스마트주문기
서버기능 명세 및 API

2015년 05월 17일

총괄 책임	
개발 기간	~ 2015-05-17
제출자	임승한
검수자	
참여인원	임승한, 김영성, 이상현, 송민아, 김희승

문서 이력

파일명	작성일(버전)	작성자	특이사항	비고
스마트주문기_서버기능명세_및_API_v1.0_임승한	2014-09-24(1.0)	임승한	초안 작성	초안
스마트주문기_서버기능명세_및_API_v1.1_임승한	2014-11-18(1.1)	임승한	재구성, 메뉴추가	
스마트주문기_서버기능명세_및_API_v2.0_임승한	2015-01-26(2.0)	임승한	DB설계변경	재구성
스마트주문기_서버기능명세_및_API_v2.1_임승한	2015-01-27(2.1)	임승한	태블,모바일 완료	
스마트주문기_서버기능명세_및_API_v2.2_임승한	2015-01-28(2.2)	임승한	추가(모바일, 웹)	
스마트주문기_서버기능명세_및_API_v2.3_임승한	2015-01-29(2.3)	임승한	웹, 결제 추가	
스마트주문기_서버기능명세_및_API_v2.4_임승한	2015-02-01(2.4)	임승한	통계기능 추가	
스마트주문기_서버기능명세_및_API_v2.5_임승한	2015-04-14(2.5)	임승한	결제, 좋아요 추가	
스마트주문기_서버기능명세_및_API_v2.6_임승한	2015-04-28(2.6)	임승한	별점, 할인 추가	
스마트주문기_서버기능명세_및_API_v2.7_임승한	2015-05-03(2.7)	임승한	menuFinder수정	
스마트주문기_서버기능명세_및_API_v2.8_임승한	2015-05-17(2.8)	임승한	종업원 호출 추가	

목 차

1. 개발 및 운영 환경

1.1. 개발 환경

서버 : Ubuntu 12.04 LTS

서버 주소 : http://117.16.231.212:8010/

사용 언어 : Node.js

사용 웹서버 : Express(Node.js)

사용 데이터베이스 : Mysql 5.5

1.2 전체 API 요약 설명

1.2.1 서버기능 명세 - Tablet

메뉴	설명
메뉴 출력	간단한 메뉴정보 출력
메뉴 상세출력	메뉴 상세정보 출력 - 메뉴 클릭 시 상세내용 출력
주문 입력	- 주문 입력(메뉴, 가격, 고객 명수, 테이블 번호, 주문한 종업원, 회원 고객 ID) - 주문 입력할 때 total과 서버에서 계산한 값을 비교해서 같으면 입력된다. - 주문 입력 후 종업원 앱에 PUSH 알람을 해줌.
추가 주문	- 이미 주문한 테이블에서 추가주문을 할 경우 사용 - 전체 주문리스트에서 해당 주문목록을 찾아 메뉴를 추가하고, 총액을 재계산한다. - 위의 과정을 하기 위해서 초기 주문시 리턴받은 '**orderID**'를 인자로 넘겨줘야 한다. - 추가 주문 시 테이블의 상태가 '추가 주문'으로 변경된다.
서빙 완료	- 초기주문 혹은 추가주문의 서빙을 완료 했을 때 호출. - 해당 테이블의 상태가 '식사중'으로 변경된다.
테이블 정리 완료	- 종업원이 테이블 정리를 완료 하면 NFC 태그를 찍어서 호출 - 해당 테이블의 상태가 '빈좌석'으로 변경된다.

1.2.2 서버기능 명세 - Andriod(주방)

메뉴	설명
새로운 주문 확인	- 태블릿에서 주문을 받을 경우, 주방 앱에서 PUSH알림을 받는다. - PUSH 알림을 받거나, 새로고침 버튼을 통해 현재 주문을 확인하기 위해 사용 - 조리가 필요한 메뉴들만 보인다. - 즉, 조리가 필요없는 메뉴(술 등)는 주방앱이 아닌 종업원(홀)앱에서 확인. - 리턴해주는 값 중 **orderMenuID**는 조리완료 호출 시 필요하므로 저장해놓아야 함.
조리 완료	- 주문확인 했을 때, 리턴 받은 **orderMenuID**를 사용한다. - 메뉴의 조리가 완료 되었을 때 종업원(주방)앱에서 호출 - 호출 시에 해당 메뉴가 조리 완료 되었다고 종업원(홀)앱에 PUSH 알림

1.2.3. 서버기능 명세 - Andriod(홀)

<table>
<tr><th>메뉴</th><th>설명</th></tr>
<tr><td>조리 완료된 메뉴 확인</td><td>- 조리완료되었다는 PUSH알림을 받을 때, 또는 새로고침 할 때 호출
- 종업원(홀)앱에서 조리완료(즉, 서빙해야 할)된 메뉴들의 목록을 반환해준다.</td></tr>
<tr><td>테이블 상태 확인</td><td>- 모든 테이블의 상태를 반환해준다.
- 테이블의 상태는 아래와 같이 총 5개의 상태가 존재.
> 주문완료(order), 식사중(have), 추가주문완료(add), 정리중(clean), 빈좌석(empty)
- 테이블의 상태가 변경되는 시점은 아래와 같다.
<table>
<tr><th>변경시점</th><th>테이블 상태</th></tr>
<tr><td>주문 할 때</td><td>주문완료(order)</td></tr>
<tr><td>모든 주문 서빙완료</td><td>식사 중(have)</td></tr>
<tr><td>추가주문 할 때</td><td>추가주문 완료(add)</td></tr>
<tr><td>계산 할 때</td><td>정리 중(clean)</td></tr>
<tr><td>정리완료</td><td>빈 좌석(empty)</td></tr>
</table>
</td></tr>
<tr><td>메뉴별 통계 출력</td><td>보류</td></tr>
</table>

1.2.4. 서버기능 명세 - Web

메뉴	설명
통계 페이지 출력	대시보드 통해 관리자가 통계를 확인 할 수 있는 웹 페이지
메뉴별 판매수량 출력	- 정해진 기간의 메뉴별 판매수량 출력 - 많이 팔린 개수 순으로 정렬
매출확인(연 기준)	연을 입력받고 해당 연의 각각 월들을 기준으로 매출 반환
매출확인(연-월 기준)	연, 월을 입력받고 해당 월의 각각 일들을 기준으로 매출 반환
직원 노동 통계	주어진 기간에 직원별 테이블 치운 횟수를 반환

1.2.5 서버기능 명세 - Mobile Web

메뉴	설명
메뉴 출력	메뉴정보(id, 이름, 가격, 그림url, 재료, 카테고리) 출력
길 안내	길 안내
대기시간 확인	대기시간 확인
예약	예약

1.2.6 결제

메뉴	설명
결제	- 현금 또는 카드로 결제한다. - 결제 완료시 해당 테이블을 '빈자리'로 변경한다.

2. 서버기능 명세 - Tablet

Tablet의 모든 API는 **"서버주소/URL"**로 요청을 하고 **JSON타입**으로 반환

2.1 메뉴 출력

메뉴 출력 API는 **"서버주소/URL"**로 요청을 하고 **JSON타입**으로 반환

기능 : 메뉴정보(id, 이름, 가격, 그림url, 재료, 카테고리) 출력
method : GET
URL : /menuFinder
// ex) /menuFinder?category=스테이크
Param :
category // 카테고리(옵션)
결과 :
result : 1(성공), 0(실패)
data : 아래와 같이 배열형태로 데이터 값 반환

결과예시 :


```
{
  "result": 1,
  "data": [
    {
      "id": 1,
      "krName": "머쉬룸 서로인 스테이크 포 유",
      "enName": "Mushroom Sirloin Steak for You",
      "cookTime": 15,
      "price": 29000,
      "picture": "/images/menu/steak/grid_20130523122803.png",
      "compose": "소고기, 탑서로인, 호주산, 250g",
      "starAvg": 4.5,        // 평가 평점
      "avgCnt": 2,           // 평가한 사람 수
      "category": "스테이크"
    },
    {
      "id": 2,
      "krName": "갈릭 립아이",
      "enName": "Garlic Ribeye",
      "cookTime": 15,
      "price": 34500,
      "picture": "/images/menu/steak/grid_20130523134625.png",
      "compose": "소고기, 꽃등심, 호주산, 280g",
      "category": "스테이크"
```

"starAvg": 5,
"avgCnt": 2
},
{
.....
},
.....
]
}

2.2 메뉴 상세출력

메뉴 상세출력 API는 **"서버주소/URL"**로 요청을 하고 **JSON타입**으로 반환

기능 : 메뉴 상세정보(이름, 가격, 재료, 내용, 그림, 재료설명 등) 출력
method : GET
URL : /menuFinder/detail
// ex) /menuFinder/detail?id=1
Param :
id // **menu ID(필수)**
결과 :
result : 1(성공), 0(실패)
data : 아래와 같이 JSON 타입으로 데이터 값 반환

결과 예시 :

{
"result": 1,
"data": {
"krName": "우드 화이어 그릴 스테이크 오 포와",
"enName": "Wood Fire Grill Steak Au Poivre",
"cookTime": 15,
"price": 36500,
"compose": "소고기, 채끝등심, 호주산, 280g",
"detail": "참나무로 구운 프렌치 페퍼 채끝등심 스테이크₩n(*일부매장 프레임 그릴드 방식 제공됩니다.)",
"ingre_picture": "/images/menu/ingredient/img_partofmeat_rib_strip.png",
"ingre_krName": "채끝등심",
"ingre_enName": "STRIP",
"ingre_detail": "채끝등심으로 갈비 아래부터 허리까지의 등심부위를 의미하며, 등심에 비해 지방은 다소 적으나 스테이크로서의 풍미는 상당히 좋아서 스테이크 매니아들이 즐겨 찾는 부위입니다.미국의 뉴욕주와 비슷한 모양의 채끝 등심을 특별히 뉴욕스트립으로 부르기도 합니다.",

```
    "ingre_addTitle": "KIDS LOVE AUSTRALIAN BEEF",
    "ingre_addDetail": "뉴욕스트립 스테이크 원료인 호주 청정우는 호주에서도 가장 청정한 목장에서 태어나는 순간부터 아웃백에 들어오는 순간까지 깨끗하게 관리 받고 안전하게 공급되는 쇠고기입니다.",
    "sideMenu": [
      {
        "side_name": "감자튀김",
        "side_picture": "/images/menu/sidemenu/img_side_dish6.png"
      }
    ]
  }
}
```


2.3 베스트 메뉴 확인

기능 :

- 1주일 전부터 지금 시간까지 많이 팔린 메뉴 Best 5를 반환한다.
- best 1~5 순서대로 반환한다.

method : GET

URL : /getBestMenus

파라미터 :

결과 :

result : 1(성공), 0(실패)

data : 아래와 같이 JSON 타입으로 데이터 값 반환


```
{
  "result": 1,
  "data": [
    {
      "name": "치킨 까르보나라",          // 메뉴명
      "sales": 65                        // 판매 갯수
    },
    {
      "name": "버팔로 비프 & 애플 머스타드 백 립",
      "sales": 33
    }
  ]
}
```


2.4 주문 입력

기능 :

- 주문 입력(메뉴, 가격, 고객 명수, 테이블 번호, 주문한 종업원, 회원고객 ID)
- 주문 입력할 때 total과 서버에서 계산한 값을 비교해서 같으면 입력된다.

- 주문 입력 후 종업원 앱에 PUSH 알람을 해줌.

method : POST

URL : /insertOrder

파라미터 : **JSON 타입**으로 전송한다.(예시 참조)

total	// 주문한 메뉴들 가격의 총합
id	// 회원고객의 ID
numOfMember	// 고객 명수
tableNum	// 테이블 번호 - **배열 타입** (ex. 한 팀이 3번,4번 테이블 사용 시 : [3, 4])
menu	// 주문메뉴 - **JSON 타입**이며 "메뉴명 : 주문갯수" 형태

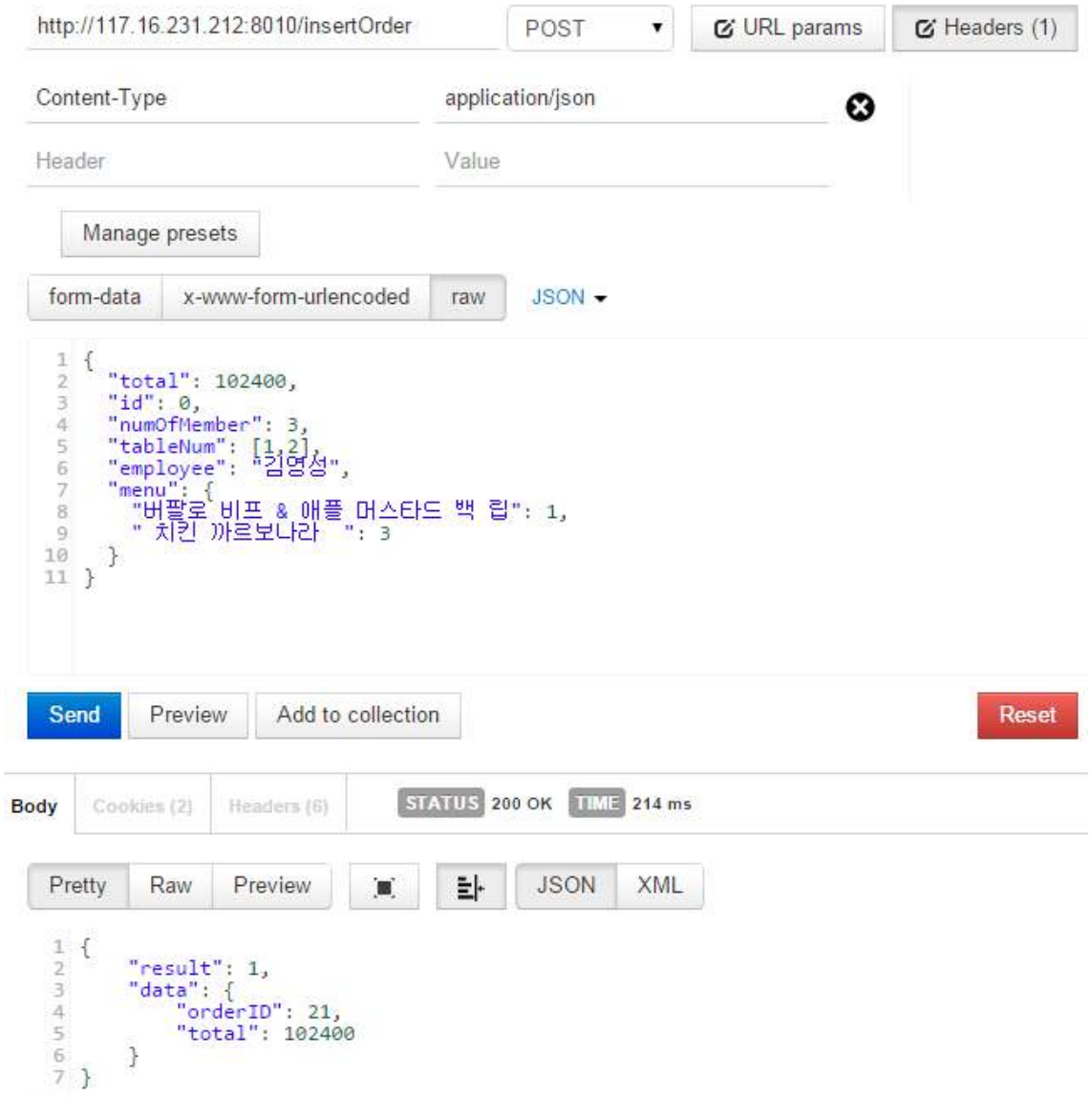
결과 :

result : 1(성공), 0(실패)

data :

orderID	// 주문목록의 고유 ID (추가주문을 포함해 추후 필요하니 저장해두길)
total	// 서버측에서 계산한 메뉴 총합(입력값과 비교검증)

예시 :


http://117.16.231.212:8010/insertOrder
POST
URL params
Headers (1)
Content-Type
application/json
Header
Value
Manage presets
form-data
x-www-form-urlencoded
raw
JSON
{
"total": 102400,
"id": 0,
"numOfMember": 3,
"tableNum": [1,2],
"employee": "김영성",
"menu": {
"버팔로 비프 & 애플 머스타드 백 립": 1,
" 치킨 까르보나라 ": 3
}
}
Send
Preview
Add to collection
Reset
Body
Cookies (2)
Headers (6)
STATUS 200 OK
TIME 214 ms
Pretty
Raw
Preview
JSON
XML
{
"result": 1,
"data": {
"orderID": 21,
"total": 102400
}
}

2.5 추가 주문

기능 :

- 이미 주문한 테이블에서 추가주문을 할 경우 사용
- 전체 주문리스트에서 해당 주문목록을 찾아 메뉴를 추가하고, 총액을 재계산한다.
- 위의 과정을 하기 위해서 초기 주문시 리턴받은 '**orderID**'를 인자로 넘겨줘야 한다.
- 추가 주문 시 테이블의 상태가 '추가 주문'으로 변경된다.

method : POST

URL : /addOrder

파라미터 :

orderID	// 주문 ID(초기 주문시 반환)
total	// 추가 주문한 메뉴들 가격의 총합
menu	// 주문메뉴 - **JSON 타입**이며 "메뉴명 : 주문갯수" 형태

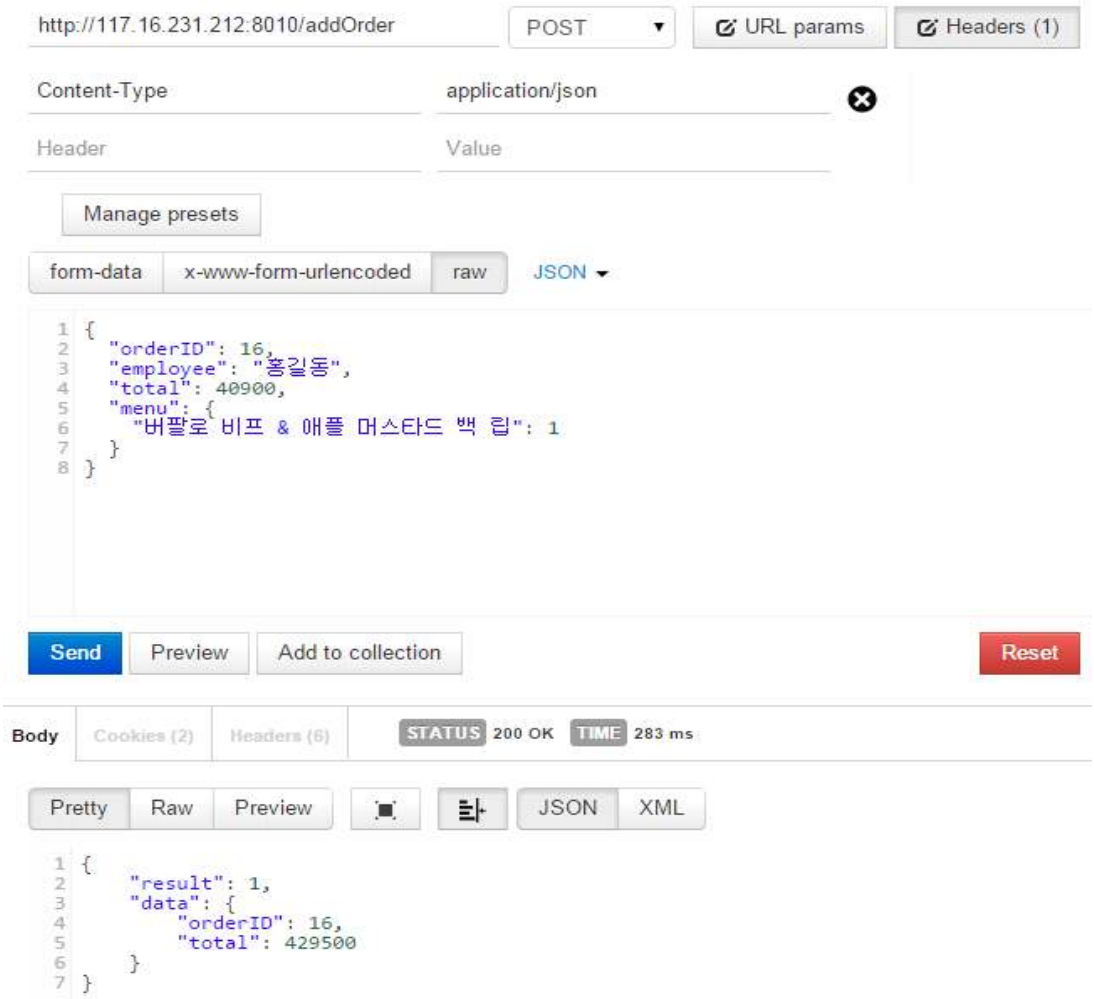
결과 :

result : 1(성공), 0(실패)

data :

orderID	// 주문ID
total	// 총액(기존 총액 + 추가 주문 총액)

예시 :



2.6 테이블 정리 완료

기능 :

- 종업원이 테이블 정리를 완료 하면 NFC 태그를 찍어서 호출
- 종업원 NFC를 각각 식별해서 종업원의 이름도 서버로 전송한다.
  (각각 NFC에 종업원들을 식별 할 정보를 넣어야 할 듯. 이건 알아서 처리 하길.
  추후에 종업원의 테이블 치우는 통계를 확인 할 때 사용)
- 해당 테이블의 상태가 '빈좌석'으로 변경된다.

method : POST

URL : /finishClean

파라미터 :

orderID // 주문 ID(초기 주문시 반환)
(orderID 기준으로 내부적으로 테이블번호 조회)

employee // 종업원이름

결과 :

result : 1(성공), 0(실패)

data : 결과 상세

2.7 종업원 호출

기능 :

- 종업원 호출 기능
- 호출 시 종업원 App으로 PUSH 알림

method : POST

URL : /callEmployee

파라미터 :

tableNum // 호출한 테이블 번호

결과 :

result : 1(성공), 0(실패)

data : 결과 상세

3. 서버기능 명세 - Android(주방)

Android의 모든 API는 **"서버주소/URL"**로 요청을 하고 **JSON타입**으로 반환

3.1 새로운 주문 확인

기능 :

- 태블릿에서 주문을 받을 경우, 주방 앱에서 PUSH알림을 받는다.
- PUSH 알림을 받거나, 새로고침 버튼을 통해 현재 주문을 확인하기 위해 사용
- 조리가 필요한 메뉴들만 보인다.
- 즉, 조리가 필요없는 메뉴(술 등)는 주방앱이 아닌 종업원(홀)앱에서 확인.
- 리턴해주는 값 중 **orderMenuID**는 조리완료 호출 시 필요하므로 저장해놓아야 함.

method : GET

URL : /getOrderList

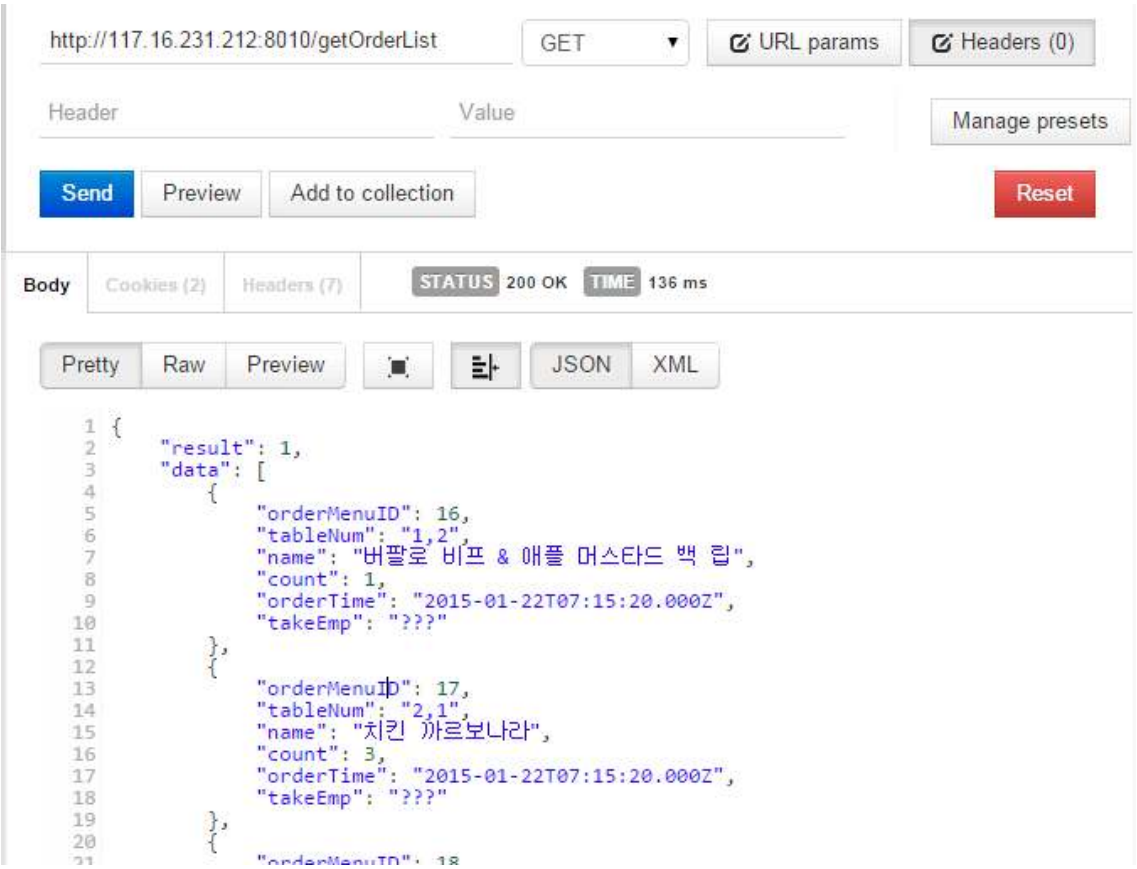
결과 :

result : 1(성공), 0(실패)

data : 아래와 같이 배열형태로 데이터 값 반환

```
[
        {
          orderMenuID : 메뉴의 ID
          tableNum : 주문한 테이블 번호
          name : 주문한 메뉴명,
          count : 주문한 메뉴 갯수,
          orderTime : 주문시각
        }
        ……
]
```

예시 :



3.2 조리 완료

기능 :

- 주문확인 했을 때, 리턴 받은 **orderMenuID**를 사용한다.
- 메뉴의 조리가 완료 되었을 때 종업원(주방)앱에서 호출
- 호출 시에 해당 메뉴가 조리 완료 되었다고 종업원(홀)앱에 PUSH 알림

method : POST

URL : /finishCook

파라미터 :

orderMenuID // 주문메뉴 ID(주문확인 시 반환)

결과 :

result : 1(성공), 0(실패)

data : 결과 상세

4. 서버기능 명세 - Android(홀)

Android의 모든 API는 **"서버주소/URL"**로 요청을 하고 **JSON타입**으로 반환

4.1 조리 완료된 메뉴 확인

기능 :

- 조리완료 되었다는 PUSH 알림을 받을 때, 또는 새로고침 할 때 호출
- 종업원(홀) 앱에서 조리완료(즉, 서빙 해야 할)된 메뉴들의 목록을 반환해준다.

method : GET

URL : /getCookedList

결과 :

result : 1(성공), 0(실패)

data : 아래와 같이 배열형태로 데이터 값 반환

```
[
    {
     orderMenuID : 메뉴의 ID
     tableNum : 주문한 테이블 번호
     name : 주문한 메뉴명,
     count : 주문한 메뉴 갯수,
     orderTime : 주문시각
    }
    ……..
]
```

4.2 메뉴 서빙 완료

기능 :

- 해당 메뉴의 서빙완료를 서버에 알림
- 서버에서는 해당 메뉴의 서빙 상태를 완료로 변경한다.

method : POST

URL : /finishServe

파라미터 :

orderMenuID // 주문메뉴 ID(주문확인 시 반환)

결과 :

result : 2(모든 메뉴 서빙완료), 1(남은 메뉴 있음), 0(실패)

data : 결과 상세

4.3 테이블 상태 확인

기능 :

- 모든 테이블의 상태를 반환해준다.
- 테이블의 상태는 아래와 같이 총 5개의 상태가 존재.
  > 주문완료(order), 식사중(have), 추가주문완료(add), 정리중(clean), 빈좌석(empty)
- 테이블의 상태가 변경되는 시점은 아래와 같다.

변경시점	테이블 상태
주문 할 때	주문완료(order)
모든 주문 서빙완료	식사 중(have)
추가주문 할 때	추가주문 완료(add)
계산 할 때	정리 중(clean)
정리완료	빈 좌석(empty)

method : GET

URL : /tableState

결과 :

result : 1(성공), 0(실패)

data : 아래와 같이 배열형태로 데이터 값 반환

```
[
    {
     tableNum : 테이블번호
     state : 테이블 상태(order, have, add, clean, empty 중 하나)
    }
    ……
]
```

5. 서버기능 명세 - Web

Web은 **"서버주소/URL"**로 요청을 하면 브라우저상에서 화면에 표시된다.

5.1 통계 페이지 출력

기능 : 대시보드 통해 관리자가 통계를 확인 할 수 있는 웹 페이지

method : GET

URL : /stat

결과 :

브라우저상에 화면 출력(client단에서 출력물 처리)

5.2 메뉴별 판매수량 출력

기능 :

- 정해진 기간의 메뉴별 판매수량 출력
- 많이 팔린 개수 순으로 정렬

method : GET

URL : /bestMenuCount/시작날짜/종료날짜

ex1) /bestMenuCount/2015-01-26/2015-01-28
(2015-01-26부터 2015-01-28까지 조회)
ex2) /bestMenuCount/2015-01-26/2015-01-26
(2015-01-26 조회)

파라미터 :

결과 : **JSON 타입**으로 리턴(데이터 받아서 그리려면 Ajax 이용해야 할 듯)

result : 1(성공), 0(실패)

data : 아래의 형식이 배열 형태로 존재

```
{
  "name" : 메뉴명,
  "sales" : 판매 개수,
  "category" : 메뉴의 카테고리
}
```

예시 :


```
{
    "result": 1,
    "data": [
        {
            "name": "치킨 까르보나라",
            "sales": 9,
            "category": "파스타"
        },
        {
            "name": "버팔로 비프 & 애플 머스타드 백 립",
            "sales": 4,
            "category": "페이버릿"
        }
    ]
}
```

5.3 매출확인(연 기준)

기능 :

- 연을 입력받고 해당 연의 각각 월들을 기준으로 매출 반환

method : GET

URL : /salesYear/연도

ex1) /salesYear/2015

(2015년 조회)

파라미터 :

결과 : **JSON 타입**으로 리턴(데이터 받아서 그리려면 Ajax 이용해야 할 듯)

result : 1(성공), 0(실패)

data : 아래의 형식이 배열 형태로 존재

{

"month" : 월,

"total" : 매출

}

예시 :

5.4 매출확인(연-월 기준)

기능 :

- 연, 월을 입력받고 해당 월의 각각 일들을 기준으로 매출 반환

method : GET

URL : /salesMonth/연도/월

ex1) /salesMonth/2015/01
(2015년 1월 조회)

파라미터 :

결과 : **JSON 타입**으로 리턴(데이터 받아서 그리려면 Ajax 이용해야 할 듯)

result : 1(성공), 0(실패)

data : 아래의 형식이 배열 형태로 존재

```
{
  "day" : 일,
  "total" : 매출
}
```

예시 :


```
{
    "result": 1,
    "data": [
        {
            "day": "2015-01-01",
            "total": 10000
        },
        {
            "day": "2015-01-22",
            "total": 1094700
        },
        {
            "day": "2015-01-23",
            "total": 941500
        },
        {
            "day": "2015-01-26",
            "total": 102400
        },
        {
            "day": "2015-01-28",
            "total": 204800
        },
        {
            "day": "2015-01-29",
            "total": 10000
        },
        {
            "day": "2015-01-30",
            "total": 10000
        },
        {
            "day": "2015-01-31",
            "total": 10000
        }
    ]
}
```

5.5 직원 노동 통계

기능 :

- 주어진 기간에 직원별 테이블 치운 횟수를 반환

method : GET

URL : /employStat/시작날짜/종료날짜

ex1) /employStat/2015-01-26/2015-01-28
(2015-01-26부터 2015-01-28까지 조회)
ex2) /employStat/2015-01-26/2015-01-26
(2015-01-26 조회)

파라미터 :

결과 : **JSON 타입**으로 리턴(데이터 받아서 그리려면 Ajax 이용해야 할 듯)

result : 1(성공), 0(실패)

data : 아래의 형식이 배열 형태로 존재

{
"name" : 직원 이름,
"cnt" : 테이블 치운 횟수
}

예시 :


```
{
    "result": 1,
    "data": [
        {
            "name": "김영성",
            "cnt": 1
        },
        {
            "name": "임승한",
            "cnt": 3
        }
    ]
}
```

6. 서버기능 명세 - Mobile Web

6.1 메뉴 출력

기능 : 메뉴정보(id, 한글이름, 영어이름, 가격, 그림url, 재료, 카테고리) 출력
method : GET
URL : /mMenuFinder
// ex) /mMenuFinder?category=스테이크
Param :
category // 카테고리(옵션)
결과 :
브라우저상에 화면 출력(client단에서 출력물 처리)
DB연동방법 :
results[i].id, // menu ID
results[i].krName, // 메뉴 한글이름
results[i].enName, // 메뉴 영어이름
results[i].cookTime, // 메뉴 조리시간
results[i].price, // 가격
results[i].picture, // 사진 URL
results[i].compose, // 음식 재료
results[i].starAvg, // 평균별점
results[i].avgCnt, // 평가한 사람 수
results[i].category // 카테고리

6.2 메뉴 상세출력

- "2.2 메뉴 상세출력" API 이용.
- jQuery로 Ajax 구성해서 modal 사용하면 될 듯합니다.

6.2 길 안내

기능 : 길 안내
method : GET
URL : /mGuideWay
// ex) /mGuideWay?classify=incheon
Param :
classify // **분류(옵션)**
parameter를 넣으면 해당 지역만, 없으면 전체 반환
현재 분류는 **'incheon', 'gungi', 'seoul'** 세 가지 존재
결과 :
브라우저상에 화면 출력(client단에서 출력물 처리)
DB연동방법 :
results[i].name, // 지점명
results[i].location, // 지점 위치

results[i].phoneNum // 지점 전화번호
results[i].classify // 지점 분류

6.3 대기시간 확인

기능 : 대기시간 확인
method : GET
URL : /mCheckTable
Param :
결과 :
브라우저상에 화면 출력(client단에서 출력물 처리)
알고리즘 :

테이블 상태	대기시간	음수일 경우
주문완료(order)	주문완료 시각 + 70분 - 현재시간	60분
식사 중(have)	초기 주문시 - 서빙완료 시각 + 50분 - 현재시간	30분
	추가 주문시 - 서빙완료 시각 + 30분 - 현재시간	30분
추가주문 완료(add)	추가 주문한 시각 + 40분 - 현재시간	40분
정리 중(clean)	계산한 시각 + 5분 - 현재시간	5분
빈 좌석(empty)	0분	

DB연동방법 :
emptyTable // 비어있는 테이블의 갯수
tableState.length // 전체 테이블 갯수
tableState[i].tableNum // 테이블 번호
(waitTime이 적은 순으로 정렬)
tableState[i].state // 테이블 상태
tableState[i].waitTime // 예상 대기 시간

반환 값 예시 :


```
{
    "emptyTable": 4,
    "tableState": [
        {
            "tableNum": 2,
            "state": "empty",
            "waitTime": "00:00:00"
        },
        {
            "tableNum": 8,
            "state": "empty",
            "waitTime": "00:00:00"
        },
        {
            "tableNum": 7,
            "state": "empty",
```

6.4 예약

기능 : 예약
method : GET
URL : /mBooking
Param :
결과 :
브라우저상에 화면 출력(client단에서 출력물 처리)
DB연동방법 :

7. 결제

7.1 결제금액 조회

기능 :

- 테이블의 결제해야 할 메뉴, 금액을 조회한다

method : GET

URL : /chargeList/테이블번호

파라미터 : URL의 '테이블번호'에 조회할 테이블번호 작성

결과 :

result : 1(성공), 0(실패)

data : {

[

menuName: 메뉴이름,

cnt: 메뉴 주문 개수,

price: 가격(소계)

],

.....

}

7.2 결제

기능 :

- 현금 또는 카드로 결제한다.
- 결제 완료시 해당 테이블을 '정리중'로 변경한다.

method : POST

URL : /charge

파라미터 : **JSON 타입**

tableNum // 테이블번호

cardPayment // 카드결제금액

cashPayment // 현금결제금액

isDiscount **// 0: 할인X, 1: 5%할인**

// 할인가 = Math.floor(합계*0.95)

결과 :

result : 1(성공), 0(실패)

data : 결과 상세

8. 좋아요

8.1 좋아요 개수 증가

기능 :

- '좋아요' 개수 증가

method : POST

URL : /like

파라미터 : 없음

결과 : **JSON 타입**

result : 1(성공), 0(실패)

data : 결과 상세

결과 예시 :

```
{
    "result": 1,
    "data": "[like]: 1 Like Count was added"
}
```

8.1 좋아요 개수 조회

기능 :

- '좋아요' 개수 조회

method : GET

URL : /getLikeCnt

파라미터 : 없음

결과 : **JSON 타입**

result : 1(성공), 0(실패)

data : {

"likeCnt": '좋아요' 개수

}

결과 예시 :

```
{
    "result": 1,
    "data": {
        "likeCnt": 4
    }
}
```

9. 별점

9.1 별점 입력

기능 :

- 메뉴에 대한 별점(평가) 입력

method : POST

URL : /putStar

파라미터 :

starCnt : 별점 개수(1~5개)

menuID : 별점을 줄 메뉴의 ID

결과 : **JSON 타입**

result : 1(성공), 0(실패)

data : 결과 상세

결과 예시 :