bigdata\\spring-workspace\\SP_SJK\\Interior3\\src\\main\\webapp\\interior\\data\\interiorIMG\\";

        File saveDirectory = new File(path);
        if (!saveDirectory.exists()) {
            saveDirectory.mkdir();
        }

        String originalFileName = new String(mf.getOriginalFilename().getBytes("8859_1"), "UTF-8");
        System.out.println("원본 파일명 : " + originalFileName);
        String saveFile = path + originalFileName;
        String FileName = originalFileName;

        mf.transferTo(new File(saveFile));
        RestTemplate restTemplate = new RestTemplate();

        String MatchPoint = restTemplate.postForObject("http://localhost:5000/bestColor" + "?image_name=" + FileName,
            null, String.class);

        System.out.println("파일 업로드 끝, MatchPoint: " + MatchPoint);
        FurnitureDTO furnitureDTO = new FurnitureDTO();
        furnitureDTO.setFurniture_img(FileName);
        furnitureDTO.setColor_score(MatchPoint);
        matchingDAO.furnitureInfo(furnitureDTO);
    }
}
```

```
@RequestMapping(value = "/showing_interior")
public @ResponseBody void showing_interior(HttpServletResponse response, String FileName) {

    String path = "D:\\bigdata\\spring-workspace\\SP_SJK\\Interior3\\src\\main\\webapp\\interior\\data\\interiorIMG\\";

    File file = new File(path + FileName);
    try {
        FileInputStream in = new FileInputStream(file);
        BufferedInputStream bis = new BufferedInputStream(in);
        byte[] buffer = new byte[65536];
        int readCnt = -1;

        OutputStream out = response.getOutputStream();
        BufferedOutputStream bos = new BufferedOutputStream(out);

        while ((readCnt = bis.read(buffer)) != -1) {
            bos.write(buffer, 0, readCnt);
        }

        bos.close();
        bis.close();
    } catch (Exception e) {

    }
}
```

스프링 환경을 활용하여 웹 서버를 구축하였고 파일을 업로드하면 R,G,B 값으로 label을 분류 받을 수 있도록 PyCham과 연결하였음.

이미지 업로드 시 해당 이미지와 같은 label의 인테리어 이미지들을 보여줌



가구 매칭 WEB 개발

개발자: 김선종

```
@Repository
public class MatchingDAOImpl implements MatchingDAO {
    private static final Logger logger = LoggerFactory.getLogger(MatchingDAOImpl.class);
    @Inject
    private SqlSession sqlSession;
    private static final String namespace = "kr.co.ots.mapper.MatchMapper.";
    @Override
    public List<ImgListVO> imgList() {
        logger.info("ImgList impl start");
        return sqlSession.selectList(namespace+"imgList");
    }

    @Override
    public void furnitureInfo(FurnitureDTO furnitureDTO) {
        logger.info("furniture insert impl start");
        sqlSession.insert(namespace+"furnitureInfo", furnitureDTO);
    }

    @Override
    public List<ColorListVO> colorList(String matchPoint) {
        logger.info("colorList impl start");
        return sqlSession.selectList(namespace+"colorList", matchPoint);
    }

    @Override
    public List<ColorListVO> moreList(MoreListVO moreListVO) {
        logger.info("moreList impl start");
        System.out.println(moreListVO);
        return sqlSession.selectList(namespace+"moreList", moreListVO);
    }
}
```

MySQL에 데이터를 저장 및 같은 label의
이미지들을 불러옴



가구 매칭 WEB 개발

개발자: 김선중

```
@RequestMapping(value = "/other_color_score")
public @ResponseBody String other_color_score(String furnitureName) {

    System.out.println();
    RestTemplate restTemplate = new RestTemplate();

    String res = restTemplate.postForObject("http://localhost:5000/colorList" + "?image_name=" + furnitureName,
        null, String.class);

    JsonParser parser = new JsonParser();
    JsonElement jsonElement = parser.parse(res);

    JsonObject jsonObject = jsonElement.getAsJsonObject();

    System.out.println(jsonObject.toString());

    return jsonObject.toString();
}
```

```
@RequestMapping(value = "/otherColor")
public String otherColor(String otherNumber, Model model) {
    List<ColorListVO> otherColorList = matchingDAO.colorList(otherNumber);

    for (ColorListVO other : otherColorList) {
        System.out.print("  inum : " + other.getInum() + ", score : " + other.getColor_score());
    }

    model.addAttribute("OtherColorList", otherColorList);

    return "matching_otherColor";
}
```

이미지를 업로드하면 업로드 된 이미지의 R,G,B 값과
label을 분석된 모델로부터 가져오고,
다른 색 클릭 시 해당하는 label의
이미지들을 보여줌

```
@RequestMapping(value = "/showing_furniture")
public @ResponseBody void showing_furniture(HttpServletResponse response, String fileName) {

    String path = "D:\\bigdata\\spring-workspace\\SP_3K\\Interior\\src\\main\\webapp\\interior\\data\\furniture\\";

    File file = new File(path + fileName);
    try {
        FileInputStream in = new FileInputStream(file);
        BufferedInputStream bis = new BufferedInputStream(in);
        byte[] buffer = new byte[65536];
        int readCnt = -1;

        OutputStream out = response.getOutputStream();
        BufferedOutputStream bos = new BufferedOutputStream(out);

        while ((readCnt = bis.read(buffer)) != -1) {
            bos.write(buffer, 0, readCnt);
        }

        bos.close();
        bis.close();
    } catch (Exception e) {
    }
}
```

자신이 업로드한 가구와 같이 비교해 줄 수 있도록 가구
이미지도 같이 보여줌



가구 매칭 WEB 개발

개발자: 김선중

```
50 * create table tb_interior(  
51     inum integer not null auto_increment,  
52     R float,  
53     G float,  
54     B float,  
55     color_name text,  
56     interior_img text,  
57     interior_url text,  
58     color_score integer,  
59     primary key(inum)  
60 *)ENGINE=InnoDB DEFAULT CHARSET=utf8;  
61 * select * from tb_interior;  
62  
63 * LOAD DATA LOCAL INFILE 'D:/bigdata/spring-workspace/SP_53K/tb_interior_data.csv' INTO TABLE web_board.tb_interior FIELDS TERMINATED BY ',';  
64
```

Python으로 정제한 Interior 데이터를 csv로 저장한 뒤 MySQL에 테이블에 입력

inum	R	G	B	color_name	interior_img	interior_url	color_score
17	47.81294328	50.5156636	46.43657775	black	20181219038black.jpg	https://hshou.se/boards/19196/detail?effect_id=0&effect_type=CardIndex	4
18	69.5014627	59.90386805	45.64881852	black	20181219133black.jpg	https://hshou.se/boards/131559/detail?effect_id=0&effect_type=CardIndex	14
19	51.54964548	40.56039603	31.4054054	black	20181219261black.jpg	https://hshou.se/boards/198399/detail?effect_id=0&effect_type=CardIndex	6
20	42.4094268	41.59824136	31.71446752	black	20181219381black.jpg	https://hshou.se/boards/198033/detail?effect_id=0&effect_type=CardIndex	6
21	143.6661918	105.8382756	67.94624027	black	20181219523black.jpg	https://hshou.se/boards/80023/detail?effect_id=0&effect_type=CardIndex	3
22	27.53421453	18.60090822	11.53465648	black	20181219984black.jpg	https://hshou.se/boards/130996/detail?effect_id=0&effect_type=CardIndex	11
23	38.23571802	34.70068715	27.30398222	black	20181219988black.jpg	https://hshou.se/boards/123863/detail?effect_id=0&effect_type=CardIndex	2
24	3.385781262	5.385115296	4.76722304	black	20181230019black.jpg	https://hshou.se/boards/20072/detail?effect_id=0&effect_type=CardIndex	11
25	70.90302965	45.90457686	31.02960996	black	20181230057black.jpg	https://hshou.se/boards/112238/detail?effect_id=0&effect_type=CardIndex	14
26	134.5294272	115.6242109	91.28924402	black	20181230048black.jpg	https://hshou.se/boards/113955/detail?effect_id=0&effect_type=CardIndex	3
27	160.4615346	123.2187623	86.35338196	black	20181230058black.jpg	https://hshou.se/boards/20743/detail?effect_id=0&effect_type=CardIndex	13

tb_furniture 테이블을 활용하여 앞으로 업로드 되는 이미지에 대한 정보를 저장

```
65 * create table tb_furniture(  
66     fnum integer not null auto_increment,  
67     furniture_img text,  
68     write_time timestamp default current_timestamp,  
69     color_score integer,  
70     primary key(fnum)  
71 *)ENGINE=InnoDB DEFAULT CHARSET=utf8;  
72 * select * from tb_furniture;  
73
```

fnum	furniture_img	write_time	color_score
5	37238368132906910 928473207.jpg	2018-10-26 15:46:06	6
6	37238368132906910 928473207.jpg	2018-10-26 15:48:30	6
7	37238368132906910 928473207.jpg	2018-10-26 15:50:25	6
8	74848667040086372 -1341002183.jpg	2018-10-26 16:03:26	3
9	74848667040086372 -1341002183.jpg	2018-10-26 16:04:24	3
10	15657577137.4.20181012105552.jpg	2018-10-26 16:04:25	6
11	34036894148726753 1525219766.PNG	2018-10-26 16:06:26	9
12	74848667040086372 -1341002183.jpg	2018-10-26 18:53:42	3
13	37238368132906910 928473207.jpg	2018-10-29 16:37:23	3
14	74848667040086372 -1341002183.jpg	2018-10-29 16:42:43	3



가구 매칭 WEB 개발

개발자: 김선중



홈 블로그 가구추천 가구매칭

 COLOR Matching !

사진을 업로드하면 인터레이 컬러 매칭이 시작됩니다.



Matching START !

가구 이미지를 업로드

```
<script>
```

```
$(document).ready(function(){  
    var userInfo = $("#user").val();  
    //alert(userInfo);  
    console.log(userInfo+"#");  
    if (userInfo == ""){  
        alert("로그인이 필요합니다.^^");  
        $("#fileButton").attr('disabled',"disabled");  
        $("#fileSubmit").attr('disabled',"disabled");  
        $("#fileSubmit").attr('class',"button2 button1");  
    }  
});
```

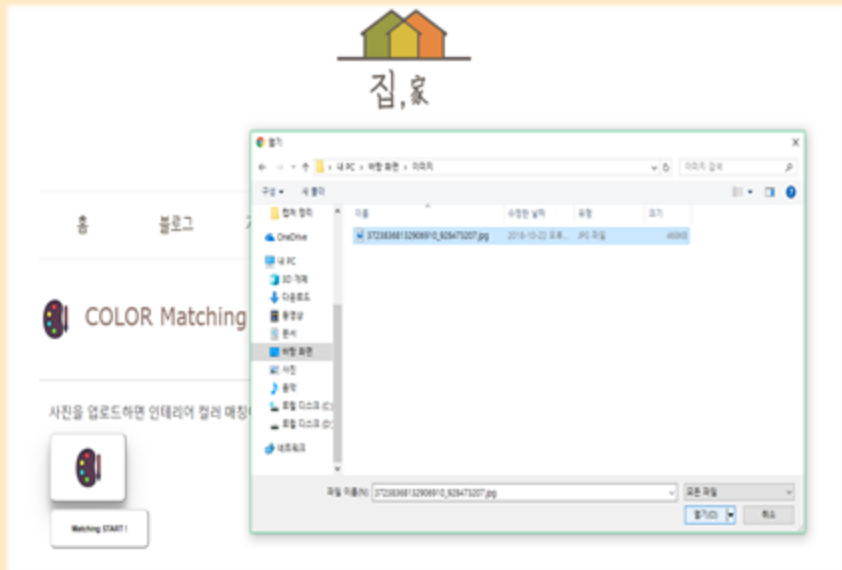
```
</script>
```

로그인이 안되어 있으면 버튼의 클래스를 동적으로 만들어서 클릭을 하지 못하도록 하였음



가구 매칭 WEB 개발

개발자: 김선중



이미지 사진을 업로드 하게 되면 매칭이 시작됨

```
<div>
  <hr width=100% color="lightgray" align="left" size=3 />
</div>
<div class="container">
  <div>
    <input type="hidden" name="user" id="user" value="${sessionScope.loginUserId}" />
    <form id="fileForm" name="fileForm" action="fileupload" method="post"
    enctype="multipart/form-data">
      <input type="file" name="file" id="file" style="display: none" />

      <b4>사진을 업로드하면 인터레어 컬러 매칭이 시작됩니다.</b4>

      <div style="float: center;">
        <button class="button2 button1" id="fileButton"
        onclick="document.getElementById('file').click()" type="button" style="float: center;">
          
        </button>

        <button class="button2" id="fileSubmit" type="submit">
          <strong>Matching START !</strong></button>
        </div>
      </form>
    </div>
  </div>
```

업로드 버튼을 숨기고 이미지 소스를 활용하여 버튼을 만들고 업로드 기능을 상속받도록 함



가구 매칭 WEB 개발

개발자 : 김선중

가구 매칭



다른 색으로 보기



매칭 리스트



업로드 된 이미지에서 색을 분류
후 모델을 통해 나온 label로
같은 label의 인테리어 이미지를
보여줌

```
<section>
  <div class="container">

    <header>
      <br> <br>
      <h4 style="display: inline-block">
        <strong> 가구 매칭 </strong> 
      </h4>
    </header>
    <br> 

    <input type="hidden" name="hide1" id="hide1" value="${furnitureName}">

    <div id="otherColorAjax" style="float: left;"></div>
  </div>
</section>
```

업로드 된 이미지를 인테리어 이미지와
비교해 볼 수 있도록 함



가구 매칭

개발자: 김선종

```
<!-- 매칭 완료된 이미지 -->
<section>
<div class="container">
  <hr width=100% color="lightgray" align="left" size=3 />
  <h4 style="display: inline-block">
    <strong> 매칭 리스트 </strong></h4>

  <div id="js-load">
    <c:forEach items="${MatchedColorList}" var="MatchedColorList">
      <div class="lists_item js-load"
        style="width: 33%; float: left; text-align: center;">
        <a href="${MatchedColorList.interior_url}"> </a>

      </div>
    </c:forEach>
  </div>
</div>
```

같은 label 값을 갖는 인테리어 이미지들을 불러옴

매칭 리스트



더보기



가구 매칭 WEB 개발

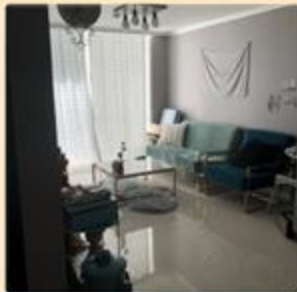
개발자: 김선중

가구 매칭



다른 색으로 보기 버튼을 클릭하면 해당 이미지의 다른 색에 해당하는 이미지들 보여줌
더보기 버튼을 클릭하면 다른 이미지들이 계속해서 나옴

매칭 리스트



더보기



가구 매칭 WEB 개발

개발자: 김선중

```
<script type="text/javascript">
    setTimeout(
        function() {
            $(document)
                .ready(
                    function() {
                        function unpack(rgb) {
                            rgb = rgb.replace(/[\^0-9,\.\,]/g, " ");
                            //rgb = rgb.replace(" ", ",");
                            rgb = "(" + rgb + ")"
                            return rgb;
                        }
                        var furnitureName = $('#hidel').val();
                        var labellist = [];
                        var jsonData = $.
                            .ajax({
                                url : "other_color_score",
                                data : {
                                    'furnitureName' : furnitureName
                                },
                                dataType : 'json',
                                async : false,
                                success : function(result) {
                                    /* alert(result.cols[1].label)
                                    alert(unpack(result.cols[0].pattern)) */
                                    var html = "<br><span>"
                                        + "<strong>다른 색으로 보기</strong><br>"
                                        + "</span>"
                                        + "<a href='/ots/matching/otherColor?otherNumber="
                                        + result.cols[0].label + "&furnitureName=" + furnitureName
                                        + "><img height='30' width='30' src='/ots/resources/image/round.png' style='background: rgb"
                                        + unpack(result.cols[0].pattern)
                                        + ";></a>"
                                        + "<a href='/ots/matching/otherColor?otherNumber="
                                        + result.cols[1].label + "&furnitureName=" + furnitureName
                                        + "><img height='30' width='30' src='/ots/resources/image/round.png' style='background: rgb"
                                        + unpack(result.cols[1].pattern)
                                        + ";></a>"
                                        + "<a href='/ots/matching/otherColor?otherNumber="
                                        + result.cols[2].label + "&furnitureName=" + furnitureName
                                        + "><img height='30' width='30' src='/ots/resources/image/round.png' style='background: rgb"
                                        + unpack(result.cols[2].pattern)
                                        + ";></a>"
                                        + "<a href='/ots/matching/otherColor?otherNumber="
                                        + result.cols[3].label + "&furnitureName=" + furnitureName
                                        + "><img height='30' width='30' src='/ots/resources/image/round.png' style='background: rgb"
                                        + unpack(result.cols[3].pattern)
                                        + ";></a>"
                                        + "<a href='/ots/matching/otherColor?otherNumber="
                                        + result.cols[4].label + "&furnitureName=" + furnitureName
                                        + "><img height='30' width='30' src='/ots/resources/image/round.png' style='background: rgb"
                                        + unpack(result.cols[4].pattern)
                                        + ";></a>"
                                    $("#otherColorAjax")
                                        .append(html);
                                }
                            })
                    })
                .responseText;
            })
        }, 2500);
    })
</script>
```

다른 색으로 보기 버튼을
ajax로 구현해서 비동기통신으로
이미지들을 불러오도록 하였음



가구 매칭 WEB 개발

개발자: 김선중

```
<script>
$(window).on('load', function() {
    load('#js-load', '9');
    $('#js-btn-wrap .button').on("click", function() {
        load('#js-load', '9', '#js-btn-wrap');
    })
});
function load(id, cnt, btn) {
    var load_list = id + " .js-load:not(.active)";
    var load_length = $(load_list).length;
    var load_total_cnt;
    if (cnt < load_length) {
        load_total_cnt = cnt;
    } else {
        load_total_cnt = load_length;
        $('#.button').hide()
    }
    $(load_list + ":lt(" + load_total_cnt + ")").addClass("active");
}
</script>
```

```
<hr width=100% color="lightgray" align="Left" size=3 />
<br>

<div id="js-btn-wrap" class="btn-wrap">
    <a href="javascript:;" class="button"><h3>더보기</h3></a>
</div>

<hr width=100% color="lightgray" align="Left" size=3 />
</section>
```

```
<style>
.js-load {
    display: none;
}
.js-load.active {
    display: block;
}
.is_comp.js-load:after {
    display: none;
}
.btn-wrap, .lists, .main {

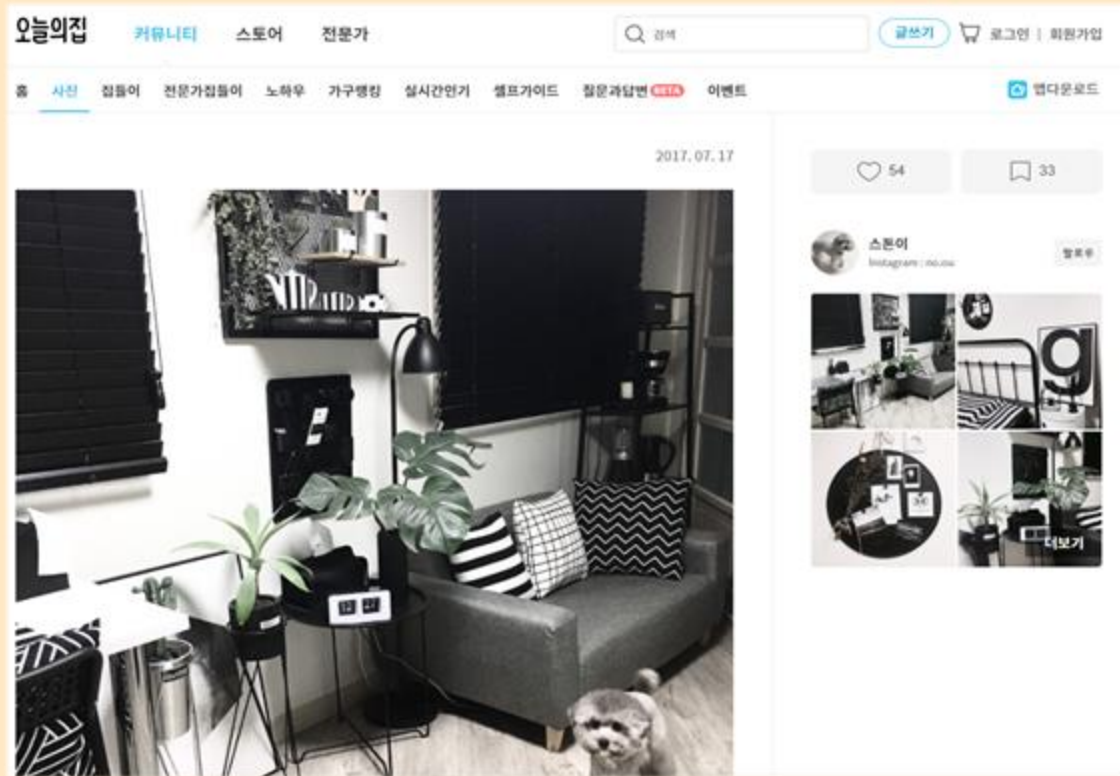
    display: block;
}
.main {
    max-width: 640px;
    margin: 0 auto;
}
.btn-wrap {
    text-align: center;
}
</style>
```

일정량의 이미지만 보여주다가 사용
자가 더 보기를 원할 경우
인테리어 사진을 더 보여줌



가구 매칭 WEB 개발

개발자: 김선중



이미지를 클릭하게 되면 해당 사이트로 이동하여 상세 정보들을 확인 할 수 있음

감사합니다



집, 家