
「KOKEE TEA 주문 시스템 개발(웹사이트)」 결 과 보 고 서

2023. 8

구로디지털훈련센터
Front-End 팀

목 차

요 약

1. 서론	3
1. 개발 동기 및 목적	3
2. 설계 목표 및 범위	3
3. 프로젝트 추진 일정	4
4. 개발 환경	5
5. 팀 구성원 역할	6
2. 본론	7
1. 메뉴 주기능	7
2. 와이어프레임	8
3. 시스템 전체구조	10
4. 웹사이트 순서도	11
3. 결론	12
1. 프로젝트 결과	12
[참고 문헌]	22
[첨부 1] 프로그램 소스 코드	22

요 약

1. 서 론

• 개발 동기 및 목적

- 음료 제품과 서비스에 대한 정보를 쉽고 빠르게 접근할 수 있는 웹사이트 필요
- 고정비 증가로 인한 어려움을 해소함.
- COVID-19로 인해 비대면 접촉을 선호하는 등 소비 트렌드의 변화도 있음.
- 소비자 또한 간편하고 신속한 결제 수단과 자동화된 서비스를 선호함.
- 데이터 기반 매출 정보 수집 및 관리
- 매장 인력 감축을 통한 인건비 절감
- 근무 인력 집중을 통한 서비스 품질 향상
- 웹을 통해 24시간 비대면 주문을 통한 고객 만족도 증대

• 설계 목표 및 범위

- 새로운 카페 홈페이지와 키오스크 구축을 통해 고객들에게 더욱 쾌적하고 편리한 주문 경험을 제공함으로써 재방문률과 고객 충성도 증가
- 자동화된 주문 시스템 도입으로 인해 주문 정확도 향상 및 주문 처리 시간 단축
- 직원들이 주문 처리에 더 집중할 수 있게 되어 업무 효율성 향상 및 생산성 증대
- 웹사이트와 키오스크를 활용한 디지털 마케팅 기회 확대로 더 많은 고객에게 다가갈 수 있음
- 키오스크와 주문 데이터 분석을 통한 재고 관리 및 원가 절감으로 경영 효율성 향상
- 새로운 기술 도입으로 인한 경쟁력 강화와 브랜드 이미지 향상

• 프로젝트 추진 일정

단계	활동	1주		2주		3주		4주		5주	
		8/1	8/3	8/7	8/10	8/14	8/17	8/20	8/24	8/27	8/30
		~ 8/2	~ 8/6	~ 8/9	~ 8/13	~ 8/16	~ 8/19	~ 8/23	~ 8/26	~ 8/29	~ 8/31
계획	WBS 작성										
	프로젝트 수행계획서										
	요구사항 정의										
분석	요구사항 분석										
	시스템 환경 구축										
설계	화면 UI 설계										
	주문 기능 설계										
	DB 설계										
구현	메뉴_옵션 기능 구현										
	장바구니 기능 구현										
	결제 기능 구현										
	주문번호 출력 기능 구현										
	DB 연동										
시스템 평가	테스트										
	피드백 및 개선										
종료	최종보고서 제출										
	프로젝트 종료										

- **개발 환경**

- (1) 개발 환경

- **고객관리 기능 :**

회원가입 시 회원정보 입력 폼, 가입한 회원을 대상으로 로그인/로그아웃 화면, 마이페이지 화면에서 회원정보 조회/수정 기능을 보여주는 것으로 해당 기능들은 HTML/CSS/JAVASCRIPT와 React.js로 구현함.

- **고객센터 기능 :**

공지사항 및 FAQ의 화면에서 조회/검색 기능, 오시는 길에서 지도 Web API 연동을 통한 코키티 위치 제공, 고객센터 화면에서 문의사항 게시/조회/수정/삭제 기능을 보여주는 것으로 해당 기능들은 HTML/CSS/JAVASCRIPT와 React.js로 구현함.

- **거래관리 기능 :**

상품 주문 기능, 카드 결제/가상 계좌, 장바구니/주문내역 확인 기능을 보여주는 것으로 해당 기능들은 HTML/CSS/JAVASCRIPT와 React.js로 구현함.

- **데이터베이스 :**

Spring, JPA, MySQL로 구축함

• 팀 구성원 역할

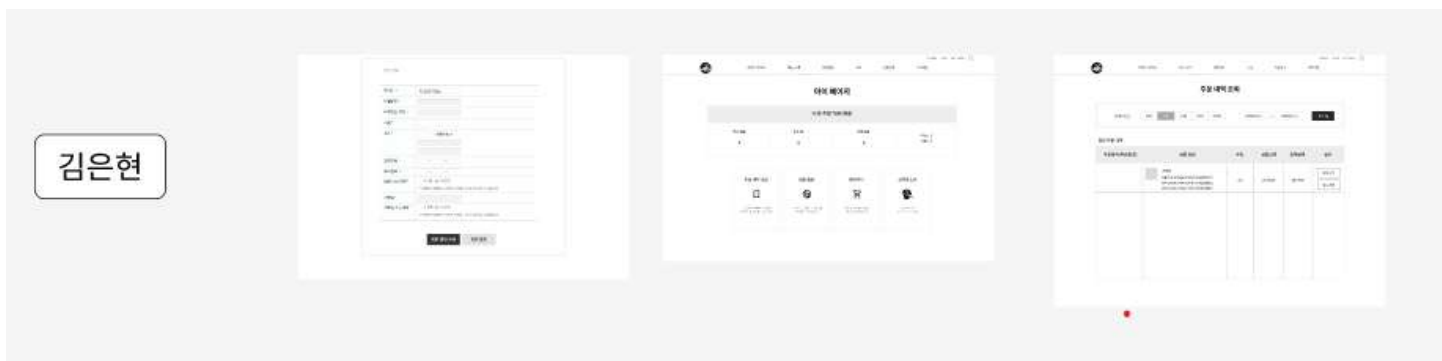
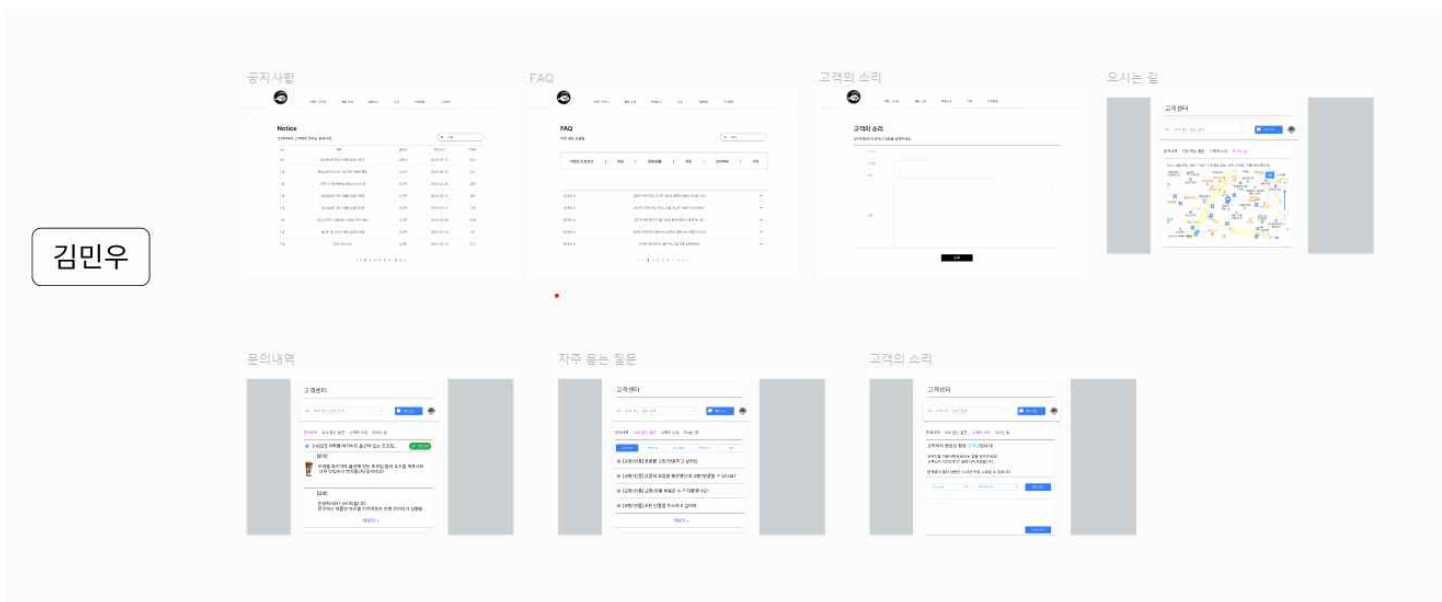
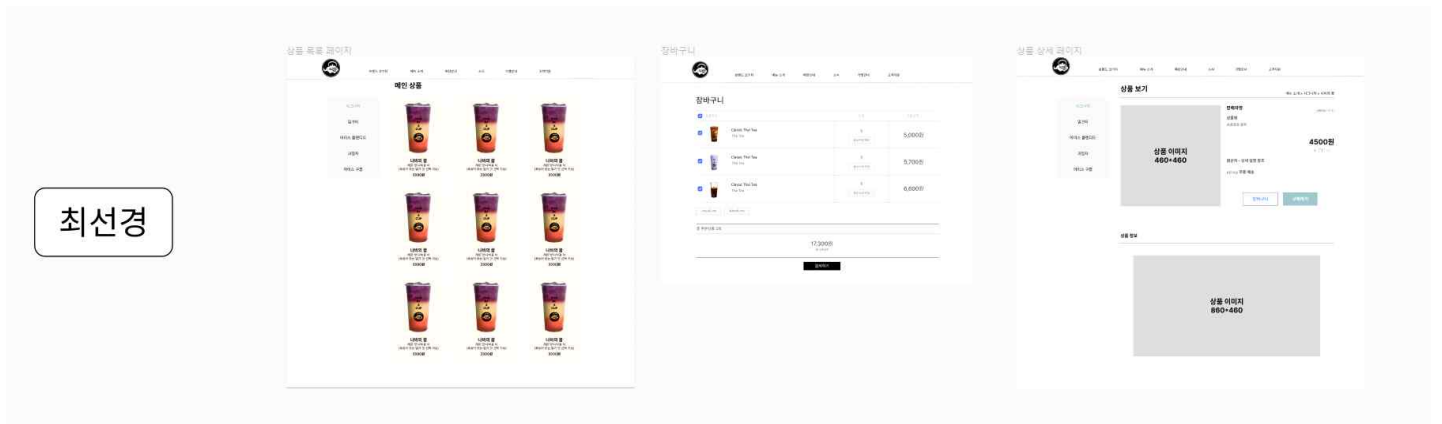
구분	역할 및 책임
최선경 (팀장)	시스템 총괄 및 개발 <ul style="list-style-type: none"> 음료 메뉴 소개 페이지 구현 주문하기/취소하기 기능 구현 장바구니 기능 구현
김도영	데이터베이스 <ul style="list-style-type: none"> 쿼리 등록 문서 기획
김민우	시스템 총괄 및 개발 <ul style="list-style-type: none"> 공지사항 페이지 구현 FAQ 자주하는 질문 페이지 구현 고객의 소리 페이지 구현
김은현	시스템 설계 및 개발 <ul style="list-style-type: none"> 마이페이지 구현 주문조회 페이지 구현 회원정보 수정/탈퇴 기능 구현
노유리	화면 설계 및 개발 <ul style="list-style-type: none"> 메인페이지 구현 챗봇, 유튜브, 카카오지도 api 기능 구현 회원가입 페이지 구현 로그인/로그아웃 기능 구현
조인제	시스템 설계 및 개발 <ul style="list-style-type: none"> 결제하기 페이지 구현 키오스크 구축

2. 본 론

• 메뉴 주기능

구분	내용
메인 화면	<ul style="list-style-type: none"> ○ KOKEE TEA 메인 <ul style="list-style-type: none"> - 상품 배너 안내 - 공지사항 - 홍보 영상 - 코키티 SNS ○ 상품소개 <ul style="list-style-type: none"> - 카테고리 항목 소개 - 상품 썸네일 소개 - 상품 상세 소개 ○ 메뉴 <ul style="list-style-type: none"> - 코키티 스토리(브랜드 소개) - 메뉴소개(밀크티, 시그니처, 과일차, 콜드 클라우드, 아이스 블렌디드) - 매장안내(매장찾기) - 가맹안내(가맹문의) - 고객지원(공지사항, FAQ, 고객의 소리) ○ 챗봇 <ul style="list-style-type: none"> - 24시간 고객 실시간 응답
고객관리	<ul style="list-style-type: none"> ○ 회원가입/탈퇴하기 기능 ○ 로그인/로그아웃 기능 ○ 마이페이지 <ul style="list-style-type: none"> - 주문 내역 조회 / 회원정보 / 장바구니 / 고객의 소리 ○ 회원정보 데이터베이스 저장/관리 등
거래관리	<ul style="list-style-type: none"> ○ 주문관리 <ul style="list-style-type: none"> - 주문하기 - 취소하기 - 장바구니 - 주문내역 확인 ○ 결제하기 <ul style="list-style-type: none"> - 카드 결제 - 가상 계좌
고객지원	<ul style="list-style-type: none"> ○ 공지사항 ○ FAQ ○ 고객의 소리
데이터베이스	<ul style="list-style-type: none"> ○ MySQL과 Spring Boot를 연동하여 데이터 저장

• 와이어프레임



노유리

메인(완료)



회원가입(완료)



로그인(완료)



아이디 찾기



비밀번호 찾기

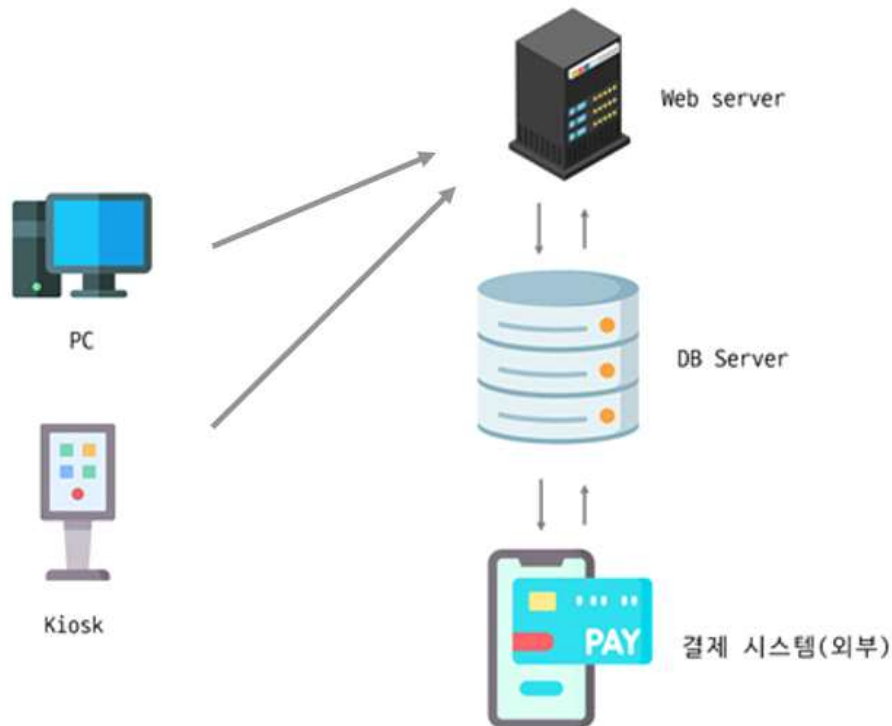


조인제

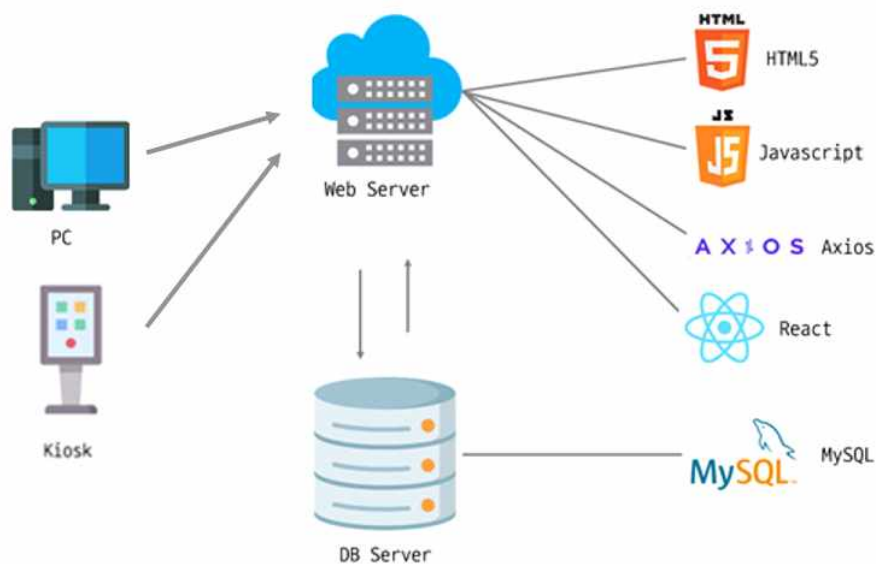


• 시스템 전체 구조

- 1) 주문과 결제 내역을 스프링 연동 DB로 관리되는 시스템으로 구성되어 있음
- 2) 고객은 웹사이트를 통해 음료별 선택 정보를 선택해 주문 기능을 수행함
- 3) 주문한 결과에 따라 결제 시 카드 결제기능을 수행함

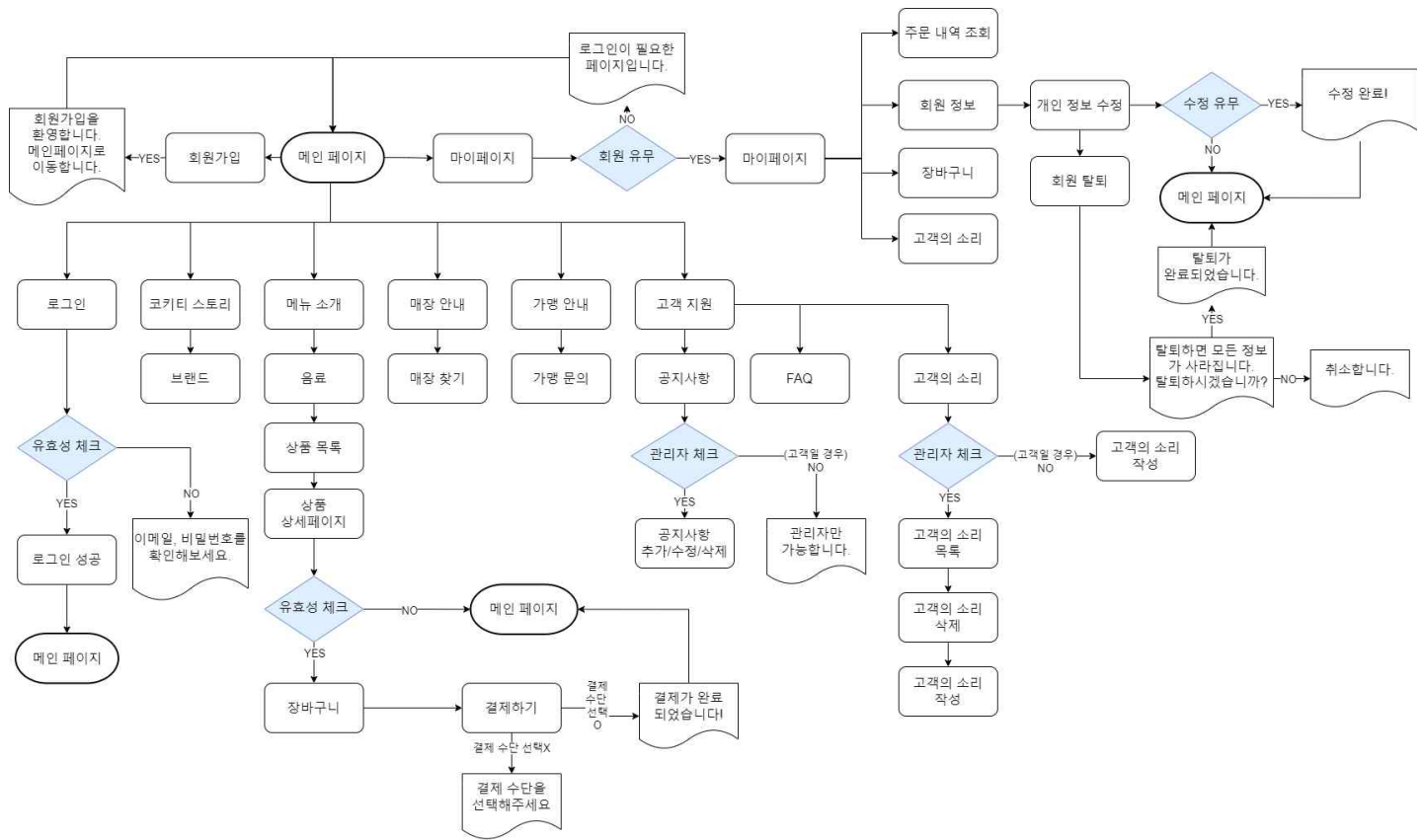


서버에서 ajax를 통해 json 형태로 주문 이력 데이터를 백엔드 api 서버로 요청



Kiosk에서 웹 서버로 신호를 송신하고, 데이터베이스 서버는 웹 서버와 외부 결제 시스템에 데이터를 주고받음.

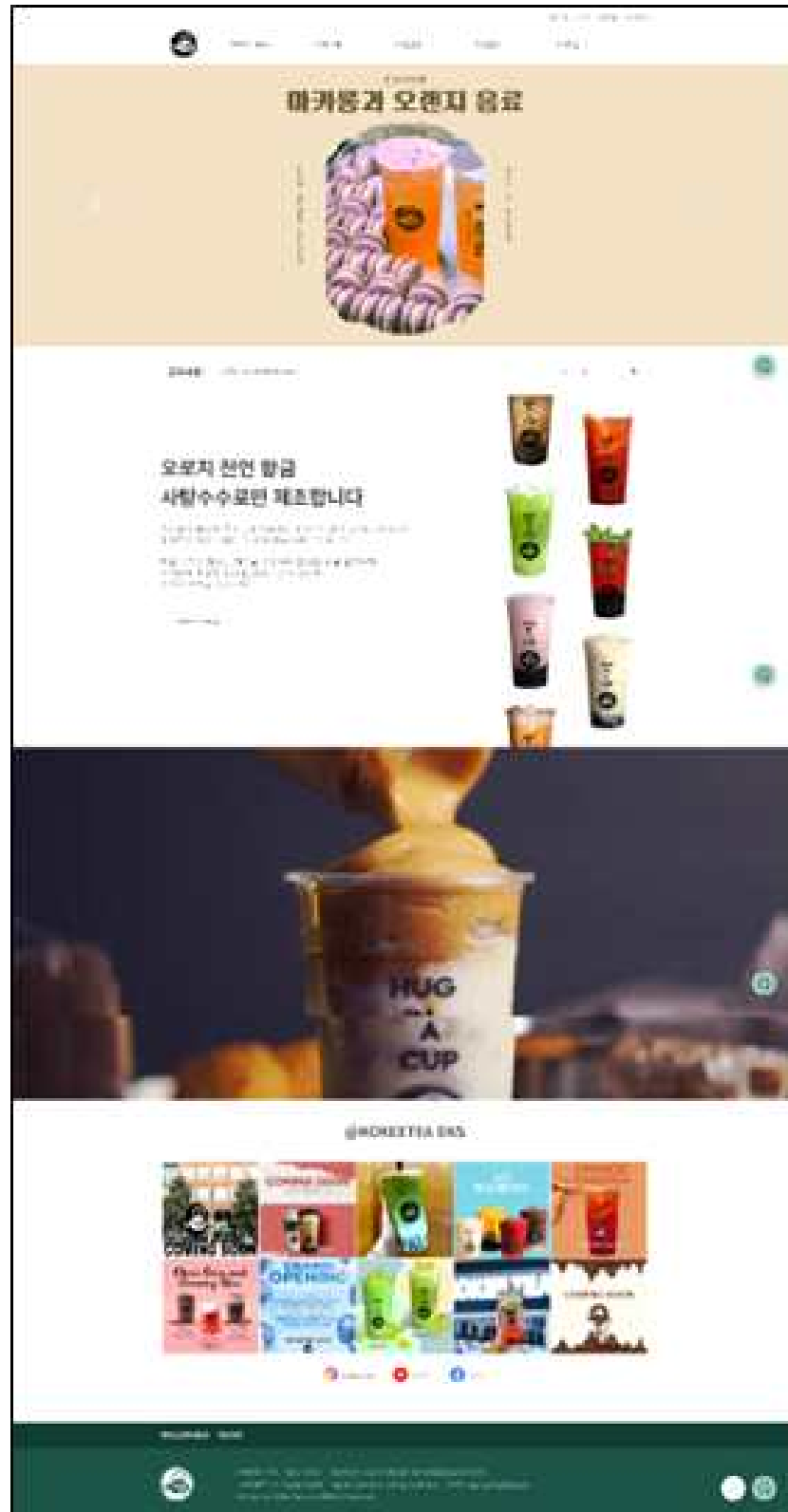
• 웹사이트 순서도



3. 결 론

- 프로젝트 결과

1. 메인 사이트



2. 메뉴 소개

로그인회원가입마이페이지

코키티 스토리

메뉴소개

매장안내

가맹안내

고객지원

시그니처

콜드 클라우드

아이스 블렌디드

KOKEE 과일 차

밀크티

시그니처

블랙 리치

5,000원

드래곤과 장미

5,000원

나비의 꿈

5,000원

조지아 온 마이 하인드

5,000원

망고 페션후르츠

5,000원

피플 러브

5,000원

샌프란시스코의 장미

5,000원

개인정보처리방침

이용약관

(주)KOKEE TEA 대표 : 김정준 서울특별시 구로구 디지털로 300 자벌리비즈플라자 11층
사업자등록번호 123-45-678901 TEL 02-1234-5678 / FAX 02-1234-5678 이메일 nugo123@cale100.com
© kokee tea Coffee Company. All Rights Reserved.

2-1. 음료 종류

시그니처

콜드 클라우드

아이스 블렌디드

KOKEE 과일 차

밀크티

콜드 클라우드

브라운슈가 콜드브루

5,000원

브라운슈가 달고나라떼

5,000원

오레오 브라운슈가

5,000원

숭사탕 콜드브루

5,000원

숭사탕 그린티

5,000원

2-3. 음료 선택

시그니처 쿨드 클라우드 아이스 블렌디드 KOKEE 과일 차 밀크티



브라운슈가 달고나라떼

5,000원

- 1 +

총 상품 금액

총 수량 1개 5,000원

장바구니

2-4. 장바구니 목록

장바구니

상품정보	담은시간	수량	주문금액	삭제
 <div> <div>드래곤과 장미</div> <div>size - Regular</div> </div>	2023-08-31 15:51:09	<div>1</div> <div>수정</div>	5,000원	×

장바구니 비우기

총 주문상품 1개

5,000원

총 주문금액

결제하기

2-5. 음료 결제하기

주문 상품 정보



드래곤과 장미

5,000원

2023-08-31 15:51:09

주문 요약

총 주문금액

5,000원

결제 수단

☐ 카드결제


☐ 계좌이체

결제하기

주문자 이메일

admin@admin.com

3. 매장 안내




코키티 스토리 메뉴소개 매장안내 가맹안내 고객센터


관리자님 로그인 회원가입 마이페이지

매장 찾기


가까운 코키티 매장을 찾아보세요.
매장 위치와 주소를 확인하실 수 있습니다.




개인정보처리방침 이용약관



(주)KOEKKEE TEA 대표 : 김정준 서울특별시 구로구 디지털로 300 자별리비즈플라자 11층
사업자등록번호 123-45-678901 TEL 02) 1234-5678 / FAX 02) 1234-5678 이메일 nugo123@kale100.com
© kokee tea Coffee Company. All Rights Reserved.



4. 가맹 문의



코키티 스토리 메뉴소개 매장안내 가맹안내 고객센터

관리자님 로그인 회원가입 마이페이지

가맹 문의

코키티 가맹 문의는 '1:1 맞춤 상담'으로 진행됩니다.
가맹 상담부터 오픈, 사후 관리까지 토털 서비스를 지원합니다.

이름

이름을 입력해주세요.

연락처

연락처를 입력해주세요.

희망 지역


희망 지역을 입력해주세요.

문의 내용


문의 내용을 입력해주세요.

문의하기


개인정보처리방침 이용약관



(주)KOEKKEE TEA 대표 : 김정준 서울특별시 구로구 디지털로 300 자별리비즈플라자 11층
사업자등록번호 123-45-678901 TEL 02) 1234-5678 / FAX 02) 1234-5678 이메일 nugo123@kale100.com
© kokee tea Coffee Company. All Rights Reserved.



5. 공지사항



코키티 스토리 메뉴소개 매장안내 가맹안내 고객센터

공지사항


코키티에서 고객에게 전하는

no	title	writer	date
7	4월 베스트 리뷰 이벤트 당첨자 발표	관리자	2023-05-10 09:00
<p>👉 축하드립니다! 👉</p> <p>4월 베스트 리뷰 이벤트의 당첨자는 'sweetcookie123'님의 '코키티의 신제품 "클라우드"에 관한 고찰' 리뷰입니다. 🍵👏</p> <div>수령 삭제</div> <p>📦 상품은 다음 주 월요일에 발송됩니다. 📦</p>			
6	팝업스토어 인스타그램 이벤트 발표	관리자	2023-05-03 09:00
5	코키티 X 현대백화점 팝업스토어 안내	관리자	2023-04-25 09:00
4	3월 베스트 리뷰 이벤트 당첨자 발표	관리자	2023-04-10 09:00
3	2월 베스트 리뷰 이벤트 당첨자 발표	관리자	2023-03-10 09:00
2	코키티 신메뉴의 이름을 지어주세요	관리자	2023-03-03 09:00
1	1월 베스트 리뷰 이벤트 당첨자 발표	관리자	2023-02-10 09:00


Write

< 1 >


개인정보처리방침 이용약관



(주)KOCKEE TEA 대표 : 김정준 서울특별시 구로구 디지털로 300 자벌리비즈플라자 11층
사업자등록번호 123-45-678901 TEL 02) 1234-5678 / FAX 02) 1234-5678 이메일 nugo123@cafe100.com
© kokee tea Coffee Company. All Rights Reserved.



5-1. FAQ



코키티 스토리 메뉴소개 매장안내 가맹안내 고객센터


이벤트/프로모션 | 회원 | 결제/환불 | 매장 | 배송/리베리 | 기타

Topic


Question

이벤트/프로모션 [이벤트/프로모션] 베스트 리뷰 이벤트 당첨자는 매월 언제쯤 발표하나요?

개인정보처리방침 이용약관




(주)KOCKEE TEA 대표 : 김정준 서울특별시 구로구 디지털로 300 자벌리비즈플라자 11층
사업자등록번호 123-45-678901 TEL 02) 1234-5678 / FAX 02) 1234-5678 이메일 nugo123@cafe100.com
© kokee tea Coffee Company. All Rights Reserved.



- 16 -

5-2. 고객의 소리



[코키티 스토리](#)
[매뉴소개](#)
[매장안내](#)
[가맹안내](#)
[고객지원](#)

[관리자님 로그인](#)
[회원가입](#)
[마이페이지](#)

고객의 소리

코키티에게 의견이나 칭찬을 남겨주세요.
(작성해주신 내역 및 답변은 이메일로 발송될 수 있습니다.)

제목

제목을 입력하세요.


내용

내용을 입력하세요.


등록

목록


[개인정보처리방침](#)
[이용약관](#)



(주)KOCKEE TEA 대표 : 김정준 서울특별시 구로구 디지털로 300 차별리비즈플라자 11층
 사업자등록번호 123-45-67890 TEL 02-1234-5678 / FAX 02-1234-5678 이메일 nugo123@cafe100.com
 © kokee tea Coffee Company. All Rights Reserved.




7. 마이페이지



코키티 스토리 메뉴소개 매장안내 가맹안내 고객센터


노유리님 로그인 회원가입 마이페이지

마이 페이지




주문 내역 조회

주문하신 상품의 주문내역을 확인 할 수 있습니다.




회원 정보

고객님의 개인정보를 관리하는 공간입니다.



장바구니


장바구니에 담은 상품의 목록을 보며드립니다.




고객의 소리

FAQ 페이지로 바로 이동할 수 있습니다.


개인정보처리방침 이용약관



(주)KOKEE TEA 대표 : 김정준 서울특별시 구로구 디지털로 300 자벨리비즈플라자 11층
사업자등록번호 123-45-678901 TEL 02) 1234-5678 / FAX 02) 1234-5678 이메일 nugo123@cafe100.com
© kokee tea Coffee Company. All Rights Reserved.



7-1. 주문내역 조회



코키티 스토리 메뉴소개 매장안내 가맹안내 고객센터

노유리님 로그인 회원가입 마이페이지

주문 내역 조회


조회 기간

오늘 7일 15일 1개월 3개월


2023-08-31 ~ 2023-08-31

조회


최근 주문 내역

지점명	주문일자	주문번호	상품정보	가격	수량
구로점	2023-08-31 16:31:02	1693467062	 드레곤과 장미	5000	1

개인정보처리방침 이용약관




(주)KOKEE TEA 대표 : 김정준 서울특별시 구로구 디지털로 300 자벨리비즈플라자 11층
사업자등록번호 123-45-678901 TEL 02) 1234-5678 / FAX 02) 1234-5678 이메일 nugo123@cafe100.com
© kokee tea Coffee Company. All Rights Reserved.



- 19 -

7-2. 개인정보 수정



[코키티 스토리](#)
[메뉴소개](#)
[매장안내](#)
[가맹안내](#)
[고객지원](#)

[노유리님 로그인](#)
[회원가입](#)
[마이페이지](#)

개인 정보 수정

기본 정보

아이디

nyr1001 [변경불가]

이메일

nyr1001@naver.com [변경불가]

현재 비밀번호*

[확인]

새 비밀번호*

(영문 대소문자/숫자/특수문자 중 2가지 이상 조합, 10~16자)

새 비밀번호 확인*

(영문 대소문자/숫자/특수문자 중 2가지 이상 조합, 10~16자)

이름*

노유리

휴대전화*

010-3721-6027

SNS 수신 여부*

☒ 수신함
 ☐ 수신안함

카톡에서 제공하는 이벤트 소식을 SMS로 받을 수 있습니다.

이메일 수신 여부*

☒ 수신함
 ☐ 수신안함

카톡에서 제공하는 이벤트 소식을 이메일로 받을 수 있습니다.


수정

취소


회원 탈퇴

개인정보처리방침


이용약관



(주)KOEKEE TEA 대표 : 김정준 서울특별시 구로구 디지털로 300 자벨라비즈플라자 11층
 사업자등록번호 123-45-678901 TEL 02) 1234-5678 / FAX 02) 1234-5678 이메일 nugo123@kale100.com
 © kckee tea Coffee Company. All Rights Reserved.



8. 로그인



[코키티 스토리](#)
[메뉴소개](#)
[매장안내](#)
[가맹안내](#)
[고객지원](#)

[로그인](#)
[회원가입](#)
[마이페이지](#)

로그인

이메일

비밀번호

☐ 로그인 상태 유지

로그인

회원가입

아이디 · 비밀번호 찾기

또는




비회원 주문 조회


SIGNATURE MILK TEA

코키티 시그니처 밀크티

공지사항

등록된 공지사항이 없습니다.



9. 키오스크



[참고 문헌]

[첨부 1] 프로그램 소스 코드

• 웹사이트 화면 소스 코드

노유리

// 회원가입 폼 구현

```
const Form = () => {
  const [userId, setUserId] = useState("");
  const [userPw, setUserPw] = useState("");
  const [userPwCheck, setUserPwCheck] = useState("");
  const [userName, setUserName] = useState("");
  const [phone02, setPhone02] = useState("");
  const [phone03, setPhone03] = useState("");
  const [email01, setEmail01] = useState("");
  const [email02, setEmail02] = useState("");
  const [locale, setLocale] = useState("");
  const [isAllChecked, setIsAllChecked] = useState(false);
  const [agree1, setAgree1] = useState(false);
  const [agree2, setAgree2] = useState(false);
  const [agree3, setAgree3] = useState(false);
  const [agree4, setAgree4] = useState(false);
  const [validId, setValidId] = useState(false);
  const [validPw, setValidPw] = useState(false);
  const navigate = useNavigate();

  const onChangeUserId = (e) => {
    const userIdRegex = /^[a-z]+[a-z0-9]{4,12}$/g;
    if (!!e.target.value || (userIdRegex.test(e.target.value))) {
      setValidId(false);
    } else {
      setValidId(true);
    }
    setUserId(e.target.value);
  };
};
```

```
const onChangeUserPw = (e) => {  
  const userPwRegex = /^(?=.*Wd)(?=.*[a-zA-Z])[0-9a-zA-Z]{8,16}$/;  
  if (!!e.target.value || (userPwRegex.test(e.target.value))) {  
    setValidPw(false);  
  } else {  
    setValidPw(true);  
  }  
  setUserPw(e.target.value);  
};
```

```
const onJoin = (event) => {  
  event.preventDefault();  
  if (  
    userId === "" ||  
    userPw === "" ||  
    userPwCheck === "" ||  
    userName === "" ||  
    phone02 === "" ||  
    phone03 === "" ||  
    email01 === "" ||  
    email02 === "" ||  
    locale === ""  
  ) {  
    alert("회원가입은 빈칸이 없게 모두 입력해주세요!!");  
  } else if (  
    isAllChecked === false ||  
    agree1 === false ||  
    agree2 === false ||  
    agree3 === false ||  
    agree4 === false  
  ) {  
    alert("비어있는 동의란을 확인해주세요!!");  
  } else if (userPw !== userPwCheck) {  
    alert("비밀번호가 일치하지 않아요.");  
  } else {
```

```

    axios
      .post("http://localhost:8080/kokee/join", {
        locale: locale,
        userId: userId,
        userPw: userPw,
        userPwCheck: userPwCheck,
        userName: userName,
        phone02: phone02,
        phone03: phone03,
        email01: email01,
        email02: email02,
      })
      .then((res) => {
        console.log(res);
        if (res.data === "success") {
          alert("회원가입을 환영합니다. 메인페이지로
이동합니다.")
          navigate("/");
        }
      })
      .catch((err) => {
        if (err.response.data === "failed") {
          alert(`입력하신 아이디와 이메일은 이미 가입된 회원
입니다.\n다른 내용으로 가입해주세요.`)
        } else {
          alert("알수 없는 에러가 발생했습니다. 관리자에게
문의하세요.")
        }
      });
  }
};

function allSelect() {
  if (isAllChecked) {
    setAgree1(false);
    setAgree2(false);
  }
}

```



```

        setAgree3(false);
        setAgree4(false);
    } else {
        setAgree1(true);
        setAgree2(true);
        setAgree3(true);
        setAgree4(true);
    }
}

```

// 로그인 기능 구현

```

const Header = ({ isLoggedIn, setIsLoggedIn }) => {
    const navigate = useNavigate();
    const userEmail = document.querySelector("#user_email");
    const userPass = document.querySelector("#user_pass");
    const [email, setEmail] = useState(""); //이메일 입력란 처음공백
    const [pass, setPass] = useState(""); //이메일 입력란 처음공백
    const [currentItem, setCurrentItem] = useState(null);
    const [headerLoggedIn, setHeaderLoggedIn] = useState(false);

    useEffect(() => {
        if (localStorage.getItem("realname")) {
            setHeaderLoggedIn(true);
        }
    }, [])

    async function fn_login() {
        const regExp = /[ㄱ-ㅎㅏ-ㅣ가-힣]/g;
        const exptext = /^[A-Za-z0-9_\.W-]+@[A-Za-z0-9W-]+\.W[A-Za-z0-9W-]+/;

        if (email == "") {
            alert("이메일이 공백입니다.");
            userEmail.focus();
            return;
        }
        if (pass == "") {

```

```
    alert("비밀번호가 공백입니다.");
    userPass.focus();
    return;
}

if (regExp.test(email)) {
    alert("이메일에 한글을 입력하실 수 없습니다.");
    return;
}

if (exptext.test(email)) {
} else {
    alert("이메일 형식을 맞춰주세요. Ex) exaple@example.com");
    return;
}

try {
    const result = await axios.post("http://localhost:8080/kokee/login", {
        email: email,
        password: pass
    })
    console.log(result);
    if (result.status === 200) {
        setIsLoggedIn(!isLoggedIn);
        setHeaderLoggedIn(!headerLoggedIn);
        document.querySelector(".sec_modal").classList.remove("active");
        localStorage.setItem("email", email);
        localStorage.setItem("realname", result.data.name);
        localStorage.setItem("locale", result.data.locale);
        alert(`${localStorage.getItem("realname")}님 환영합니다!`)
        navigate("/");
    }
} catch (error) {
    alert("이메일과 비밀번호를 확인해보세요")
}
}
```

```
function logoutFunction() {
  alert(`로그아웃 합니다. ${localStorage.getItem("realname")}님 안녕히 가세요.`);
  localStorage.clear();
  setIsLoggedIn(!isLoggedIn);
  navigate("/");
  window.location.reload();
}
```

```
function fn_open() {
  document.querySelector(".sec_modal").classList.add("active");
}
```

```
function fn_close() {
  document.querySelector(".sec_modal").classList.remove("active");
}
```

김은현

마이페이지 소스 코드

**** 로그인 안 되어 있을 경우 메인페이지로 가는 코드 ****

```
useEffect(() => {
  if (isLoggedIn) {
    // pass를 의미함
  } else {
    alert("로그인이 필요한 메뉴 입니다.");
    navigate("/");
  }
}, [isLoggedIn, navigate]);
```

**** html 구현 코드 ****

```
function Mypage({ isLoggedIn }) {
  console.log(isLoggedIn);
  const navigate = useNavigate();
```

```

useEffect(() => {
  if (isLoggedIn) {
    // pass를 의미함
  } else {
    alert("로그인이 필요한 메뉴 입니다.");
    navigate("/");
  }
}, [isLoggedIn, navigate]);

return (
  <>
    <hr />
    <div className="inner">
      {/* 본문 시작 */}
      <h2 className="mypage">마이 페이지</h2>
      <div className="shortCut">
        <div className="sc" onClick={() => navigate("../orderList")}>
          <div className="customer">
            <p>주문 내역 조회</p>
          </div>
          <div className="icons">
            <IoDocumentTextOutline size={72} />
          </div>
          <div className="context">
            <span>
              주문하신 상품의 주문내역을 <br />
              확인 할 수 있습니다.
            </span>
          </div>
        </div>
      </div>
      <div className="sc" onClick={() => navigate("../Editprofile")}>
        <div className="customer">
          <p>회원 정보</p>
        </div>
        <div className="icons">
          <IoProfile size={72} />
        </div>
      </div>
    </div>
  </>
)

```

```

    </div>
    <div className="context">
        <span>고객님의 개인정보를 관리하는 공간입니다.</span>
    </div>
</div>
<div className="sc" onClick={() => navigate("../basket")}>
    <div className="customer">
        <p>장바구니</p>
    </div>
    <div className="icons">
        <CgShoppingCart size={72} />
    </div>
    <div className="context">
        <span>장바구니에 담은 상품의 목록을 보여드립니다.</span>
    </div>
</div>
<div className="sc" onClick={() => navigate("../faq")}>
    <div className="customer">
        <p>고객의 소리</p>
    </div>
    <div className="icons">
        <RiCustomerService2Line size={72} />
    </div>
    <div className="context">
        <span>FAQ 페이지로 바로 이동할 수 있습니다.</span>
    </div>
</div>
</div>
<div className="test" />
</>
);
}

```

****개인정보 수정 페이지 소스 코드****

**** 개인정보 데이터 가져오는 코드 ****

```

useEffect(() => {
  async function getMember() {
    try {
      const result = await axios.get(
        `http://localhost:8080/kokee/get_member/${localStorage.getItem(
          "email"
        )}`
      );
      console.log(result);
      setUserId(result.data.username);
      setUserEmail(result.data.email);
      setNowPassword(result.data.password);
      setRealName(result.data.realName);
      setUserPhone(result.data.phoneNumber);
      setIsChecked(true);
    } catch (error) {
      alert("네트워크 오류가 있어요.");
    }
  }
  getMember();
}, []);

```

**** 개인정보 수정하는 코드 ****

```

async function clickUpd() {
  if (newPassword1 !== newPassword2) {
    alert("현재 비밀번호와 새로운 비밀번호가 일치하지 않습니다.");
  } else if (newPassword1 === "" || newPassword2 === "") {
    alert("비밀번호란은 비어 있으면 안됩니다.");
  } else {
    try {
      await axios.put("http://localhost:8080/kokee/update_member", {
        password: newPassword1,
        realname: realName,
        phone: userPhone,
        email: userEmail,

```

```

    });
    alert("수정 완료!");
    navigate("/");
  } catch (error) {
    alert("네트워크 오류가 있어요.");
  }
}
}
}

```

**** 회원 탈퇴하는 코드 ****

```

async function clickDel() {
  if (
    window.confirm(
      "탈퇴하시면, 모든 정보(회원정보, 장바구니, 주문내역 등)가 사라집니다. 탈퇴하시겠습니까?"
    )
  ) {
    await axios.delete(
      `http://localhost:8080/kokee/delete_member/${localStorage.getItem(
        "email"
      )}`
    );
    localStorage.clear();
    alert("탈퇴가 완료되었습니다.");
    navigate("/");
  } else {
    alert("취소합니다.");
  }
}
}

```

** 주문 내역 페이지 **

**** 날짜 형식 표시하는 코드 ****

```

const getFormattedDate = (date) => {
  const year = date.getFullYear();
  const month = String(date.getMonth() + 1).padStart(2, "0");

```

```
const day = String(date.getDate()).padStart(2, "0");
return `${year}-${month}-${day}`;
};
```

```
const getFormattedDateTime = (date) => {
  const year = date.getFullYear();
  const month = String(date.getMonth() + 1).padStart(2, "0");
  const day = String(date.getDate()).padStart(2, "0");
  const hours = String(date.getHours()).padStart(2, "0");
  const minutes = String(date.getMinutes()).padStart(2, "0");
  const seconds = String(date.getSeconds()).padStart(2, "0");
  return `${year}-${month}-${day} ${hours}:${minutes}:${seconds}`;
};
```

**** 주문 내역 조회하는 코드 ****

```
async function findOrders() {
  if (!startDate || !endDate) {
    alert("조회할 날짜를 선택하세요!");
  } else {
    const teststart = getFormattedDateTime(startDate);
    const testend = getFormattedDateTime(endDate);
    try {
      const result = await axios.get(
        `http://localhost:8080/kokee/orders/${localStorage.getItem("email")}?&start=${teststart}&end=${testend}`
      );
      let tempObject = {};
      let tempArray = [];
      JsonData.map((item) => {
        result.data.map((item2) => {
          if (item.name === item2.productName) {
            tempObject = {
              ...item2,
              image: item.image,
            };
          }
        });
      });
    } catch (error) {
      console.log(error);
    }
  }
}
```



```

        tempArray.push(tempObject);
    }
    });
    setResultData(tempArray);
    console.log(tempArray);
} catch (error) {
    alert("네트워크 에러가 발생했어요");
}
}
}

```

**** 조회한 데이터를 화면으로 구현하는 코드 ****

```

<div className="inner recentOrder">
  <h6 className="recent-orders">최근 주문 내역</h6>
  <table className="productTable">
    <thead className="productHeader">
      <tr>
        <th>지점명</th>
        <th>주문일자</th>
        <th>주문번호</th>
        <th>상품정보</th>
        <th>가격</th>
        <th>수량</th>
      </tr>
    </thead>
    <tbody className="productBody">
      {resultData &&
        resultData.map((order) => (
          <tr key={order.id}>
            <td>
              {localStorage.getItem("locale") === "guro"
                ? "구로점"
                : "영등포점"}
            </td>
            <td>{moment(order.date).format("YYYY-MM-DD HH:mm:ss")}</td>

```

```

        <td>{moment(order.date).format("X")}</td>
        <td>
            <div className="productInfo">
                <img
                    className="productList"
                    src={order.image}
                    alt="..."
                    style={{
                        width: "60px",
                        height: "90px",
                        border: "1px solid #DEE2E6",
                        padding: "5px",
                    }} // 이미지 크기 조정
                />
                <div className="productDetail">
                    <br />
                    <h3>{order.productName}</h3>
                </div>
            </div>
        </td>
        <td>{order.price}</td>
        <td>{order.mount}</td>
    </tr>
    )))
</tbody>
</table>
{resultData.length === 0 && (
    <div>
        <span>주문 내역이 없습니다.</span>
    </div>
)}
</div>

```

최선경

**** 음료 데이터 코드 ****

```
function Drink({ product }) {
  const navigate = useNavigate();
  function onClick() {
    navigate("/Detailpage?data=" + JSON.stringify(product));
  }

  return (
    <div className="product" onClick={onClick}>
      <div className="product_image">
        <img src={product.image} alt={product.name} />
      </div>

      <div className="product_name">
        <span>{product.name}</span>
      </div>

      <div className="product_price">
        <span className="price">
          {product.price.toString().replace(/WB(?=(\Wd{3})+(?!Wd))/g, ",")}
        </span>
        <span className="unit">원</span>
      </div>
    </div>
  );
}
```

export default Drink;

**** 상품 목록 페이지 코드 ****

```
function List(props) {
```

```

const [products, setProducts] = useState([]);
const navigate = useNavigate();
const [selectedCategory, setSelectedCategory] = useState(null);
const location = useLocation();

const queryParams = new URLSearchParams(location.search);
const dataParam = queryParams.get("category");

let data = null;

```

**** 상품 상세 페이지 코드 ****

```

function DetailPage({ isLogined }) {
  const navigate = useNavigate();
  const [carts, setCarts] = useState([]);
  const [count, setCount] = useState(1);
  const [basket, setBasket] = useState([]);
  const location = useLocation();
  const queryParams = new URLSearchParams(location.search);
  const dataParam = queryParams.get("data");
  const [data, setData] = useState([]);
  const [loaded, setLoaded] = useState(false);

```

// 상품 데이터를 불러오는 코드

```

useEffect(() => {
  if (dataParam) {
    try {
      setData(JSON.parse(dataParam));
      setLoaded(true);
    } catch (error) {
      console.error("Error parsing data from URL:", error);
    }
  }
}, []);

```

//상품 수량 조절

```
function changeCount(n) {
  if (count + n >= 1) {
    setCount(count + n);
  }
}
```

//장바구니 페이지로 이동하는 코드, 로그인이 되어있지 않으면 네트워크 에러 발생

```
async function pay() {
  if (isLoggedIn) {
    const cart = {
      product_name: data.name,
      mount: count,
      price: data.price * count,
      email: localStorage.getItem("email"),
    };
    try {
      await axios.post("http://localhost:8080/kokee/carts", cart);
      navigate("/basket");
    } catch (error) {
      alert("네트워크 에러 발생");
    }
  } else {
    alert("로그인이 필요한 메뉴 입니다.");
    navigate("/");
  }
}
```

```
const handleCategoryClick = (category) => {
  navigate(-1);
};
```

```
return (
  <div className="inner">
    <div className="detail-page">
      <Sidebar onCategoryClick={handleCategoryClick} />
      <div className="product-detail">
```

```

<div className="detail-img">
  <div className="d-img">
    <img src={data.image} alt="product" />
  </div>
</div>

<div className="detail-info">
  <div className="detail-drink">
    <p className="d-name">{data.name}</p>
    <span className="d-price">
      {loaded &&
        data.price.toString().replace(/WB(?=(\Wd{3})+(?!Wd))/g, ", ")}
    <span className="d-price-fix">원</span>
  </span>
</div>

<div className="d-line" />

<div className="d-amount">
   changeCount(-1)}
  />

  <div className="detail_count">
    <span>{count}</span>
  </div>
   changeCount(1)}
  />
</div>

```

```

    <div className="d-line" />

    <div className="d-sum">
      <div>
        <span className="d-sum-price">총 상품 금액</span>
      </div>
    </div>

    <div className="total-info">
      <span className="total">
        총 수량 <span className="t-count">{count}개 </span>
      </span>
      <span className="t-price">
        {loaded &&
          (data.price * count)
            .toString()
            .replace(/WB(?=(\Wd{3})+(?!Wd))/g, ", ")}
        <span className="t-price-fix">원</span>
      </span>
    </div>

    <div className="d-btn">
      <button className="d-btn-btn" onClick={pay}>
        장바구니
      </button>
    </div>
  </div>
</div>
<div className="testBox" />
</div>
);
}
export default DetailPage;

```

**** 장바구니 코드 ****

```
function Basket({ isLogined }) {
  const navigate = useNavigate();
  const [basket, setBasket] = useState([]);
  const [totalPrice, setTotalPrice] = useState(0);

  // 장바구니에 아이템 추가
  useEffect(() => {
    axios
      .get(`http://localhost:8080/kokee/carts/${localStorage.getItem("email")}`)
      .then((response) => {
        let tempObject = {};
        let tempArray = [];
        jsonData.map((item) => {
          response.data.map((item2) => {
            if (item.name === item2.productName) {
              tempObject = {
                ...item2,
                image: item.image,
              };
              tempArray.push(tempObject);
            }
          });
        });
        setBasket(tempArray);
        let temp = 0;
        tempArray.map((item) => {
          temp += item.price;
        });
        setTotalPrice(temp);
      })
      .catch((error) => {
        console.error("Error fetching carts:", error);
      });
  });
}
```



```
}, [totalPrice, basket]);
```

// 장바구니에 담은 음료의 수량 조절

```
const editcount = async (name, mount, price, id) => {
  let newcount = prompt("수량을 입력하세요");
  let perprice,
      newprice = 0;
  if (newcount) {
    if (newcount >= 1) {
      newcount = parseInt(newcount);
      perprice = price / mount;
      newprice = perprice * newcount;
      const result = await axios.put(
        `http://localhost:8080/kokee/carts/update/${localStorage.getItem(
          "email"
        )}`,
        {
          id: id,
          productName: name,
          updateMount: newcount,
          updatePrice: newprice,
        }
      );
      console.log(result);
    } else {
      alert("수량은 1개 이상이어야 합니다.");
    }
  } else {
    alert("수량을 입력해주세요");
  }
};
```

// 결제하기로 이동

```
const topurchase = () => {
  navigate("/buy", {
    state: { basket },
  });
};
```

```

    });
};

// 장바구니 비우기
const clearbasket = async () => {
  try {
    await axios.delete(
      `http://localhost:8080/kokee/carts/delete/${localStorage.getItem(
        "email"
      )}`
    );
    setBasket([]);
  } catch (error) {
    alert("네트워크 에러입니다.");
  }
};

```

```

// 장바구니 삭제
async function deleteProduct(id) {
  try {
    console.log("삭제삭제");
    await axios.delete(`http://localhost:8080/kokee/carts/delete_one/${id}`);
  } catch (error) {
    alert("네트워크 에러입니다.");
  }
}

```

```

return (
  <>
  <div className="inner">
    <div>
      <div className="title-cart">
        <h2>장바구니</h2>
      </div>
      <div className="cart-container">
        <table>

```

```

<thead>
  <tr className="">
    <th className="table_basket goleft" colSpan={2}>
      <span>상품정보</span>
    </th>
    <th className="table_basket">담은시간</th>
    <th className="table_basket">수량</th>
    <th className="table_basket">주문금액</th>
    <th className="table_basket table_basket_right">삭제</th>
  </tr>
</thead>
<tbody>
  {basket &&
    basket.map((item, index) => (
      <tr className="basket_row" key={index}>
        <td className="table_basket goleft table_smol ">
          <img
            src={item.image}
            height={90}
            style={{
              marginLeft: "50px",
            }}
            alt="이미지"
          />
        </td>
        <td className="table_basket goleft">
          <div>
            <b>{item.productName}</b>
            <p>size - Regular</p>
          </div>
        </td>
        <td className="table_basket goleft">
          <div>
            <b>
              {moment(item.date).format("YYYY-MM-DD HH:mm:ss")}
            </b>
          </div>
        </td>
      </tr>
    ))
  }

```

```

        </div>
    </td>
    <td className="table_basket">
        <p>{item.mount}</p>
        <div>
            <button
                className="btn-btn-control"
                onClick={() =>
                    editcount(
                        item.productName,
                        item.mount,
                        item.price,
                        item.id
                    )
                }
            >
                수정
            </button>
        </div>
    </td>
    <td className="table_basket table_basket_right">
        {item.price
            .toString()
            .replace(/WB(?=(Wd{3})+(?!Wd))/g, ",")}
        원
    </td>
    <td className="table_basket">
        <div
            className="basketDelete"
            onClick={() => deleteProduct(item.id)}
        >
            <FiX />
        </div>
    </td>
</tr>
)}}

```

```

        </tbody>
    </table>
</div>

<div className="button-control">
    <div className="d-flex">
        <button
            type="button"
            className="btn btn-control ms-2"
            onClick={clearbasket}
        >
            장바구니 비우기
        </button>
    </div>

    <div className="pay-container">
        <table>
            <tbody>
                <tr>
                    <td className="table_bottom" colSpan="2">
                        총 주문상품 {basket.length}개
                    </td>
                </tr>
                <tr>
                    <td colSpan="2" className="pay-cell table_bottom">
                        <span className="pay_sum_sum">
                            {totalPrice &&
                                totalPrice
                                    .toString()
                                    .replace(/WB(?=(\Wd{3})+(?!Wd))/g, ",")}
                            원
                        </span>
                        <p>총 주문금액</p>
                    </td>
                </tr>
            </tbody>
        </table>
    </div>

```

```

        </table>
      </div>
      <button
        type="button"
        className="btn-btn-control-pay"
        onClick={topurchase}
      >
        결제하기
      </button>
    </div>
  </div>
  <div className="testBox" />
</div>{" "}
</>
);
}

```

```
export default Basket;
```

김민우

**** Faq 코드 ****

```

function Faq() {
  // 클릭한 FAQ 항목의 인덱스를 관리하는 상태
  const [selectedQuestionIndex, setSelectedQuestionIndex] = useState(null);

  // FAQ 항목을 클릭하면 호출되는 함수
  const handleQuestionClick = (index) => {
    // 이미 선택된 FAQ 항목을 다시 클릭하면 선택을 취소
    if (selectedQuestionIndex === index) {
      setSelectedQuestionIndex(null);
    } else {
      setSelectedQuestionIndex(index);
    }
  };
}

```

**** 공지사항 코드 ****

//관리자 계정만이 공지사항 게시글을 관리할 수 있게 함

```
const isAdmin = localStorage.getItem("email") === "admin@admin.com";
```

```
function checkAdmin() {  
  if (localStorage.getItem("email") === "admin@admin.com") {  
    navigate("/noticeboardwrite");  
  } else {  
    alert("관리자만 가능합니다.");  
  }  
}
```

//공지사항 글 게시

```
async function addNotice() {  
  try {  
    await axios.post("http://localhost:8080/kokee/notice/add", {  
      subject: subject,  
      content: content,  
      email: localStorage.getItem("email")  
    })  
    navigate(-1);  
  } catch (error) {  
    console.log(error)  
  }  
}
```

//공지사항 글 조회

```
useEffect(() => {  
  async function getNoticeList() {  
    try {  
      const result = await axios.get(  

```

```

        "http://localhost:8080/kokee/notice/list"
    );
    setResultNotice(result.data);
  } catch (error) {
    alert("네트워크 에러가 있어요");
  }
}
getNoticeList();
}, []);

```

//공지사항 글 삭제

```

async function deleteNotice() {
  if (localStorage.getItem("email") === "admin@admin.com") {
    try {
      await axios.delete(
        `http://localhost:8080/kokee/notice/delete/${notice.id}`
      );
      navigate("/");
    } catch (error) {
      alert("네트워크 에러가 있어요");
    }
  } else {
    alert("관리자만 가능합니다.");
  }
}

```

//공지사항 글 수정

```

function updateNotice() {
  if (localStorage.getItem("email") === "admin@admin.com") {
    navigate("/noticeboardmodify", {
      state: notice,
    });
  } else {
    alert("관리자만 가능합니다.");
  }
}

```



```

    }
}

```

**** 고객의 소리 코드 ****

```

function VoiceOfCustomer() {

    const [subject, setSubject] = useState("");
    const [content, setContent] = useState("");
    const navigate = useNavigate();

    const isAdmin = localStorage.getItem("email") === "admin@admin.com";

    //관리자 계정만이 고객의 소리 리스트 게시판에 접근할 수 있음

    function moveBoard() {
        if (localStorage.getItem("email") === "admin@admin.com") {
            navigate("/VoiceOfCustomerBoard")
        } else {
            alert("관리자만 접근 가능합니다.")
        }
    }

    async function addBoard() {
        try {
            await axios.post("http://localhost:8080/kokee/feed_back/add", {
                subject: subject,
                content: content,
                email: localStorage.getItem("email")
            })
            navigate("/")
        } catch (error) {
            alert("네트워크에 오류가 있어요.");
        }
    }
}

```

//고객의 소리 리스트 게시판 조회

```
useEffect(() => {
  async function getBoard() {
    try {
      const result = await axios.get("http://localhost:8080/kokee/feed_back/list")
      console.log(result.data);
      setResultData(result.data);
    } catch (error) {
      alert("네트워크 오류가 있어요.")
    }
  }
  getBoard();
}, [])
```

• 키오스크 화면 소스 코드

조인제

**** 장바구니 넣기 기능, 중복 있을시 이미 있던 상품의 수량만 늘어남 ****

```
function puttobasket() {
  let temp = {
    name: current.name,
    price: current.price,
    count: count,
    image: current.image,
    tempature: isiced ? "ICED" : "HOT",
    size: regular ? "Regular" : "Extra",
  };
  let flag = false;
  let temp2 = [...basket];
  temp2.forEach((item) => {
    if (
      item.name === temp.name &&
      item.tempature === temp.tempature &&
      item.size === temp.size
```

```

    ) {
        item.count += temp.count;
        flag = true;
    }
});
if (!flag) {
    temp2.push(temp);
}
setBasket(temp2);
closeModalHandler();
}

```

**** axios로 서버에 주문 데이터를 보내는 기능 ****

```

async function payFunction() {
    try {
        const result = await axios.post(
            "http://gambit.kro.kr:8080/kokee/kiosk",
            data
        );
        navigate("/done");
    } catch (error) {
        alert("네트워크에 오류가 있어요.");
    }
}

```

**** 영수증 출력 기능 ****

```

const onPrintHandler = () => {
    try {
        setPrinted(true);
        const printContents = ReactDOM.findDOMNode(a4Notice.current).innerHTML;
        const windowObject = window.open(
            "",
            "PrintWindow",
            "width=10, height=10, top=0, left=0, toolbars=no, scrollbars=no, status=no,
            resizale=yes"
        );
    }
}

```

```
    windowObject.document.writeln(printContents);
    windowObject.document.close();
    windowObject.focus();
    windowObject.print();
  } catch (e) {
    console.log(e);
    setCookie("basket", [], { path: "/" });
    navigate("/");
  }
};
```

**** 자동으로 홈페이지로 가면서 쿠키 초기화 ****

```
setTimeout(() => {
  setCookie("basket", [], { path: "/" });
  navigate("/");
}, 3000);
```