BALANCED ONE

Requirements Specification



|  |  |
| --- | --- |
| Student Number | Name |
| 2016314843 | 김윤재 |
| 2015312802 | 김두원 |
| 2016313088 | 조원이 |
| 2016313760 | 너드르 |
| 2015313016 | 이재현 |
| 2016313352 | 모하메드 |

Contents

[1. Preface 7](#_Toc23118035)

[1.1. Objective 7](#_Toc23118036)

[1.2. Readership 7](#_Toc23118037)

[A. User Requirements Readership 7](#_Toc23118038)

[B. System Requirements Readership 7](#_Toc23118039)

[1.3. Document Structure 8](#_Toc23118040)

[A. Introduction 8](#_Toc23118041)

[B. Glossary 8](#_Toc23118042)

[C. User Requirements Definition 8](#_Toc23118043)

[D. System Architecture 8](#_Toc23118044)

[E. System Requirements Specification 8](#_Toc23118045)

[F. System Models 9](#_Toc23118046)

[G. System Evolution 9](#_Toc23118047)

[H. Appendices 9](#_Toc23118048)

[I. Index 9](#_Toc23118049)

[2. Introduction 10](#_Toc23118050)

[2.1. Needs 10](#_Toc23118051)

[2.2 System Overview 11](#_Toc23118052)

[2.3 Expected Effects 13](#_Toc23118053)

[A. 소비자 정보 접근 불균형 해소 13](#_Toc23118054)

[B. 판매자 광고 비용 절감 13](#_Toc23118055)

[3 Glossary 14](#_Toc23118056)

[4 User Requirements Definition 16](#_Toc23118057)

[4.1. Functional Requirements 16](#_Toc23118058)

[A. User Sign up 16](#_Toc23118059)

[B. Seller Sign up 16](#_Toc23118060)

[C. User Login 17](#_Toc23118061)

[D. Seller Login 17](#_Toc23118062)

[E. Add Food 17](#_Toc23118063)

[F. Recommendation 18](#_Toc23118064)

[G. My page 18](#_Toc23118065)

[H. Order 18](#_Toc23118066)

[I. Add Product 19](#_Toc23118067)

[J. Product Management 19](#_Toc23118068)

[4.2. Non-functional Requirements 19](#_Toc23118069)

[A. Product requirements 19](#_Toc23118070)

[B. Organizational requirements 21](#_Toc23118071)

[C. External requirements 21](#_Toc23118072)

[5 System Architecture 22](#_Toc23118073)

[5.1 Frontend Architecture 22](#_Toc23118074)

[5.2 Backend Architecture 23](#_Toc23118075)

[5.3 Recommendation System 25](#_Toc23118076)

[5.4 Token Distribution System 26](#_Toc23118077)

[6 System Requirements Specification 27](#_Toc23118078)

[6.1 Functional Requirements 27](#_Toc23118079)

[A. Login 27](#_Toc23118080)

[B. Sign up(User) 28](#_Toc23118081)

[C. Sign up(Seller) 28](#_Toc23118082)

[D. Camera 29](#_Toc23118083)

[E. Image Classification 29](#_Toc23118084)

[F. Show Food Candidate List 30](#_Toc23118085)

[K. Food Nutrition Database 30](#_Toc23118086)

[L. Food Nutrition Information 31](#_Toc23118087)

[M. Radar graph indicator 31](#_Toc23118088)

[N. Body status accumulator 32](#_Toc23118089)

[O. Food recommendation 32](#_Toc23118090)

[P. Order menu 33](#_Toc23118091)

[Q. Product listing 33](#_Toc23118092)

[R. Add product 34](#_Toc23118093)

[S. Product management 34](#_Toc23118094)

[6.2 Non-functional Requirements 36](#_Toc23118095)

[A. Product requirements 36](#_Toc23118096)

[B. Organizational requirements 37](#_Toc23118097)

[C. External requirements 38](#_Toc23118098)

[6.3 Scenario Examples 39](#_Toc23118099)

[A. Login Scenario 39](#_Toc23118100)

[B. Searching food nutrition information Scenario 39](#_Toc23118101)

[C. Checking body nutrition status Scenario 40](#_Toc23118102)

[D. Seller marketing Scenario 41](#_Toc23118103)

[7 System Models 43](#_Toc23118104)

[7.1 Context models 43](#_Toc23118105)

[A. Context Diagram 43](#_Toc23118106)

[B. Process Diagram 44](#_Toc23118107)

[7.2. Interaction models 45](#_Toc23118108)

[A. Use case Diagram 45](#_Toc23118109)

[B. Tabular Description of Use case Diagram 46](#_Toc23118110)

[8. System Evolution 50](#_Toc23118111)

[A. Augmented Reality (AR) 50](#_Toc23118112)

[B. SNS Signup / Login 51](#_Toc23118113)

[C. Own Image Classification Model / Re-training System 51](#_Toc23118114)

[9. Appendices 52](#_Toc23118115)

[9.1 Hardware requirements 52](#_Toc23118116)

[9.2 Software requirements 52](#_Toc23118117)

[9.3 Database requirements 52](#_Toc23118118)

[A. User 사용자 정보를 저장하는 엔티티 53](#_Toc23118119)

[B. Product 구입 가능한 상품 정보를 저장하는 엔티티 53](#_Toc23118120)

[C. Food 음식의 영양소 정보를 저장하는 엔티티 53](#_Toc23118121)

[D. UserFood 각 유저가 섭취한 음식에 대한 정보를 저장하는 엔티티 53](#_Toc23118122)

[E. Recommendation 각 유저에 대한 추천 음식 정보를 저장하는 엔티티 53](#_Toc23118123)

[10. Index 54](#_Toc23118124)

[10.1. Tables 54](#_Toc23118125)

[10.2. Figures 54](#_Toc23118126)

[10.3. Diagrams 55](#_Toc23118127)

[11. References 56](#_Toc23118128)

# Preface

## Objective

Preface에서는 본 명세서의 예상 독자층을 밝히고, 각 챕터 별 내용을 간단하게 기술하고 있다. 또한, 문서 작성 과정에서의 각 version과 변경 사항, 새로운 버전을 만드는데 필요한 이론적 근거에 대해 서술한다.

## Readership

본 문서의 각 부분이 어떤 예상 독자층을 가지고 있는 지 설명한다.

### User Requirements Readership

User requirements는 본 시스템의 사용자를 예상 독자로 생각한다. 해당 챕터는 요구사항을 사용자가 쉽게 이해할 수 있게 작성하는 것을 목적으로 하며, 전문 용어의 사용을 지양하고, 도표, 도식 등의 시각 자료를 동반한 자연어로 요구사항을 서술한다.

### System Requirements Readership

System Requirements는 개발에 참여하는 개발자와 프로젝트 참여자를 예상 독자로 생각한다. 해당 챕터는 요구사항을 실제 개발 과정에서 참고 자료로 사용하거나, 계약서에서 사용될 수 있게 작성하는 것을 목적으로 한다. 전문 용어를 사용하며, 상세하고 체계적으로 구조적 언어를 사용해 요구사항을 서술한다.

## Document Structure

1. Introduction  
   Introduction에서는 개발하고자 하는 시스템의 필요성에 대해 논한다. 시스템이 제안되는데 바탕이 된 니즈를 설명하고 Balanced one! 서비스의 개략적인 구조와 기능에 대해 서술한다. 마지막으로 본 서비스를 시장에 도입함으로써 기대되는 효과에 대해 논한다.
2. Glossary  
   Glossary에서는 본 문서에서 사용되고 있는 기술적 용어들을 정의하고, 해당 용어가 어떤 뜻을 가졌으며, 어떤 맥락에서 사용되는지를 설명한다. 이는 사용자와 이해당사자 등 배경지식이 없는 독자들이 본 문서를 이해하는데 도움을 주기 위함이며, 가능한 자세하게 전문 용어의 사용을 지양하며 용어에 대해 설명한다.
3. User Requirements Definition  
   User Requirement Definition에서는 사용자에게 시스템이 어떤 서비스를 제공하는지에 대하여 설명한다. Functional requirement와 non-functional requirement 두 가지로 분류해 서술하며, 시스템을 구성하는 기능들을 모두 포함하고 있다. 사용자가 쉽게 문서를 이해할 수 있도록 자연어와 도식을 사용하며, 따라야 하는 제품이나 프로세스의 표준을 명시한다.
4. System Architecture  
   System Architecture에서는 시스템의 구조를 설명한다. 다이어그램을 이용하여 시스템의 전체적인 구조와 서브시스템들 간의 관계를 설명하고, 각 시스템의 기능과 구조를 설명한다.
5. System Requirements Specification  
   System Requirements Specification에서는 User Requirements Definition에서 간략히 기술한 Functional Requirements와 Non-functional Requirement의 요구사항을 구체적으로 명시한다. 이때 본 챕터는 개발 프로세스 전반에서 사용되므로 다이어그램 등을 사용해 구체적이고 자세하게 기술한다.
6. System Models  
   System Models에서는 본 시스템과 시스템의 구성요소, 외부 동작 환경 사이의 관계들을 시각적인 모델로 표현한다. 시각적인 모델은 Context Diagram, Process Diagram, Use-case Diagram등 다양한 Diagram을 포함한다.
7. System Evolution  
   System Evolution은 본 시스템이 기반으로 하는 기본적인 가정들을 정의하고, 현 시스템의 개발상 한계와 시스템을 운영하면서 발생할 수 있는 변경사항들을 기술한다. 더불어, 예상 가능한 변경사항들에 맞춰 발전시킬 방향성에 대해 논한다. 이는 추후 시스템을 변경할 때 제약이 될 수 있는 design decision을 피하기 위함이다.
8. Appendices  
   Appendices에서는 본문에서 구체적으로 기술하지 않은 참고자료 등을 제공한다. 시스템 안에서 사용되는 하드웨어, 데이터베이스 요구사항, 개발 환경 요구사항 등을 포함하고 있다.
9. Index  
   Index에서는 본 문서에서 사용된 그림, 표, 다이어그램 등의 시각자료에 대한 색인을 철자 순서대로 기술한다.

# Introduction

이 챕터에서는 본 시스템을 둘러싼 니즈를 설명하고, 시스템의 기능을 비롯해 본 시스템이 다른 시스템과 어떻게 상호작용하는지를 간략히 서술한다.

## 2.1. Needs

최근에는 단순히 싸고 양이 많은 음식보다는 비싼 가격에도 불구하고 훌륭한 맛을 가지거나, 몸에 좋은 음식이 각광받다. 이와 더불어 체중 감량이나 웰빙에 대한 사람들의 관심이 높아지면서 음식의 칼로리나 영양 정보에 대한 필요성이 점점 증가하고 있다.

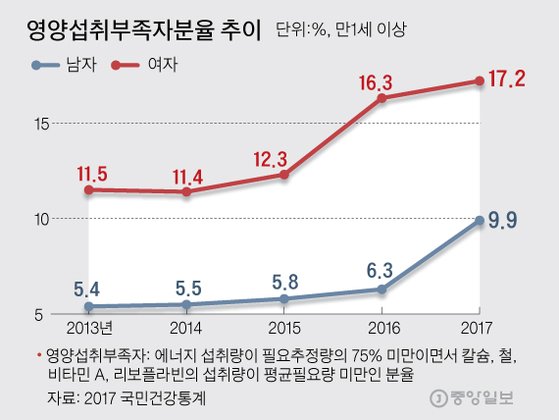
 

Figure 건강 기능식품 국내 시장규모 Figure 2 영양 섭취 부족자 분율 추이

사회 전반적인 트렌드 이외에도 영양 정보에 관심있는 계층은 남녀노소를 가리지 않고 광범위하게 존재한다. 영양 섭취가 건강과 직결되는 유아, 노인 계층, 편향적인 영양 섭취 경향이 있는 자취생, 직장인들, 성장기에 균형 잡힌 영양이 필요한 청소년층, 건강에 문제가 있는 환자들은 사회에 다수 존재한다. 사회적인 트렌드, 교육, 상식적인 측면에서 우리 모두는 몸의 영양 불균형이 건강에 심각한 문제를 끼칠 수 있다는 것을 알고 있으며, 영양적으로 균형 잡힌 건강한 몸을 만들기 위해서는 그 무엇보다도 음식의 영양 정보를 알고 식단을 관리해야 한다고 알고 있다. 이렇게 그 어느 때보다 많은 사람들이 자신의 몸 상태와 영양 밸런스에 대해 관심이 많으며, 자신의 몸에 부족한 영양 정보나 이를 보완하기 위해 섭취해야 하는 음식을 알기를 원한다.

그러나 온라인에 수많은 영양정보들이 있음에도, 특별히 수고를 들여 이를 검색하지 않고 서는 이 정보들을 얻기가 쉽지가 않으며 그 마저도 부정확한 경우가 많다. 실제로 ‘김치찌개’의 영양성분을 알기 위해서는 1) 구글 검색 2) 식품 영양 데이터베이스에서 검색 3) 후보자 선택 등 최소 세단계의 검색과정은 거쳐야 하며, 그 영양 정보 마저도 전문가가 아니면 쉽게 알아보기 힘들고 불친절하다. 이렇게 영양정보를 찾기가 어렵기 때문에 많은 소비자가 이를 활용하기 까다롭게 여기거나 불편하게 느낀다. 우리가 개발하고자 하는 시스템인 “Balanced one”은 이러한 정보들을 바탕으로 쉽고 빠르게 영양 섭취 정보를 기록, 확인, 보완할 수 있어 앞서 서술한 많은 사람들의 필요를 충족할 수 있는 영양 관리 시스템이 될 수 있다.

## System Overview

‘Balanced one’은 소비자들이 원하는 음식의 영양정보들을 빠르고 쉽게 제공하고 부족한 영양을 보완, 관리할 수 있게 도와주는 영양관리 시스템이다. 소비자들은 섭취하고자 하는 음식의 영양정보를 단 한번의 사진 촬영을 통해 손쉽게 얻을 수 있으며, 지금까지 섭취한 음식의 영양 정보를 종합하여 자신의 몸의 영양 균형을 명확하게 확인할 수 있다. 또한 건강한 식품들을 판매하는 판매자와 소비자들을 이어주는 소셜 마켓의 기능도 가지고 있어 소비자들이 신속하게 부족한 영양을 보완할 수 있는 음식을 추천 받고 구매할 수 있다.

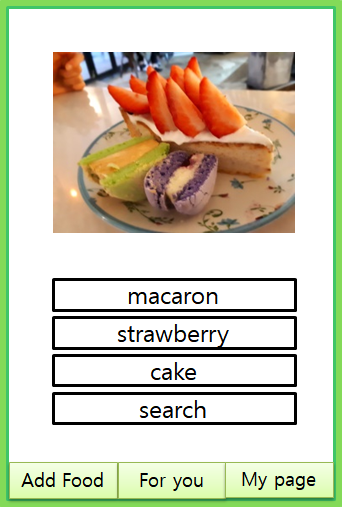
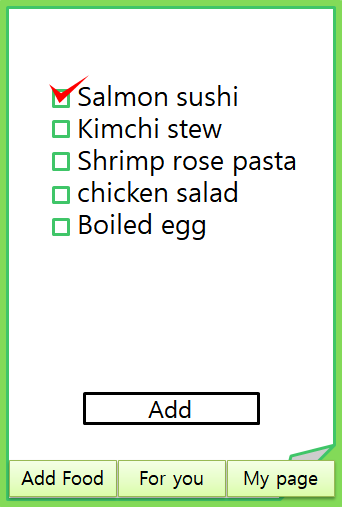
 

Figure 3 카메라를 이용한 음식 인식 및 선택

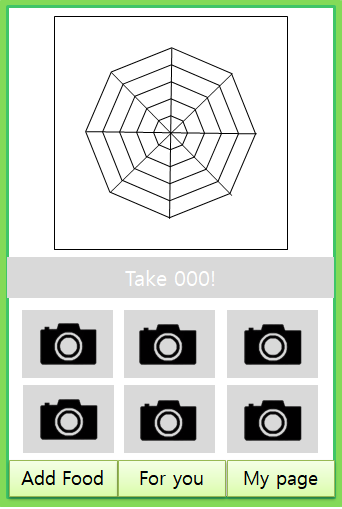
 

Figure 4 몸의 영양 상태 확인 Figure 5 음식의 영양 정보 확인

‘Balanced one’의 주요 기능은 세 가지인데, 카메라를 이용해 음식 사진 촬영 후 인식을 통해 섭취하고자 하는 음식을 빠르게 검색하고 영양 정보를 알 수 있다. 두 번째로 섭취한 영양정보를 그래프로 알기 쉽게 알려준다. 세 번째로 몸의 영양 통계를 바탕으로 부족한 영양분을 보완할 수 있는 음식을 추천해주고 쉽게 구매할 수 있도록 연결해 준다.

## Expected Effects

### 소비자 정보 접근 불균형 해소

기존의 단일 기준 정렬에 기반해 소비자가 일일이 리뷰를 확인해 가면서 유용성을 판단하는 프로세스는 매우 비효율적이고 소비자의 경험과 기반 지식에 따라 획득할 수 있는 정보의 질과 양에 차이가 매우 심각하다. 그러나 본 시스템을 사용할 경우 사용자들이 남긴 리뷰를 자동으로 분석해 소비자의 관심사과 중요도에 따라 주요 키워드를 나열해 주고, 그에 기반한 상품 점수 표시를 통해 쇼핑 경험에 무관하게 객관적이고 정확한 상품 리뷰를 확인할 수 있게 되었다.

### 판매자 광고 비용 절감

오픈마켓은 우선적으로 자사와 제휴해 광고료를 지불하는 업체의 상품을 우선적으로 노출하기 때문에 판매자 입장에서는 상품 판매 실적을 높이기 위해서 광고료를 지불할 수밖에 없고, 이는 수익성 악화로 이어진다. 그러나 본 시스템은 사용자 리뷰를 기반으로 높은 점수를 얻은 순서대로 상품을 노출하므로 판매자가 광고 비용을 지불할 필요가 없다.

# Glossary

Glossary에서는 문서에서 사용되는 용어에 대해 설명한다.

|  |  |
| --- | --- |
| Term | Description |
| Diet | 식습관과 식이요법을 의미한다. 단순히 체중을 줄이는 것뿐만이 아닌 에너지, 영양소 등의 균형 잡힌 식습관을 위한 식이요법을 포괄한다. |
| Nutrient | 영양소를 의미한다. 음식물 속에 들어 있는 에너지원이나 몸의 구성 성분이 되는 물질을 의미한다. |
| Signup | 회원가입을 의미한다. 사용자를 이를 통해 본인의 정보를 시스템에 등록하고, 시스템을 사용할 수 있는 권한을 부여 받는다. |
| Login | 로그인을 의미한다. 사용자는 이를 통해 세션을 부여 받고 이를 이용하여 시스템을 사용한다. |
| Logout | 로그아웃을 의미한다. 사용자는 부여 받은 세션을 반납하고 시스템 이용을 종료한다. |
| Session | 세션을 의미한다. 로그인 시 부여 받고, 로그아웃시 반납한다. 세션을 소유한 동안 시스템에 대한 사용 권한을 취득하며, 매 로그인 시 새로운 세션을 발급받는다. |
| Add Food | 먹은 음식 추가 기능을 의미한다. 사용자는 먹은 음식의 사진을 찍어 데이터베이스에 등록한다. |
| Image Classification | 기계학습 인공지능의 일종으로, 이미지를 학습하여 입력된 이미지가 무엇과 관련된 이미지인지 분류한다. 해당 시스템에서는 ‘Add Food’에서 이용된다. |
| For you | 음식 추천 기능을 의미한다. 사용자가 등록한 음식을 분석하여 사용자에게 부족한 영양소를 파악하고, 이를 기반으로 음식을 추천하여 사용자에게 보여준다. |
| Order | 음식 주문하기 기능을 의미한다. 사용자는 시스템이 추천한 음식을 셀러가 등록한 음식 중에서 골라 구입할 수 있다. |
| Database | 데이터베이스를 의미한다. 해당 공간에 시스템에서 사용되는 데이터(정보)들이 저장된다. |
| API | Application Programming Interface로, 응용 프로그래밍 인터페이스를 의미한다. API를 통해 외부에 정의되어 있는 시스템을 이용할 수 있다. |
| REST API | Representational State Transfer API로, CRUD 연산을 수행하기 위한 API이다. CRUD에 대해서는 아래에서 서술한다. |
| CRUD | Create, Read, Update, Delete를 의미한다. 데이터베이스에 저장되어 있는 정보에 대한 특정 동작 및 연산을 의미한다. |
| Frontend | 시스템에서 사용자가 볼 수 있으며, 사용자와의 상호작용을 담당하는 부분을 의미한다. |
| Android | 대표적인 모바일 운영체제로, 구글에서 개발하였다. 크게는 운영체제와 미들웨어, 사용자 인터페이스, 어플리케이션, MMS 서비스 등을 하나로 묶어 서비스를 제공하는 어플리케이션 플랫폼이라고 볼 수 있다. 해당 시스템에서는 안드로이드 플랫폼에 올라가는 어플리케이션으로 프론트엔드를 구축한다. |
| React | 페이스북에서 제공해주는 프론트엔드 라이브러리로, 사용자와 상호작용할 수 있는 UI를 손쉽게 개발할 수 있도록 해준다. |
| Backend | 시스템에서 사용자가 볼 수 없으며, 시스템 운영을 위한 전반적인 작업이 수행되는 부분을 의미한다. 흔히 서버라고 불리운다. |
| AWS | Amazon Web Service로, 아마존사에서 운영하는 클라우드 컴퓨팅 플랫폼이다. 해당 플랫폼을 이용하여 서버, 데이터베이스 등을 클라우드 형태로 운영할 수 있다. |
| MySQL | 전세계적으로 가장 널리 사용되고 있는 오픈 소스 데이터베이스이다. SQL을 사용하는 RDBMS 형태의 데이터베이스이다. SQL과 RDBMS에 대해서는 아래에서 서술한다. |
| SQL | Structured Query Language로, RDBMS에 저장된 데이터와 통신하기 위해 필요한 프로그래밍 언어이다. |
| RDBMS | Relational Database Management System으로, 관계형 데이터베이스를 생성, 갱신, 관리하기 위한 시스템이다. |
| Node.js | JavaScript를 기반으로 하는 서버 프레임워크이다. 해당 시스템에서는 이를 이용하여 백엔드 서버를 구축한다. |

Table Glossary

# User Requirements Definition

본 챕터에서는 시스템이 사용자에게 제공하는 서비스에 대해 서술하고, 사용자가 시스템을 사용하는 데 있어 중요한 비기능적 요구사항에 대해 설명한다. 이번 챕터의 독자는 사용자이므로 전문지식이 없는 독자도 이해할 수 있도록 기술용어의 사용을 자제하고, 자연어와 시각자료를 활용해 각 요구사항에 대해 설명한다.

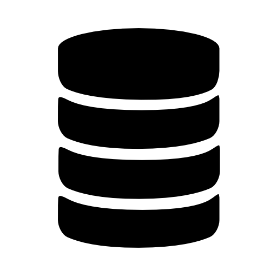
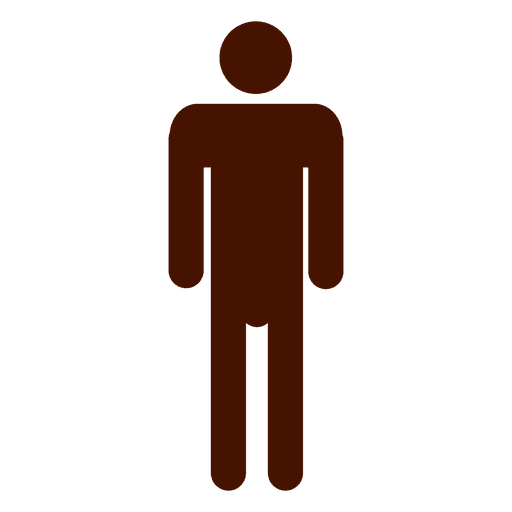
## 4.1. Functional Requirements

### User Sign up

이 요구사항은 유저가 카카오 계정을 통한 인증 과정을 걸쳐 일반 유저로서 해당 시스템을 사용하는데 필요한 권한을 얻는 과정이다. 인증 과정을 통해 ID와 Password가 생성되고 데이터베이스에 저장된다. 회원가입 과정에서 카카오 계정을 사용하기 때문에 별도의 인증 과정 없이 간편하게 사용할 수 있다.

### B. Seller Sign up

이 요구사항은 유저가 카카오 계정을 통한 인증 과정을 걸쳐 음식 판매자로서 해당 시스템을 사용하는데 필요한 권한을 얻는 과정이다. 인증 과정을 통해 ID와 Password가 생성되고 데이터베이스에 저장된다. 회원가입 과정에서 카카오 계정을 사용하기 때문에 별도의 인증 과정 없이 간편하게 사용할 수 있다.



ID, Password,  
Personal Information

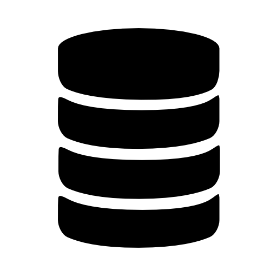
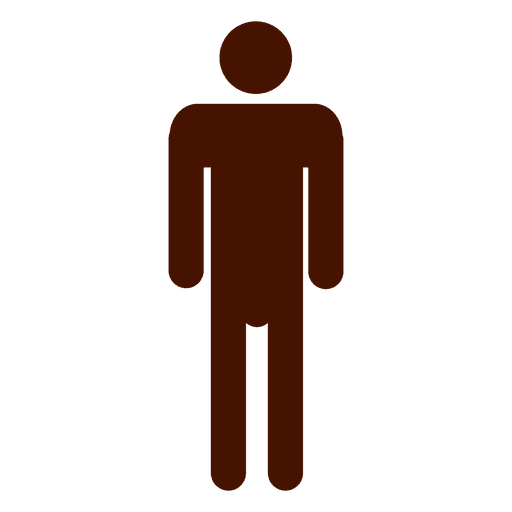
Figure 6 Login process

### C. User Login

이 요구사항은 유저가 인증된 ID와 password를 이용하여 해당 시스템을 사용하는 권한을 가져오는 것이다. 입력한 ID과 password가 database에 등록된 정보와 일치하면 로그인할 수 있다. 성공적으로 로그인하면 해당 유저로 인증되며, 해당 유저의 다른 개인 정보에 접근, 수정 할 수 있다. Balanced one에서의 일반 유저는 자신이 먹은 음식을 등록하고, 영양 상태를 확인하며 맞춤형 음식을 추천 받거나 음식을 주문 할 수 있다.

### D. Seller Login

이 요구사항은 유저가 인증된 ID와 password를 이용하여 해당 시스템을 사용하는 권한을 가져오는 것이다. 입력한 ID과 password가 database에 등록된 정보와 일치하면 로그인할 수 있다. 성공적으로 로그인하면 해당 유저로 인증되며, 해당 유저의 다른 개인 정보에 접근, 수정 할 수 있다. Balanced one에서의 셀러는 자신이 판매할 음식에 대한 정보를 등록, 수정하고, 음식을 판매할 수 있다.



ID, PW

Figure 7 Signup process

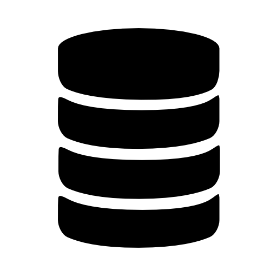
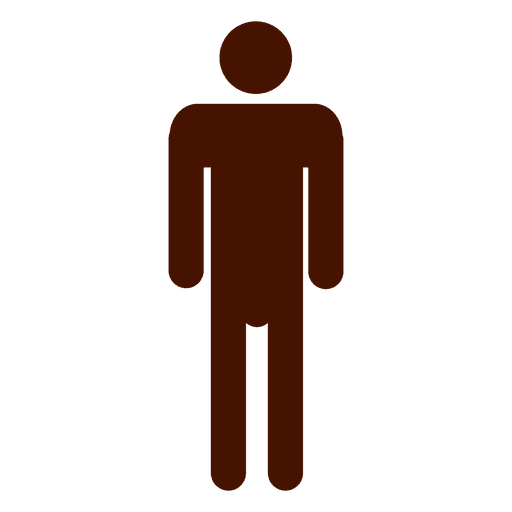
### E. Add Food

이 요구사항은 유저가 핸드폰에 내장된 카메라를 이용해 자신이 먹은 음식을 등록할 수 있는 기능이다. 유저는 핸드폰 카메라로 사진을 찍어 서버에 등록하면, 시스템이 해당 사진을 분석해 어떤 음식인지 후보지를 제시한다. 유저는 해당 음식 이름을 선택해 자신이 먹은 음식을 등록할 수 있다. 자신이 먹은 음식을 등록하면, 해당 음식의 영양 성분이 데이터 베이스에 등록되며, 이는 음식을 추천하는 데 사용된다.



User Nutrition

Food Nutrition



Food

Figure 8 Add Food

### F. Recommendation

이 요구사항은 유저가 먹은 음식의 영양 성분을 바탕으로 부족한 영양소를 파악하고, 해당 영양소들이 포함된 음식을 추천해주는 기능이다. 유저가 추가한 음식의 영양 성분 중, 가장 기본이 되는 칼로리와 쉽게 챙기기 어려운 영양소인 단백질, 비타민A, 비타민B, 비타민C, 칼슘, 아연을 기준으로 분석한다. 해당 영양소를 하루 권장섭취량에 따른 비율을 계산하여 그래프로 보여주고, 이에 따른 추천 음식을 제공한다.



Recommended Foods

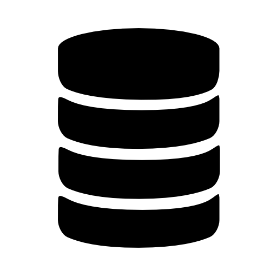
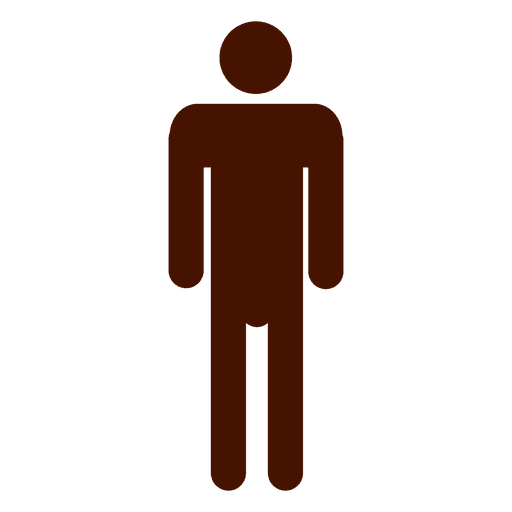


Figure 9 Recommendation

### G. My page

이 요구사항은 유저가 이름, 성별, 나이 정보를 수정할 수 있는 기능을 한다. 성별 정보는 남성과 여성을 선택할 수 있고, 나이 정보는 연령대별로 10대, 20대, 30대, 40대, 50대, 60대 이상으로 선택할 수 있다. 이 페이지는 로그인한 유저만 접근 가능하다.

### H. Order

이 요구사항은 유저가 추천 받은 음식을 한번의 클릭으로 주문하는 기능을 한다. 추천 음식의 사진과 이름, 가격 정보를 보여주고, 음식의 칼로리, 탄수화물, 단백질, 지방, 비타민 등 영양정보를 보여준다. 또한 수량을 입력하는 곳과 주문 버튼을 통해 원하는 수량만큼 한번의 클릭으로 주문이 가능하다.

### Add Product

이 요구사항은 음식 셀러가 판매하고자 하는 음식을 판매 리스트에 등록하는 기능을 한다. 음식 이름, 설명, 이미지, 가격, 재고 수량 정보와 영양 정보를 입력할 수 있고, 판매 음식 등록 버튼을 통해 판매 리스트에 등록이 가능하다. 음식 등록이 완료되면, 셀러는 판매 음식이 추가되었다는 알림을 받을 수 있다. 등록하고자 하는 음식의 정보가 불충분할 경우, 판매 음식 등록을 할 수 없다.

### Product Management

이 요구사항은 음식 셀러가 이미 등록한 상품의 정보를 수정하는 기능을 한다. 셀러는 판매 리스트에 등록된 자신의 상품을 선택하고, 선택 시 나오는 이름, 설명, 이미지, 가격, 재고 수량 정보와 영양 정보를 선택하여 수정하기 버튼을 통해 수정할 수 있다. 판매 리스트에 등록하지 않은 음식 정보는 수정할 수 없다.

## Non-functional Requirements

### Product requirements

1. Usability

Usability는 성공적인 전자상거래 어플리케이션을 만드는데 필수적이다. 사용자의 편의를 위해서 직관적이고 간단한 UI(User Interface)를 제공하되, 원하는 기능을 실행하는데 까지 많은 행동을 요구해서는 안 된다. Balanced one은 심플한 디자인과 최소한의 수의 버튼과 탭을 구현하여 사용자가 불편함 없이 빠르게 다른 여러 섹션에 도달 할 수 있게 도와준다.

또한, 본 시스템은 모바일 어플로 접속하기 때문에 모바일에 맞춘 접속 환경을 제공해야 하며, 사용자가 쉽게 어플에 접근할 수 있게 해야 한다.

정보를 등록하고 음식을 추천 받고 구매하는 사용자와 판매할 음식을 등록하고 판매하는 셀러를 분리하며, 각 입장에 맞는 기능만을 제공해 불필요하게 복잡해지는 것을 막아야 한다.

1. Performance

유저가 입력한 사진으로 음식을 입력하는 기능에서 음식을 정확하게 구분해내는 기술이 필요하다. 그래서 유저가 한 번에 음식을 등록할 수 있도록 굉장히 많은 종류의 음식 이미지 데이터를 보유하고, 정확하게 구분해 내는 학습이 되어있는 Image Classification 기술이 필요하다.

또한, 음식 추천 알고리즘은 유저가 먹은 음식의 영양소를 정확하게 분석하고 이를 바탕으로 부족한 영양소를 포함한 음식을 추천해야 하며, 추천된 음식과 구매 기능까지의 연결이 매끄러워야 한다. 또한, 음식에 따른 판매자 목록을 제공하며 간단하게 구매를 완료할 수 있어야 한다.

1. Dependability

우리 프로젝트는 모든 action을 atomic하게 만들어 dependability를 최대화한다. 즉, 유저가 음식을 등록, 주문하거나, 셀러가 판매할 음식을 등록하는 중간에 시스템 충돌이 발생하면, 서버와의 연결이 끊어져 진행되면 모든 것들이 취소되며 진행 중이던 액션의 데이터가 사라진다. 이를 통해 asynchronous modification을 방지한다. 또한, 하드웨어 문제로 예기치 못한 상황에 어플리케이션이 종료되면, 유저가 접속 하자마자 모든 것을 원래대로 돌려 놓는다.

1. Security

본 시스템은 카카오 계정을 연동하여 로그인하기 때문에 사용자의 정보가 외부로 유출되면 안 되며 외부의 접근에 대한 방어가 필요하다. 또한, 회원 가입 당시의 정보 뿐만 아니라, 사용자가 시스템을 사용하면서 만들어지는 데이터가 저장되는 데이터베이스 역시 잘 관리되어야 한다. 마지막으로, 사용자의 결제 수단에 대한 보안도 잘 관리 되어야 한다.

### Organizational requirements

1. Operational

현재 지속적으로 운영할 수 있는 하드웨어 자원이 없으므로, 상용 클라우드 서버에 배포할 것을 전제로 한다. 따라서 다양한 클라우드 서버에 배포할 수 있도록 hardware-dependent한 구현을 자제하고, 서버의 주소가 바뀌어도 대응할 수 있게 시스템을 설계해야 한다.

1. Development  
   보다 빠른 기간에 시스템 개발이 가능하도록 Parallel Development를 도입하기 위해 개발 단위는 프론트엔드와 백엔드로 나눈다. 또한, 정확한 이미지 분석을 위해 Image Classification이라는 Machine Learning AI Technology인 Vision AI를 사용한다.

### External requirements

1. Regulatory Requirement

본 시스템은 사용자에게 로그인을 요구하기에 앞서 먼저 개인정보를 수집함에 대한 동의를 구하고, 취급방침에 대해 알려야 한다. 또한, 정부를 외부에 제공하게 될 경우엔 사용자에게 동의를 먼저 구해야 하며, 취급방침을 벗어나게 사용자의 정보를 사용해서는 안 된다.

1. Safety/Security Requirement  
   본 시스템은 음식 등록 기능 및 추천 기능을 제공하기 위해 회원제로 운영함에 따라 사용자의 개인정보를 수집한다. 이러한 개인정보가 유출되지 않도록 해야 하며, 결제 시스템에도 오류가 있으면 안 된다.

# System Architecture

System Architecture에서는 시스템의 전체적인 구성을 서술한다. 각 서브시스템의 구성, 서브시스템간의 관계를 개략적으로 서술하며, 각 구조는 다이어그램을 첨부하여 이해를 돕는다.

## Frontend Architecture

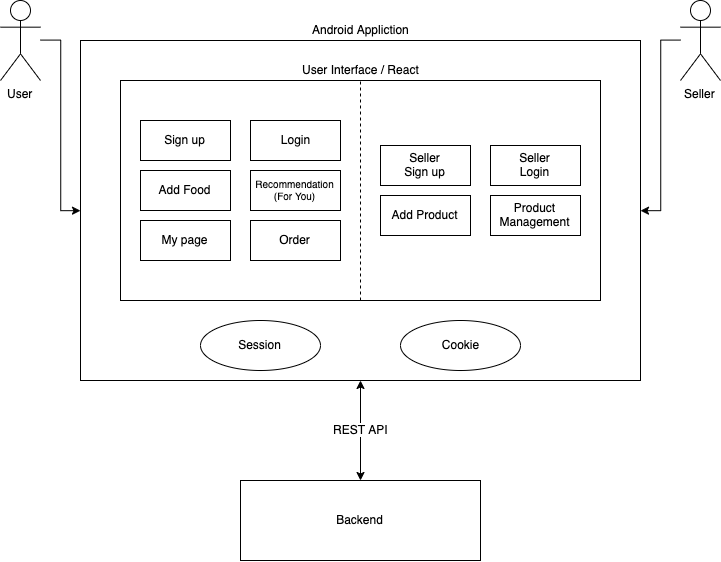


Figure 10 Frontend Architecture

Frontend Architecture는 User Interface에 대한 구성을 중점적으로 기술한다. Frontend는 Android Application으로 개발되며, 내부적으로 React Framework를 이용하여 User Interface를 구축한다. Application에서는 Session과 Cookie를 사용한다. 사용자는 Normal User(다이어그램 상에서의 User)와 Seller User(다이어그램 상에서의 Seller)로 구분되어진다. User Interface 내에 Normal User를 위한 Interface로 Sign up Interface, Login Interface, Add Food Interface, Recommendation(For You) Interface, My Page Interface, Order Interface가 제공되며 Seller User를 위하여는 Seller Sign up Interface, Seller Login Interface, Add Product Interface, Product Management Interface가 제공된다. Frontend System은REST API를 통하여 Backend System과 통신한다.

## Backend Architecture

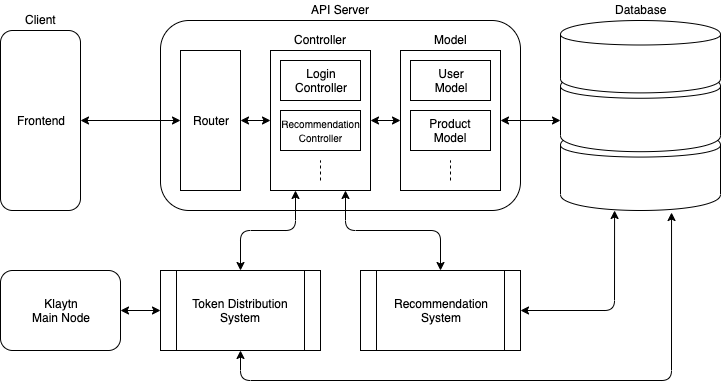


Figure 11 Backend Architecture

Backend Architecture는 API Server에 대한 구성을 중점적으로 서술한다. Backend에 해당하는 Recommendation System과 Token Distribution System은 별도의 항목으로 서술한다.

앞서 Frontend Architecture에서 서술하였던 바와 같이 Frontend(Android Application)은 REST API를 통해 Backend Server와 통신한다. Backend Server Architecture로는 MVC Model이 채택되었으나, REST API Server로만 사용될 것이기 때문에 View Component는 사용되지 않는다.

Frontend에서 REST API를 통해 API Server에 요청을 보내면 Router가 각각의 요청에 해당하는 Controller를 호출한다. Controller에서는 요청에 해당하는 동작을 수행하며, 필요에 따라 외부 시스템인 Recommendation System과 Token Distribution System을 호출한다.

Controller에서는 데이터가 필요할 경우 필요에 따라 Model을 통해 DB와 통신하며, 통신 결과를 또한 필요에 따라 가공하여 Router를 통해 Frontend에 전송한다.

## Recommendation System

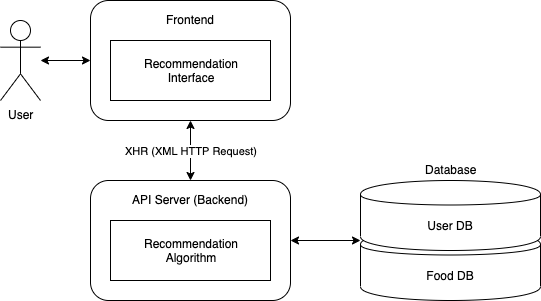


Figure 12 Recommendation System Architecture

Recommendation System은 User DB에 저장되어 있는 사용자가 지금까지 입력한 먹은 음식 정보와 Food DB에 저장되어 있는 각 음식에 대한 영양소 정보를 Recommendation Algorithm을 통해 분석하여 부족한 영양소를 파악하고, 사용자에게 필요한 음식을 추천한다.

## Token Distribution System

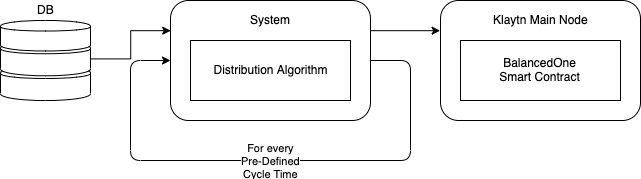


Figure 13 Token Distribution System Architecture

Token Distribution System은 미리 정해진 일정 사이클 시간마다 호출된다. 매번 호출될 때마다 DB에서 사용자의 섭취 영양소 정보를 받아온다. 받아온 영양소 정보를 계산하여 사용자의 영양 수준이 어느정도 균형 잡혀 있는지, 불균형 상태인지를 계산하여 일정 척도에 따라 그에 해당하는 토큰을 사용자에게 제공한다. 사용자는 제공받은 토큰을 이용하여 상품(음식)을 구입하는데 포인트처럼 사용할 수 있다. 토큰은 Klaytn ERC-20 Token을 구현하여 사용하며, BalancedOne Smart Contract를 Klaytn Baobab Network에 올려 사용자에게 제공한다.

# System Requirements Specification

이 챕터에서는 유저 요구사항 명세에서 정의된 요구사항을 바탕으로 세부적인 Functional requirements와 Non-functional requirements를 명시한다. 각 requirements는 Name, description, inputs, source, outputs, destination, action, requirements, pre-condition, post-condition, side effects를 포함해 최대한 상세히 기술한다.

## Functional Requirements

### Login

|  |  |  |
| --- | --- | --- |
| Name | Login Function | |
| Description | 사용자 인증 및 권한 부여 | |
| Inputs | ID | Id string of user |
| Password | Password string of user |
| Source (Input) | 유저 입력 | |
| Outputs | 인증된 사용자 정보 개체 | |
| Destination (Output) | 사용자 단말(모바일 어플) | |
| Action | 유저가 입력한 ID를 사용해 데이터베이스에 저장된 유저 정보를 찾는다. 입력한 비밀번호와 데이터베이스에 저장된 해당 ID의 비밀번호를 비교한다. 일치하면, 저장된 다른 정보들을 포함한 유저 object를 생성해 어플리케이션으로 보낸다. | |
| Requirements | ID와 비밀번호가 모두 입력되고 일치해야 한다. | |
| Pre-condition | 로그인 및 회원가입 창을 띄운다. | |
| Post-condition | 어플의 모든 영역을 사용할 수 있다. | |
| Side effects | 로그인하지 않으면 어플을 사용할 수 없다. | |

Table 2 Login Function

### Sign up(User)

|  |  |  |
| --- | --- | --- |
| Name | User Sign up function | |
| Description | 일반 사용자 인증 | |
| Inputs | ID | Id string of user’s kakao account |
| Password | Password string of user’s kakao account |
| Source (Input) | 유저 입력 | |
| Outputs | 사용자 인증 및 사용자 권한 부여 | |
| Destination (Output) | 사용자 단말(모바일 어플), Database(인증된 사용자 정보) | |
| Action | 유저가 입력한 카카오 계정을 통해 카카오에서 인증된 사용자임을 확인 한 뒤, database에 ID와 Password를 저장한다. | |
| Requirements | 유효하고 인증된 kakao 계정의 ID와 비밀번호가 모두 필요하다. | |
| Pre-condition | - | |
| Post-condition | 유저의 로그인 권한이 생긴다. | |
| Side effects | 카카오 계정이 없으면 사용이 불가능하다.  카카오 계정을 확인하기 위한 카카오API가 필요하다. | |

Table 3 User Sign Up Function

### Sign up(Seller)

|  |  |  |
| --- | --- | --- |
| Name | Seller Sign up function | |
| Description | 판매자 인증 | |
| Inputs | ID | Id string of user’s kakao account |
| Password | Password string of user’s kakao account |
| Source (Input) | 유저 입력 | |
| Outputs | 사용자 인증 및 사용자 권한 부여 | |
| Destination (Output) | 사용자 단말(모바일 어플), Database(인증된 사용자 정보) | |
| Action | 유저가 입력한 카카오 계정을 통해 카카오에서 인증된 사용자임을 확인 한 뒤, database에 ID와 Password를 저장한다. | |
| Requirements | 유효하고 인증된 kakao 계정의 ID와 비밀번호가 모두 필요하다. | |
| Pre-condition | - | |
| Post-condition | 유저의 로그인 권한이 생긴다. | |
| Side effects | 카카오 계정이 없으면 사용이 불가능하다.  카카오 계정을 확인하기 위한 카카오API가 필요하다. | |

Table 4 Seller Sign Up Function

### Camera

|  |  |  |
| --- | --- | --- |
| Name | Camera function | |
| Description | 유저가 음식 사진 촬영한 후 데이터 전송 | |
| Inputs | Add Button push | 사용자가 Add button을 누른다. |
| Source (Input) | 핸드폰 카메라를 이용한 유저 입력 | |
| Outputs | 먹은 음식의 이미지 파일 | |
| Destination (Output) | Image Classification Machine learning | |
| Action | 핸드폰에 내장된 카메라가 켜지고, 유저가 음식을 촬영한다. 촬영한 이미지를 Image Classification 함수에 input으로 입력한다. | |
| Requirements | - | |
| Pre-condition | 유저가 음식 등록 버튼을 누른다. | |
| Post-condition | Image Classification 함수 실행 | |
| Side effects | Frontend will be changed into the chosen ranking list view. | |

Table 5 Camera Function

### Image Classification

|  |  |  |
| --- | --- | --- |
| Name | Image Classification Machine Function | |
| Description | 입력된 이미지를 이미지 분류 머신러닝 AI를 통해 음식 구분 | |
| Inputs | Food\_Pic | 먹을 음식의 사진 |
| Source (Input) | Camera Function | |
| Outputs | 음식 후보 이름들 | |
| Destination (Output) | 유저 단말기(모바일 어플) | |
| Action | 서버에서 유저가 전송한 이미지 파일을 찾아 Vision AI를 이용해 입력된 사진과 가장 유사한 음식 네 가지를 추출한다. 추출된 네 가지 음식을 다음에 이어질 Candidate Indicator에 input으로 입력한다. | |
| Requirements | 음식 이미지에 대한 방대한 데이터베이스 필요 | |
| Pre-condition | 유저가 카메라로 음식 사진을 촬영한다. | |
| Post-condition | Show Food Candidate List Function | |
| Side effects | 사진에 따라 음식이 정확하게 구분되지 않을 수 있다. | |

Table 6 Image Classification Function

### F. Show Food Candidate List

|  |  |  |
| --- | --- | --- |
| Name | Show Food Candidate List Function | |
| Description | 유저가 촬영한 사진의 음식 후보지 띄우기 | |
| Inputs | Candidate\_Name | Image Classification 결과 |
| Source (Input) | Image Classification Function, 사용자의 버튼 클릭으로 발생한 프론트엔드 요청 | |
| Outputs | 음식 이름 | |
| Destination (Output) | database | |
| Action | 이미지 분류 머신러닝 결과 네 가지의 음식 이름과 직접 입력 버튼을 화면에 띄운다. 유저가 고른 음식을 데이터 베이스 속 먹은 음식 데이터에 추가한다.  만약, 유저가 직접 입력을 고른다면, 음식 이름을 통해 데이터 베이스에서 검색 후 데이터를 추가한다. | |
| Requirements | - | |
| Pre-condition | Image Classification Function | |
| Post-condition | - | |
| Side effects | 직접 입력을 고른다면, 데이터베이스에서 직접 검색하는 새로운 UI가 필요하다. | |

Table 7 Show Food Candidate List

### Food Nutrition Database

|  |  |  |
| --- | --- | --- |
| Name | Food Nutrition Database | |
| Description | 개발자가 음식 영양소 정보 관리 | |
| Inputs | Food\_Name | String representing the name of the food |
| Food\_Cal | Floating number representing the calories |
| Food\_Pro | Floating number representing the protein |
| Food\_VitA | Floating number representing the Vitamin A |
| Food\_VitB | Floating number representing the Vitamin B |
| Food\_VitC | Floating number representing the Vitamin C |
| Food\_Cal | Floating number representing the Calcium |
| Food\_Zn | Floating number representing the Zinc |
| Source (Input) | 개발자 입력 | |
| Outputs | Food Object | |
| Destination (Output) | Database | |
| Action | 개발자가 음식의 이름과 해당 음식의 여덟 가지 영양소인 칼로리, 단백질, 비타민A, 비타민B, 비타민 C, 칼슘, 아연의 함유량을 입력한다. 입력된 정보는 Food Object로 묶여 저장된다. 미입력시 0으로 간주한다. | |
| Requirements | 정확한 영양 성분이 필요하다. | |

Table 8 Food Nutrition Database

### Food Nutrition Information

|  |  |  |
| --- | --- | --- |
| Name | Food Nutrition Information Function | |
| Description | 선택한 음식의 영양소를 표시한다. | |
| Inputs | Food Name | 음식 이름 |
| Source (Input) | 사용자의 버튼 클릭으로 발생한 프론트엔드 요청 | |
| Outputs | Food Object | |
| Destination (Output) | 사용자 단말 (모바일 어플) | |
| Action | 유저가 클릭한 음식 이름을 입력 받아 데이터 베이스에서 해당 음식의 object에 접근한다. 음식 object에서 영양 정보를 프론트 엔드로 가져와 유저에게 보여준다. | |
| Requirements | 유저의 음식 입력과 등록된 음식 영양 정보가 필요하다. | |
| Pre-condition | 유저가 음식 정보 확인을 위해 음식 이름을 클릭했다. | |
| Post-condition | 화면에 음식 정보가 뜬다. | |

Table 9 Food Nutrition Information Function

### Radar graph indicator

|  |  |  |
| --- | --- | --- |
| Name | Radar graph indicator | |
| Description | Show radar graph by calculating user body nutrition status | |
| Inputs | Body nutrition status | User body nutrition status which made by accumulating several food data before. |
| Source (Input) | Body nutrition status. | |
| Outputs | Radar graph indicator | |
| Destination (Output) | User terminal (application) | |
| Action | When the user clicks my page, backend calculates user body nutrition status and produce it as a radar graph. | |
| Requirements | User body nutrition should be specified | |
| Pre-condition | - | |
| Post-condition | - | |
| Side effects | - | |

Table 10 radar graph indicator

### Body status accumulator

|  |  |  |
| --- | --- | --- |
| Name | Body status accumulator | |
| Description | A function which accumulates body status by food nutrition information user going to eat. | |
| Inputs | food nutrition information | Food nutrition information. |
| Source (Input) | Food nutrition database | |
| Outputs |  | |
| Destination (Output) | Radar graph indicator, food recommendation. | |
| Action | It automatedly accumulate food nutrition information to user body nutrition status when the food photo is taken. . | |
| Requirements | Proper food information needed. | |
| Pre-condition | - | |
| Post-condition | - | |
| Side effects | Data it produces will be shown to user as a graph. Also, the recommendation algorithm will use this information. | |

Table 11 Body status accumulator

### Food recommendation

|  |  |  |
| --- | --- | --- |
| Name | Food recommendation | |
| Description | It recommends several food which compensate unbalanced body nutrition status. | |
| Inputs | food nutrition information | Food nutrition information. |
| Body nutrition status | User body nutrition status which made by accumulating several food data before. |
| Source (Input) | Food nutrition database  Body nutrition status | |
| Outputs | For you menu, food recommendation. | |
| Destination (Output) | For you menu, food recommendation. | |
| Action | It recommends several food which compensate unbalanced body nutrition status, using food recommend algorithm.. | |
| Requirements | Proper Body nutrition status required. | |
| Pre-condition | Body nutrition status should be made before. | |
| Post-condition | - | |
| Side effects | User can access order menu by accepting recommendation. | |

Table 12 Food recommendation

### Order menu

|  |  |  |
| --- | --- | --- |
| Name | Order menu | |
| Description | An order menu which lists several nutritional products supplied by seller. User can order listed food easily. | |
| Inputs | Product list | List of data containing product information. |
| Source (Input) | Product list | |
| Outputs | Order menu. | |
| Destination (Output) | Order menu. | |
| Action | It lists product information. User can easily buy the product by selecting what they need. | |
| Requirements | Products information should be uploaded by sellers | |
| Pre-condition | Products information should be uploaded by sellers | |
| Post-condition | - | |
| Side effects | - | |

Table 13 Order menu

### Product listing

|  |  |  |
| --- | --- | --- |
| Name | Product listing | |
| Description | Product should be listed in proper order. It may can be cost, nutrition and recentness. This part is required to show a readable list for order menu | |
| Inputs | Product information data | data containing product information. |
| Source (Input) | Product database | |
| Outputs | Order menu. | |
| Destination (Output) | Order menu. | |
| Action | It lists products information by several rule and order. | |
| Requirements | Products information should be uploaded by sellers | |
| Pre-condition | Products information should be uploaded by sellers | |
| Post-condition | - | |
| Side effects | - | |

Table 14 Product listing

### Add product

|  |  |  |
| --- | --- | --- |
| Name | Add product | |
| Description | A function adding product information descripted by sellers. | |
| Inputs | Product name | Name of the product. |
| Price | Price of the product |
| Product nutrition | Nutrition data of the product. |
| Source (Input) | Seller input | |
| Outputs | Product database | |
| Destination (Output) | Product database | |
| Action | It updates data written by seller to product database. | |
| Requirements | Proper information should be made by sellers | |
| Pre-condition | Proper information should be made by sellers | |
| Post-condition | -. | |
| Side effects | - | |

Table 15 Add product

### Product management

|  |  |  |
| --- | --- | --- |
| Name | Product management | |
| Description | Product information could be modified and deleted by seller made it before. Also system manager can get authority to delete the information if it contains unproper information. | |
| Inputs | Product information data | data containing modified product information. |
| Source (Input) | Seller input  System manager | |
| Outputs | Product database | |
| Destination (Output) | Product database | |
| Action | It updates modified data written by seller or system manager.  It also can be deleted by them.. | |
| Requirements | Proper information. | |
| Pre-condition | Proper information. | |
| Post-condition | - | |
| Side effects | - | |

Table 16 Product management

## Non-functional Requirements

### Product requirements

1. Usability

Usability는 성공적인 전자상거래 어플리케이션을 만드는데 필수적이다. 사용자의 편의를 위해서 직관적이고 간단한 UI(User Interface)를 제공하되, 원하는 기능을 실행하는데 까지 많은 행동을 요구해서는 안 된다. Balanced one은 심플한 디자인과 최소한의 수의 버튼과 탭을 구현하여 사용자가 불편함 없이 빠르게 다른 여러 섹션에 도달 할 수 있게 도와준다.

또한, 본 시스템은 모바일 어플로 접속하기 때문에 모바일에 맞춘 접속 환경을 제공해야 하며, 사용자가 쉽게 어플에 접근할 수 있게 해야 한다.

정보를 등록하고 음식을 추천 받고 구매하는 사용자와 판매할 음식을 등록하고 판매하는 셀러를 분리하며, 각 입장에 맞는 기능만을 제공해 불필요하게 복잡해지는 것을 막아야 한다.

1. Performance

유저가 입력한 사진으로 음식을 입력하는 기능에서 음식을 정확하게 구분해내는 기술이 필요하다. 그래서 유저가 한 번에 음식을 등록할 수 있도록 굉장히 많은 종류의 음식 이미지 데이터를 보유하고, 정확하게 구분해 내는 학습이 되어있는 Image Classification 기술이 필요하다.

또한, 음식 추천 알고리즘은 유저가 먹은 음식의 영양소를 정확하게 분석하고 이를 바탕으로 부족한 영양소를 포함한 음식을 추천해야 하며, 추천된 음식과 구매 기능까지의 연결이 매끄러워야 한다. 또한, 음식에 따른 판매자 목록을 제공하며 간단하게 구매를 완료할 수 있어야 한다.

1. Dependability

우리 프로젝트는 모든 action을 atomic하게 만들어 dependability를 최대화한다. 즉, 유저가 음식을 등록, 주문하거나, 셀러가 판매할 음식을 등록하는 중간에 시스템 충돌이 발생하면, 서버와의 연결이 끊어져 진행되면 모든 것들이 취소되며 진행 중이던 액션의 데이터가 사라진다. 이를 통해 asynchronous modification을 방지한다. 또한, 하드웨어 문제로 예기치 못한 상황에 어플리케이션이 종료되면, 유저가 접속 하자마자 모든 것을 원래대로 돌려 놓는다.

1. Security

본 시스템은 카카오 계정을 연동하여 로그인하기 때문에 사용자의 정보가 외부로 유출되면 안 되며 외부의 접근에 대한 방어가 필요하다. 또한, 회원 가입 당시의 정보 뿐만 아니라, 사용자가 시스템을 사용하면서 만들어지는 데이터가 저장되는 데이터베이스 역시 잘 관리되어야 한다. 마지막으로, 사용자의 결제 수단에 대한 보안도 잘 관리 되어야 한다.

### Organizational requirements

1. Operational

현재 지속적으로 운영할 수 있는 하드웨어 자원이 없으므로, 상용 클라우드 서버에 배포할 것을 전제로 한다. 따라서 다양한 클라우드 서버에 배포할 수 있도록 hardware-dependent한 구현을 자제하고, 서버의 주소가 바뀌어도 대응할 수 있게 시스템을 설계해야 한다.

1. Development  
   보다 빠른 기간에 시스템 개발이 가능하도록 Parallel Development를 도입하기 위해 개발 단위는 프론트엔드와 백엔드로 나눈다. 또한, 정확한 이미지 분석을 위해 Image Classification이라는 Machine Learning AI Technology인 Vision AI를 사용한다.

### External requirements

1. Regulatory Requirement

본 시스템은 사용자에게 로그인을 요구하기에 앞서 먼저 개인정보를 수집함에 대한 동의를 구하고, 취급방침에 대해 알려야 한다. 또한, 정부를 외부에 제공하게 될 경우엔 사용자에게 동의를 먼저 구해야 하며, 취급방침을 벗어나게 사용자의 정보를 사용해서는 안 된다.

1. Safety/Security Requirement  
   본 시스템은 음식 등록 기능 및 추천 기능을 제공하기 위해 회원제로 운영함에 따라 사용자의 개인정보를 수집한다. 이러한 개인정보가 유출되지 않도록 해야 하며, 결제 시스템에도 오류가 있으면 안 된다.

## Scenario Examples

### Login Scenario

1. Initial Assumption  
   The user has already created an account earlier and wants to login with the same information (Email and ID) he used for registration.
2. Normal flow of events  
   The user inputs his information and hits the login button. In the meantime, the database checks for matching information and finally the user can enter the platform with a customized interface according to his personal usage.
3. What can go wrong  
   One of the very obvious unprecedented scenarios will be if the user is not registered and the database. This thing will obviously generate an error msg.
4. System state on completion  
   The user is welcomed with his customized interface where he can see his information and filters of his preference.

### Searching food nutrition information Scenario

1. Initial Assumption  
   사용자는 자신이 섭취하고자 하는 음식의 영양분 정보를 확인하기 위해 본 시스템에 접속하였다. 사용자는 해당 음식의 이름은 정확히 알고 있지만 어떠한 영양분을 가지고 있는지는 알지 못한다.   
   또한 사용자는 본 시스템에 이미 등록하고 이전에 수 차례 서비스를 이용한 사용자이다.
2. Normal flow of events  
   사용자가 본 시스템에 접속해 로그인하면, 3개의 상위 메뉴가 표시된다. 사용자가 Add food 메뉴를 누르면, 스마트폰의 카메라 기능이 켜진다. 사용자는 카메라로 자신의 음식을 촬영한다. Image classification machine learning technology를 활용하여 시스템이 인식한 음식 후보들을 알 수 있다. 사용자는 이 후보들 중에서 자신의 음식을 선택한다. 시스템은 온라인 데이터 베이스에 접속하여 해당하는 음식의 영양정보를 가져와 가공해 사용자에게 보여준다.
3. What can go wrong  
   촬영된 음식을 시스템이 제대로 인식하지 못할 수도 있다.  
   사용자가 검색하고자 하는 음식이 데이터베이스에 저장되어 있지 않을 수 있다.
4. System state on completion  
   사용자가 검색한 음식이 섭취한 음식 기록에 남아 추후에도 다시 확인할 수 있다.   
   검색한 음식의 영양분 정보가 사용자의 몸 영양 상태 데이터에 추가된다.

### Checking body nutrition status Scenario

1. Initial Assumption  
   사용자는 이전에 섭취한 음식 정보를 바탕으로 계산된 자신의 몸의 영양 상태를 확인하기 위해 보 시스템에 본 시스템에 접속하였다.  
   또한 사용자는 본 시스템에 이미 등록하고 이전에 수 차례 서비스를 이용한 사용자이다.
2. Normal flow of events  
   사용자가 본 시스템에 접속해 로그인하면, 3개의 상위 메뉴가 표시된다. 사용자가 Mypage 메뉴를 누르면 radar graph형식으로 사용자의 신체 영양 정보를 사용자에게 보여준다. 이전에 섭취한 다양한 음식 사진들이 그 밑에 나열되어 있으며, 이 사진을 누르면서 이전에 먹은 음식 정보를 손쉽게 얻을 수 있다.
3. What can go wrong  
   이전에 시스템을 사용하지 않았을 경우 빈 정보를 가진 body status를 나타낼 수밖에 없다.   
   장기간 시스템을 사용하면 획득하 영양 정보의 오차가 작지 않게 생길 수 있다.
4. System state on completion  
   사용자의 body nutrition status를 활용하여 부족한 영양분을 보충할 수 있는 음식을 추천하는데 사용할 수 있다.

### Seller marketing Scenario

1. Initial Assumption  
   영양적으로 훌륭하거나 특정 영양소에 대해 보완 역할을 할 수 있는 음식 제품을 홍보하고 싶은 판매자는 본 시스템에 접속하여 자신의 제품을 등록할 수 있다.  
   판매자는 판매자용 계정을 사용하여 본 시스템에 로그인한다.
2. Normal flow of events  
   판매자가 판매자용 계정으로 본 시스템에 접속해 로그인하면, 판매자 전용 페이지가 보인다.  
   이 페이지에서는, 판매할 음식의 정보, 사진, 가격 등 판매에 필요한 모든 정보를 입력할 수 있다. 필요한 모든 정보를 입력하면 정상적으로 제품 등록이 완료되어 구매자가 손쉽게 물건을 구매할 수 있다.
3. What can go wrong  
   판매자가 판매자용 계정을 가지고 있지 않으면 시스템에 자신의 제품을 등록할 수 없다.
4. System state on completion  
   판매자의 제품이 등록되어 이를 원하는 일반 사용자가 손쉽게 물건을 구매할 수 있다.

# System Models

System Models 에서는 각 시스템 컴포넌트, 전체 시스템, 환경 사이의 관계를 서술한다.

## Context models

### Context Diagram

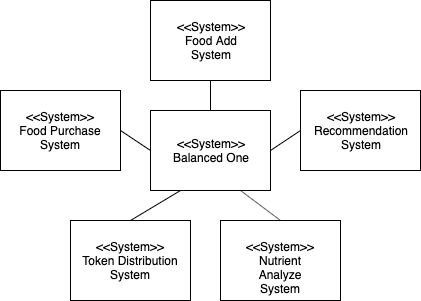


Figure 14 Overall context diagram

Context Diagram에서는 시스템의 경계를 정의하고 그 구성요소 및 외부와의 연동 관계를 나타낸다.

본 시스템에서는 메인 시스템으로 Balanced One 시스템이 동작하고 있으며, 서브 시스템으로 Food Add System, Recommendation System, Nutrient Analyze System, Token Distribution System, Food Purchase System이 있다.

### Process Diagram

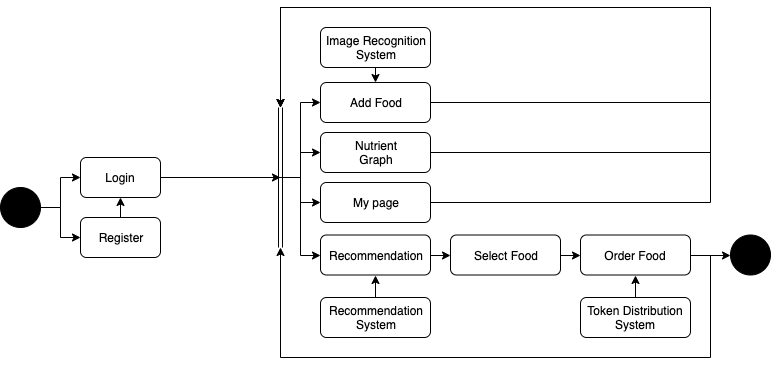


Figure 15 Overall process diagram

## 7.2. Interaction models

### Use case Diagram

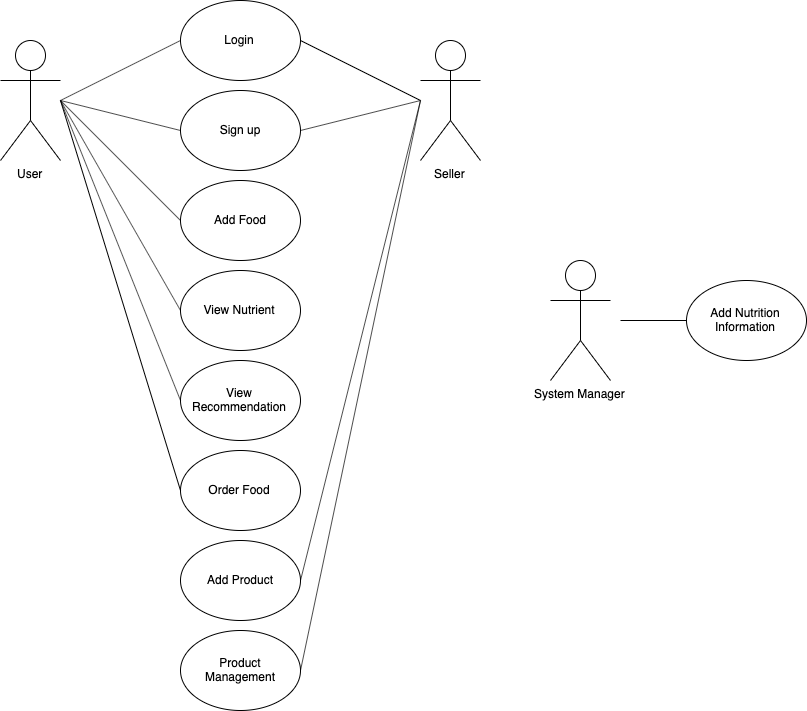


Figure 16 Use case diagram

### Tabular Description of Use case Diagram

1. Login

|  |  |
| --- | --- |
| Use case | Login |
| Actor | User, Seller |
| Description | 회원가입 여부 확인을 통해 사용자가 시스템을 이용할 수 있는 권한이 있는지 확인하고, 권한이 존재할 경우 세션을 통해 사용자에게 시스템 이용 권한을 부여한다. |
| Trigger | 사용자가 아이디와 비밀번호를 입력한다. |
| Success Response | 입력한 아이디가 데이터베이스에 존재하고, 입력한 비밀번호의 해시 값이 데이터베이스에 저장되어 있는 해당 아이디의 비밀번호 해시 값과 일치할 경우 세션을 반환한다. |
| Failure Response | 입력한 아이디가 데이터베이스에 존재하지 않거나, 비밀번호가 아이디와 일치하지 않을 경우 실패 메세지를 반환한다. |

Table 17 Login

1. Sign up

|  |  |
| --- | --- |
| Use case | Sign up |
| Actor | User, Seller |
| Description | 사용자가 시스템을 사용할 수 있도록 회원가입을 통해 사용자 정보를 시스템에 등록한다. 이는 카카오 API를 통해 이루어진다. |
| Trigger | 사용자가 카카오API 로그인을 통해 회원가입을 요청한다. |
| Success Response | 회원가입이 완료될 경우 완료 메세지를 반환하고 메인 화면으로 이동한다. |
| Failure Response | 이미 회원가입된 사용자거나, 카카오 API 로그인에 실패할 경우 실패 메세지를 반환한다. |

Table 18 Sign up

1. Add food

|  |  |
| --- | --- |
| Use case | Add food |
| Actor | User |
| Description | 유저가 먹고자 하는 음식의 영양 성분을 알고자 시스템에 접속한다. 사용자가 카메라를 이용해 해당 음식 촬영을 하면 시스템이 이를 인식하여 음식 영양정보 데이터 베이스에서 해당하는 정보를 가져와 사용자에게 보여준다. |
| Trigger | 사용자가 시스템에 접속해 Add food 메뉴에 들어간다. |
| Success Response | 시스템이 인식한 음식의 영양 정보를 사용자에게 보여준다.  이 음식을 사용자가 섭취했다고 가정하고 영양 정보를 사용자의 신체 영양 상태에 추가한다. |
| Failure Response | 인식할 음식의 정보를 찾을 수 없거나, 음식 인식이 올바르게 되지 않았으므로 Add food 메뉴 초기 상태로 돌아간다. |

Table 19 Add food

1. View nutrient

|  |  |
| --- | --- |
| Use case | View nutrient |
| Actor | User |
| Description | 사용자가 자신의 신체 영양상태를 알고자 함. |
| Trigger | 사용자가 시스템에 접속해 Mypage 메뉴를 들어 감 |
| Success Response | Database에 저장된 사용자의 신체 영양 상태 정보를 불러와 사용자에게 Radar graph를 이용해 보여줌. |
| Failure Response | 이전에 저장된 정보가 없을 경우 초기화된 정보를 보여줌.  올바른 정보를 불러오는데 실패할 경우 실패 메세지를 반환한다. |

Table 20 View nutrient

1. View recommendation

|  |  |
| --- | --- |
| Use case | View recommendation |
| Actor | User |
| Description | 사용자의 신체 영양 상태 중 부족한 부분을 참조하여 이를 보완할 수 있는 음식을 알고자 함. |
| Trigger | 사용자가 For you 메뉴에 들어감. |
| Success Response | Database에 저장된 사용자의 신체 영양 상태 정보를 불러와 부족한 부분을 보완할 수 있는 추천 음식을 보여줌. |
| Failure Response | 올바른 정보를 불러오는데 실패할 경우 실패 메세지를 반환한다. |

Table 21 View recommendation

1. Order Food

|  |  |
| --- | --- |
| Use case | Order Food |
| Actor | User |
| Description | 권한을 가진 사용자가 추천 받은 음식 중 선호하는 것을 고른다. 고른 음식을 판매하는 셀러의 리스트에서 가격 등의 정보를 확인하고 원하는 셀러를 골라 주문한다. |
| Trigger | 주문할 음식을 클릭한다. |
| Success Response | 유저가 권한을 가지고 있고, 해당 음식을 파는 셀러가 존재하면 음식이 주문된다. 유저의 주문 내용이 셀러의 데이터 베이스에 추가된다. |
| Failure Response | 선택한 음식을 하는 셀러가 존재하지 않는 경우 주문이 안 된다. |

Table 22 Order Food

7. Add Product

|  |  |
| --- | --- |
| Use case | Add Product |
| Actor | Seller |
| Description | 셀러가 음식을 팔기 위해 음식 이름, 가격 등 필요한 정보를 입력한 뒤 판매 리스트에 등록한다. |
| Trigger | 셀러가 판매 음식 등록 버튼을 클릭한다. |
| Success Response | 음식 등록이 완료된 경우, 입력한 정보가 데이터 베이스에 추가 되고, 판매 음식이 추가 되었음을 알린다. |
| Failure Response | 필수로 입력해야 하는 음식의 이름, 가격 정보가 입력되지 않는 경우 반환한다. |

Table 23 Add Product

1. Product Management

|  |  |
| --- | --- |
| Use case | Product Management |
| Actor | Seller |
| Description | 판매자가 본인이 등록한 상품의 정보를 수정한다. 상품의 이름, 설명, 이미지, 가격, 재고 수량 등을 수정할 수 있다. |
| Trigger | 수정할 상품을 선택하고, 이름, 설명, 이미지, 가격, 재고 수량 등 여러 항목 중 수정할 항목을 수정한 후 수정하기 버튼을 클릭한다. |
| Success Response | 수정이 완료될 경우 데이터베이스에서 값을 수정한 후 완료 메세지를 반환한다. |
| Failure Response | 수정하고자 하는 상품이 본인이 등록한 상품이 아니거나, 수정하고자 하는 항목이 올바르지 못하게 수정될 경우 실패 메세지를 반환한다. |

Table 24 Product Management

1. Add Nutrition Information

|  |  |
| --- | --- |
| Use case | Add Nutrition Information |
| Actor | System Manager |
| Description | 시스템 관리자가 새로운 음식에 대한 영양소 정보를 추가한다. |
| Trigger | 추가할 음식에 대한 정보를 입력한 후 추가하기 버튼을 클릭한다. |
| Success Response | 추가가 완료될 경우 데이터베이스에 정보를 추가하고 완료 메세지를 반환한다. |
| Failure Response | 추가하고자 하는 음식이 이미 데이터베이스에 있거나, 영양소 정보가 형식에 맞지 않게 입력된 경우 실패 메세지를 반환한다. |

Table 25 Add Nutrition Information

# System Evolution

System Evolution에서는 시스템을 둘러싼 환경과, 이를 배포하고 운영하며 발생할 수 있는 여러가지 변화를 예측하고, 이에 대한 대응 방안에 대해 기술한다. 이를 통해 추후 시스템의 수정 시 발생할 수 있는 설계 변경에 대한 비용을 절감한다.

### Augmented Reality (AR)

먹은 음식 입력의 간편화를 위하여, AR 기술을 적용하여 기존 시스템 방식대로 사진을 업로드 하는 것이 아닌 AR을 통해 바로 음식을 인식하고 업로드 할 수 있는 시스템을 구축한다.



Figure 17 AR example

### SNS Signup / Login

현재 시스템은 카카오 API를 통한 회원가입/로그인 만을 지원한다. 다양한 사용자에 대한 수용 및 해외 사용자에 대한 수용을 통해 네이버 API, 페이스북 API, 구글 API를 통한 로그인 또한 지원하도록 한다.

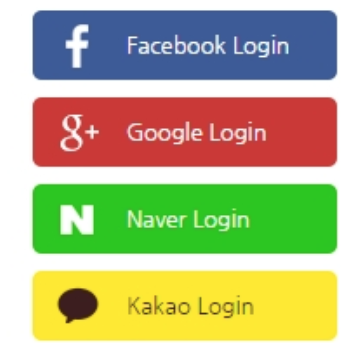


Figure 18 SNS signup/ login example

### Own Image Classification Model / Re-training System

현재 시스템에서는 개발에 대한 시간적인 한계를 이유로 Image Classification을 위한 Model을 직접 학습시키지 않고 Google Vision에서 사용하는 Pre-trained된 모델을 사용한다. 이 때문에 사용자가 업로드하는 사진을 추가한 재 학습이 불가능한데, 직접 Model을 구축하여 학습시키고, 일정 기간마다 사용자가 업로드하는 사진을 재 학습시킨다. 사용자가 사진을 찍는 구도, 색감 등은 사용자에 따라 일정한 패턴으로 특징성을 갖고 나타나는데, 이러한 재 학습을 통해 각각의 사용자에 대한 Image Recognition을 통한 결과를 향상시킬 수 있다.

# Appendices

Appendices에서는 시스템 개발에 관련된 세부 정보를 서술한다. 시스템 운영 환경에 필요한 하드웨어 요구사항, 소프트웨어 요구사항, 시스템에 사용되는 데이터베이스의 개략적 구조에 대해 서술한다.

## Hardware requirements

본 시스템은 안드로이드에서 동작하는 Native Application이다. 본 시스템은 React Framework를 이용하여 UI를 구현하는데, 이를 위하여 Android 4.1(API 16), Jellybean 이상 버전의 Android를 지원하는 모바일 하드웨어 기기가 필요하다.

또한 본 시스템은 기기 내부 DB를 사용하지 않고, REST API를 사용하여 서버를 통해 DB와 통신하므로 항상 네트워크에 연결되어 있어야 한다.

## Software requirements

본 시스템은 Node.js server를 통해 API Server 및 각종 System을 구축한다. 이 중 Token Distribution System에서 Klaytn Main Node와 통신하기 위하여 Caver.js라는 모듈을 사용하는데, 이를 위하여 v10의 Node.js가 설치되어야 한다.

## Database requirements

본 시스템은 RDBMS인 MySQL에 데이터를 저장한다. 데이터베이스를 구성하는 각 엔티티는 다음과 같다. 세부적인 데이터베이스 설계는 Design Specification에 기술한다

### User 사용자 정보를 저장하는 엔티티

### Product 구입 가능한 상품 정보를 저장하는 엔티티

### Food 음식의 영양소 정보를 저장하는 엔티티

### UserFood 각 유저가 섭취한 음식에 대한 정보를 저장하는 엔티티

### Recommendation 각 유저에 대한 추천 음식 정보를 저장하는 엔티티

# 10. Index

## 10.1. Tables

Table 1: Glossary…………………………………………………………………………………………………………15

Table 2: Login Function………………………………………………………………………………………………27

Table 3: User Sign Up Function………………………………………………………………………………….28

Table 4: Seller Sign Up Function………………………………………………………………………………..28

Table 5: Camera Function…………………………………………………………………………………………..29

Table 6: Image Classification Function………………………………………………………………………29

Table 7: Show Food Candidate List…………………………………………………………………………...30

Table 8: Food Nutrition Database………………………………………………………………………………31

Table 9: Food Nutrition Information Function…………………………………………………………..31

Table 10: radar graph indicator………………………………………………………………………………….32

Table 11: Body status accumulator…………………………………………………………………………….32

Table 12: Food recommendation………………………………………………………………………………..33

Table 13: Order menu……………………………..………………………………………………………………….33

Table 14: Product listing…………………………………………………………………………………………….34

Table 15: Add product………………………………………………………………………………………………..34

Table 16: Product management…………………………………………………………………………………35

Table 17: Login……………………………………………………………………………………………………………46

Table 18: Sign up………………………………………………………………………………………………………..46

Table 19: Add food……………………………………………………………………………………………………..47

Table 20: View nutrient……………………………………………………………………………………………….47

Table 21: View recommendation………………………………………………………………………………...47

Table 22: Order Food…………………………………………………………………………………………………..48

Table 23: Add Product…………………………………………………………………………………………………48

Table 24: Product Management………………………………………………………………………………….49

Table 24: Add Nutrition Information…………………………………………………………………………..49

## 10.2. Figures

Figure 1: 건강 기능식품 국내 시장 규모……………………………………………………………………10

Figure 2: 영양 섭취 부족자 분율 추이……………………………………………………………………….10

Figure 3: 카메라를 이용한 음식 인식 및 선택…………………………………………………………..12

Figure 4: 몸의 영양 상태 확인……………………………………………………………………………………12

Figure 5: 음식의 영양 정보 확인……………………………………………………………………………….12

Figure 6: Login process……………………………………………………………………………………………….16

Figure 7: Signup process…………………………………………………………………………………………….17

Figure 8: Add Food……………………………………………………………………………………………………..18

Figure 9: Recommendation…………………………………………………………………………………………18

Figure 10: Frontend Architecture………………………………………………………………………………..22

Figure 11: Backend Architecture…………………………………………………………………………………23

Figure 12: Recommendation System Architecture…………………………………………………….25

Figure 13: Token Distribution System Architecture…………………………………………………..26

Figure 14: Overall context diagram……………………………………………………………………………43

Figure 15: Overall process diagram……………………………………………………………………………44

Figure 16: Use case diagram………………………………………………………………………………………45

Figure 17: AR example………………………………………………………………………………………………..50

Figure 18: SNS signup/login example………………………………………………………………………..51

# 11. References

김영은. (2018년 05월 03일). “온라인쇼핑 거래액 추이.” 연합뉴스: https://www.yna.co.kr/view/GYH20180503001700044에서 검색됨

김윤미, 채선애, 송으뜸, & 윤덕환. (2017). “2017 소비자 리뷰 영향력 조사.” 리서치보고서.