

---

# 레스토랑 운영을 위한 실시간 재고관리 솔루션

F&B ERP program **Club-ee**



김민섭, 이상윤, 신창훈, 한지웅

# 목차

---

## 1. 솔루션 개요

- 1.1 기획안
- 1.2 개발 환경 및 업무분담

## 2. 프로젝트 진행 순서

- 2.1 DB 생성
  - 2.1.1 DB 구성
  - 2.1.2 DB 연결
  - 2.1.3 DB 생성
- 2.2 웹 구성
  - 2.2.1 웹 구성도
  - 2.2.2 Use Case
  - 2.2.3 Logic Process

## 3. 페이지 레이아웃

- 3.0 Home 화면 구조
- 3.1 메인페이지(대시보드)
- 3.2 발주 현황 페이지
- 3.3 발주 신청 페이지
- 3.4 매출관리 페이지
- 3.5 마이 페이지
- 3.6 키오스크

## 4. 시연

- 4.1 DB 생성
- 4.2 기본 Data 입력
- 4.3 manager mypage
- 4.4 cusOrd 더미 입력
- 4.5 메뉴 판매량
- 4.6 발주 신청
- 4.7 머신러닝
- 4.8 주문

## 5. 보완할 점

---

# 1. 솔루션 개요

1.1 기획안

1.2 개발 환경 및 업무분담



# 1.솔루션 개요

## 1.1 기획안

구분	내용
기획내용	<ul style="list-style-type: none"><li>Django Rest Framework를 활용해 재고관리 시스템 만들기</li></ul>
기획목적	<ul style="list-style-type: none"><li>레스토랑에서 사용되는 재료들을 예측 솔루션 API를 통한 재고관리 시스템으로 관리</li></ul>
기대효과	<ul style="list-style-type: none"><li>사용자가 가게 재고 상황을 한 눈에 파악 가능 및 편리한 발주신청 등 사용자 편의 향상</li><li>예측 API를 통해 다음날 나갈 메뉴들을 예측하여 예측 발주량을 계산, 남은 재료를 줄여 폐기율을 줄임</li></ul>
기능요약	<ul style="list-style-type: none"><li>고객 메뉴 주문 시, 재료 재고 실시간 현황 업데이트</li><li>메뉴를 주문했던 데이터를 통해 메뉴 예측, 예측한 데이터로 재료 소비를 예측, 추천 발주량을 산정</li><li>웹 내, 발주, 재고관리, 매출관리 등 레스토랑 운영에 필요한 웹페이지로 편리화</li></ul>
기타사항	<ul style="list-style-type: none"><li>키오스크 페이지(고객용)와 대시보드(관리자용) 분리</li><li>Django Rest Framework를 사용, 다양한 플랫폼에서 활용 가능</li><li>Bootstrap 사용</li></ul>

# 1.솔루션 개요

---

## 1.2 개발 환경 및 업무 분담

### 구축 환경



### 통합 개발 환경



### 개발 언어



### 커뮤니케이션



### 분석 & 머신러닝



### 저장 및 연동



### Web 개발



## 1.솔루션 개요

---

### 1.2 개발 환경 및 업무 분담

한지웅	DB구축 및 관리, Django Back-end, 머신러닝, 발표
김민섭	Django Back-end, jQuery 동적 테이블 , Rest-API 구축, 머신러닝
이상윤	웹 Front-end, Django Back-end, 산출문 작성
신창훈	웹 Front-end, 시각화, Django Back-end, 산출문 작성

---

## 2. 프로젝트 진행 순서

### 2.1 DB

- 2.1.1 DB 구성

- 2.1.2 DB 연결

- 2.1.3 DB 생성

### 2.2 웹 구성

- 2.2.1 웹 구성도

- 2.2.2 웹

### 2.3 Django Back-End



## 2.1 DB

### 2.1.1 DB 설계

ord(발주 테이블)		
이름	발주 번호	발주 시간
데이터 예시	100000001	2022-09-01 18:00
컬럼명	ordNum	inTime
데이터 타입	int	datetime
널	not null	not null
제약조건	pk	default=sysdate

inStock_(입고 테이블)					
이름	입고 번호	입고 시간	발주 번호	자재 ID	입고수량
데이터 예시	200000001	2022-09-01 18:00	100000001	600000001	1000
데이터 예시	200000002	2022-09-01 18:00	100000001	600000002	500
컬럼명	inNum	inTime	ordNum	matelId	inQuan
데이터 타입	int	datetime	int	int	int
널	not null	not null	not null	not null	not null
제약조건	pk	default=sysdate	fk	fk	default=0



## 2.1 DB

### 2.1.1 DB 설계

menu (menu 테이블)				
이름	메뉴 ID	메뉴 사진	메뉴 이름	메뉴 가격
데이터 예시	500000001	사진.jpg	짬뽕	12000
컬럼명	menuId	menuPic	menuName	menuPri
데이터 타입	int	jpeg	str	int
널	not null	null	not null	not null
제약조건	pk			

recipe (recipe 테이블)			
이름	메뉴 ID	자재 ID	사용량(g)
데이터 예시	500000001	600000001	200
데이터 예시	500000001	600000002	800
컬럼명	menuId	matelId	mateUsage
데이터 타입	int	int	int
널	not null	null	null
제약조건	fk	fk	default=0

## 2.1 DB

### 2.1.1 DB 설계

cusOrd(주문 테이블)			
이름	주문 번호	주문 시간	메뉴 ID
데이터 예시	300000001	2022-09-01 18:00	500000001
컬럼명	cusOrdNum	outTime	menuId
데이터 타입	int	datetime	int
널	not null	not null	not null
제약조건	pk	default=sysdate	fk

outStock (출고 테이블)					
이름	출고 번호	출고 시간	주문 번호	자재 ID	출고수량
데이터 예시	400000001	2022-09-01 18:00	300000001	600000001	500
데이터 예시	400000002	2022-09-01 18:00	300000001	600000002	700
컬럼명	outNum	outTime	cusOrdNum	matelId	outQuan
데이터 타입	int	datetime	int	int	int
널	not null	not null	null	not null	not null
제약조건	pk	default=sysdate	fk	fk	default=0

## 2.1 DB

### 2.1.1 DB 설계

Material_(재료 테이블)							
이름	자재 ID	자재 이름	대분류	중분류	소분류	재료 단가	재고
데이터 예시	600000001	양파	농산물	채소	양파	12000	500
컬럼명	matelId	mateName	lCat	mCat	sCat	unitCost	stock
데이터 타입	int	str	str	str	str	int	int
널	not null	not null	null	null	null	null	not null
제약조건	pk		default='입력값없음'	default='입력값없음'	default='입력값없음'	api로 가져오기	default=0


Manager_(매니저 테이블)								
이름	회원 ID	manager name	password	전화번호	주소	e-mail	안전율	사업자 번호
데이터 예시	900000001	이정현	1234	1012345678	서울시	manager@joongang.com	1.5	123456789
컬럼명	manId	manName	manPw	manPhone	manAddr	manMail	manSafe	bizNum
데이터 타입	int	str	str	str	str	sttr	float	str
널	not null	not null	not null	not null	not null	null	null	null
제약조건	pk							uniq

## 2.1 DB


---

### 2.1.2 DB 연결 – MySQL – Django

- settings.py

```
_cakd_erp >  settings.py > ...  
92     DATABASES = config.DATABASES  
93
```

- config.py

```
 config.py > ...  
1     # config.py  
2     DATABASES = {  
3         'default': {  
4             'ENGINE': 'django.db.backends.mysql',  
5             'NAME': 'cakd_erp',  
6             'USER': 'root',  
7             'PASSWORD': '1234',  
8             'HOST': '127.0.0.1',  
9             'PORT': '3306',  
10        }  
11    }
```

## 2.1 DB

### 2.1.3 DB 생성 – SQL python manage.py inspectdb

```
-- DB Name : cakd7_erp
-- DB Create / cakd7_erp --
CREATE DATABASE cakd7_erp DEFAULT CHARACTER SET UTF8MB4;
USE cakd7_erp;

-- TABLE Create --
-- material --
CREATE TABLE IF NOT EXISTS material(
  mateId int PRIMARY KEY AUTO_INCREMENT,
  mateName varchar(20) NOT NULL,
  lCat varchar(10) DEFAULT 'etc',
  mCat varchar(10) DEFAULT 'etc',
  sCat varchar(10) DEFAULT 'etc',
  unitCost int,
  stock int NOT NULL DEFAULT 0
);
ALTER TABLE material AUTO_INCREMENT=6000000001;

-- manager --
CREATE TABLE IF NOT EXISTS manager(
  manId int PRIMARY KEY AUTO_INCREMENT,
  manName varchar(10) NOT NULL,
  manPw varchar(20) NOT NULL,
  manPhone varchar(15) NOT NULL,
  manAddr varchar(50) NOT NULL,
  manMail varchar(50),
  manSafe FLOAT NOT NULL DEFAULT 1.2,
  bizNum varchar(15),
  CONSTRAINT manager_bizNum_uk UNIQUE(bizNum)
);
ALTER TABLE manager AUTO_INCREMENT=9000000001;

-- menu --
CREATE TABLE IF NOT EXISTS menu(
  menuId int PRIMARY KEY AUTO_INCREMENT,
  menuPic varchar(10),
  menuName varchar(20) NOT NULL,
  menuPri int NOT NULL
);
ALTER TABLE menu AUTO_INCREMENT=5000000001;

-- recipe --
CREATE TABLE IF NOT EXISTS recipe(
  menuId int NOT NULL,
  mateId int NOT NULL,
  mateUsage int DEFAULT 0,
  CONSTRAINT recipe_menuId_menu_menuId_fk FOREIGN KEY (menuId) REFERENCES menu(menuId),
  CONSTRAINT recipe_mateId_material_mateId_fk FOREIGN KEY (mateId) REFERENCES material(mateId)
);
```

```
-- ord --
CREATE TABLE ord(
  ordNum int PRIMARY KEY AUTO_INCREMENT,
  inTime datetime NOT NULL DEFAULT now()
);
ALTER TABLE ord AUTO_INCREMENT=1000000001;

-- inStock --
CREATE TABLE IF NOT EXISTS inStock(
  inNum int PRIMARY KEY AUTO_INCREMENT,
  inTime datetime NOT NULL DEFAULT now(),
  ordNum int NOT NULL,
  mateId int NOT NULL,
  inQuan int NOT NULL DEFAULT 0,
  CONSTRAINT inStock_ordNum_ord_ordNum_fk FOREIGN KEY (ordNum) REFERENCES ord(ordNum),
  CONSTRAINT inStock_mateId_material_mateId_fk FOREIGN KEY (mateId) REFERENCES material(mateId)
);
ALTER TABLE inStock AUTO_INCREMENT=2000000001;

-- cusOrd --
CREATE TABLE IF NOT EXISTS cusOrd(
  cusOrdNum int PRIMARY KEY AUTO_INCREMENT,
  outTime datetime NOT NULL DEFAULT now(),
  menuId int NOT NULL,
  CONSTRAINT cusOrd_menuId_menu_menuId_fk FOREIGN KEY (menuId) REFERENCES menu(menuId)
);
ALTER TABLE cusOrd AUTO_INCREMENT=3000000001;

-- outStock --
CREATE TABLE IF NOT EXISTS outStock(
  outNum int PRIMARY KEY AUTO_INCREMENT,
  outTime datetime NOT NULL DEFAULT now(),
  cusOrdNum int NOT NULL,
  mateId int NOT NULL,
  outQuan int NOT NULL DEFAULT 0,
  CONSTRAINT outStock_cusOrdNum_cusOrdNum_fk FOREIGN KEY (cusOrdNum) REFERENCES cusOrd(cusOrdNum),
  CONSTRAINT outStock_mateId_material_mateId_fk FOREIGN KEY (mateId) REFERENCES material(mateId)
);
ALTER TABLE outStock AUTO_INCREMENT=4000000001;
```

## 2.1 DB

### 2.1.3 DB 생성 – Django ORM

```
from django.db import models

class Menu(models.Model):
    menu_id = models.AutoField(db_column='menuId', primary_key=True)
    menu_pic = models.ImageField(
        db_column='menuPic', upload_to='erp/menu/imgaes/', blank=True, null=True)
    menu_name = models.CharField(db_column='menuName', max_length=20)
    menu_pri = models.IntegerField(db_column='menuPri')
    menu_cnt = models.IntegerField(
        db_column='menuCnt', default=0) # 주문카운트
    menu_sum = models.IntegerField(
        db_column='menuSum', default=0) # 합계

    class Meta:
        db_table = 'menu'

    def __str__(self):
        return f'{self.menu_id}:{self.menu_name}'

class Cusord(models.Model):
    cus_ord_num = models.AutoField(db_column='cusOrdNum', primary_key=True)
    out_time = models.DateField(db_column='outTime', auto_now_add=True)
    menu_id = models.ForeignKey('Menu', models.DO_NOTHING, db_column='menuId')

    class Meta:
        db_table = 'cusOrd'

    def __str__(self):
        return str(self.cus_ord_num)

class Instock(models.Model):
    in_num = models.AutoField(db_column='inNum', primary_key=True)
    in_time = models.DateField(db_column='inTime', auto_now_add=True)
    ord_num = models.ForeignKey('Ord', models.DO_NOTHING, db_column='ordNum')
    mate_id = models.ForeignKey(
        'Material', models.DO_NOTHING, db_column='mateId')
    in_quan = models.IntegerField(db_column='inQuan', default=0)
    in_total = models.IntegerField(db_column='inTotal', default=0)

    class Meta:
        db_table = 'inStock'

    def __str__(self):
        return str(self.in_num)
```

```
class Manager(models.Model):
    man_id = models.AutoField(db_column='manId', primary_key=True)
    man_name = models.CharField(db_column='manName', max_length=10)
    man_pw = models.CharField(db_column='manPw', max_length=20)
    man_phone = models.CharField(db_column='manPhone', max_length=15)
    man_addr = models.CharField(db_column='manAddr', max_length=50)
    man_mail = models.CharField(
        db_column='manMail', max_length=50, blank=True, null=True)
    man_safe = models.FloatField(db_column='manSafe')
    biz_num = models.CharField(
        db_column='bizNum', unique=True, max_length=15, blank=True, null=True)

    class Meta:
        db_table = 'manager'

    def __str__(self):
        return self.man_name

class Material(models.Model):
    mate_id = models.AutoField(db_column='mateId', primary_key=True)
    mate_name = models.CharField(db_column='mateName', max_length=20)
    l_cat = models.CharField(
        db_column='lCat', max_length=10, blank=True, null=True)
    m_cat = models.CharField(
        db_column='mCat', max_length=10, blank=True, null=True)
    s_cat = models.CharField(
        db_column='sCat', max_length=10, blank=True, null=True)
    unit_cost = models.IntegerField(
        db_column='unitCost', blank=True, null=True)
    stock = models.IntegerField(db_column='stock', default=0)

    class Meta:
        db_table = 'material'

    def __str__(self):
        return str(self.mate_name)
```

```
class Ord(models.Model):
    ord_num = models.AutoField(db_column='ordNum', primary_key=True)
    in_time = models.DateField(db_column='inTime', auto_now_add=True)

    class Meta:
        db_table = 'ord'

    def __str__(self):
        return str(self.ord_num)

class Outstock(models.Model):
    out_num = models.AutoField(db_column='outNum', primary_key=True)
    out_time = models.DateField(db_column='outTime', auto_now_add=True)
    cus_ord_num = models.ForeignKey(
        'Cusord', models.DO_NOTHING, db_column='cusOrdNum')
    mate_id = models.ForeignKey(
        'Material', models.DO_NOTHING, db_column='mateId')
    out_quan = models.IntegerField(db_column='outQuan', default=0)

    class Meta:
        db_table = 'outStock'

    def __str__(self):
        return str(self.out_num)

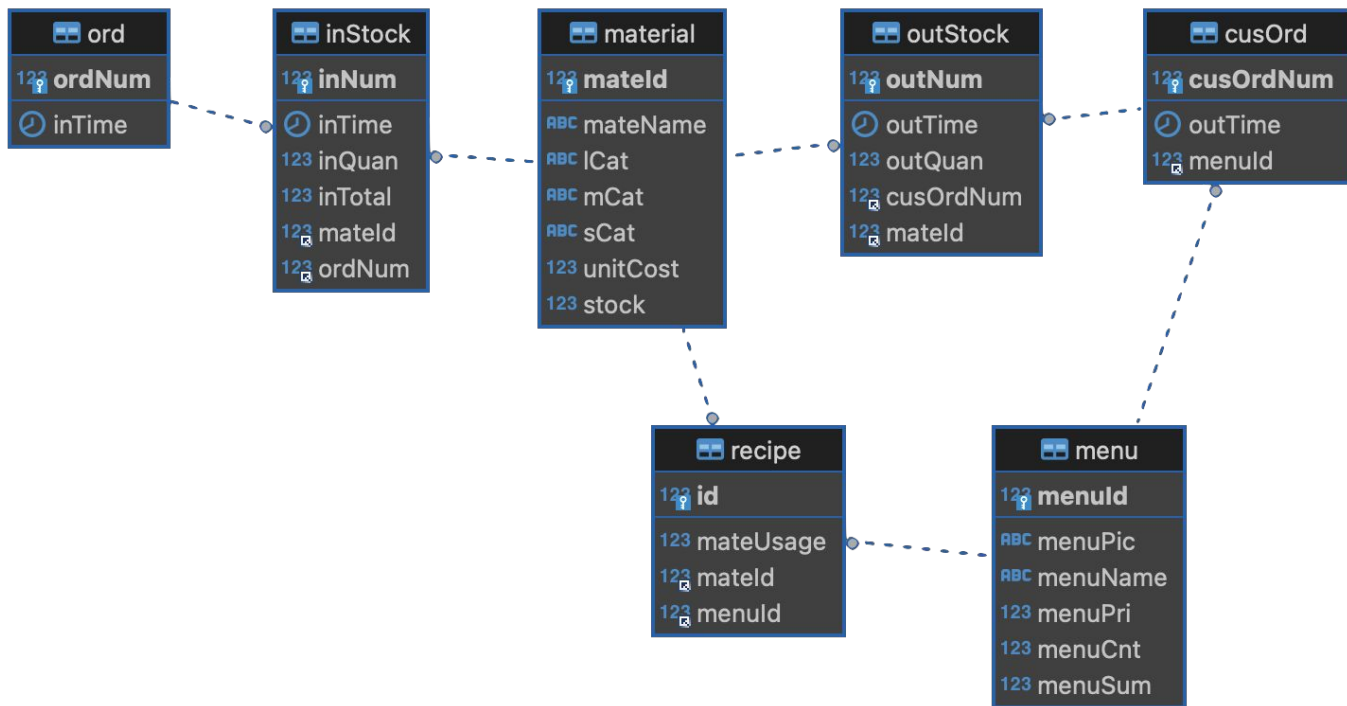
class Recipe(models.Model):
    menu_id = models.ForeignKey(Menu, models.DO_NOTHING, db_column='menuId')
    mate_id = models.ForeignKey(
        'Material', models.DO_NOTHING, db_column='mateId')
    mate_usage = models.IntegerField(
        db_column='mateUsage', blank=True, null=True)

    class Meta:
        db_table = 'recipe'

    def __str__(self):
        return str(self.menu_id)
```

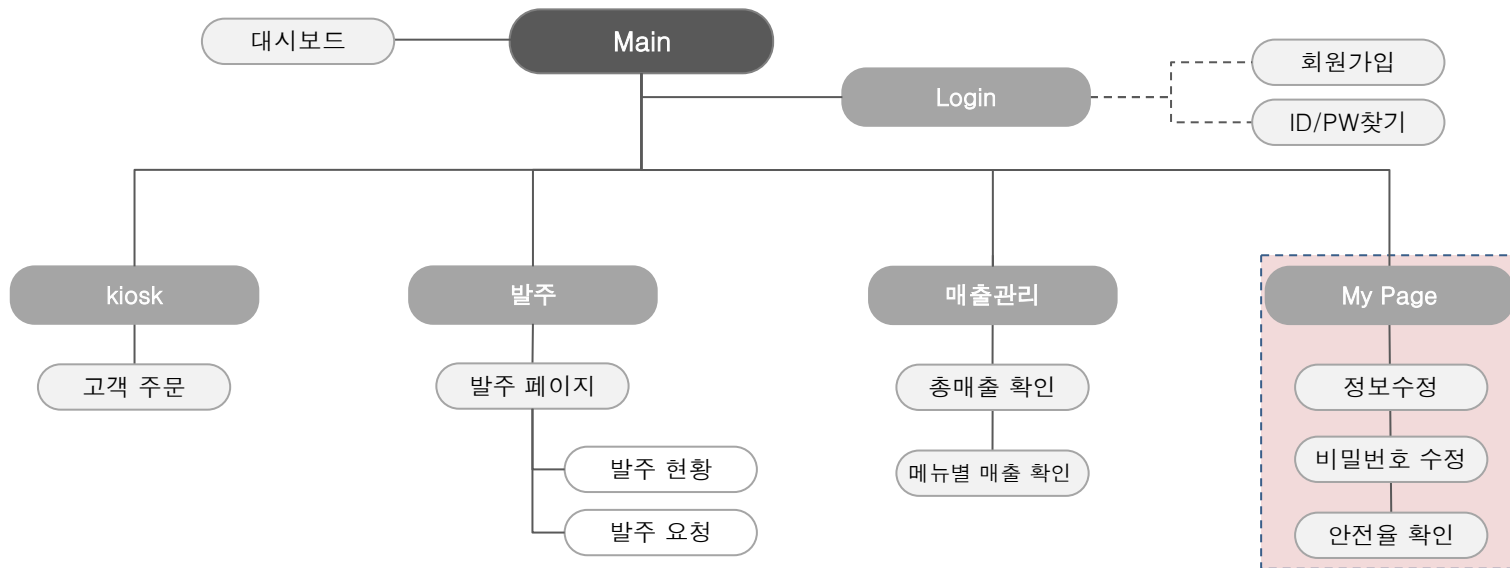
## 2.1 DB

### 2.1.3 DB 생성 – Django ERM



## 2.2 웹 구성

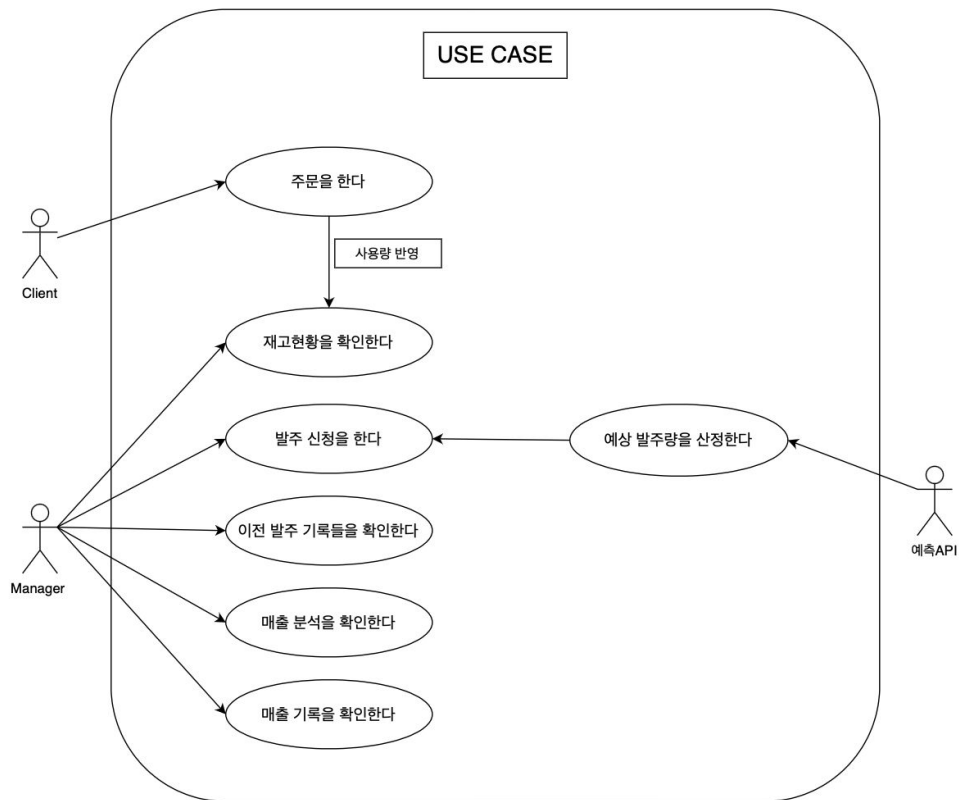
### 2.2.1 웹 구성도





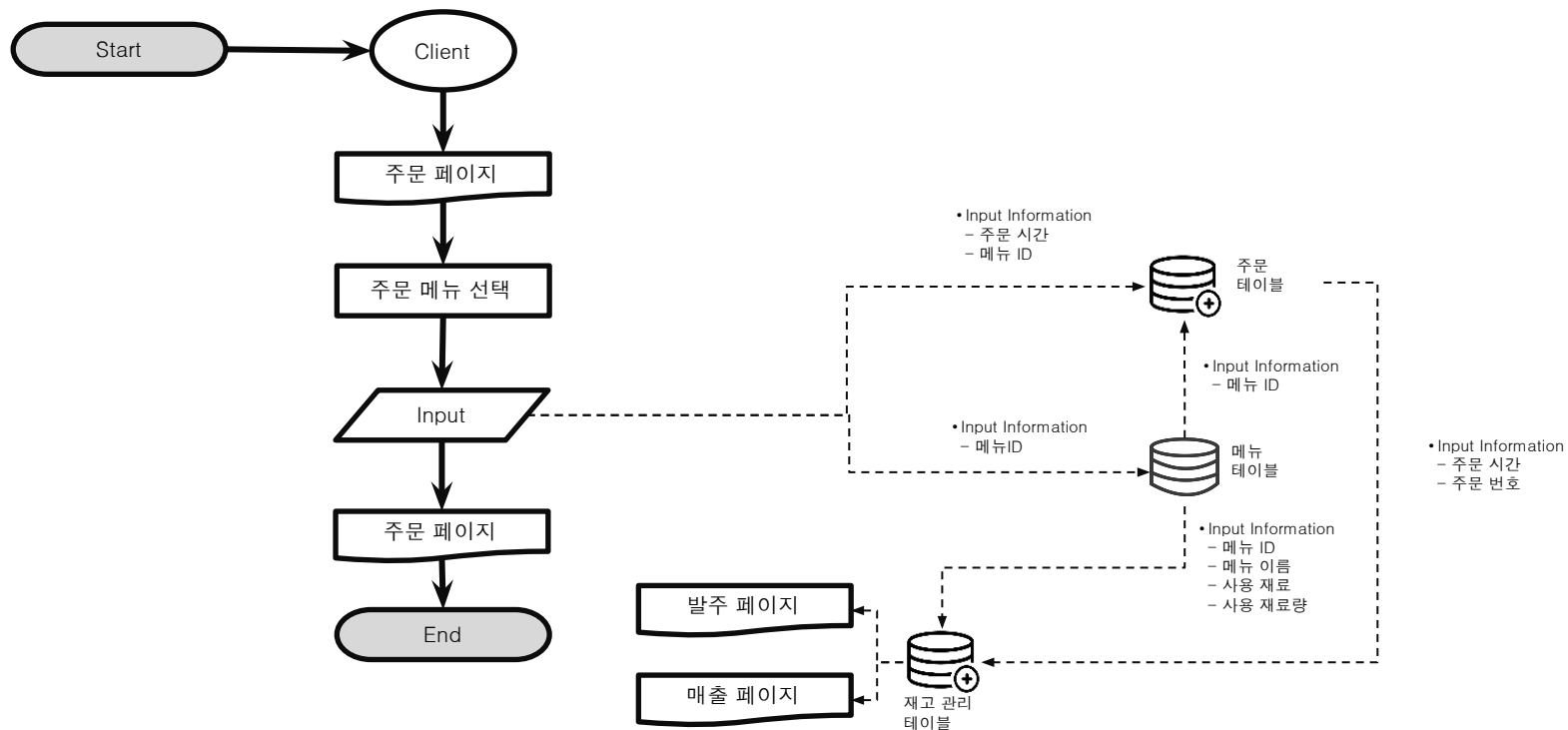
## 2.2 웹 구성

### 2.2.2 Use Case



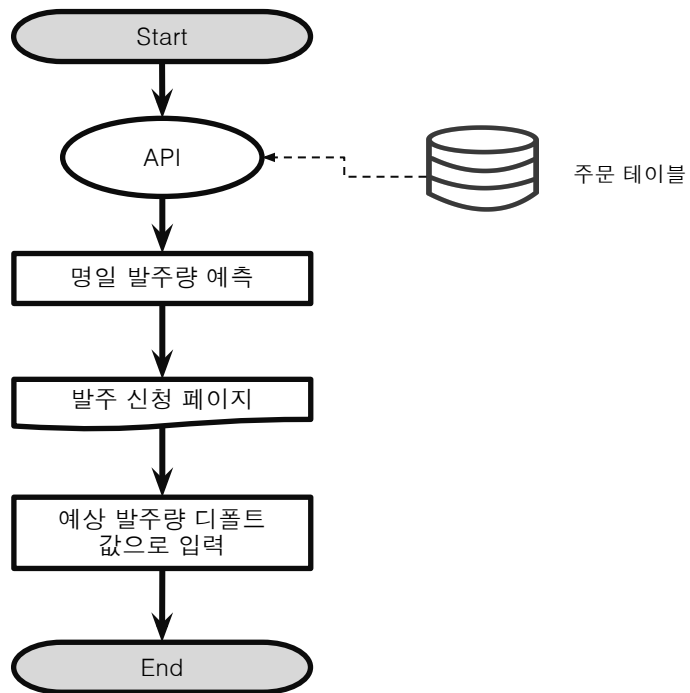
## 2.2 웹 구성

### 2.2.3 Logic process – Customer



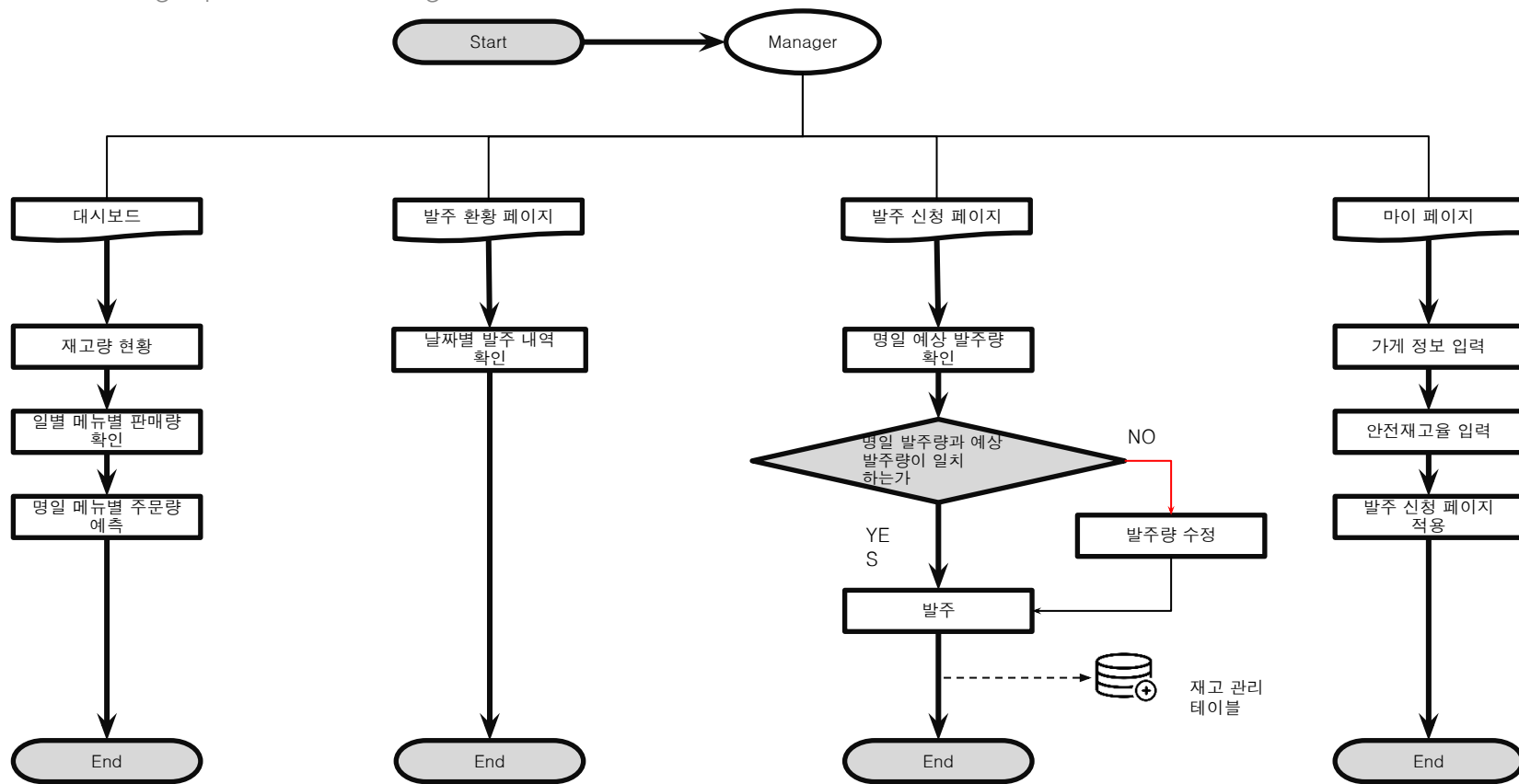
## 2.2 웹 구성

### 2.2.3 Logic process – API



## 2.2 웹 구성

### 2.2.3 Logic process – Manager



## 2.3 Django Back-End

### 2.2.3 erp/view.py

```
erp > views.py > ...
1 from django.shortcuts import render, redirect
2 from rest_framework.response import Response
3 from django.views.generic import ListView, DetailView, CreateView, UpdateView
4 from django.contrib.auth.mixins import LoginRequiredMixin, UserPassesTestMixin
5 from .models import Cusord, Instock, Manager, Material, Menu, Ord, Outstock, Recipe
6 from .serializers import CusordSerializer, InstockSerializer, ManagerSerializer,\
7     MaterialSerializer, MenuSerializer, OrdSerializer, OutstockSerializer, RecipeSerializer
8 from rest_framework import generics
9 from rest_framework import viewsets
10
11 class CusordViewSet(viewsets.ModelViewSet):
12     queryset = Cusord.objects.all()
13     serializer_class = CusordSerializer
14
15 class InstockList(viewsets.ModelViewSet):
16     queryset = Instock.objects.all()
17     serializer_class = InstockSerializer
18
19 class ManagerList(viewsets.ModelViewSet):
20     queryset = Manager.objects.all()
21     serializer_class = ManagerSerializer
22
23 class MaterialList(viewsets.ModelViewSet):
24     queryset = Material.objects.all()
25     serializer_class = MaterialSerializer
26
27 class MenuList(viewsets.ModelViewSet):
28     queryset = Menu.objects.all()
29     serializer_class = MenuSerializer
30
31 class OrdList(viewsets.ModelViewSet):
32     queryset = Ord.objects.all()
33     serializer_class = OrdSerializer
34
35 class OutstockList(viewsets.ModelViewSet):
36     queryset = Outstock.objects.all()
37     serializer_class = OutstockSerializer
38
39 class RecipeList(viewsets.ModelViewSet):
40     queryset = Recipe.objects.all()
41     serializer_class = RecipeSerializer
42
```

## 2.3 Django Back-End

### 2.2.3 erp/serializers.py

```
erp > serializers.py > ...
1 from dataclasses import field
2 from rest_framework import serializers
3 from erp.models import Cusord, Instock, Manager, Material, Menu, Ord, Outstock, Recipe
4 from drf_queryfields import QueryFieldsMixin
5
6 # Menu Serializer
7 class MenuSerializer(QueryFieldsMixin, serializers.ModelSerializer):
8     class Meta:
9         model = Menu
10        fields = '__all__'
11
12 # Material Serializer
13 class MaterialSerializer(QueryFieldsMixin, serializers.ModelSerializer):
14     class Meta:
15         model = Material
16        fields = '__all__'
17
18 # Ord Serializer
19 class OrdSerializer(QueryFieldsMixin, serializers.ModelSerializer):
20     class Meta:
21         model = Ord
22        fields = '__all__'
23
24 # Cusord Serializer
25 class CusordSerializer(serializers.ModelSerializer):
26     menu_id = MenuSerializer(read_only=True)
27
28     class Meta:
29         model = Cusord
30        fields = '__all__'
```

```
erp > serializers.py > ...
22     menu_id = MenuSerializer(read_only=True)
23
24     class Meta:
25         model = Cusord
26        fields = '__all__'
27
28 # InStock Serializer
29 class InstockSerializer(QueryFieldsMixin, serializers.ModelSerializer):
30     mate_id = MaterialSerializer(read_only=True)
31
32     class Meta:
33         model = Instock
34        fields = '__all__'
35
36 # Outstock Serializer
37 class OutstockSerializer(QueryFieldsMixin, serializers.ModelSerializer):
38     mate_id = MaterialSerializer(read_only=True)
39     cus_ord_num = CusordSerializer(read_only=True)
40
41     class Meta:
42         model = Outstock
43        fields = '__all__'
44
45 # Recipe Serializer
46 class RecipeSerializer(QueryFieldsMixin, serializers.ModelSerializer):
47     mate_id = MaterialSerializer(read_only=True)
48     menu_id = MenuSerializer(read_only=True)
49
50     class Meta:
51         model = Recipe
52        fields = '__all__'
53
54 # Manager Serializer
55 class ManagerSerializer(QueryFieldsMixin, serializers.ModelSerializer):
56     class Meta:
57         model = Manager
58        fields = '__all__'
```

## 2.3 Django Back-End

### 2.2.3 erp/serializers.py

Django REST framework

Api Root / Cusord Viewset List

### Cusord Viewset List

OPTIONS GET

GET /rest/api\_cusord/

HTTP 200 OK  
Allow: GET, POST, HEAD, OPTIONS  
Content-Type: application/json  
Vary: Accept

```
[
  {
    "cus_ord_num": 1,
    "menu_id": {
      "menu_id": 2,
      "menu_pic": null,
      "menu_name": "채육볶음",
      "menu_pri": 10000,
      "menu_cnt": 10,
      "menu_sum": 100000
    },
    "out_time": "2022-09-01"
  },
  {
    "cus_ord_num": 2,
    "menu_id": {
      "menu_id": 3,
      "menu_pic": null,
      "menu_name": "비빔밥",
      "menu_pri": 10000,
      "menu_cnt": 8,
      "menu_sum": 80000
    },
    "out_time": "2022-09-01"
  },
  {
    "cus_ord_num": 3,
```

## 2.3 Django Back-End

---

### 2.2.3 kiosk/views.py

```
kiosk > views.py > ...
22 from django.http import JsonResponse
23 from django.contrib import messages
24
25
26 def index(request):
27     return render(request, "kiosk/kiosk.html")
28
29
30 def cusorder(request):
31
32     if request.method == "POST":
33
34         menuid = request.POST.get('menuid')
35         cusord = Cusord(menu_id=Menu.objects.get(menu_id=menuid))
36         cusord.save()
37
38     return render(request, 'kiosk/kiosk.html', messages.info(request, "주문이 완료 되었습니다."))
39
```



## 2.3 Django Back-End

### 2.2.3 erp/signal.py

```
erp > signal.py > ...
1  from django.db.models.signals import post_save
2  from django.dispatch import receiver
3  from erp.models import Cusord, Instock, Outstock, Recipe
4
5
6  @receiver(post_save, sender=Cusord)
7  def Cusord_post_save(sender, **kwargs):
8      menu_id = kwargs['instance'].menu_id
9      menu_id.menu_cnt += 1
10     menu_id.save()
11
12     ord_menu_recipe = Recipe.objects.filter(
13         menu_id=kwargs['instance'].menu_id.menu_id)
14
15     for i in ord_menu_recipe:
16         # menuid = i.menu_id
17         mateid = i.mate_id
18         usage = i.mate_usage
19         Outstock(cus_ord_num=kwargs['instance'], mate_id=mateid, out_quan=usage).save()
20         mateid.stock -= usage
21         mateid.save()
22
23
24  @ receiver(post_save, sender=Instock)
25  def Cusord_post_save(sender, **kwargs):
26     instock_ = kwargs['instance'].mate_id
27     instock_.stock += kwargs['instance'].in_quan
28     instock_.save()
29
30
```

---

## 3. 페이지 레이아웃

- 3.0 Home 화면 구조
- 3.1 메인페이지(대시보드)
- 3.2 발주 현황 페이지
- 3.3 발주 신청 페이지
- 3.4 매출관리 페이지
- 3.5 마이 페이지
- 3.6 키오스크



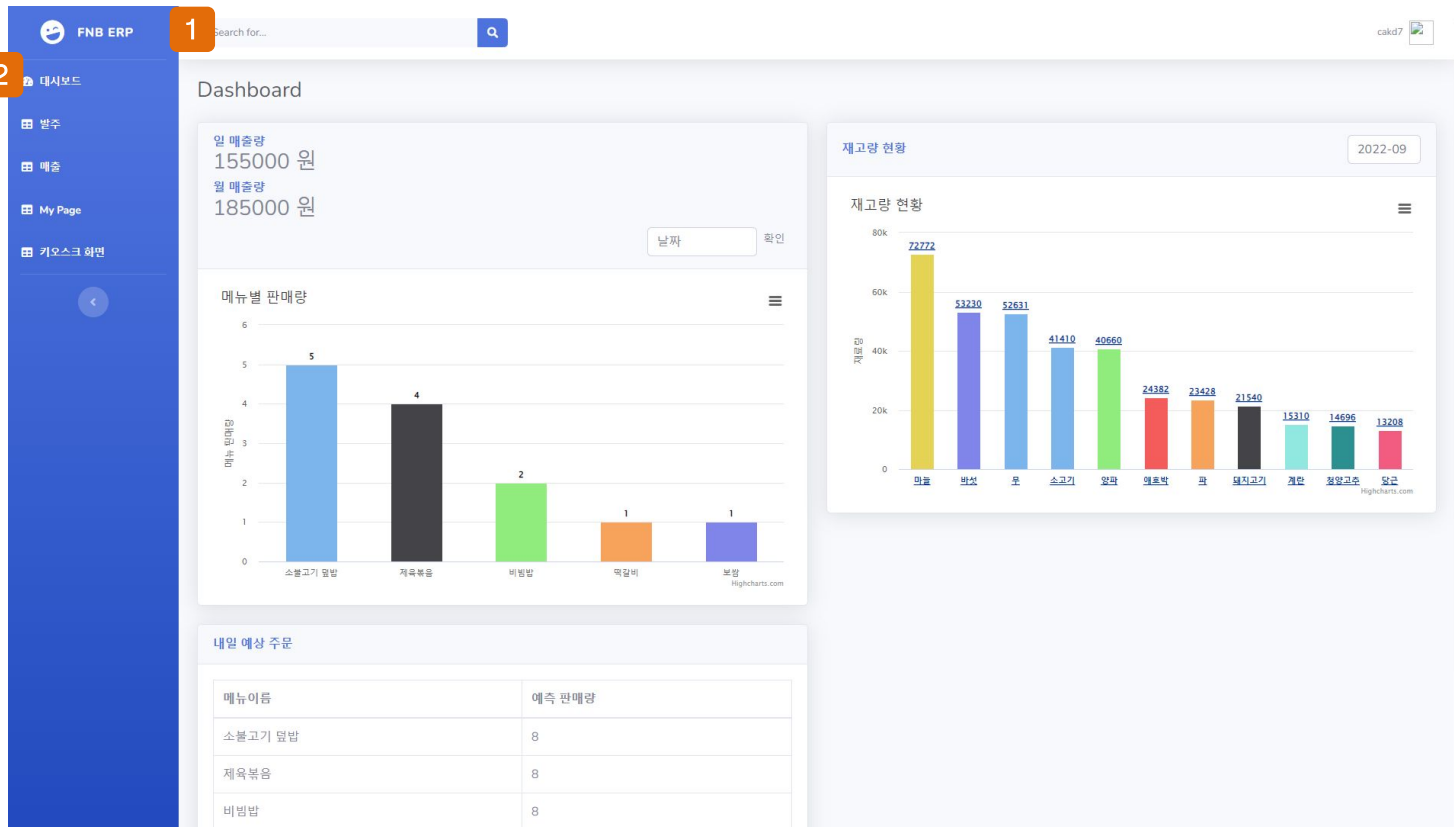
### 3. 페이지 레이아웃

#### URL

```
_cakd_erp > urls.py > ...
17 from importlib.resources import path
18 from django.conf.urls import url, include
19 from django.contrib import admin
20 from erp import views
21 from rest_framework import routers
22 from django.urls import path
23
24 router = routers.DefaultRouter()
25 router.register(r'api_cusord', views.CusordViewSet)
26 router.register(r'api_instock', views.InstockList)
27 router.register(r'api_manager', views.ManagerList)
28 router.register(r'api_material', views.MaterialList)
29 router.register(r'api_menu', views.MenuList)
30 router.register(r'api_ord', views.OrdList)
31 router.register(r'api_outstock', views.OutstockList)
32 router.register(r'api_recipe', views.RecipeList)
33
34
35 urlpatterns = [
36     url(r'^rest/', include(router.urls)),
37     url(r'^admin/', admin.site.urls),
38     path('tmep/', include('template.urls')),
39     path('', include('kiosk.urls')),
40 ]
41
```

### 3. 페이지 레이아웃

#### 3.0 Home 화면 구조

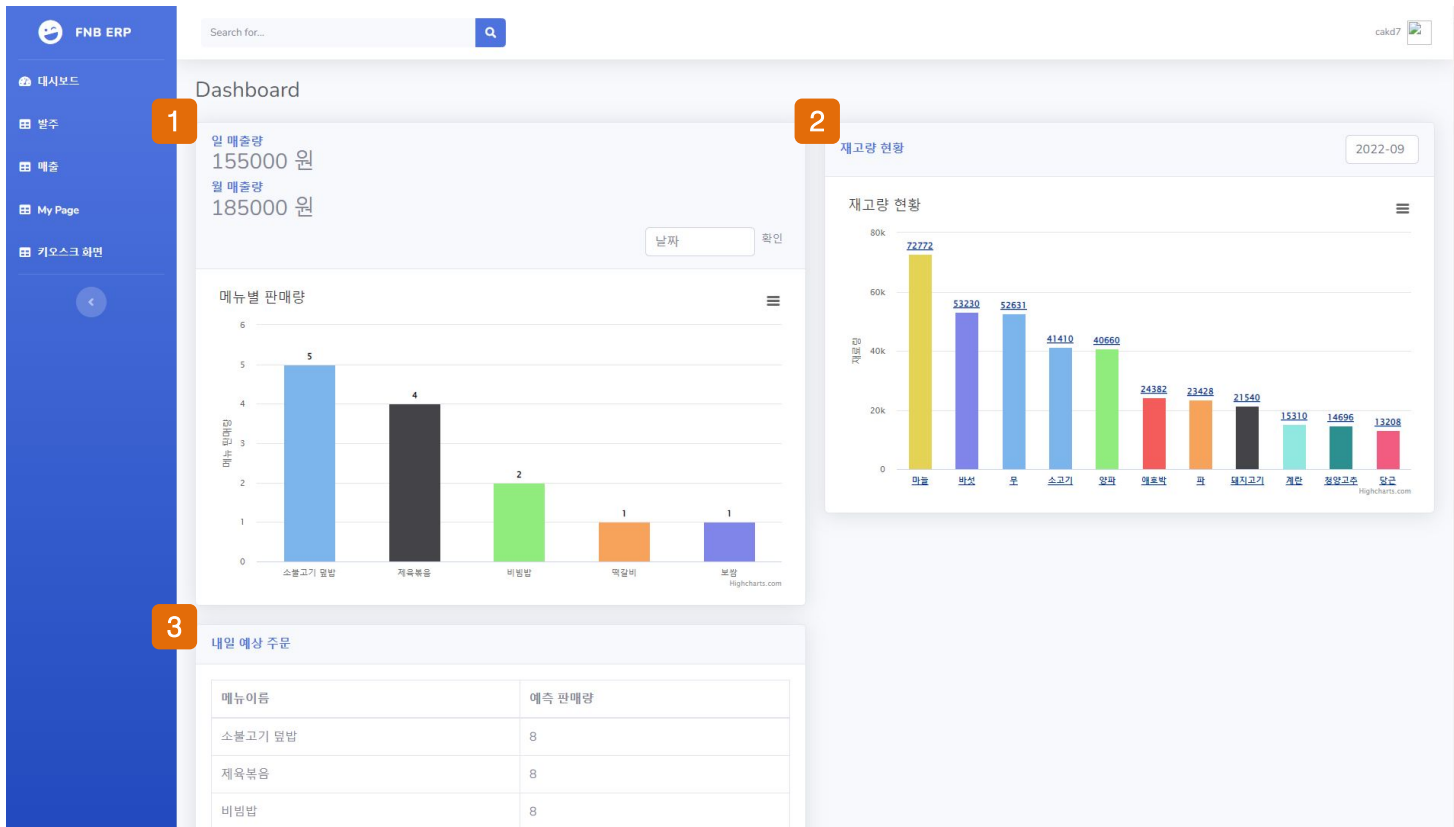


#### Description(화면 설명)

1. Search Bar
2. nav bar 세로형

### 3. 페이지 레이아웃

#### 3.1 메인 페이지 대시보드



#### Description(화면 설명)

1. 월 메뉴 판매량
  - 날짜별 조회 기능 (드롭다운)
  - 일 매출량
  - 월 매출량
  - 막대그래프
  - x축: 메뉴 종류
  - y축: 메뉴 판매량
2. 재고량 현황
  - 날짜별 조회 기능 (드롭다운)
  - 막대그래프
  - x축: 재료 종류
  - y축: 사용량
3. 내일 예상 주문량
  - 메뉴별 예상 판매 수: @@메뉴 00개,...

### 3. 페이지 레이아웃

#### 3.2 발주 현황 페이지

대시보드

발주

매출

My Page

키오스크 화면

Search for...

Q

cakd7

발주 현황

발주 신청

Right now

Show  entries Search:

날짜	발주 번호	대분류	상품명	단가	주문수량	공급가액
Sept. 27, 2022	1	축산물	소고기	300	5000	1500000
Sept. 27, 2022	1	축산물	돼지고기	170	5000	850000
Sept. 27, 2022	1	농산물	양파	30	5000	150000
Sept. 27, 2022	1	농산물	파	50	5000	250000
Sept. 27, 2022	1	농산물	바섯	40	5000	200000
Sept. 27, 2022	1	농산물	당근	50	5000	250000
Sept. 27, 2022	1	농산물	마늘	100	5000	500000
Sept. 27, 2022	1	농산물	청양고추	170	5000	850000
Sept. 27, 2022	1	농산물	애호박	80	5000	400000
Sept. 27, 2022	1	축산물	계란	40	5000	200000

#### Description(화면 설명)

1. 발주 현황 버튼
2. 발주 신청 페이지 버튼
3. 발주 현황 표
  - 날짜
  - 발주 번호
  - 대분류
  - 상품명
  - 단가가
  - 주문 수량
  - 공급가액

## 3. 페이지 레이아웃

### 3.3 발주 신청 페이지

대시보드

발주

매출

My Page

기오스크 화면

1

발주 현황

2

발주 신청

3

발주 신청 폼

4

발주신청

발주 신청

발주번호

중액 302688 원

Showing 1 to 11 of 11 entries

상품번호	상품명	재고(g)	단가(원/10g)	주문수량(g)	공급가액(원)
1	소고기	5400	300	3360	100800
2	돼지고기	4800	170	9120	155040
3	양파	4800	30	3360	10080
4	파	4950	50	1728	8640
5	바섯	5000	40	480	1920
6	당근	4990	50	768	3840
7	마늘	5040	100	672	6720
8	청양고추	5040	170	96	1632
9	애호박	5030	80	192	1536
10	계란	4950	40	960	3840
11	무	50050	30	2880	8640

Showing 1 to 11 of 11 entries

Previous 1 Next

#### Description(화면 설명)

1. 발주 현황 버튼
2. 발주 신청 페이지 버튼
3. 발주 신청 폼
  - 상품번호
  - 상품명
  - 재고
  - 단가
  - 주문수량
  - 공급가액
4. 발주신청 버튼

### 3. 페이지 레이아웃

#### 3.4 매출관리 페이지

대시보드

발주

매출

My Page

키오스크 화면

Search for...

Q

cakd7

매출관리

1 날짜

총매출 #

Show 5 entries Search:

메뉴 ID	메뉴명	메뉴가격	수량	총 판매액
1	소불고기 덮밥	10000	1	10000
2	제육볶음	10000	1	10000
3	비빔밥	10000	1	10000
4	떡갈비	15000	0	0
5	보쌈	30000	0	0
3 총매출			30000 원	

Showing 1 to 5 of 5 entries

Previous 1 Next

총매출

#### Description(화면 설명)

1.날짜 선택

2.매출 관리 표


- 메뉴ID
- 메뉴명
- 메뉴 가격
- 수량
- 총 판매액

3.일일 총 매출액



### 3. 페이지 레이아웃

#### 3.5 마이페이지

 FNB ERP


대시보드

발주

매출


My Page

키오스크 화면



Search for...

Q

cakd7

My page

Right now

1

MY\_PAGE

사업자등록번호	105-91-95789
가게 주소	서울특별시
가게 전화번호	02-313-1711
대표자명	중앙정보처리가든
재고 안전율 초기값	1.2

#### Description(화면 설명)

1. 레스토랑 정보 관련 입력 테이블

### 3. 페이지 레이아웃

#### 3.6 키오스크

1



소불고기 덮밥  
10,000원 12,000원

2

주문



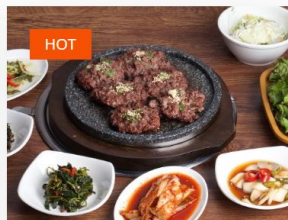
제육볶음  
10,000원 12,000원

주문



비빔밥  
10,000원 12,000원

주문



떡갈비 정식  
15,000원 18,000원

주문



보쌈  
30,000원 35,000원

주문

3

관리자페이지



#### Description(화면 설명)

1. 키오스크 메뉴 이미지
2. 주문 버튼
3. 관리자페이지 버튼  
- (대시보드로 이동)

---

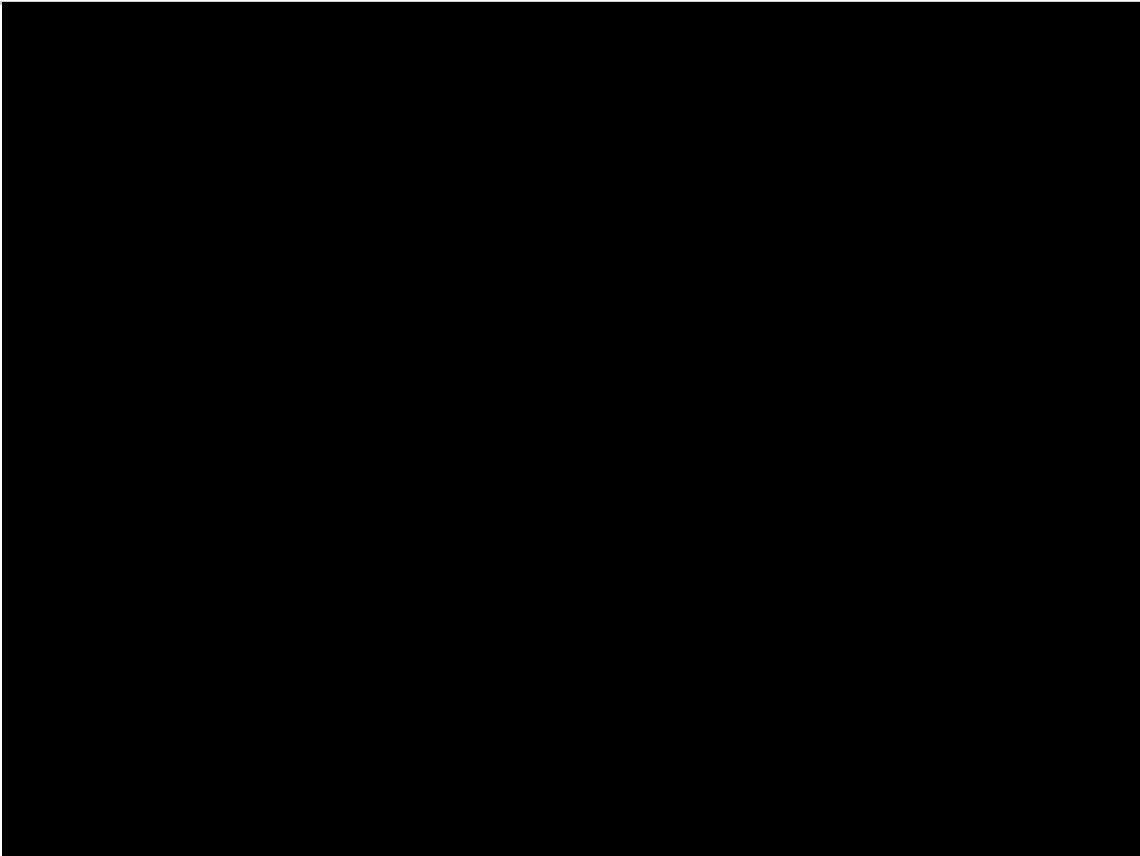
## 4. 시현

- 4.1 DB 생성
- 4.2 기본 Data 입력
- 4.3 manager mypage
- 4.4 cusOrd 더미 입력
- 4.5 메뉴 판매량
- 4.6 발주 신청
- 4.7 머신러닝
- 4.8 주문



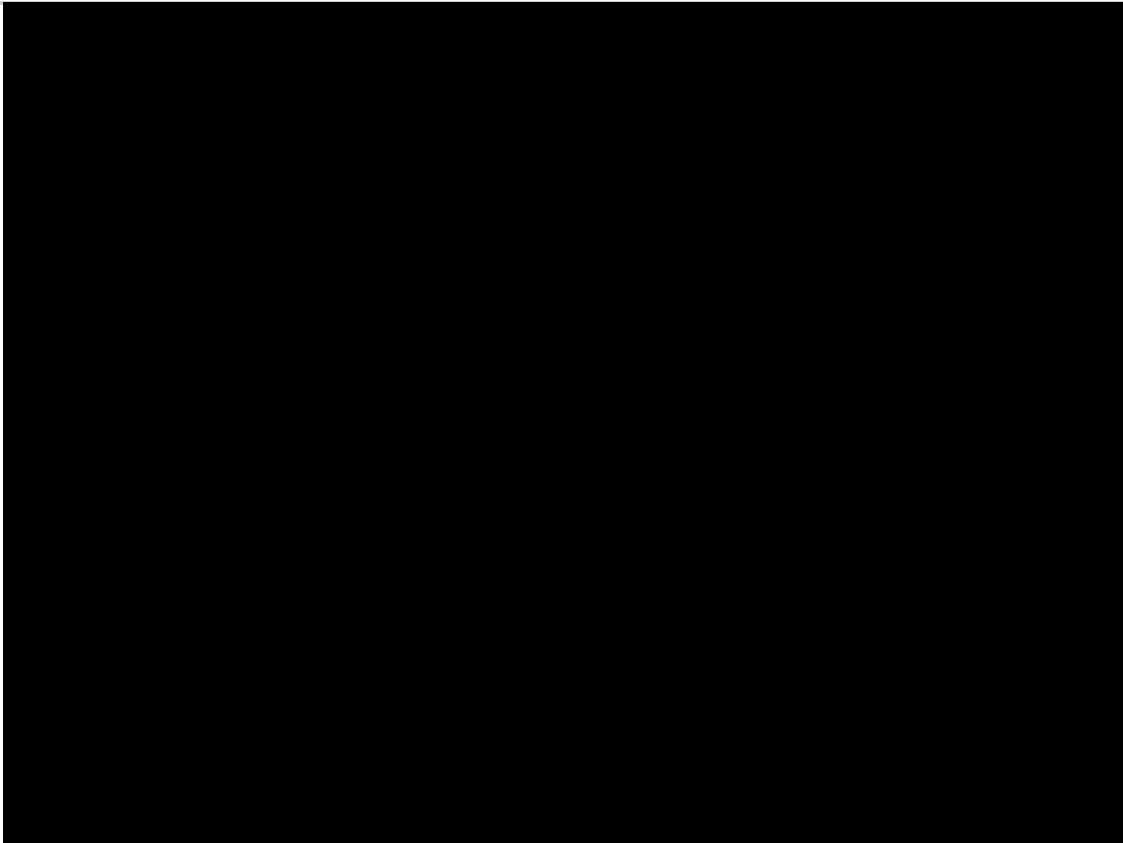
---

## 4.1 DB\_생성

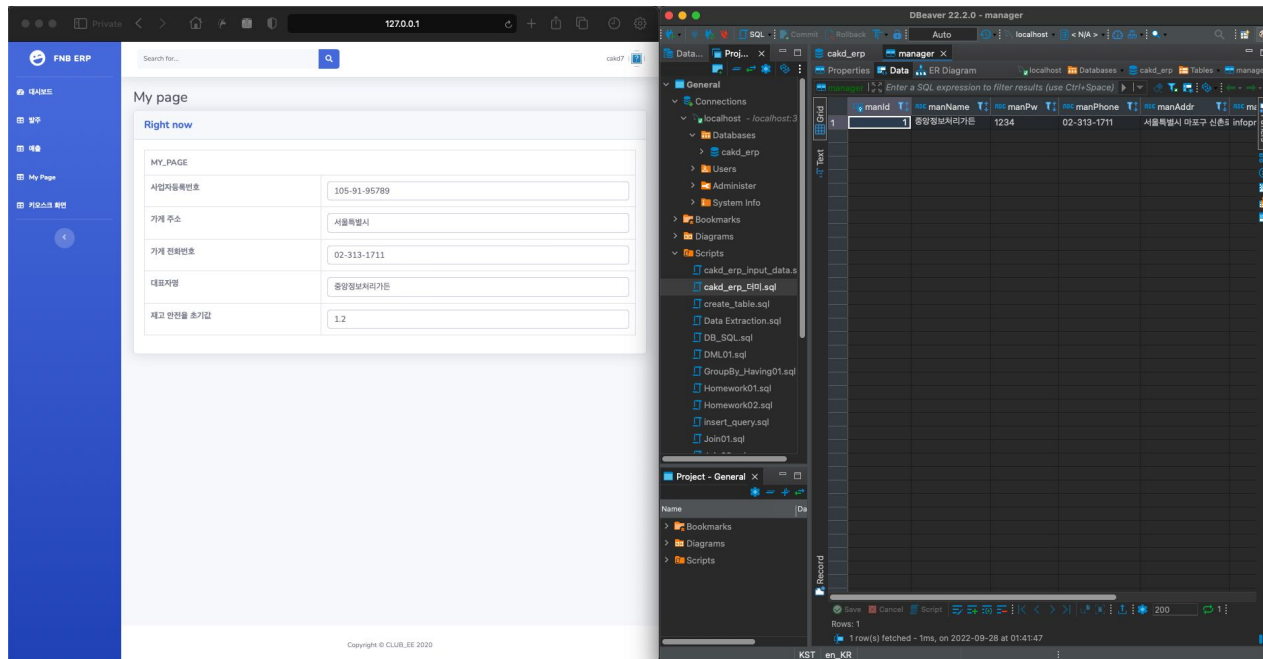


---

## 4.2 기본 Data 입력

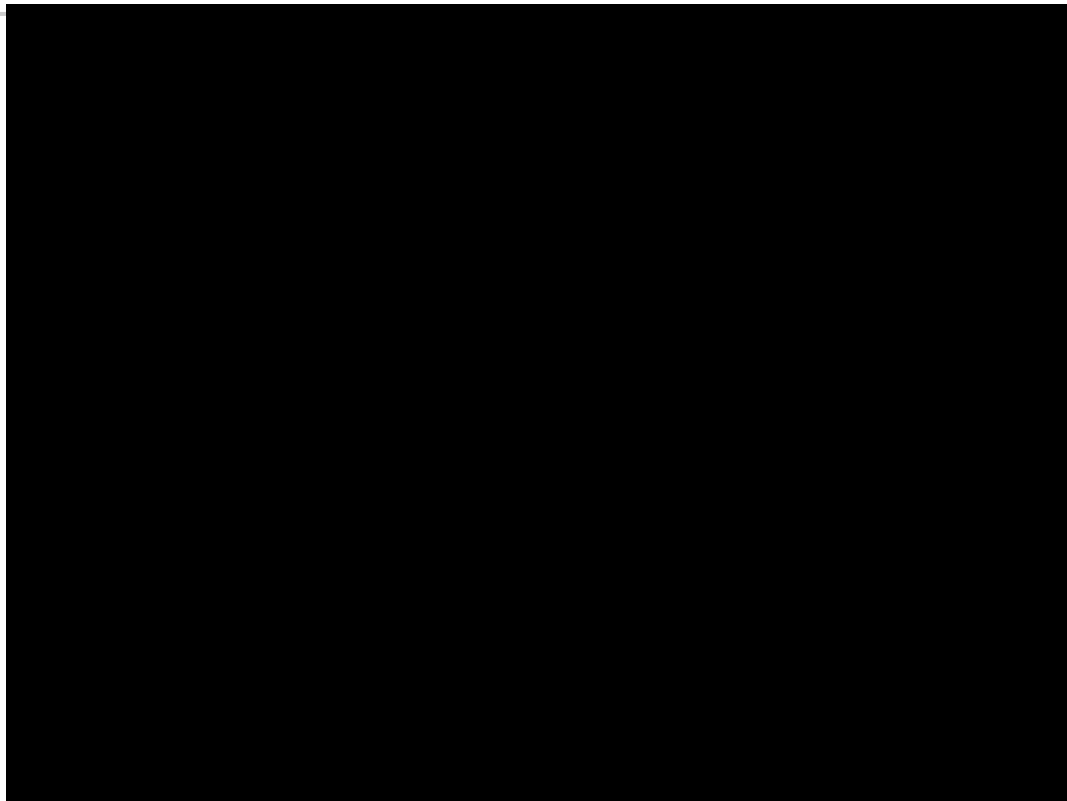


## 4.3 manager\_mypage



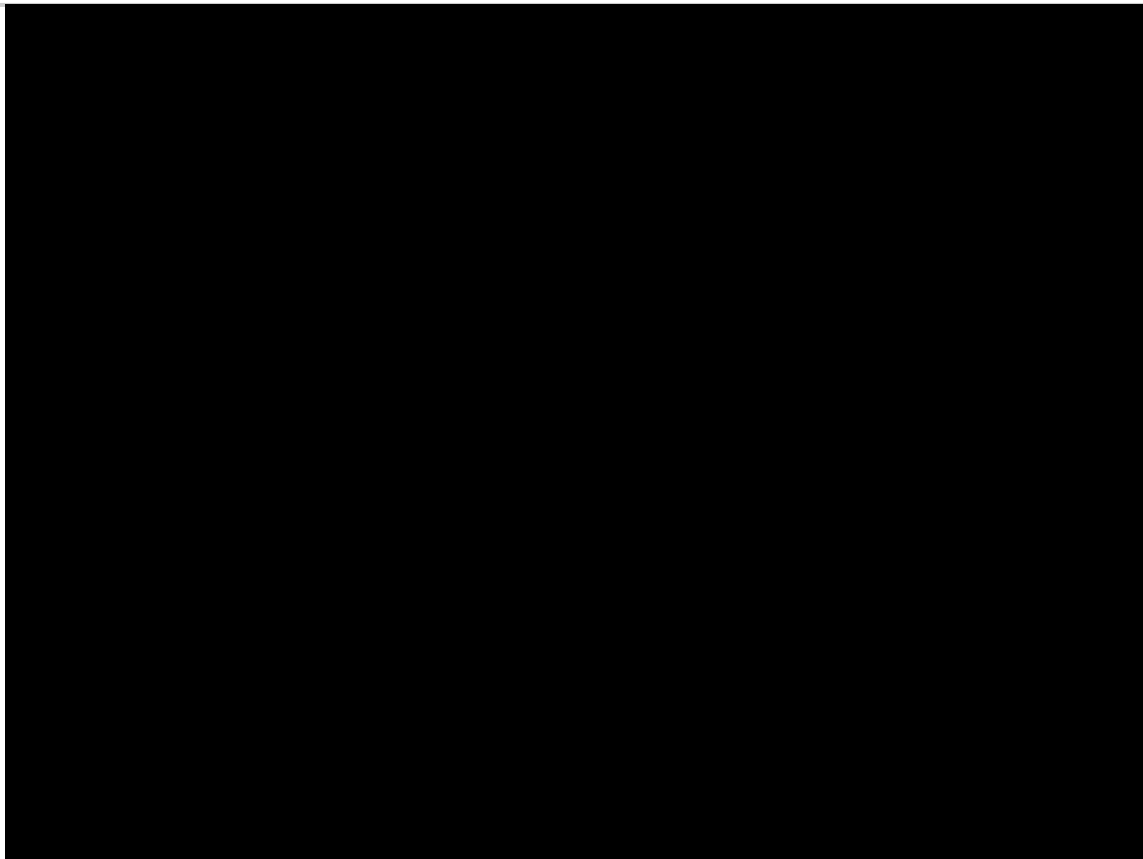
---

## 4.4 cusOrd 더미 입력



---

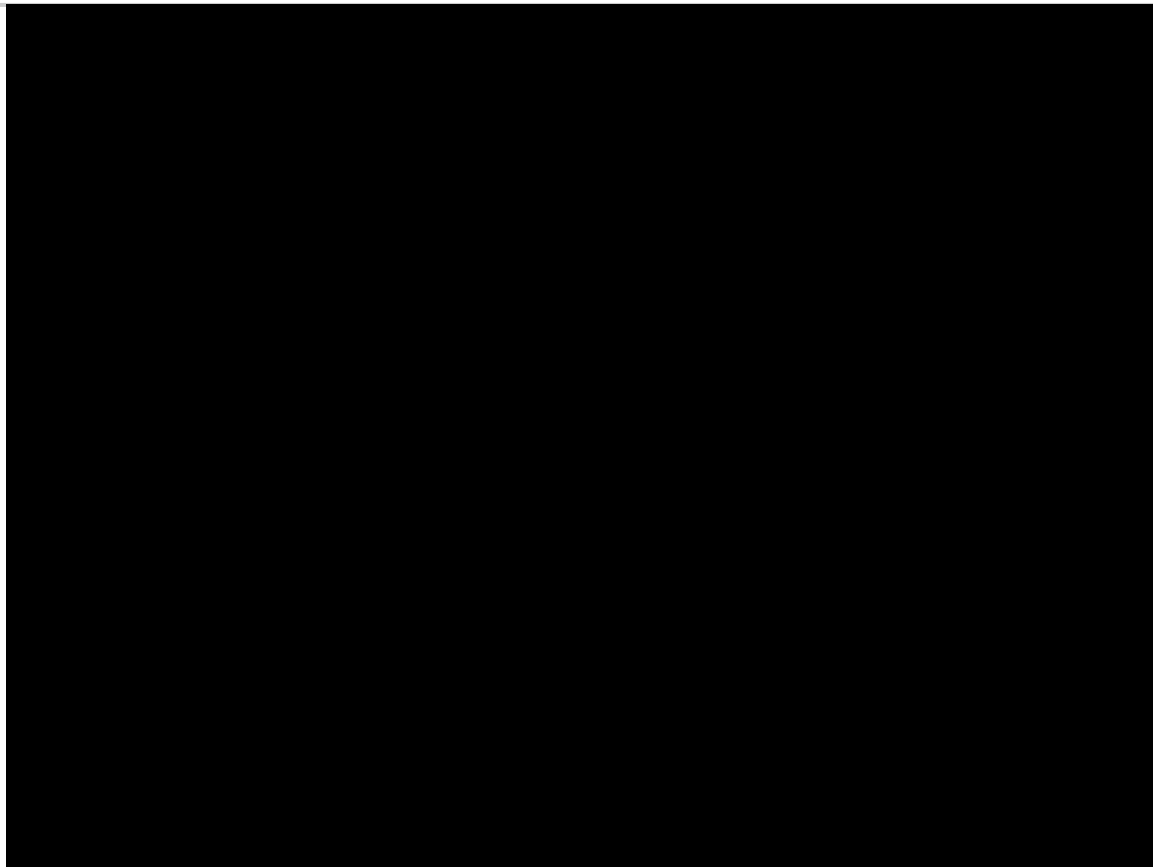
## 4.5 메뉴\_판매량





---

## 4.6 발주 신청



## 4.7 머신러닝

- 원본 컬럼 ['메뉴', '오전/오후', '요일', '주문 수']
- 메뉴, 오전/오후, 요일 데이터로 >> 메뉴 판매 갯수를 학습

```
[17]: df_g_count
```

```
[17]:
```

	outTime	menuId	APM	weekday	count
0	2022-01-01	1	0	5	3
1	2022-01-01	1	1	5	1
2	2022-01-01	2	0	5	6
3	2022-01-01	2	1	5	5
4	2022-01-01	3	0	5	5

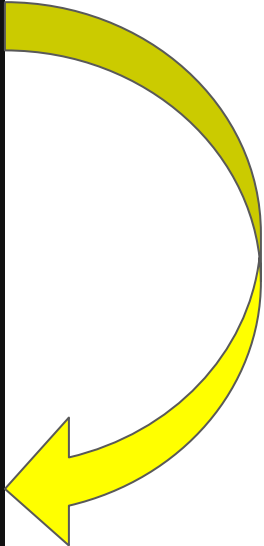
```
[171]: display(feature, target)
```

	menuId	APM	weekday
0	1	0	5
1	1	1	5
2	2	0	5
3	2	1	5
4	3	0	5
...	...	...	...
2596	3	1	3
2597	4	0	3
2598	4	1	3
2599	5	0	3
2600	5	1	3

2601 rows x 3 columns

0	3
1	1
2	6
3	5
4	5
...	...
2596	3
2597	5
2598	3
2599	1
2600	2

```
Name: count, Length: 2601, dtype: int64
```



## 4.7 머신러닝

- 엘라스틱넷 회귀모델을 이용

적용 파라미터

- L1 ratio = 0.7
- alpha = 3

엘라스틱 회귀

```
[45]: from sklearn.model_selection import cross_val_score
      from sklearn.linear_model import Lasso, ElasticNet

      # alpha값에 따른 회귀 모델의 폴드 평균 RMSE를 출력하고 회귀 계수값들을 DataFrame으로 반환
      def get_linear_reg_eval(model_name, params=None, X_data_n=None, y_target_n=None,
                             verbose=True, return_coeff=True, l1_ratio=None):
          coeff_df = pd.DataFrame()
          if verbose: print('##### ', model_name, '#####')
          if model_name == 'ElasticNet':
              print(f'<l1_ratio:{l1_ratio}>')
          for param in params:
              if model_name == 'Ridge': model = Ridge(alpha=param)
              elif model_name == 'Lasso': model = Lasso(alpha=param)
              elif model_name == 'ElasticNet': model = ElasticNet(alpha=param, l1_ratio=l1_ratio)
              neg_mse_scores = cross_val_score(model, X_data_n,
                                              y_target_n, scoring="neg_mean_squared_error", cv = 5)
              avg_rmse = np.mean(np.sqrt(-1 * neg_mse_scores))
              print(f'alpha {0}일 때 5 폴드 세트의 평균 RMSE: {1:.3f} '.format(param, avg_rmse))
              # cross_val_score는 evaluation metric만 반환하므로 모델을 다시 학습하여 회귀 계수 추출

              model.fit(X_data_n, y_target_n)
              if return_coeff:
                  # alpha에 따른 피쳐별 회귀 계수를 Series로 반환하고 이를 DataFrame의 컬럼으로 추가.
                  coeff = pd.Series(data=model.coef_, index=X_data_n.columns)
                  colname = 'alpha:' + str(param)
                  coeff_df[colname] = coeff

          return coeff_df, model
      # end of get_linear_reg_eval
```

```
[48]: elastic_alphas = [ 0.07, 0.1, 0.5, 1, 3]
      coeff_elastic_df_1, elastic_1 = get_linear_reg_eval('ElasticNet', params=elastic_alphas,
                                                         X_data_n=feature, y_target_n=target, l1_ratio=0.7)
      coeff_elastic_df_2, elastic_2 = get_linear_reg_eval('ElasticNet', params=elastic_alphas,
                                                         X_data_n=feature, y_target_n=target, l1_ratio=0.3)

      ##### ElasticNet #####
      <l1_ratio:0.7>
      alpha 0.07일 때 5 폴드 세트의 평균 RMSE: 1.910
      alpha 0.1일 때 5 폴드 세트의 평균 RMSE: 1.910
      alpha 0.5일 때 5 폴드 세트의 평균 RMSE: 1.909
      alpha 3일 때 5 폴드 세트의 평균 RMSE: 1.909

      <l1_ratio:0.3>
      alpha 0.07일 때 5 폴드 세트의 평균 RMSE: 1.910
      alpha 0.1일 때 5 폴드 세트의 평균 RMSE: 1.910
      alpha 0.5일 때 5 폴드 세트의 평균 RMSE: 1.909
      alpha 1일 때 5 폴드 세트의 평균 RMSE: 1.909
      alpha 3일 때 5 폴드 세트의 평균 RMSE: 1.909
```

## 4.7 머신러닝

- 결과

사용 모델



```
[53]: # 모델 객체 만들
import pickle
import joblib

filename = 'elastic_model.pkl'
joblib.dump(elastic_1, filename)

[53]: ['elastic_model.pkl']

[54]: elastic = joblib.load('elastic_model.pkl')
elastic

[54]: ElasticNet(alpha=3, l1_ratio=0.7)

[63]: data = [1,0,5]
data_2 = [1,1,5]

df = pd.DataFrame(columns=['menuId', 'APM', 'weekday'])
df.loc[0,:] = data
df.loc[1,:] = data_2
display(df)
y_pred = elastic.predict(df)
y_pred

   menuId  APM  weekday
0        1    0        5
1        1    1        5

[63]: array([3.84467512, 3.84467512])
```



예시)  
소불고기(메뉴:1)는  
금요일에  
오전/오후 3.8개씩 판매  
예측

## 4.7 머신러닝

- 장고 view.py에서 모델 로드

내일 메뉴별  
판매량 예측

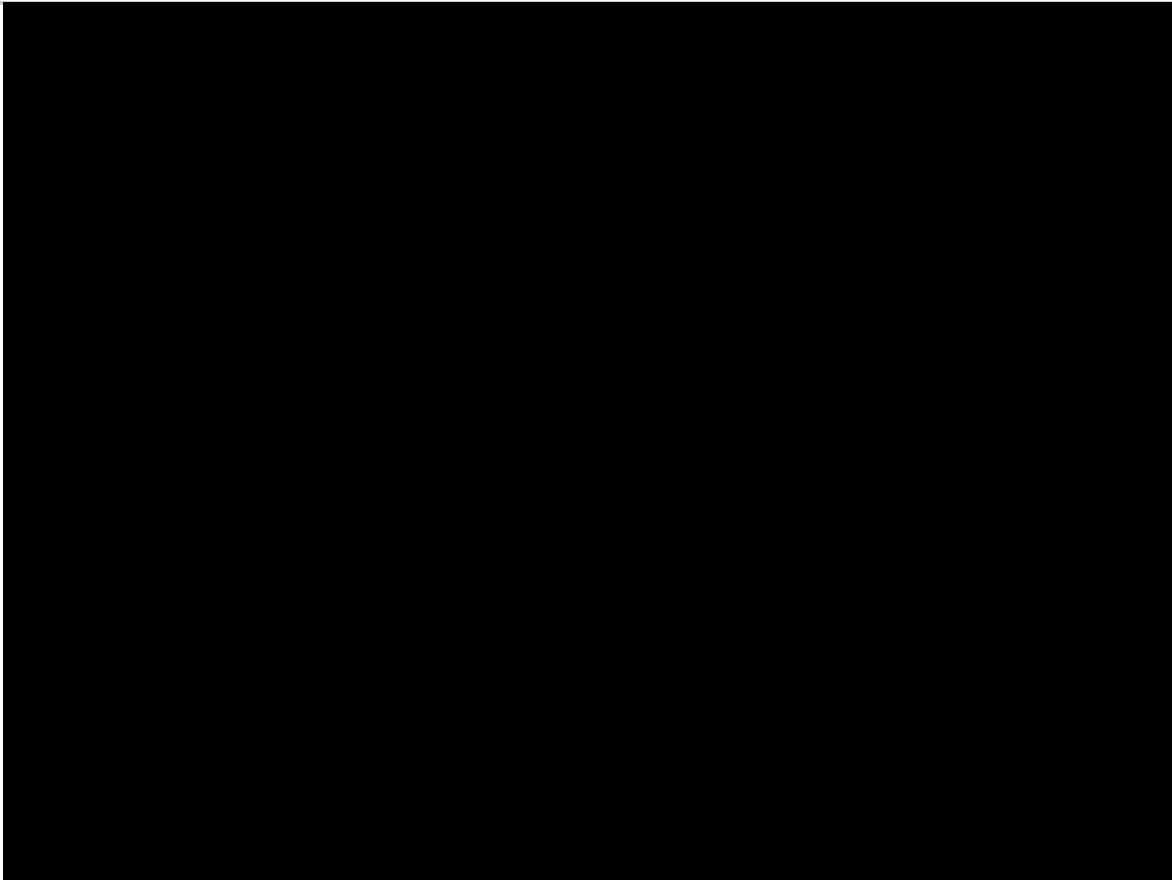
```
template > views.py > MyAPIView
56 def dash(request):
57
58     queryset1 = Material.objects.all()
59     if request.method == 'POST':
60         month = request.POST.get('month')
61         menu_li = Cusord.objects.filter(out_time=month)
62         menu_ = [str(i.menu_id.menu_id) for i in list(menu_li)]
63
64     from collections import Counter
65
66     menu_ = Counter(menu_)
67     menu_list = []
68     for i in [str(j) for j in range(1, 6)]:
69         menu_list.append(menu_[i])
70
71     menu_name_list = []
72     for i in range(1, 6):
73         menu_name_list.append(Menu.objects.get(menu_id=i))
74
75     menu_zip = zip(menu_name_list, menu_list)
76
77     elastic = joblib.load('template/elastic_model.pkl')
78     df = pd.DataFrame(columns=['menuId', 'APM', 'weekday'])
79     global sum_list
80     sum_list = []
81     menu_name_li = []
82     next_day = datetime.now().weekday() + 1
83     for i in range(1, 6):
84         data = [i, 0, next_day]
85         data_2 = [i, 1, next_day]
86         df.loc[0, :] = data
87         df.loc[1, :] = data_2
88         y_pred = elastic.predict(df)
89         predict = round(sum(y_pred))
90         sum_list.append(predict)
91         menu_name_li.append(Menu.objects.get(pk=i).menu_name)
92
93     zip_list = zip(menu_name_li, sum_list)
94
95     return render(request, 'dash.html', {'material': queryset1
96
```

### 내일 예상 주문

메뉴이름	예측 판매량
소불고기 덮밥	8
제육볶음	8
비빔밥	8
떡갈비	8
보쌈	8

---

## 4.8 주문



## 5. 보완할 점

---

- 머신러닝 고도화
- 로그인 / 회원가입
- 세밀한 필터링 구현
- PEP8 규정에 맞는 코딩

## 느낀점

---

한지웅	가장 대중적인 RDBMS인 MySQL을 사용하여 프로젝트에 적용할 RDB를 구성하고, Django를 사용하여 CRUD를 구현해 보았습니다. 또한 수집된 데이터를 사용하여 미래의 수요량을 자동으로 발주해 주는 머신러닝 예측 시스템을 만들어 보았습니다. DB 구축부터 데이터 수집, 적재, 가공 후 웹에 배포까지 일련의 과정을 모두 경험해 볼 수 있는 프로젝트였습니다. 각자의 역할에 충실하게 임해준 조원들 덕에 짧은 기간에도 불구하고 계획했던 모든 기능들을 성공적으로 구현해 냈다는 점에서 더 뜻깊었습니다.
김민섭	백엔드 프론트엔드를 넘나들며 데이터 처리, 렌더링 등 풀스택 개발자가 되기 위한 도메인 지식들을 다양하게 학습할 좋은 기회였습니다. 하면 할수록 부족함을 많이 느꼈습니다. 스스로 더 발전하는 과정속에서 앞으로도 기억이 많이 남을 좋은 프로젝트였습니다.
이상윤	안보이는 길을 하나하나 더듬어가며 가는 느낌이었습니다. DB공부를 더 해야될 것 같습니다.
신창훈	MySQL을 사용하면서 처음 DB와 SQL문을 접했는데 굉장히 효율적인 언어라고 생각되었습니다. 또한 DB의 정보를 가져와서 활용하는 코드를 만들면서 DB와 웹이 연결되는 과정을 알았고 시각화를 진행하면서 자바스크립트를 쓰게 되었는데 자바스크립트에 대한 지식이 모자라서 원하는대로 구현을 못했던 게 아쉬웠습니다.



End of Document

감사합니다!