Rookie Hero Project

미니 요기요 - 3차 스프린트

태형 - 3차 스프린트 구현 내용

- **미니 요기요**
  - 주문 내역 페이지
  - 주문 내역 디테일 페이지

- **랜덤 메뉴 Pick**
  - **메뉴 점수 계산 함수 작성**(조회수, 좋아요, 주문수, 사장님 추천메뉴의 합계)
  - **내 취향 기반 랜덤 고르기** 기능
  - **내가 먹었던 메뉴들 중 랜덤 고르기** 기능(점수 반영(X))

태형 - 랜덤 메뉴 Pick

- **내 취향 중 랜덤**
  - 단순히 랜덤으로 메뉴를 골라주는 것이 아닌,
    1. **사용자 취향**(#매운맛, #매콤한맛 등등)
    2. **메뉴 점수**(조회수, 좋아요, 주문수, 사장님 추천메뉴)

    를 고려한 **랜덤 선택 기능**

- **내가 먹었던 메뉴 중 랜덤**
  - 내가 먹었던 메뉴들 중, **메뉴 점수가 높은 메뉴를 필터링 하고**, 랜덤으로 선택해준다.

태형 - 언제 사용하면 좋을까?

- **내 취향 중 랜덤**
  1. **내 취향에 맞는 음식**을 **찾는 노력 없이** 선택해서 주문 하고 싶을 때,
  2. 굳이 리뷰를 보지 않더라도, **내 취향에 맞는 인기메뉴**를 주문하고 싶을 때,

- **내가 먹었던 메뉴 중 랜덤**
  1. 신 메뉴 보다는 **기존에 먹었던 메뉴들 중**에서, **인기메뉴**를 고르고 싶을 때,

태형 - 4차 스프린트 계획

- **미니 요기요**
  - 주문 내역 디테일 페이지 내 재주문 기능
  - 메뉴 좋아요, 조회수 관련 API 작성
  - 기존 코드에 추가된 쿠폰 관련 코드 반영

- **랜덤 메뉴 Pick**
  - 취향 **AND, OR 조건** 선택 가능
  - **내 취향** 을 **랜덤메뉴Pick 화면에서 간편 수정할 수 있도록**
  - 취향 **해시태그**에 대한 **메뉴 순위 리스트**
  - 내가 먹었던 메뉴 랜덤 선택에서 **메뉴 점수 반영**

- **타임라인**

명훈 - 3차 스프린트 구현 내용

1. 날씨와 관계 없이 잘 팔린 메뉴 제외, 잘 팔린 메뉴 Top10을 위한 더보기 버튼
2. 기간별 주문 수량에 대한 가중치를 부여해서 메뉴 정렬
3. 레스토랑 구독하기
4. 타임라인 페이지에서 지난 1시간 동안 많이 팔린 메뉴 보여주기

명훈 - 1. 날씨와 관계 없이 잘 팔린 메뉴는 날씨별 메뉴 추천에서 제외, 제외된 메뉴 하단 표시, 더보기 버튼을 누르면 잘 팔린 메뉴 Top10을 볼 수 있음

1. 현재 동네의 날씨와 다른 날씨에 많이 팔린 메뉴를 센다.
2. 메뉴가 3번 중복되면 날씨와 무관한 메뉴라고 판단해서 날씨 카테고리에서 제외한다.
3. 제외된 메뉴들은 메뉴 리스트 하단에 텍스트로 보여준다.

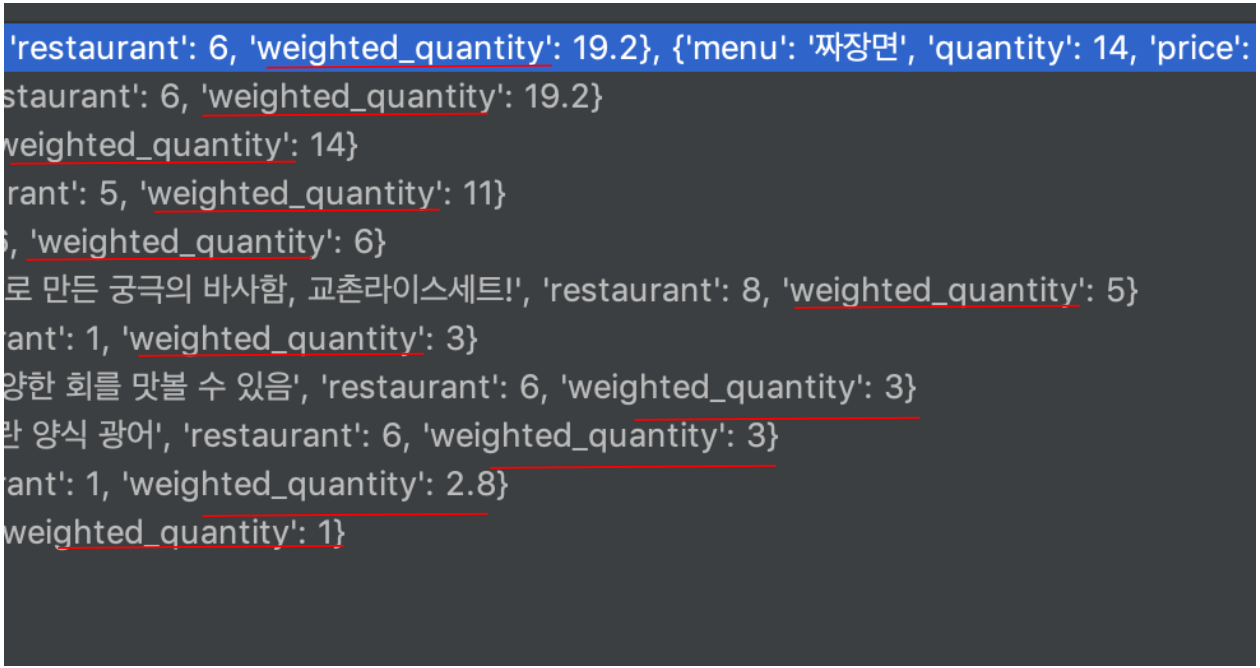
명훈 - 1. 날씨와 관계 없이 잘 팔린 메뉴는 날씨별 메뉴 추천에서 제외

```
weather_list.remove(user_dong_weather)
other_weathers = weather_list
for other_weather in other_weathers:
    menu_list = (Cart.objects
                 .prefetch_related('order')
                 .filter(order__address__contains=user_dong, order__weather=other_weather.value)
                 .values('cartitem__menu__name')
                 .annotate(menu_quantity=Sum('cartitem__quantity'), menu=F('cartitem__menu__name'))
                 .order_by('-menu_quantity')[:5]
                 .values('menu', quantity=F('menu_quantity'),
                         price=F('cartitem__menu__price'), img=F('cartitem__menu__img'),
                         id=F('cartitem__menu__id'), detail=F('cartitem__menu__detail'),
                         restaurant=F('cartitem__menu__restaurant__id'),
                         )
                 )
    for menu in menu_list:
        if count_duplicated_weather_order.get(menu['menu']):
            count_duplicated_weather_order[menu['menu']] += 1
            menu_quantity_to_exclude_or_not[menu['menu']] += menu['quantity']
        else:
            count_duplicated_weather_order[menu['menu']] = 1
            menu_quantity_to_exclude_or_not[menu['menu']] = menu['quantity']

try:
    for menu_name, menu_cnt in count_duplicated_weather_order.items():
        if menu_cnt >= 3:
            excluded_menu_count[menu_name] = count_duplicated_weather_order[menu_name]
            excluded_menu_quantity[menu_name] = menu_quantity_to_exclude_or_not[menu_name]
```

명훈 - 2. 이번달, 지난달, 지지난달 주문 수량에 대한 가중치를 부여해서 메뉴 정렬

1. 이번달, 지난달, 지지난달의 주문수에 대해 가중치를 부여해서 정렬 후 출력한다.



2. 이번달, 지난달,

지지난달 주문 수량에
대한 가중치를
부여해서 메뉴 정렬

```
menu_list = (Cart.objects
             .prefetch_related('order')
             .filter(order__address__contains=user_dong, order__weather=user_dong_weather)
             .values('cartitem__menu__name')
             .annotate(menu=F('cartitem__menu__name'), quantity=Sum('cartitem__quantity'))
             .exclude(menu__in=excluded_menu_count)
             .order_by('-quant
             .values('menu', '
                     price=F('
                     id=F('car
                     restauran
                         'cart
                     )
             )[:10]
```

```
weighted_quantity = {}
for menu in menu_list:
    for month in range(this_month, this_month - Month.three_month, -1):
        queryset = (Order.objects
                    .filter(cart__cartitem__menu__name=menu['menu'], created_time__month=month,
                            weather=user_dong_weather)
                    .select_related('cart', 'cart__cartitem__menu')
                    .prefetch_related('cart__cartitem')
                    .annotate(name=F('cart__cartitem__menu__name'),
                              quantity=F('cart__cartitem__quantity'))
                    .values('name', 'quantity', 'created_time__month')
                    )
        for q in queryset:
            if month == this_month:
                if q['name'] not in weighted_quantity:
                    weighted_quantity[q['name']] = q['quantity']
                else:
                    weighted_quantity[q['name']] += q['quantity']
            elif month == this_month - Month.one_month:
                if q['name'] not in weighted_quantity:
                    weighted_quantity[q['name']] = round(
                        q['quantity'] * Weight.one_month_ago_weight, 2)
                else:
                    weighted_quantity[q['name']] += round(
                        q['quantity'] * Weight.one_month_ago_weight, 2)
            elif month == this_month - Month.two_month:
                if q['name'] not in weighted_quantity:
                    weighted_quantity[q['name']] = round(
                        q['quantity'] * Weight.two_month_ago_weight, 2)
                else:
                    weighted_quantity[q['name']] += round(
                        q['quantity'] * Weight.two_month_ago_weight, 2)
    if menu['menu'] in weighted_quantity.keys():
        menu['weighted_quantity'] = weighted_quantity[menu['menu']]
menu_list = sorted(list(menu_list), key=operator.itemgetter('weighted_quantity'), reverse=True)
```

명훈 - 3. 레스토랑 구독, 취소, mypage에서 구독 조회, 취소 시 남은 레스토랑만 출력

1. 레스토랑 디테일 페이지에서 구독, 취소
2. mypage에서 구독 중인 레스토랑 리스트 확인, 취소
3. 취소버튼 누르면 나머지 레스토랑만 출력

3. 레스토랑 구독, 취소, mypage에서 구독 조회, 취소 시 남은 레스토랑만 출력

```
restaurant = Restaurant.objects.get(pk=restaurant_id)
if user.subscribed_restaurants.filter(pk=restaurant_id):
    user.subscribed_restaurants.remove(restaurant)
    json_data = {
        "message": "구독 취소",
        "subscribe_flag": True,
    }
    return JsonResponse(
        json_data,
    )
else:
    user.subscribed_restaurants.add(restaurant)
    user.save()
    json_data = {
        "message": "구독 성공",
        "subscribe_flag": False,
    }
```

명훈 - 4. 지난 1시간 동안 많이 팔린 메뉴

1. 타임라인 페이지에서 현재 동네에서 가장 많이 주문된 메뉴 보여주기
2. 메뉴 클릭 시 이미지 및 주문표 추가 버튼이 있는 모달 출력

4. 지난 1시간 동안 많이 팔린 메뉴

```
today = datetime.now().date()
one_hour = 1
current_hour = datetime.now().hour
created_hour = current_hour - one_hour
try:
    order = (
        CartItem.objects
            .select_related('cart', 'menu')
            .values(
            address=F('cart__order__address'),
            delivery_completed=F('cart__order__delivery_status'),
            created_time=F('cart__order__created_time'),
        )
            .filter(
            address__contains=user_address,
            delivery_completed=DeliveryStatus.COMPLETE,
            created_time__date=today,
            created_time__hour=created_hour,
        )
            .values(
            'menu__name',
        )
            .annotate(
            menu=F('menu__name'),
            menu_quantity=Sum('quantity')
        )
            .values(
            'menu',
            'menu_quantity',
            'menu_id',
            'menu__img',
        )
            .order_by('-menu_quantity')[:1]
    )
```

명훈 - 개선할 내용

- Django Background Tasks 사용해서 1시간마다 주문수 많은 것 수집
  - 어제 하루동안 잘 팔린 메뉴<Yes(terday) Menu>에 대해 보여주기
- 잘 팔린 메뉴 모달 창에서 메뉴 정보 더 넣기

명훈

- 다음 스프린트 계획
  - 6월 1, 2주차 :
    - 레스토랑 알리미 관련 기능 구현

소라

- 3차 스프린트 구현 내용 - 요기요 gift coupon 기능
  - 요기요 gift coupon
    - 보낸 요기요 gift coupon 내역(보낸 쿠폰)
      - 선물한 쿠폰(구매한 쿠폰)& 양도한 쿠폰 보기
    - 요기요 gift coupon 양도하기
      - 받은 gift coupon이 유효기간이 남아 있는 경우 양도 가능
      - 양도할 사람의 **아이디** 또는 **이메일**(미니 요기요사이트에 등록한)을 입력하여 양도 가능
      - 양도시 양도 받은 사람에게 표시
    - 양도받은 사람에게 알림
      - 쿠폰을 양도받은 사용자는 양도받은 쿠폰에 대하여 상단에 표기된 숫자를 볼 수 있음
      - 쿠폰을 양도받은 사용자가 **새로 양도받은 쿠폰 리스트** 확인하면 상단에 표기된 숫자 지워짐
    - 쿠폰 적용하여 주문하기

소라

- 개선할 내용
  - 한 쿠폰에 대해 여러번 양도가 진행된 경우, 보낸 쿠폰 보기에서 정렬

- 4차 스프린트 계획
  - 구독한 restaurant 알리미
    - 구독할 restaurant 추천

    - 현재 구독하고 있는 restaurant과 유사한 restaurant 추천(구독 기준)
      - **Item-based-collaborative filtering**을 사용한 restaurant추천
      - 구독한 레스토랑간의 유사도를 측정하며, 사용자가 어떤 restaurant 선호하면 유사한 다른 restaurant을 추천하는 방식을 사용할 예정

레스토랑 알리미

- 유튜브 채널 ‘구독' 시, 채널 주인으로 부터, **채널과 관련된 공지사항, 소식 등등을 타임라인으로 볼 수 있는 것**을 벤치마킹

- **기능**
  - **구독 레스토랑 알림**
    - 사장님 공지사항
    - 신 메뉴 추가 or 기존 메뉴 가격 변경 등 소식
    - 이벤트 소식
    - 레스토랑 정보 변경 소식
  - **구독한 레스토랑과 유사한 레스토랑 추천**
  - **현재 주문이 많은 메뉴를 알림**