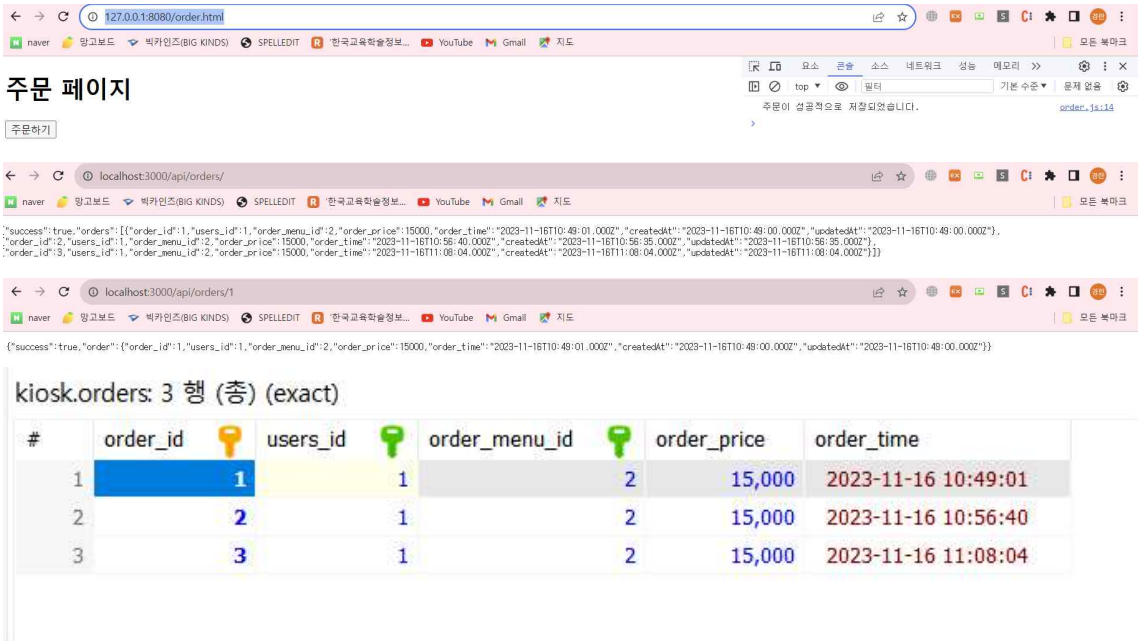
주문 시나리오:
첫번째 방문:
고객: 키오스크 초기화면 클릭 -> 전화번호 입력창 작성 -> 키오스크 주문 화면
사장님: 메뉴 입력 사이트에서 메뉴 입력 -> 데이터베이스 저장 -> 데이터베이스 정보를 가져와서 키오스크에 등록
고객관리시스템: 고객이 키오스크로 주문 -> 주문 기록 및 전화번호 데이터베이스 저장
두번째 방문:
고객: 키오스크 초기화면 클릭 -> 전화번호 입력창 작성
백엔드: 데이터베이스에서 이전 주문 이력 정보 가져오기
추천 알고리즘: 추천 알고리즘을 사용하여 메뉴를 추천
프론트엔드: 키오스크 주문 화면에 추천된 메뉴 표시

1. 주문 기록 시스템:
<주요 내용>
프론트엔드:
주문한 메뉴와 수량, 주문 시간 등을 보기 쉽게 표시하는 대시보드를 제공합니다.
백엔드:
주문이 완료되면 해당 주문 정보를 데이터베이스에 저장합니다.
데이터베이스:
주문 정보를 저장하는 테이블을 관리하고, 새로운 주문이 들어오면 해당 정보를 추가합니다.
<상세 설명>
1. 소개
주문 기록 시스템은 음식 주문과 관련된 정보를 저장하고 조회하는 웹 어플리케이션입니다. 이 시스템은 Express.js와 Sequelize를 사용하여 개발되었으며, MySQL 데이터베이스를 통해 주문 기록을 관리합니다.
2. 시스템 구성 요소
2.1. 기술 스택
Express.js: 웹 어플리케이션 개발을 위한 웹 프레임워크
Sequelize: Node.js에서 사용하는 ORM(Object-Relational Mapping) 라이브러리
MySQL: 데이터베이스 시스템
2.2. 데이터 모델
주문 정보 (OrderRecord)
order_id: 주문 고유 식별자
users_id: 주문한 사용자의 식별자
order_menu_id: 주문한 음식 메뉴의 식별자
order_price: 주문한 음식의 가격
order_time: 주문이 발생한 시각
3. 실행 방법
1. node server.js , http-server 명령 프롬프트에 실행
2. http://127.0.0.1:8080/order.html , http://localhost:3000/api/orders , http://localhost:3000/api/orders/1

[node server.js vs http-server]
node server.js:
목적: Node.js 어플리케이션을 실행하는 데 사용됩니다.
동작: node 명령어를 통해 JavaScript 파일 (server.js와 같은)을 실행합니다. 주로 서버 사이드 코드를 실행하는 데 사용됩니다.
예시: 프로젝트의 백엔드 서버를 실행할 때 사용됩니다.
http-server:
목적: 정적 파일을 간단한 HTTP 서버로 제공하는 데 사용됩니다.
동작: 현재 디렉토리를 기준으로 간단한 HTTP 서버를 실행합니다. 디렉토리 내의 파일들을 브라우저에서 접근 가능하게 만듭니다.
예시: 프로젝트의 정적 파일(HTML, CSS, JavaScript 등)을 로컬에서 테스트할 때 사용됩니다.
따라서, 두 명령어는 서로 다른 목적을 가지고 있습니다. node server.js는 주로 서버 사이드 코드를 실행하는 데 사용되고, http-server는 정적 파일을 제공하는 간단한 로컬 HTTP 서버를 실행하는 데 사용됩니다.



kiosk.orders: 3 행 (총) (exact)

#	order_id	users_id	order_menu_id	order_price	order_time
1	1	1	2	15,000	2023-11-16 10:49:01
2	2	1	2	15,000	2023-11-16 10:56:40
3	3	1	2	15,000	2023-11-16 11:08:04

2. 고객관리 시스템:
프론트엔드:
고객 주문 정보 및 이력을 확인하는 대시보드 및 화면을 개발합니다.
고객 정보를 검색하고 관리할 수 있는 검색 창과 필터를 구현합니다.
백엔드:
키오스크로부터 오는 주문 정보를 받아 고객 정보 및 주문 이력을 데이터베이스에 저장합니다.
사장님 확인용 대시보드에서는 데이터베이스에서 고객 정보 및 주문 이력을 가져와 보여주는 API를 구현합니다.

데이터베이스:
고객 정보와 주문 이력을 저장할 테이블을 생성합니다.
주문 정보에는 메뉴, 수량, 주문 일시 등이 포함됩니다.

 # 주문 기록 시스템과 고객관리시스템의 차이점
1.주문 기록은 단순히 주문에 대한 정보를 저장하고 조회하는 역할을 합니다. 반면에 고객관리 시스템은 이전 주문 이력을 통합적으로 관리하고, 해당 정보를 검색하고 표시하는 역할을 합니다.
2. 두 시스템은 함께 작동하여 고객의 주문 이력을 추적하고, 이를 기반으로 서비스를 제공합니다. 종종, 주문 기록 시스템은 주문 정보에 대한 세부사항을 저장하고, 고객관리 시스템은 해당 주문 정보를 특정 고객과 연결하여 고객 관리에 활용됩니다.
3. 전화번호 연동:
<주요 내용>
프론트엔드:
전화번호를 입력하는 화면을 키오스크에 제공하고, 입력된 전화번호를 백엔드로 전송합니다.
백엔드:
전화번호를 받아 해당 고객의 주문 이력을 데이터베이스에서 조회합니다.
데이터베이스:
전화번호를 기반으로 해당 고객의 주문 이력을 검색하고 반환합니다.
<상세 내용>
기술 스택
Frontend: React.js
Backend: Node.js (Express.js 프레임워크)
Database: MySQL
호스팅 및 배포: AWS (Amazon RDS, EC2), Docker
[실행방법]
1. PS C:\project\customer_management> node server.js

kiosk.users: 4 행 (총) (exact)

#	users_id	phone_number	createdAt	updatedAt
1	1	01036889161	2023-11-16 14:50:08	2023-11-16 14:50:08
2	3	01012345678	2023-11-16 14:53:57	2023-11-16 14:53:57
3	4	01011111111	2023-11-16 14:59:59	2023-11-16 14:59:59
4	5	01022222222	2023-11-16 15:10:36	2023-11-16 15:10:36

4. 추천 기능:
프론트엔드:
추천된 메뉴를 표시하는 화면을 구현합니다.
백엔드:
고객의 이전 주문 이력과 추천 알고리즘을 활용하여 새로운 메뉴를 추천합니다.
데이터베이스:
고객의 이력 정보를 바탕으로 추천 알고리즘에 필요한 데이터를 관리합니다.

5. 메뉴 시스템

프론트엔드:

메뉴 화면 디자인: 사용자가 주문할 수 있는 메뉴를 시각적으로 표시하는 화면을 디자인합니다. 이 화면은 키오스크 장치에서 사용자에게 보여집니다.

주문 폼: 메뉴에서 선택한 항목을 주문할 수 있는 폼을 구현합니다. 이 폼은 주문 버튼과 함께 메뉴 항목, 수량, 옵션 등을 포함해야 합니다.

주문 정보 표시: 주문한 내용을 사용자에게 확인할 수 있는 화면을 구현합니다.

백엔드:

메뉴 정보 관리: 사장님이 입력한 메뉴 정보(메뉴명, 가격, 설명 등)를 저장하고 관리하는 API를 구현합니다.

주문 처리: 사용자가 주문한 내용을 받아와 데이터베이스에 저장하고, 주문 내역을 확인할 수 있는 API를 제공합니다.

사장님 대시보드: 사장님이 메뉴를 관리하고 주문을 확인할 수 있는 대시보드를 만듭니다.

데이터베이스:

메뉴 정보 저장: 메뉴에 관련된 정보를 저장하는 테이블을 생성하고, 사장님이 입력한 메뉴 정보를 저장합니다.

주문 내역 저장: 사용자가 주문한 내용을 저장하는 테이블을 생성하고, 주문된 메뉴, 수량, 시간 등을 저장합니다.

<기술 스택>

프론트엔드: HTML, JavaScript, Fetch API

백엔드: Node.js, Express.js

데이터베이스: MySQL

ORM(Object-Relational Mapping): Sequelize

<데이터 테이블 설명>

menu 테이블은 주문 시스템에서 메뉴 정보를 저장하는 데이터베이스 테이블입니다.

menu_id (메뉴 고유 식별자):

데이터 타입: INTEGER

속성: 자동 증가 및 기본 키로 설정되어 있어 각 메뉴를 고유하게 식별하는 데 사용됩니다.

menu_title (메뉴 제목):

데이터 타입: STRING

속성: 메뉴의 이름 또는 제목을 나타냅니다. 이는 메뉴를 사용자에게 표시하는 데 사용됩니다.

menu_price (메뉴 가격):

데이터 타입: INTEGER

속성: 메뉴의 가격을 나타냅니다. 주문 시에는 각 메뉴의 가격이 계산되어 총 주문 가격에 기여합니다.

menu_explain (메뉴 설명):

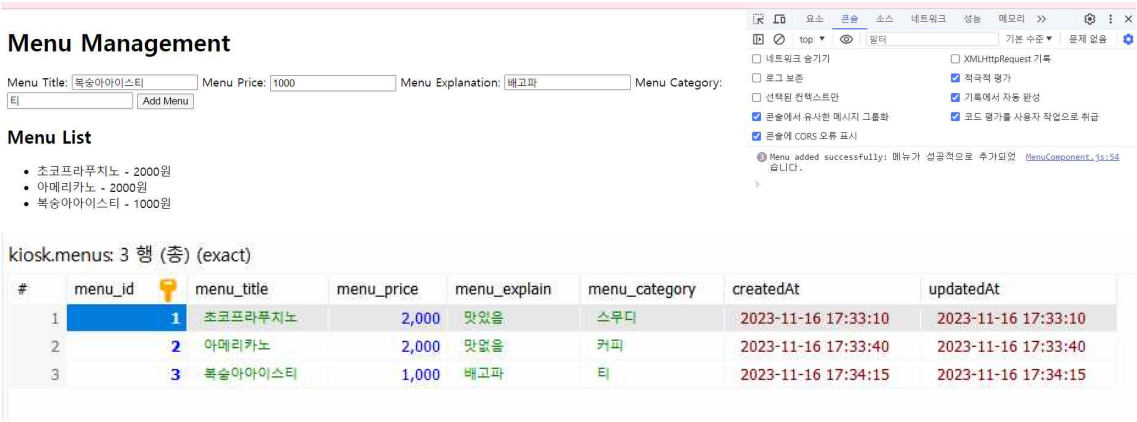
데이터 타입: STRING

속성: 각 메뉴에 대한 설명이나 추가 정보를 나타냅니다. 사용자에게 메뉴에 대한 자세한 내용을 제공하는 데 사용될 수 있습니다.

menu_category (메뉴 카테고리):

데이터 타입: STRING

속성: 메뉴를 그룹화하거나 분류하기 위한 카테고리를 나타냅니다. 이는 주문 시에 특정 유형의 메뉴를 쉽게 찾을 수 있도록 도와줍니다.
createdAt 및 updatedAt (생성 및 업데이트 날짜):
데이터 타입: DATETIME
속성: 각 레코드의 생성 및 마지막 업데이트 날짜 및 시간을 나타냅니다. Sequelize에서 자동으로 관리됩니다.



kiosk.menus: 3 행 (총) (exact)

#	menu_id	menu_title	menu_price	menu_explain	menu_category	createdAt	updatedAt
1	1	초코프라푸치노	2,000	맛있음	스무디	2023-11-16 17:33:10	2023-11-16 17:33:10
2	2	아메리카노	2,000	맛없음	커피	2023-11-16 17:33:40	2023-11-16 17:33:40
3	3	복숭아아이스티	1,000	배고파	티	2023-11-16 17:34:15	2023-11-16 17:34:15

[실행방법]
1. PS C:\project\customer_management\menus> node server.js
2. http://localhost:5000/menu.html

5. 연결 방법:
RESTful API:
고객 주문 정보 및 이력에 대한 CRUD(Create, Read, Update, Delete) 작업을 수행할 RESTful API를 구현합니다.
HTTP 통신:
프론트엔드와 백엔드 간의 통신은 HTTP 프로토콜을 사용하며, 데이터는 주로 JSON 형식으로 교환됩니다.
ORM (옵션):
데이터베이스와의 상호작용을 편리하게 하기 위해 ORM을 사용할 수 있습니다.

<추천기능 알고리즘>
고객이 주문할 때마다 해당 메뉴의 카운트를 증가시켜 매번 오셨을 때 주문 횟수가 많은 음료를 추천하는 기능
주문 이력 저장:
데이터베이스에 주문 이력을 저장하는 테이블을 만듭니다.
이 테이블은 사용자 ID, 메뉴 ID, 주문 횟수, 주문 일자 등을 포함합니다.
주문 시 카운트 증가:
사용자가 주문할 때마다 해당 메뉴의 주문 횟수를 증가시킵니다.
이 정보를 데이터베이스에 업데이트합니다.
주문 이력을 기반으로 추천:
사용자가 새로운 주문을 할 때, 해당 사용자의 이전 주문 이력을 확인합니다.
주문 횟수가 많은 메뉴를 찾아 추천 리스트에 추가합니다.