안녕하세요, 저는 이번 기말 발표를 맡은 오민진입니다.

대부분 사람들은 유통기한이 지난 음식은 먹을 수 없다는 인식을 갖고 있습니다.
유통기한은 상품이 시중에 유통될 수 있는 기한을 의미하는 것일 뿐 그때가 지났다고 하더라도 무조건 못 먹는 것은 아닙니다.
하지만 잘못된 인식으로 인해 매년 버려지는 식품 폐기물이 엄청나다고 합니다.

이러한 문제를 해결하기 위해 국내에서는 2023년부터 유통기한이 아닌 소비기한을 표시한다고 합니다.
소비기한은 보관 조건을 지키면서 식품을 소비하면 안전에 이상이 없는 기한을 의미합니다.
저희는 이런 소비기한을 사용하여 식품을 관리하는 서비스를 제공함으로써 불필요하게 버려지는 식품 폐기물을 줄이고, 보관 기관이 길어짐에 따라 불필요한 지출 또한 최소화하는 것을 목표로 개발을 진행하였습니다.

주요 요구사항으로 기능적 요구사항, 품질적 요구사항, 제약사항을 보도록 하겠습니다.

기능적 요구사항으로는 사용자가 영수증 이미지를 입력하면,
시스템이 영수증에 있는 품목을 추출하여 품목에 맞는 소비기한을 계산하여 사용자 보유 품목 리스트에 등록하고,
품목 리스트에 대해 조회하고, 품목을 수정, 삭제하는 것입니다.

품질적 요구사항으로는 OCR이 영수증 텍스트를 90% 이상의 정확도로 인식하는 것과 영수증에서 추출한 품목에 대한 소비기한을 90% 이상의 정확도로 제공하는 것입니다.
속도 측면에서는 모든 기능을 수행하는데 시스템의 반응 속도가 3초 이내로 이루어지는 것입니다.

제약사항으로는 3-4개월 이내로 프로젝트가 수행되어야 하며
안드로이드 스튜디오 개발 환경에서 코틀린 언어로 개발을 진행해야 합니다.

다음은 유스케이스 다이어그램입니다.
사용자는 외부 액터인 핸드폰을 사용하여 시스템의 기능 4가지, 등록, 수정, 삭제, 조회를 수행할 수 있습니다.

품목을 등록하는 유스케이스는 영수증으로부터 텍스트를 추출하여 추출한 품목에 대한 소비기한과 함께 리스트에 등록하는 것입니다.
텍스트를 추출하는 기능인 OCR은 품목을 등록하기 위해 반드시 실행되어야 하므로 include 관계로 표시하였고,
텍스트 추출을 위해서는 반드시 사진 첨부가 이루어져야 하므로 include 관계로 표시하였습니다.
또한, 사진 첨부는 외부 액터 카메라를 사용하므로 연관 관계로 표시하였습니다.

품목을 수정하는 유스케이스는 보관 방법을 수정하는 것과 등록 날짜를 수정하는 방법 두 가지가 선택적으로 실행될 수 있습니다.
따라서 보관 방법을 수정하는 것과 등록 날짜를 수정하는 것을 품목을 수정하는 유스케이스와 extend 관계로 표시하였습니다.

이번 발표에서는 품목을 등록한다와 삭제한다에 대한 유스케이스 명세서와 시퀀스 다이어그램을 자세히 다루겠습니다.

먼저 품목을 등록한다에 대한 유스케이스 명세서입니다.
액터는 사용자이고, 사전 조건으로는 사용자가 메인 화면에서 '영수증 촬영하기' 버튼을 클릭하는 것입니다.
기본 흐름을 간략하게 소개하자면 사용자가 카메라 버튼을 클릭하여 품목을 추출하고자 하는 영수증을 촬영하고
시스템이 해당 영수증에 대해 OCR 기능을 수행하여 품목을 추출하면 소비기한을 계산한 품목 리스트를 사용자에게 보여주는 것입니다.

대체 흐름으로는 영수증을 입력할 때 갤러리에서 사진을 불러오는 방법과
품목을 등록하는 최종 확인 과정에서 취소 버튼을 클릭하여 품목 등록이 되지 않는 흐름이 있습니다.

예외 흐름으로는 카메라나 갤러리 접근 허용을 거부하는 등의 흐름이 있습니다.

시퀀스 다이어그램은 MVC 기법을 활용하여 작성하였습니다.
View 클래스는 사용자 유저 인터페이스를 담당하는 클래스로,
Model 클래스로부터 정보를 얻어옵니다.
Controller 클래스는 View와 Model 사이의 매개, 중재 역할을 수행하는 클래스로,
View로부터 사용자 액션을 받아서 Model 클래스의 상태를 변경합니다.
Model 클래스는 데이터베이스 등 저장 매체 입출력을 담당하는 클래스입니다.

품목을 등록한다에 대한 시퀀스 다이어그램에서
View 클래스로는 품목 추출과 카메라, 갤러리를,
Controller 클래스로는 품목 추출, 카메라, 갤러리, OCR, 품목 등록을
Model 클래스로는 보유 품목 DB, 냉장 소비기한 DB, 실온 소비기한 DB를 생성하였습니다.

다음은 품목을 등록한다의 기본 흐름 시퀀스 다이어그램입니다.
사용자 클래스를 추가로 생성하여 시스템과의 상호작용을 이해하기 쉽도록 작성하였으며,
사용자의 액션은 View 클래스를 관리하는 Controller로 전달하여 기능을 수행하도록 하였습니다.
카메라 Controller는 외부 액터 카메라를 사용하여 사진 촬영을 수행하도록 하였으며
OCR Controller는 외부 액터 시스템 타이머를 사용하여 소비기한을 계산할 당시 날짜를 받아와 수행하도록 하였습니다.

먼저, 사용자가 카메라 버튼을 클릭하면, 해당 이벤트를 품목 추출 Controller로 전달합니다.
품목 추출 Controller는 카메라 Controller에게 카메라를 요청하고
카메라 Controller는 접근 허용을 위한 팝업창을 user에게 제공합니다.
사용자에게 보여지는 화면은 View 클래스에서 담당하므로, 카메라 controller의 동작을 품목 추출 Controller, 품목 추출 View의 순서로 전달하여 사용자에게 제공하게 됩니다.
이후, 사용자가 팝업창의 확인 버튼을 클릭하면, 해당 이벤트를 카메라 Controller로 전달하여 카메라 View를 실행하도록 합니다.
이때, 사용자가 팝업창의 취소 버튼을 클릭할 경우 시스템이 '팝업을 허용해주세요'라는 메시지를 제공하고 카메라가 실행되지 않습니다.
카메라가 실행되면 사용자는 추출하고자 하는 영수증을 촬영하고 그 결과물을 카메라 View에서 보여주어 원하는 부분만 잘라서 이미지를 활용하도록 합니다.
이 동작은 사용자가 촬영을 다시 시도하는 경우에 반복해서 작업할 수 있으므로 loop 박스로 묶어두었습니다.
사용자가 촬영 사진을 확정하면, '잠시만 기다려주세요'라는 메시지를 사용자에게 보여줌과 동시에 카메라 Controller에서 OCR Controller로 사진을 전달합니다.
OCR Controller에서 이미지 전처리, 텍스트 추출을 수행하고
추출한 텍스트에 대한 정보를 냉장 소비기한 DB로 요청을 하여 정보를 받습니다.
해당 정보를 가지고 품목에 대한 소비기한을 계산하여 그 결과를 사용자에게 보여줍니다.
사용자가 결과를 보고 '확인' 버튼을 클릭하면 추출한 품목을 보유 품목 DB에 저장함으로써 품목을 등록하는 기능이 끝이 납니다.
이때 사용자가 '취소' 버튼을 클릭하면 품목 등록이 취소되며 메인 화면으로 돌아가게 됩니다.

다음 시퀀스 다이어그램은 '품목을 등록한다'에 대한 대체 흐름을 나타낸 것으로,
카메라가 아닌 갤러리로부터 영수증 사진을 불러와 텍스트 추출을 수행하는 것입니다.
갤러리 Controller는 외부 액터 갤러리로부터 사진을 가져와 기능을 수행합니다.

다음은 품목을 삭제한다에 대한 유스케이스 명세서입니다.
액터는 사용자이고 유스케이스를 실행하기 위한 사전 조건으로는 사용자 보유 품목 리스트가 존재해야 하는 것입니다.
기본 흐름은 시스템이 사용자에게 현재 보유 품목 리스트를 보여주고,
사용자가 삭제하고자 하는 품목의 삭제 버튼을 클릭하면,
시스템이 '삭제하시겠습니까'라는 메시지를 보여줍니다.
해당 메시지에 대해 사용자가 '확인' 버튼을 클릭하면 기능 수행이 완료됩니다.
예외 흐름은 '삭제하시겠습니까'라는 메시지에 대해 '취소' 버튼을 클릭하는 것으로, 그 결과 품목이 삭제되지 않은 리스트를 보여줍니다.

'품목을 삭제한다'에 대한 시퀀스 다이어그램도 MVC 기법을 활용하여 작성하였습니다.
View 클래스로는 보유 품목 리스트를,
Controller 클래스로는 보유 품목 리스트와 품목 삭제를
Model 클래스로는 보유 품목 DB를 생성하였습니다.

시퀀스 다이어그램을 보면,
먼저 보유 품목 리스트 View가 사용자에게 현재 보유 품목 리스트를 보여줍니다.
사용자가 리스트 중 삭제하고자 하는 품목의 삭제 버튼을 클릭하면
이벤트를 품목 삭제 Controller로 전달하여 사용자에게 '삭제하시겠습니까'라는 메시지를 전달합니다.
해당 메시지에 대해 사용자가 '확인' 버튼을 클릭하면 품목 삭제 Controller가 보유 품목 DB에서 해당 품목을 삭제하고
삭제가 완료된 품목 리스트를 사용자에게 보여줍니다.
메시지에 대해 사용자가 '취소' 버튼을 클릭하면 삭제 기능이 취소되며 기존의 보유 품목 리스트를 사용자에게 보여줍니다.

다음은 클래스 다이어그램입니다.
저희는 지난 리팩토링 과제에서 받은 피드백을 반영하고,
매주 개발을 진행하면서 다이어그램에 많은 변화가 있었습니다.
왼쪽은 수정 전 클래스 다이어그램이고, 오른쪽은 수정 후 클래스 다이어그램입니다.

전체적으로 MVC 기법을 따라 각 클래스들을 View, Controller, Model로 나누었고
코드 작성 당시 사용했던 이름으로 클래스, 객체, 메소드 이름을 수정하여 각 클래스를 구성하도록 하였습니다.

View 클래스와 Controller 클래스의 관계는 Controller의 변화에 따라 View에 영향을 미치기 때문에 의존 관계로 표시하였습니다.

수정 전에는 Main Controller에서 Delete의 기능을 수행하였으나
수정 후에는 Delete Controller를 따로 생성하여 삭제 기능을 수행하도록 하였습니다.
Delete의 경우 새로운 View가 아닌 Main View에서 진행하는 기능으로 Delete View를 따로 생성하지 않았습니다.

품목을 등록할 때 품목 추출, 카메라, 갤러리, OCR 클래스를 사용합니다.
지난 번 리팩토링 당시 피드백을 반영하여
기존 OCR 클래스에 있던 텍스트 추출과 데이터베이스에 저장하는 두 기능을 단일 책임의 원리에 맞게 분리하여
InsertItem 클래스를 새롭게 생성하여 데이터베이스에 저장하는 기능을 수행하도록 했습니다.

저희는 Json 형식 파일을 사용하여 소비기한 데이터베이스를 구축하였고
보관 방법에 따라 실온과 냉장 데이터베이스로 나누었습니다.(1)
두 개의 클래스는 JsonDB 클래스를 구성하므로 합성 관계로 표시하였습니다.(2)
JsonDB 클래스를 따로 생성한 이유는 사용자의 개인 보유 품목을 관리하는 내부 데이터베이스인 MyList와 구별하기 위한 것입니다.(3)
JsonDB 클래스와 MyList 클래스는 item 클래스와 일반화 관계로 표현하였습니다.(4)

다음은 품질 시나리오입니다.

첫 번째 시나리오는 '사용자가 OCR 기능을 사용하여 영수증 품목이 90% 이상의 정확도로 추출되기를 원한다'입니다.
이를 만족시키기 위해 J tess box editor를 사용하여 직접 학습 데이터를 만들어서 OCR을 학습시켜 텍스트 추출을 진행하였습니다.
하지만 기존 OCR 기능이 최신화가 되어있어 오히려 더 낮은 정확도로 텍스트가 추출된 것을 알 수 있었습니다.
따라서 저희는 다른 추가적인 코딩 없이 최적화된 OCR 라이브러리를 사용함으로써
해당 시나리오를 만족시켰습니다.

두 번째 시나리오는 '사용자가 시스템을 통해 영수증 품목에 대한 소비기한 정보를 90% 이상의 정확도로 얻기를 원한다'입니다.
아직 우리나라의 경우 유통기한을 사용하고 있어 소비기한에 대한 정확한 정보가 나와있지 않았습니다.
따라서 저희는 현재 소비기한을 사용하는 외국의 데이터베이스를 활용하여 데이터베이스를 구축하였고,
이를 품목에 따라 소비기한을 계산하여 사용자에게 정보를 제공하도록 하였습니다.
소비기한의 경우 나라마다, 보관 방법에 따라 다르기 때문에
시나리오를 만족했다고 단정 지을 수는 없지만, 품목에 따라 지정된 소비기한을 제공하는 것은 달성했습니다.
따라서 2023년부터 적용될 우리나라의 소비기한을 차후에 적용한다면, 이 시나리오를 달성했다고 할 수 있을 것이라 예상됩니다.

마지막 시나리오는 '사용자가 시스템이 제공하는 각 기능이 3초 이내에 처리되기를 원한다'입니다.
이를 만족시키기 위해 저희는 Imagecrop 기능을 추가하였습니다.
이 기능은 촬영한 이미지를 잘라서 원하는 부분만 OCR의 input으로 사용하는 방법입니다.
실제 테스트를 진행한 결과 이미지를 자르지 않고 텍스트를 추출했을 때는 약 3000-4000ms가 소요되었지만,
필요한 부분의 이미지만 잘라서 텍스트를 추출했을 때는 약 2000ms가 줄어든 것을 볼 수 있었습니다.

저희는 원활한 개발 진행을 위해 매주 수업이 끝난 후 비대면 회의를 통해
발표 당시 받은 피드백을 반영하여 개발 상황을 조정하였고,
해당 주차의 과제를 수행하기 위한 역할을 분담하였습니다.
또한, 각자 맡았던 개발 진행 상황을 브리핑하여 진행 속도를 확인하였으며
모든 회의에 대해 회의록을 작성하여 깃허브에 업로드하였습니다.
매일 필요한 정보나 어려움, 의견 조율을 위해서 카카오톡 단체 채팅방을 활용하였고,

각자 진행 상황을 한눈에 파악할 수 있도록 깃허브의 칸반 보드를 활용하였습니다.
이를 사용함으로써 서로 맡은 사항이 어디까지 진행되었는지,
각자가 진행해야 할 과제가 무엇인지 빠르게 파악할 수 있었고
계획대로 큰 어려움 없이 개발이 잘 진행되었습니다.

개발을 진행하면서 저희 조가 잘한 점은
각자 주어진 과제와 역할을 정해진 시간 내에 수행하였고,
팀원과의 소통이 원활하게 이루어졌습니다.
또한, 각자 의견을 자유롭게 제시하여 피드백을 많이 주고받음으로써 더 좋은 해결책을 도출할 수 있었으며
매주 과제 수행 시 관련 자료를 검색하고 활용하여 더 정확한 결과물을 제출하기 위해 노력하였습니다.

잘못한 점으로는
초기 계획 수립 당시 파이어베이스를 사용하여 데이터를 구축해야 한다는 생각에 사로잡혀
시간을 낭비한 것입니다.
저희가 개발한 앱의 경우 외부와 따로 소통하는 것이 아니기 때문에 굳이 파이어베이스를 사용하지 않아도 내부 데이터베이스만으로도 작동할 수 있음을 개발 도중에 깨달아서 초기 계획 수립을 잘못한 것으로 판단하였습니다.
또한, 깃허브 사용에 미숙하여 각자 개발을 다른 파일로 진행하였다는 것입니다.
개발 초기 당시 맡은 바를 수행하기 위해 각자 다른 파일을 생성하였고
이를 하나로 합치기 위해 많은 시간이 소요되었습니다.
깃허브의 'branch' 기능을 활용하였더라면 하나로 합치는 과정을 생략하고 더 빠르게 개발을 진행할 수 있었을 것이라 생각하였습니다.

마지막으로 향후 다르게 개선할 점입니다.
현재 저희가 개발한 시스템을 향후에 보완한다면,
2023년부터 사용될 소비기한 데이터베이스를 활용하여 품질적 요구사항을 100% 만족하는 시스템을 개발할 수 있을 것입니다.
또한, 이번 개발을 진행하면서 잘못한 점을 반영하여
깃허브를 사용할 때, branch를 활용하여 개발을 진행하고
초기 설계를 잘 수립하여 불필요한 시간 낭비를 최소화한다면
더 좋은 프로젝트 결과물을 도출할 수 있을 것이라 생각합니다.

이상 발표를 마치겠습니다. 감사합니다