발표 스크립트

1

안녕하세요 빱조입니다.
지금부터 발표 시작하겠습니다.(클릭)

2

포스코인터내셔널과 함께
구내식당/카페 이용내역 확인용 모바일 앱을 만들었습니다.

저는 발표를 맡게 된 조혜원입니다. (클릭)

3

다음과 같은 순서로
발표를 진행하도록 하겠습니다. (클릭)

4

기획과정입니다(클릭)

5

현재
포스코 인터내셔널의 임직원분들은

사내에서 사원증 Tag를 통해
사내 카페에서의 (클릭)음료와 (클릭)식사 비용 등을 결제한다고 합니다.(클릭)

6

각 사업자가
임직원들이 한달 간 태깅한 사용실적을

(클릭)

엑셀 형태로
당사 총무부서에 전달하면 (클릭)

7

(클릭)전달받은
담당자가 내용검토 후 (클릭)
급여시스템에 등록하고
(클릭)
사용금액에 대해 차월 월급에서 차감되는 형태로 이루어지고 있다고 합니다. (클릭)

8

이렇다보니

임직원은 실시간 사용 내역을 확인하기 어려운 문제가 있다고 하셨습니다. (클릭)

9

회사에서 지원되는 금액과
본인이 부담하게 되는 금액을
바로바로
눈으로 확인하고 싶은데 말이죠(클릭)

10

그래서
사내 임직원들이 실시간으로
사원증을 이용한 결제 내역을

바로 확인할 수 있는 앱을 만들게 되었습니다. (클릭)

11

앱에 들어갈
필수 구현기능으로
(클릭)로그인과 (클릭) 지원금내역 표시 그리고
(클릭)모바일 알림 발송이 있고(클릭)

추가 구현기능으로
급여시스템에 반영할 수 있는 관리자 화면이 있었습니다(클릭)

12

이에 저희는(클릭)
다음과 같이 많은 소통과 멘토링을 통해(클릭)

13

앱과 키오스크 기능에 내기 및 오더기능도 추가하게 되었습니다. (클릭)

14

구체화된 서비스를 소개드리겠습니다.

저희 서비스는 앱 / 키오스크 / 관리자 세 부분으로 나뉘어있습니다. (클릭)

15

먼저, 앱에는(클릭)

1. 식당 메뉴 확인(클릭)
2. 카페 오더(클릭)
3. 사용 상세내역을 지원금과 함께 확인할 수 있는 사용내역 기능이 있습니다.(클릭)

16

키오스크에는
주문기능이 기본적으로 있고(클릭)
내기 기능과(클릭)얼굴 인식결제기능을 추가했습니다.(클릭)

17

마지막으로 지원금 관리자는 사원별 사용내역을
기간별로도 조회가능하며
급여 반영도 가능하도록 했습니다. (클릭)

18

그럼
저희 앱의 홍보영상 한번 보실까요?? (클릭)

19

영상 끝난 다음에 클릭

20

그럼 BBAP에 대해
시연과 함께 설명드리도록 하겠습니다.

(시연 화면으로 넘어가면서 자리 이동)

21

시연 잘보셨나요?
저희가 제공해드린 서비스를 만들기 위해 적용한 기술들을 소개드리겠습니다. (클릭)

22

프론트에서는 PWA로 빠르게 개발을 진행하였고 (클릭)
다음과 같은 피드백 요청서를 통해 자세한 피드백을 받을 수 있었으며
멘토님과 사원님의 피드백을 반영하여 보다 완벽한 서비스를 만들기 위해 노력했습니다.(클릭)

23

얼굴인식 결제기능을 위해
(클릭)OpenCV를 활용한 얼굴 검출 및
(클릭)KNN분류를 활용한 얼굴 인식 기술을 사용하였습니다.(클릭)

24

얼굴인식 결제 시의 보안을 위해(클릭)
타인의 이미지가 기존에 학습되어 있는 사람의 얼굴로 인식되지 않도록 하였고
총 5821장의 이미지 테스트를 통해 이를 확인했습니다.

추가적으로 (클릭) 사진으로 타인의 정보를 이용하는 스푸핑 사례를 방지하기 위해
얼굴 인식 시 눈과 입의 움직임을 감지해 이미지와 실제 사람을 분류하도록 하였습니다. (클릭)

25

주문방에 적용된 기술입니다.

먼저 (클릭) 각 사용자가 동시에 같은 화면을 볼 수 있도록
실시간 양방향 통신 기술인 WebSocket을 사용하였습니다.

둘째, (클릭) 메시지 기반의 프로토콜을 구현할 수 있게 도와주는 STOMP는
pub/sub 구조로 메시지를 처리할 수 있어 클라이언트와 서버 간의 메시지 전송을 더욱 효율적으로 관리할 수 있습니다.

세 번째, Redis (클릭) 입니다.
빠른 데이터 접근과 pub/sub 메시지 기능을 제공하여 여러 서버 간의 메시지 전달을 가능하게 하는 확장성을 제공합니다.

마지막으로 Kafka (클릭) 입니다.
각 파티션 내에서 메시지를 순서대로 기록 및 소비하여 메시지를 주문 이벤트가 발생한 순서대로 정확하게 처리되어 데이터 일관성을 유지할 수 있습니다.
또한, Kafka는 시스템 장애 시에도 메시지를 복구할 수 있어 높은 안정성을 제공합니다.(클릭)

26

저희 프로젝트는 (클릭) 여러개의 복잡한 서비스를 갖게 되는 구조를 지녀서 모놀리식 아키텍쳐로 구현하는 것보다,(클릭)

MSA 환경으로 구현하는 것이 더욱 효율적이라고 판단하였고
이에 (클릭)
도메인 주도 설계를 적용하여
관심사별로 총 9개의 서비스로 분리한 MSA 구조를 설계하였습니다.(클릭)

27

그 결과 다음과 같은 아키텍쳐가 나왔습니다.

효율적인 구조를 구성하기 위해 저희는 네 가지 사항을 고려하였습니다.(클릭)

28

첫 번째는 각 서비스가 단일 책임의 원칙을 준수하여 느슨한 결합을 유지할 수 있게 하는 것입니다.
Kafka 메시지 큐를 도입하여 서비스 간의 트랜잭션을 처리하는 방식으로 이를 실현하였습니다.

(클릭)
예를 들어 결제 프로세스가 진행될 때, (클릭)
메시지 큐를 통해(클릭) 결제, 알림, 식당 서비스에서 독립적으로 로직이 처리될 수 있도록 구현하였고,
그 결과 각 (클릭) 서비스의 결합도를 낮출 뿐만 아니라 (클릭) 전체적인 처리속도를 향상 시킬 수 있었습니다.(클릭)

29

두 번째는 각 서비스들이 독립적으로 빌드되고 배포하는 환경이 필요하였습니다.
이에 저희는 (클릭)쿠버네티스 도입으로 무중단 배포, (클릭)Argo와의 통합을 통한 지속적 배포를 활용하여
효율적이고 간단하게 MSA 환경을 구축할 수 있었습니다.(클릭)

30

세 번째는 안정성입니다. MSA는 각 서비스가 독립적으로 실행됨에 따라 더 많은 리소스를 필요로 합니다.
이에 대한 해결책으로 (클릭)그라파나와 (클릭)프로메테우스를 통해 모니터링 시스템을 구축하였고
(클릭)Mattermost와 알림을 연동하여 서비스에 과부하가 발생했을 때 신속하게 대응할 수 있도록 하였습니다.(클릭)

31

마지막은 처리 속도입니다.
저희는 Kafka뿐만 아니라 Java 21의 가상 쓰레드를 활용한 비동기 구조를 적용하여 처리 속도를 개선하였습니다.
또한 Redis 캐시를 활용하여 DB 접근 시간을 단축하고, 쓰기 연산을 줄여 서비스 성능을 향상시켰습니다.(클릭)

32

이렇게 완성된
저희의 서비스를 이용하게 되면,(클릭)

33

사내 임직원들께서 실제로 불편해하셨던 지원금과 본인부담금의 내역을 투명하게 확인하실 수 있으며,
총무담당자의 복잡했던 지원금 관리도 용이해집니다.
이렇게 직원들의 편의를 향상시킬 수 있는 유용한 앱이기에(클릭)

34

포스코그룹의 모든 계열사로 확장할 수 있다고도 생각합니다.(클릭)

35

그럼 이렇게 멋진 프로젝트를 함께해준 팀원 소개드리도록 하겠습니다.(클릭)

36

저희는 프로젝트 4주차에 팀원이 한분이 취업퇴소를 하게 되서

강성은
김다희
이성완
박영진
조혜원
이렇게 5명으로 프로젝트를 마무리하게 되었습니다.(클릭)

37

발표를 끝까지 집중해 들어주셔서 감사합니다. 질문있으시면 해주시기 바랍니다.