

**Team 5**

**김동현**

**김주영**

**백지연**

**유대청**

**채호정**

**Contents**

1. Preface 5
   1. Objective 5
   2. Readership 5
      1. User Requirement Readership 5
      2. System Requirement Readership 5
   3. Document structure 5
      1. Preface 5
      2. Introduction 5
      3. Glossary 5
      4. User Requirement Definition 5
      5. Systemp Architecture 6
      6. System Requirement Specification 6
      7. System Models 6
      8. System Evolutions 6
      9. Appendices 6
      10. Index 6
2. Introduction 7
   1. needs 7
   2. overviews 8
   3. expected effects 9
3. Glossary 10
   1. Objective 10
   2. Term Definition 10
4. User Requirement Definition 11
   1. Objective 11
   2. Functional Requirements 11
   3. Non-Functional Requirements 13
5. System Architecture 14
   1. Objective 14
   2. Frontend Architecture 14
   3. Backend Architecture 16
   4. Recipe Registration System 16
   5. Recipe Recommendation System 17
   6. Recipe Searching System 17
   7. Refrigerator Management System 17
   8. Products Purchasing System 18
6. System requirement specification 18
   1. Objective 18
   2. Functional Requirement 19
      1. 회원가입 19
      2. 로그인 19
      3. 냉장고 재료 추가 20
      4. 냉장고 재료 제거 20
      5. 레시피 추천 20
      6. 레시피 검색 20
      7. 레시피 등록신청 21
      8. 레세피 등록 21
      9. 재료구매 21
   3. Non-functional Requirement 21
      1. Product Requirement 21
      2. Organizational Requirement 23
      3. External Requirement 23
   4. Scenario 24
      1. Sign up / Login 24
      2. 냉장고 관리 24
      3. 레시피 추천 25
      4. 레시피 검색 25
      5. 레시피 등록 26
      6. 재료 구매 26
7. System Models 27
   1. Objectives 27
   2. Context Models 27
      1. Context model 27
      2. Process Diagram 28
   3. Interaction Models 31
      1. Use case models 31
      2. Tabular description for each use case 32
      3. Sequence Diagram 35
   4. Structural Models 39
      1. User System Class Diagram 39
   5. Behavioral Models 40
      1. Data-Driven Modeling 40
      2. Event-Driven Modeling 42
8. System Evolutions 44
   1. Objective 44
   2. Limitation and Assumption 44
   3. Evolutions of Hardware 45
   4. Evolutions of User Requirement 45
   5. Evolutions of Environment 45
9. Appendices 47
   1. Database Requirements 47
10. Index 48
    1. Tables 48
    2. Diagrams 48

**1. Preface**

**1.1 Objective**

Preface에서는 이 문서의 전체적인 개요에 대해서 설명한다. 문서 형식, 독자층, 문서의 구조에 대한 설명으로 이루어져 있다.

**1.2 Readership**

이 문서는 독자에 따라 크게 2개의 분류 User Requirement와 System Requirement로 구성되어 있다.

**1.2.1. User Requirement Readership**

사용자 요구사항에서는 제공해야 할 시스템과 준수해야 하는 시스템 제약 사항에 관해 개괄적으로 기록한다. Non-functional requirements도 함께 설명한다. 자연어, 다이어그램 등으로 보다 독자들에게 쉽게 설명한다.

**1.2.2 System Requirement Readership**

시스템 요구사항에서는 시스템 개발자의 관점에서 프로젝트에서 개발해야 하는 시스템의 특정 기능들이 어떻게 구현되야 하는지 설명한다.

**1.3 Document structure**

**1.3.1. Preface**

Preface에서는 본 문서의 예상되는 독자들과 전반적인 구조, 그리고 각 부분의 역할에 대해 설명한다.

**1.3.2. Introduction**

Introduction에서는 본 시스템의 필요성과 해당 시스템이 반영하고 있는 needs를 설명한다. 또한, 주요 기능들과 이 기능들이 목표로 하는 소비자들에게 어떠한 효과를 줄 수 있는지 설명한다.

**1.3.3. Glossary**

Glossary에서는 본 문서에서 사용되는 기술적 용어들에 대해 정의한다. 전문 배경지식이 없는 독자들이 읽더라도 이해할 수 있도록, 가능한 한 모든 용어에 대해서 설명한다.

**1.3.4. User Requirement Definition**

User Requirements Definition에서는 본 문서에서 제시한 사용자에게 제공되는 서비스에 대해서 설명한다. 목표 시스템이 가지고 있는 non-functional requirements도 설명한다. 자연어와 다이어그램과 같은 방식들을 활용하여 독자들의 이해를 돕는다.

**1.3.5. System Architecture**

System Architecture에서는 본 프로젝트의 전반적인 시스템 구조에 대해서 설명한다.

**1.3.6. System Requirement Specification**

System Requirements Specification에서는 User Requirements Definition에서 분석한 Functional requirements와 Non-functional requirements에 대하여 보다 구체적인 서술을 제시한다. 각 기능 별로 예상되는 입력과 출력 및 처리 과정에 대하여 기술한다. 중심 기능들에 대한 Scenario를 작성하여 사용자와 개발자의 기능 이해를 돕는다.

**1.3.7. System Models**

System Models에서는 System Component, System Environment 등에 대한 도식적인 이해를 제공한다. 각 기능의 입력/출력 결과 및 처리 절차 혹은 데이터베이스와의 상호작용 과정 등에 대한 시각적 자료를 제시한다.

**1.3.8. System Evolutions**

System Evolution에서는 System에 대한 주요한 가정들에 대해 설명한다. 그리고System에 일어날 수 있는 예상되는 변화들에 대해 설명한다. 이 항목은 차후에 발생할 시스템의 설계 변경에 있을 Risk를 피할 수 있도록 하는데 도움이 된다.

**1.3.9. Appendices**

Appendices에서는 개발되는 시스템에 대한 더 자세하고 구체적인 정보들을 제공한다.

**1.3.10. Index**

Index에서는 본 문서에 포함된 시각 자료들에 대한 index가 적시된다.

**2. Introduction**

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

**2.1 needs**

스크린샷이(가) 표시된 사진

자동 생성된 설명스크린샷이(가) 표시된 사진

자동 생성된 설명다양한 미디어의 영향으로 인해 요리가 대중화되면서 가정, 개인에서 직접 음식을 요리해 먹는 문화가 자리 잡고 있다. 이러한 환경에 이용자가 냉장고에 가지고 있는 재료를 사용하여 레시피를 추천해주는 이밥차, 밥타임과 같은 시스템이 사용되고 있다. 밥타임은 냉장고에 있는 재료를 관리하고 일주일치의 식단을 스마트폰을 활용하여 언제, 어디서나 관리할 수 있다는 점을 어필하여 주부나 직접 요리를 하는 사람들에게 좋은 평가를 받으면서 사용되고 있다.

한편 신선식품을 배송하는 서비스가 늘어나면서 요리를 하는 사람들이 직접 장을 보지 않더라도 신선한 재료를 배송받고 요리할 수 있는 환경이 마련되고 있다. 마켓컬리, 신세계, 티몬 등 많은 업체에서 신선식품 배송사업에 뛰어들고 있으며 이 시장은 빠른 속도로 성장하고 있다.

스크린샷이(가) 표시된 사진

자동 생성된 설명

트럭, 도로, 하늘, 실외이(가) 표시된 사진

자동 생성된 설명텍스트이(가) 표시된 사진

자동 생성된 설명

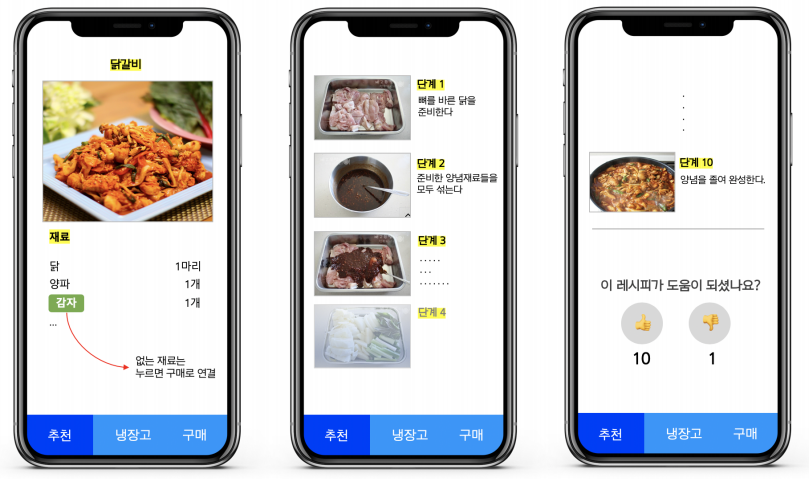
이러한 시장 상황에서 냉장고에 있는 재료를 실시간으로 관리하고 요리할 수 있는 레시피를 찾아보고 레시피에서 부족한 재료들을 즉시 구매하면, 신선식품 배송서비스를 통해 부족한 재료를 배송받고 냉장고에 자동을 재료가 추가되는 서비스를 제공한다면, 요리를 하고 싶지만 요리를 공부하거나 장을 볼 시간이 부족한 사람들이 효과적으로 사용하여 요리에 대한 접근성을 가지고 신선식품 소비를 촉진할 수 있는 서비스가 될 것이다.

**2.2 overviews**

‘냉모밀’은 이용자가 냉장고에 가지고 있는 재료들을 관리하고 가진 재료를 활용하여 만들 수 있는 레시피를 추천하고 이용자가 만들고 싶은 레시피를 검색할 수 있다. 이용자는 직접 레시피를 작성해 등록 할 수 있고 등록 신청된 레시피는 관리자의 검토를 거쳐 등록되게 된다. 이를 통해 완성도 있는 다양한 레시피를 이용자들에게 제공하고 레시피를 제공한 이용자들에게는  재료 구매시 활용 할 수 있는 포인트를 제공한다. 이 앱을 통해 재료를 구입하면 신선식품 배송 업체에 주문을 하고 주문한 재료들은 이용자의 냉장고정보에 업데이트되어 레시피 추천을 받는데에 사용될 수 있다.

스크린샷이(가) 표시된 사진

자동 생성된 설명



스크린샷이(가) 표시된 사진

자동 생성된 설명

**2.3 expected effects**

1) 요리에 대한 접근성 향상

요리와 관련된 미디어가 확산되고 요리에 대한 사람들의 관심이 커졌다. 하지만 요리에 관심을 가지더라도, 만들고 싶은 음식의 요리방법을 검색하고 필요한 음식 재료를 직접 사서 요리를 할 수 있는 시간이 부족한 사람도 있을 수 있다. 이에 레시피를 검색, 추천 받아 필요한 재료를 신선식품 배송서비스를 통해 배송 받을 수 있다면 시간이 부족하더라도 요리를 시도해 볼 수 있는 환경이 조성되어 요리에 대한 접근성이 향상되고 이는 관련 시장에 소비를 촉진하는 효과는 가질 수 있을 것이다.

2) 신선식품 배송 서비스 시장의 촉진

신선식품 배송 서비스는 이미 꾸준히 성장하고 있지만, 신선식품 배송에 대해 더 쉽게 접근 할 수 있는 서비스를 제공하여 신선식품 배송을 적극적으로 사용할 수 있는 환경을 만드는 데에 기여할 수 있을 것이다. 특히 국산 식품을 주로 다루는 배송 서비스와 제휴하여 국산식품에 대한 소비를 촉진하는데에도 기여할 수 있을 것이다.

3) 이용자들간의 정보 공유

이용자들이 서로 레시피를 공유하고 ‘좋아요’시스템을 이용하여 서로 선의의 경쟁을 하며 완성도있는 레시피를 작성하고 공유하여 요리에 대한 공통주제에 대한 정보를 공유하는 커뮤니티의 역할을 수행 할 수 있을 것이다.

**3. Glossary**

**3.1 Objective**

본 문서에서 기재되는 여러가지 기술적인 용어나 다 른 사람들에게는 생소할 수도 있는 용어들의 뜻을 정리한다. 배경지식이 없는 독자가 읽더라도 문서를 이해하는 데에 어려움이 없도록 하는 것이 그 취지이다 따라서 독자의 전문지식 수준을 추정하지 않고 가능한 한 쉽게 용어들을 풀어서 설명한다.

**3.2 Term Definition**

|  |  |
| --- | --- |
| **Term** | **Description** |
| 로그인 | 사용자가 개인 정보가 포함된 어플리케이션에 들어가기 위해 자신의 사용자명과 패스워드를 입력하는 것 |
| 프론트엔드 | 프로세스의 처음 단계를 의미하며, 사용자로부터 다양한 형태의 입력을 받아 백엔드가 사용할 수 있게끔 규격에 따라 처리하는 부분 |
| 백엔드 | 프로세스의 마지막 단계를 의미하며, 사용자는 알 필요가 없는 부분을 의미함. |
| 사용자 | ‘냉모밀’을 이용하는 모든 사용자 |
| 관리자 | ‘냉모밀’앱을 운영하고 관리하는 사람. |
| 레시피 | 요리의 조리법을 의미함 |
| 재료 | 레시피를 완성하기 위해서 필요한 모든 것들 |
| 냉장고 | 앱 내에서 재료들이 저장되고 관리되는 부분 |
| UI | 사용자와 컴퓨터가 정보를 주고받기 위해 사용자와 프로그램이 상호 작용하는 프로그램의 부분 |
| Controller | 시스템으로 들어오는 요청과 응답을 담당하는 부분 |
| 서버 | 요청 받은 서비스를 응답하는 프로세스 및 하드웨어와 관련된 서브시스템 |
| Database | 요청 받은 서비스를 제공할 수 있도록 통합하여 관리되는 데이터의 집합 |

**table 1: Glossary**

**4. User Requirement Definition**

**4.1. Objective**

프로세스동안 만들어지는 시스템 서비스와 제약사항들 중, 서비스와, 해당 서비스가 제공하는 운영 제약사항들을 자연어와 다이어그램으로 기술한다.

**4.2. Functional Requirements**

Functional Requirements로 아래와 같은 기능을 요구한다.

**1) Sign up**

User가 어플리케이션의 사용을 하기위해 회원가입이 필요하며, 그 회원가입의 절차를 진행하는 기능이다. 회원가입 정보로는 아이디, 패스워드, 그리고 어플리케이션 내에서 사용되는 닉네임이 있다.

**2) Log in**

User의 회원가입을 통해 계정이 생성되고, 생성된 계정을 이용하여 어플리케이션 시스템에 로그인하는 기능이다. 로그인 과정을 통해 로그인한 계정의 데이터베이스로부터 회원정보 및 데이터를 불러온다.

유효하지 않는 계정에 접근하거나 올바르지 않은 패스워드를 입력할 경우 그에 맞는 메시지를 User에게 전달한다.

**3) Refrigerator's Ingredients survey**

User가 현재 보유하고 있는 재료에 대해 확인하는 기능이다. Survey로부터 얻은 User의 보유하고 있는 재료에 대한 Data는 User 계정의 Database에 추가된다.

**4) Taste survey**

이후에 User가 레시피를 검색하거나 시스템이 User에게 레시피를 추천을 하는데 있어서 필터링 요소를 선택하는 기능이다. User의 기호에 따라 음식의 맵기의 정도, 제외해야 할 재료, 그리고 음식유형(한식, 중식, 일식 그리고 양식)을 선택할 수 있다.

**5) Recipe Post**

User가 레시피에 관한 정보(레시피 이름, 재료, 요리법, 음식 이미지 등)을 업로드하는 기능이다. User가 입력한 레시피 이름은 불용어 제거 알고리즘에 의해 표준형식으로 변환된 Data로 시스템에 전달된다.

**6) Recipe Searching & Rating**

User가 원하는 레시피를 검색하는 기능이다. User가 입력한 키워드에 해당하는 모든 레시피를 찾고, Refrigerator's Ingredients survey 와 Taste survey로 수집한 User data를 이용하여 Prioritization한 후 최종적으로 User에게 정보가 전달된다. 또한 User는 검색한 레시피에 대해 평가할 수 있으며, User의 평가는 레시피의 Rating Data에 적용된다.

**7) Recipe Recommendation**

User는 검색한 Refrigerator's Ingredients에서 사용하고자 하는 재료를 선택할 수 있고, 그 재료들을 사용하여 만들 수 있는 레시피를 추천해주는 기능이다. User가 입력한 키워드에 해당하는 모든 레시피를 찾고, Refrigerator's Ingredients survey 와 Taste survey로 수집한 User data를 이용하여 Prioritization한 후 최종적으로 User에게 정보가 전달된다.

**8) Shopping**

User는 원하는 재료를 구매하는 기능이다. User가 구매하고자 하는 재료를 선택한 후 구매를 요청하면 시스템은 구매에 대한 정보를 User에게 전달한다. 구매가 확정되면 구매한 재료의 정보가 Refrigerator's Ingredients Data에 적용된다.

**4.3. Non-Functional Requirements**

Non-Functional Requirements로 아래와 같은 기능을 요구한다.

**1) Product Requirement**

- Performance : User가 레시피를 검색할 경우, User에게 전달 되는 Data는 User의 필요성에 부합해야 한다. User가 레시피를 Posting하는 과정에서 User가 입력한 Data의 누락이 없어야 한다. Shopping 단계에서 구매 재료에 대한 Data에 오류가 없어야 하며, 구매 완료한 재료에 대한 Data가 Refrigerator's Ingredients에 올바르게 적용되어야 한다.

- Security : Sign up 및 Log in을 하는 과정에서 User의 개인정보가 사용되며, User의 개인정보는 유출되면 안되며, Security를 유지하기 위한 시스템적인 기능이 요구된다.

- Efficiency : User가 레시피를 검색할 경우, 검색한 키워드에 일치하는 모든 데이터가 User Data에 의해 정렬되어야 한다. 이때 정렬되는데 소요되는 시간이 짧아야 한다. 검색하는 시점과 User에게 정보가 전달되는 시점 사이의 Delay를 최소화하는 것이 관건이다.

User가 Posting을 하는 경우, User가 입력한 데이터를 시스템의 표준형식에 맞게 변환하는 시간을 단축해야 한다.

-Dependability : User의 Input에 대해 시스템은 올바른 Output을 전달해야 한다. User의 request가 Server에 전달되는데 오류가 없어야 한다. Server로부터 User로 Data가 전송되는데 오류가 없어야 한다.

- Usability : 사용자가 이용하기에 편한 UI를 사용한다. User의 Usability를 높이기 위해 간단하고, 가시성이 높은 UI를 디자인한다.

**2) Organizational Requirement**

- Environment : 안드로이드 스튜디오를 사용한 어플리케이션으로, 코드의 성능이나 버전 호환성 및 기타 문제점을 잡아내는 Lint의 기능에 최적화 되었다. 또한 구글 클라우드 플랫폼을 자체적으로 지원하여 구글 클라우드 메시징/앱 엔진과 쉽게 통합할 수 있다.

- Operation : 시스템은 User의 Taste 와 Refrigerator's Ingredients에 부합하는 정보를 User에게 전달해야 하며, 전달된 정보는 올바른 Prioritization을 통해 User가 사용하기 쉽게 Sorting되어야 한다.

**3) External Requirement**

- Safety / Security : Sign up를 통하여 생성된 User들의 개인정보 Data를 관리하며, 타인으로의 Data 유출과 개인 Data에 대한 악의적인 접근을 방지한다. Shopping 단계에서 User의 Credit Data의 유출을 막고, 결제단계에서 발생할 수 있는 오류를 차단한다.

**5. System Architecture**

**5.1. Objective**

이번 챕터에서는 시스템의 개괄적인 Architecture를 기술한다. 시스템의 전체적인 구조와 각 서브시스템의 구성, 서브시스템간 관계를 대략적으로 설명하며, 각 구조는 다이어그램을 첨부하여 이해를 돕는다.

**5.2. Frontend Architecture**

**스크린샷이(가) 표시된 사진

자동 생성된 설명**

**diagram1: Frontend Architecture**

유저가 직접적으로 이용하는 Frontend인 앱은 크게 4개의 페이지들로 이루어진다.

냉장고 관리, 레시피 검색 및 추천, 레시피 등록, 재료 구매 페이지로 이루어져 있으며, 각 페이지는 서버를 통해 데이터를 가져와 보여주고 생성 또는 수정된 데이터들을 서버로 보내 데이터베이스에 저장한다.

**5.3. Backend Architecture**

스크린샷이(가) 표시된 사진

자동 생성된 설명

**diagram 2: Backend Architecture**

백엔드 시스템은 위와 같이 구성되어 있으며, 프론트엔드로부터 받은 요청을 Controller가 처리한다. 필요한 경우 서비스 핸들러를 통해, 구매할 수 있는 상품 목록을 업데이트 한다. 처리된 요청들은 요청의 종류에 따라 데이터베이스에 저장되거나 프론트엔드인 앱으로 보내어 사용자가 볼 수 있게 한다.

데이터베이스는 사용자의 정보(냉장고 정보, 닉네임 등), 허가된 레시피, 아직 허가되지 않은 레시피를 저장하는 총 3개의 데이터베이스로 이루어져 있다.

**5.4. Recipe Registration System**

스크린샷이(가) 표시된 사진

자동 생성된 설명

**diagram 3: Recipe Registration System**

레시피 등록 시스템은, 유저는 레시피는 등록, 관리자는 레시피 등록을 허가/거부할 수 있게 하는 시스템이다. 유저가 등록을 하면 Unallowed recipe database에 레시피가 등록이 되고, 관리자는 검사 시, unallowed recipe database를 읽어와, 레시피 등록여부를 판단할 수 있다. 판단 후엔 unallowed recipe database에서 삭제 후, 등록할 만한 레시피면 allowed recipe에 레시피를 추가한다.

**5.5. Recipe Recommendation System**

**스크린샷이(가) 표시된 사진

자동 생성된 설명**

**diagram 4: Recipe Recommendation System**

레시피 추천시스템은, 유저가 자신의 냉장고 정보를 추천을 위해 입력하면, 해당 정보를 바탕으로 allowed recipe내의 레시피들을 필터링, 정렬하여 사용자에게 보여주는 시스템이다.

**5.6. Recipe Searching System**

스크린샷이(가) 표시된 사진

자동 생성된 설명

**diagram 5: Recipe Searching System**

레시피 탐색 시스템은, 유저가 찾고 싶은 레시피의 이름을 입력하면 해당 이름을 포함하는 레시피들을 모두 찾아서 유저에게 좋아요 순으로 보여준다.

**5.7. Refrigerator Management System**

**스크린샷이(가) 표시된 사진

자동 생성된 설명**

**diagram 6: Refrigerator Management System**

냉장고 관리 시스템은, 사용자가 냉장고에서 재료를 삭제/추가할 때 마다 데이터베이스의 User info에 업데이트 후, 업데이트 된 결과를 사용자에게 보여주는 시스템이다.

**5.8. Products Purchasing System**

**스크린샷이(가) 표시된 사진

자동 생성된 설명**

**diagram 7: Products Purchasing System**

상품 구매 시스템은, 사용자가 구매할 물건들을 선택하면 결제 시스템에서 결제 후, 결제가 완료된다면 사용자의 냉장고에 구매한 상품들 정보를 업데이트한 후, 사용자에게 성공적으로 구매가 되었다는 메시지를 보내주는 시스템이다.

**6. System requirement specification**

**6.1 Objective**

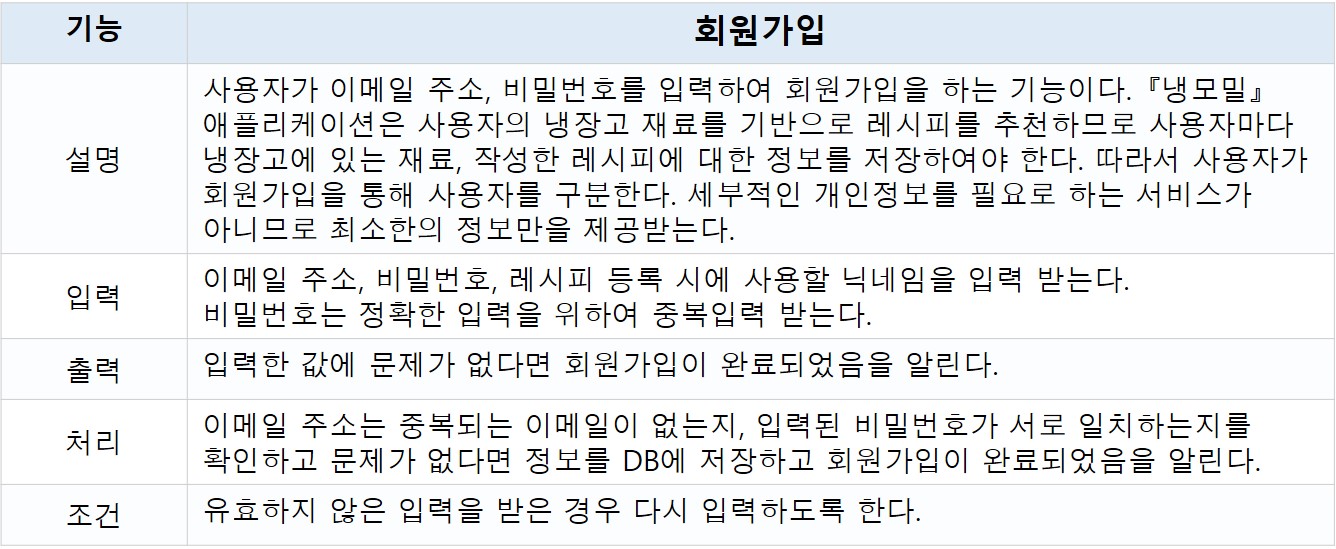
시스템 요구사항 명세서에서는 시스템이 요구하는 기능적 요구사항과 비기능적 요구사항에 대해 상세히 기술한다.

기능적 요구사항에서는 시스템에 요구되는 각 기능에 대한 설명과 입력/출력 받는 정보, 처리과정, 조건사항에 대해 상세히 기술한다.

비기능적 요구사항에서는 product requirement, organizational requirement, external requirement를 중심으로 시스템에 요구되는 비기능적 요구사항을 상세히기술한다.

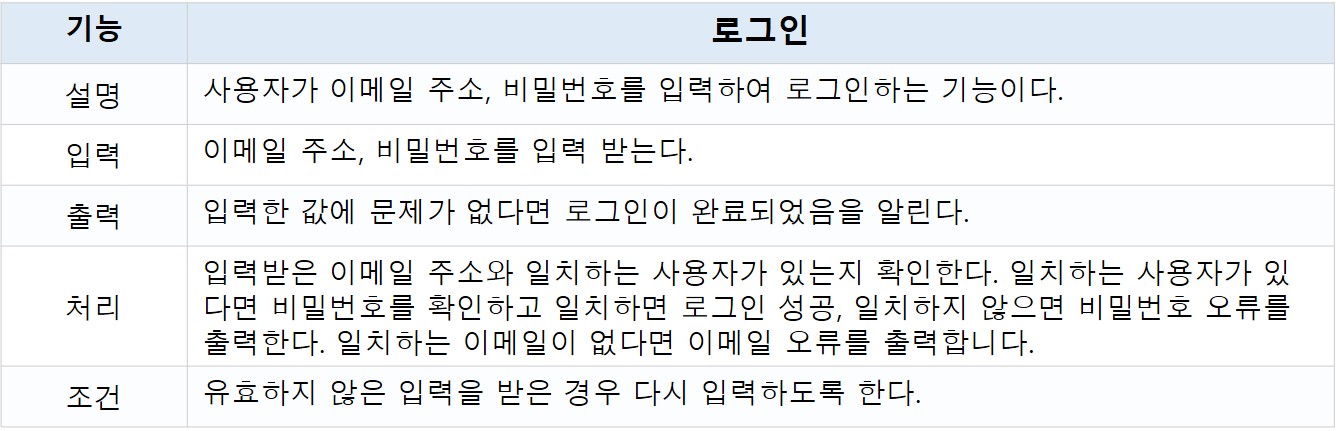
**6.2 Functional Requirement**

**6.2.1 회원가입**



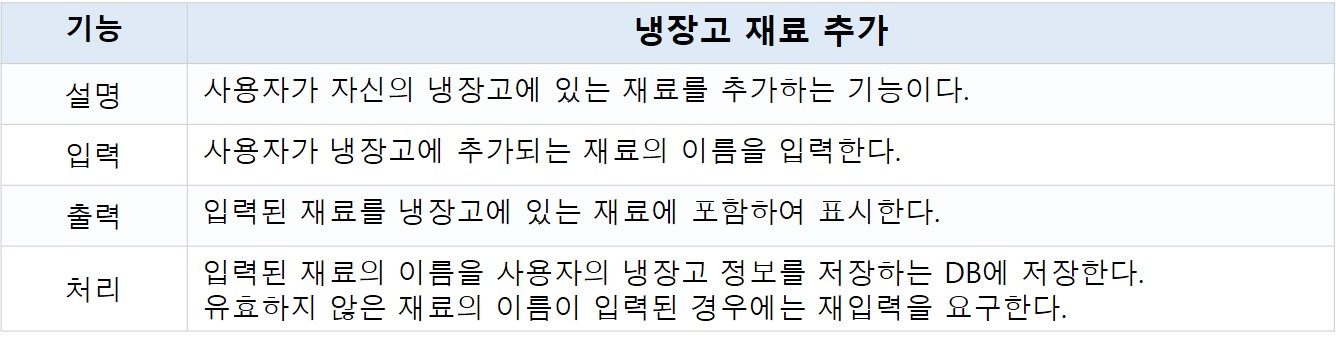
**table 2: 회원가입 Functional Requirement**

**6.2.2 로그인**



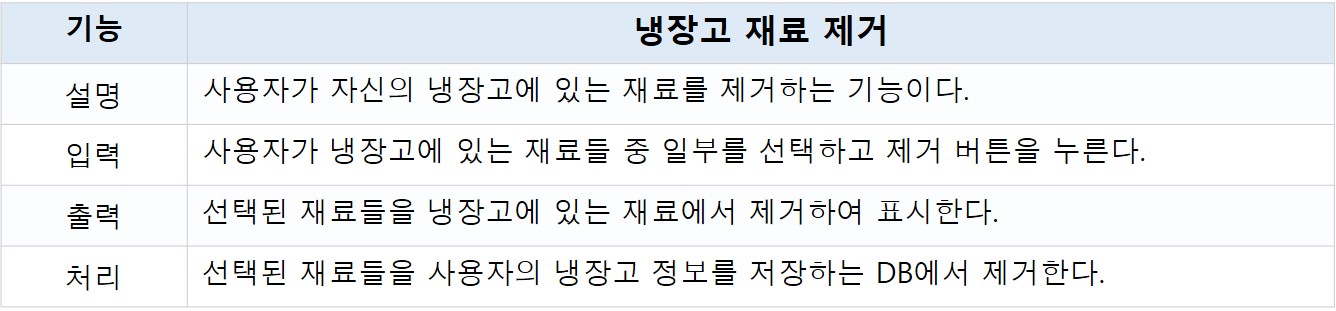
**table 3: 로그인 Functional Requirement**

**6.2.3 냉장고 재료 추가**



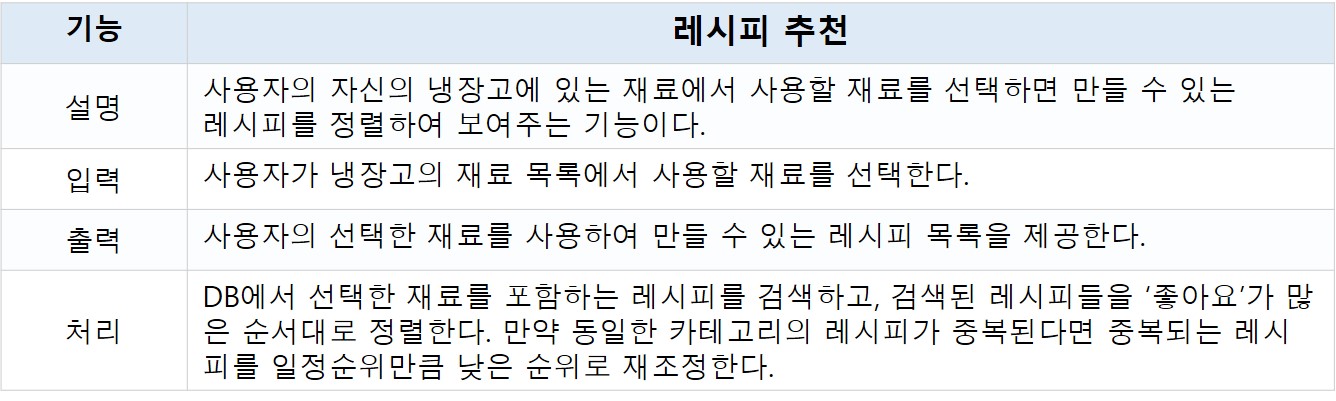
**table 4: 냉장고 재료 Functional Requirement**

**6.2.4 냉장고 재료 제거**



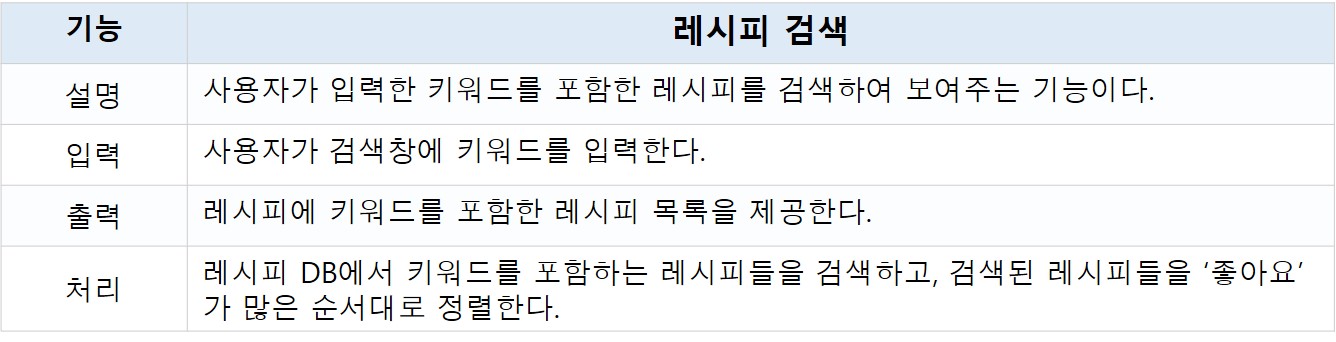
**table 5: 냉장고 재료 제거 Functional Requirement**

**6.2.5 레시피 추천**



**table 6: 레시피 추천 Functional Requirement**

**6.2.6 레시피 검색**



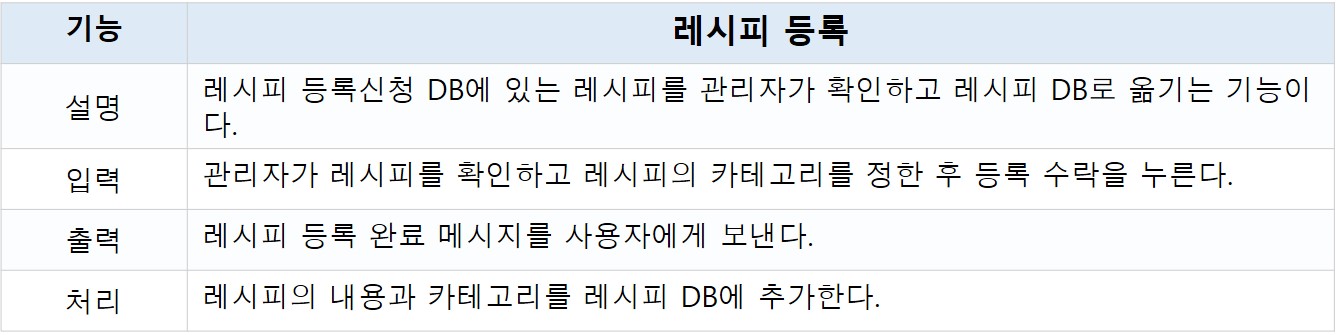
**table 7: 레시피 검색 Functional Requirement**

**6.2.7 레시피 등록신청**



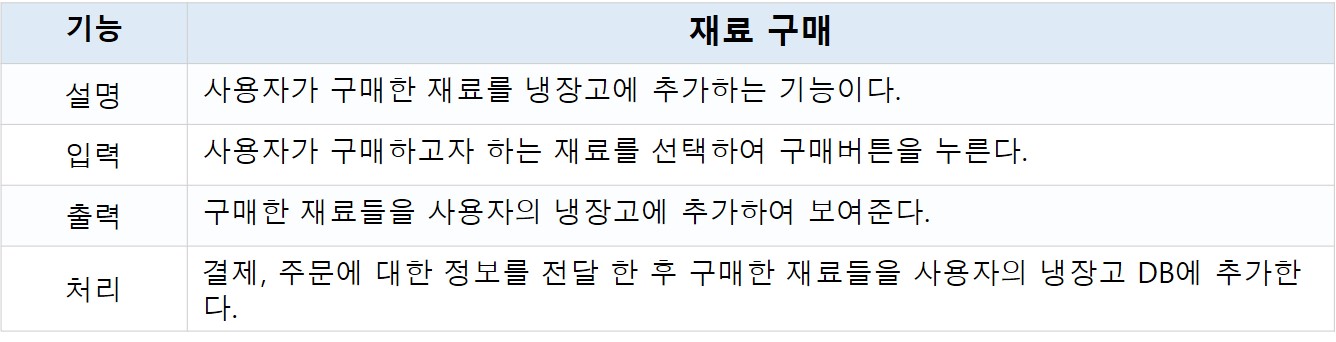
**table 8: 레시피 등록 신청 Functional Requirement**

**6.2.8 레시피 등록**



**table 9: 레시피 등록 Functional Requirement**

**6.2.9 재료구매**



**table 10: 재료 구매 Functional Requirement**

**6.3 Non-functional Requirement**

**6.3.1 Product Requirement**

**A. Performance**

시스템의 모든 기능은 입력과 동시에 DB를 검색, 입력하고 빠르게 사용자의 화면에 반영되어야 한다. 특히 레시피 추천, 레시피 검색에서의 검색, 정렬기능에서 빠르게 결과를 보여주어야 한다.

사용자가 냉장고 정보에 재료를 입력하면 냉장고 DB에 저장하고, 저장된 정보는 즉시 사용자의 화면에 반영되어야 한다. 레시피 추천기능을 사용하면 사용자가 선택한 재료를 사용하여 만들수 있는 레시피를 검색하여 ‘좋아요’ 갯수 순으로 정렬한 후 제공한다. 그 후 일정범위내에 같은 카테고리의 레시피가 연속해서 나온다면 범위를 벗어나도록 정렬 순위를 미룬다. 검색과 정렬과정을 효율적으로 하여 레시피 추천과 레시피 검색 기능이 빠르게 사용자에게 결과를 제공할 수 있어야 한다. 레시피 등록신청과 레시피 등록은 관리자 승인시 즉시 반영되어 레시피 추천, 레시피 검색 기능으로 접근가능하여야 한다. 구매한 재료도 구매와 동시에 냉장고 재료에 반영되어야 한다.

**B. Security**

사용자의 계정에 사용자의 냉장고 정보와 등록한 레시피가 연동되어 있기 때문에 계정 정보는 보호되어야 한다. 회원가입 시 동일한 이메일로는 가입할 수 없도록 하고 사용자의 비밀번호는 암호화해 관리하여 관리자도 확인할 수 없도록 한다. 레시피 등록 시 노출되는 것은 닉네임으로 사용자의 이메일도 노출되지 않도록 한다. (비밀번호 찾기 기능은 등록된 이메일 주소에 비밀번호를 보낸다.) 구매한 정보는 기록하지 않고 냉장고에 반영되도록만 한다.

**C. Efficiency**

시스템에서 가장 중요한 기능은 DB에서 레시피 검색과 정렬하는 것으로 이 시간을 줄이기 위해 정렬을 효율적으로 하는 것이 중요하다. 레시피는 ‘좋아요’가 많은 순으로 정렬한 상태로 저장하고 ‘좋아요’가 변경될 때마다 레시피를 정렬하여 기능을 실행하였을 때 검색과 정렬을 효율적으로 한다.

**D. Dependability**

시스템은 사용자가 사용하고자 하는 재료를 포함한 신뢰가능한 레시피를 제공해주어야 한다. 사용자들이 어느정도의 완성도를 갖춘 레시피를 제공할 수 있도록 레시피 등록과정을 관리자가 관리한다. (레시피 등록을 하는 사람들에게도 구매시 활용할 수 있는 포인트를 제공하여 양질의 레시피를 제공하도록 한다.)

**E. Usability**

시스템은 간단하고 직관적인 UI를 제공하여 사용자들이 쉽게 재료를 추가/제거하고 원하는 재료를 사용하는 레시피를 찾기 쉽도록 하여 서비스를 이용하는 것에 불편함이 없도록 한다. 직관적인 탭 이동과 버튼 배치 등을 통해 시스템의 기능을 사용할 수 있도록 한다.

**6.3.2 Organizational Requirement**

**A. Environment**

시스템은 안드로이드에서 사용할 수 있는 애플리케이션의 형태로 개발되어, 언제 어디에서나 쉽게 접근하고 사용할 수 있도록 한다.

**B. Operation**

시스템은 사용자들이 실제 음식을 만드는 과정에서 사용할 수 있도록 다양한 재료를 활용한 레시피를 수집하여 제공하고 사용자들도 레시피를 등록하므로서 그 다양성을 확보한다. 또한 완성도있는 레시피를 선별하여 등록하는 것으로 사용자가 보고 바로 따라 할 수 있는 레시피를 제공한다. 사용자 정보가 저장된 DB와 레시피 DB를 보안을 유지하면서 효율성있게 관리한다.

**6.3.3 External Requirement**

**A. Compatibility**

서비스는 구매하고자 하는 재료들을 외부의 신선식품 배달 서비스와 연동하여 주문할 수 있어야 한다. 이 과정에서 재료의 정보가 오류없이 냉장고 정보로 저장될 수 있어야 하며, 즉시 주문과정이 이루어 질 수 있어야 한다.

**B. Security/Safety**

서비스는 사용자 개인의 데이터를 관리하고 있으므로 이 정보에 대한 악의적인 접근 및 활용의 가능성을 차단하여야 한다. (또한 전자 결제 시스템의 경우 보안성과 신뢰성이 높은 외부 시스템을 사용하여 고객의 결제 정보가 유출되지 않도록 한다. 전자 결제 시스템이 요구하는 정보를 구매자에게 요청하고, 이를 전자 결제 시스템의 형식에 맞게 보내어 오류가 발생하지 않도록 한다.

**6.4. Scenario**

**6.4.1. Sign up / Login**

**A. Initial Assumption**

개인 냉장고 관리, 개인 레시피 등록 등을 위해서는 이용자가 로그인을 해서 데이터베이스에 따로 관리할 필요가 있다.

**B. Normal Flow of Events**

어플에 접속하면 로그인 페이지를 먼저 보게 된다. 만약 가입하지 않았다면 sign-up 페이지로 이동한다. 아이디로 사용할 email 주소, password, 그리고 닉네임을 입력하고 가입을 할 수 있다. 가입하면 상기한 정보들이 데이터베이스로 들어오게 된다. 로그인 페이지에서는 ID와 PASSWORD를 입력하면, 데이터베이스에서 이용자의 정보를 확인하고 로그인 시켜준다.

**C. What can go wrong & concurrent activities**

Sign-up 할 때, email 주소와 닉네임이 데이터베이스에 이미 존재하는 것들과 겹쳐지지 않도록 확인할 필요가 있다. 또한 password에 관한 보안 이슈가 있을 수 있다.

그리고 로그인 시, 이용자의 개인 냉장고 정보를 바르게 제공할 수 있어야 한다.

**D. System State on Completion**

Sign-up과 Login이 정상적으로 완료되면, 이용자 개인 냉장고 관리 페이지로 넘어가 자신의 냉장고 상황을 확인할 수 있다.

* + 1. **냉장고 관리**

**A. Initial Assumption**

각 이용자는 자신의 냉장고 속의 재료들을 관리할 수 있다. .

**B. Normal Flow of Events**

이용자가 정상적으로 로그인 했다면, 자신의 냉장고 정보를 데이터베이스에서 받아와 화면 상으로 확인할 수 있다. 이용자가 이미 사용하여 냉장고 속에 더 이상 없는 재료가 있다면 삭제할 수 있고, 이용자가 구입하여 냉장고 속에 생긴 재료가 있다면 추가할 수도 있다.

**C. What can go wrong & concurrent activities**

이용자의 냉장고 정보를 오류없이 데이터베이스에서 가져올 수 있어야 한다.

이용자의 재료 삭제/추가에 대해 데이터베이스에 정보를 바로바로 넘길 수 있어야 한다.

이용자가 추가하려는 재료명을 데이터베이스 속에 꼭 가지고 있어야 한다.

**D. System State on Completion**

이용자가 생각하는 냉장고 속의 상황이 어플 속 냉장고 내용으로 정확히 반영된다.

**6.4.3. 레시피 추천**

**A. Initial Assumption**

이용자는 자신의 냉장고 속에 있는 재료들을 활용하여 만들 수 있는 레시피는 무엇이 있는지 알고 싶어 한다.

**B. Normal Flow of Events**

이용자는 자신의 냉장고 속에 있는 재료들 중에서 활용하고 싶은 재료와 활용하고 싶지 않은 재료를 선택할 수 있다. 활용하고 싶은 재료를 기반으로 데이터베이스에서 적절한 레시피를 찾고 그 중에서 활용하고 싶지 않은 재료를 갖고 있는 레시피를 제외하여 이용자에게 레시피를 추천해준다.

**C. What can go wrong & concurrent activities**

이용자가 선택하는 재료가 많아 질수록 검색에 있어서 어려움이 나타날 수 있다. 이용자의 요구에 맡은 레시피를 찾을 수 없을 수 있다.

**D. System State on Completion**

이용자가 선택한 재료들을 기반으로 추천된 레시피들을 유저가 확인할 수 있다.

**6.4.4. 레시피 검색**

**A. Initial Assumption**

이용자가 가진 재료를 기반으로 레시피를 추천받는 것이 아니라, 이용자가 원하는 음식이 따로 있을 수 있다.

**B. Normal Flow of Events**

이용자가 원하는 음식을 검색창에 입력한다. 해당 음식에 대한 레시피들을 데이터베이스에서 가져온다. 해당 레시피들은 이용자들에게 좋은 평가를 받은 레시피 일수록 상위에 노출된다.

**C. What can go wrong & concurrent activities**

이용자가 원하는 음식에 대한 레시피가 아예 없을 수도 있다.

**D. System State on Completion**

이용자가 원하는 음식에 대한 다양한 레시피들을 이용자가 접할 수 있다.

**6.4.5. 레시피 등록**

**A. Initial Assumption**

이용자들도 자신의 레시피를 등록하고 알리고 싶을 수 있다.

**B. Normal Flow of Events**

이용자가 레시피 등록 버튼을 누르면 해당 페이지로 이동한다. 거기서 주어진 양식에 따라서 레시피를 입력하여 등록할 수 있다. 등록된 레시피는 바로 이용자들에게 보이지 않고, 관리자의 확인을 거친다.

**C. What can go wrong & concurrent activities**

지속적인 관리자의 관리가 필요하다. 비슷한 레시피에 대한 대책이 필요할 수 있다.

**D. System State on Completion**

이용자가 레시피를 등록하고 관리자가 확인한다. 등록된 레시피를 다른 유저들에게 평가받을 수 있다.

**6.4.6. 재료 구매**

**A. Initial Assumption**

당일배송, 새벽배송 가능한 온라인 업체(ssg.com)에서 재료를 구매할 수 있다.

**B. Normal Flow of Events**

이용자가 자신이 만들고 싶은 레시피를 찾았을 때, 이용자의 냉장고에 없는 재료가 있다면 장바구니에 바로 담고 구매할 수 있다.

이용자가 원하는 재료를 검색하여 장바구니에 담을 수도 있다.

장바구니에서 구매가 완료되면 이용자의 냉장고 속으로 재료가 자동으로 들어간다.

**C. What can go wrong & concurrent activities**

온라인 업체와 현재 연결이 되어 있지 않다. 이용자가 원하는 재료를 찾을 수 없을 수도 있다.

**D. System State on Completion**

이용자가 구입을 원하는 재료를 모두 장바구니에 담고 구매를 할 수 있다. 구매한 재료는 이용자의 냉장고 재료로 자동으로 입력된다.

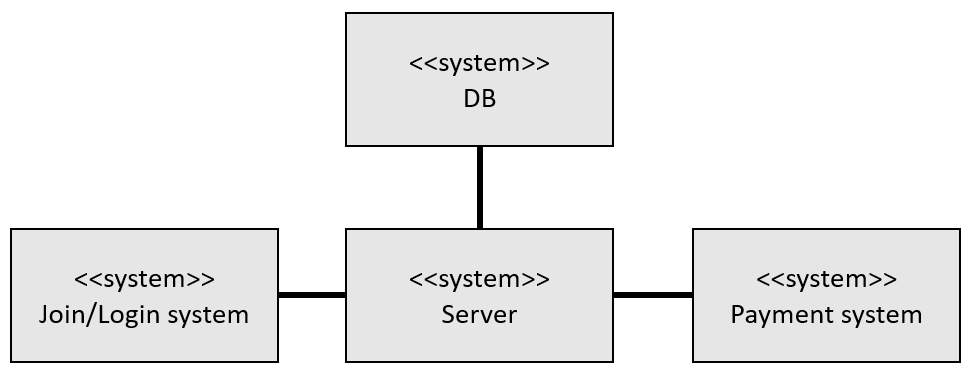
**7. System Models**

**7.1. Objectives**

System Models에서는 시스템 컴포넌트, 시스템 그리고 시스템 환경 사이의 관계를 보여준다. Context diagram과 Process diagram을 이용하여 시스템과 외부 환경의 상호작용을 나타내고, Use case diagram과 sequence diagram을 이용하여 사용자와 시스템, 또는 components 사이의 상호작용을 나타낸다.

**7.2. Context Models**

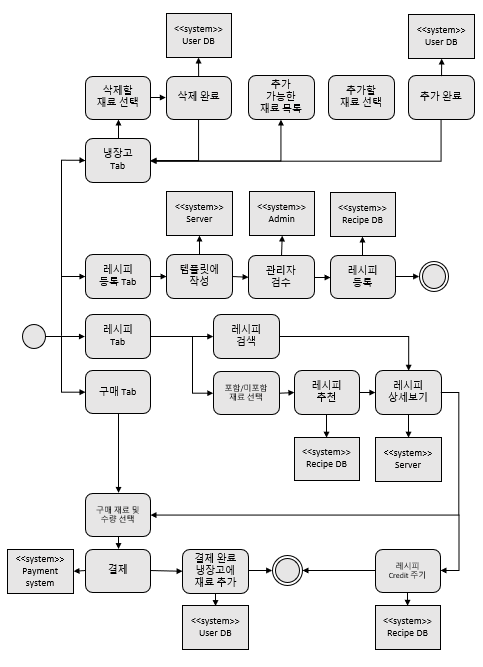
**7.2.1. Context model**



**diagram 8: Context Diagram**

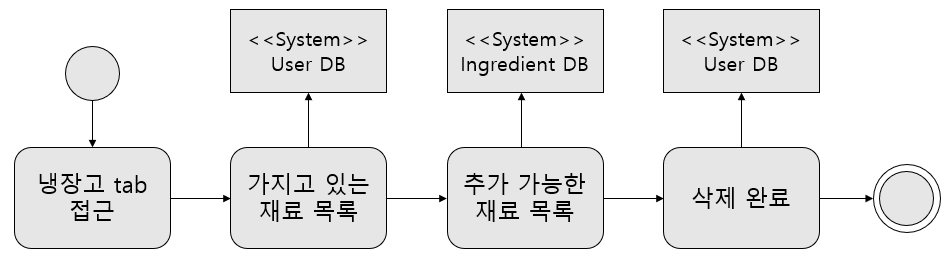
**7.2.2. Process Diagram**

**A. Overall Process**



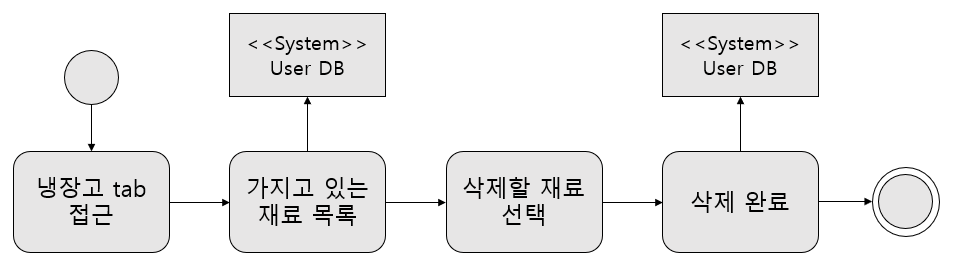
**diagram 9: Overall Process**

**B. 냉장고 재료 추가 Process**



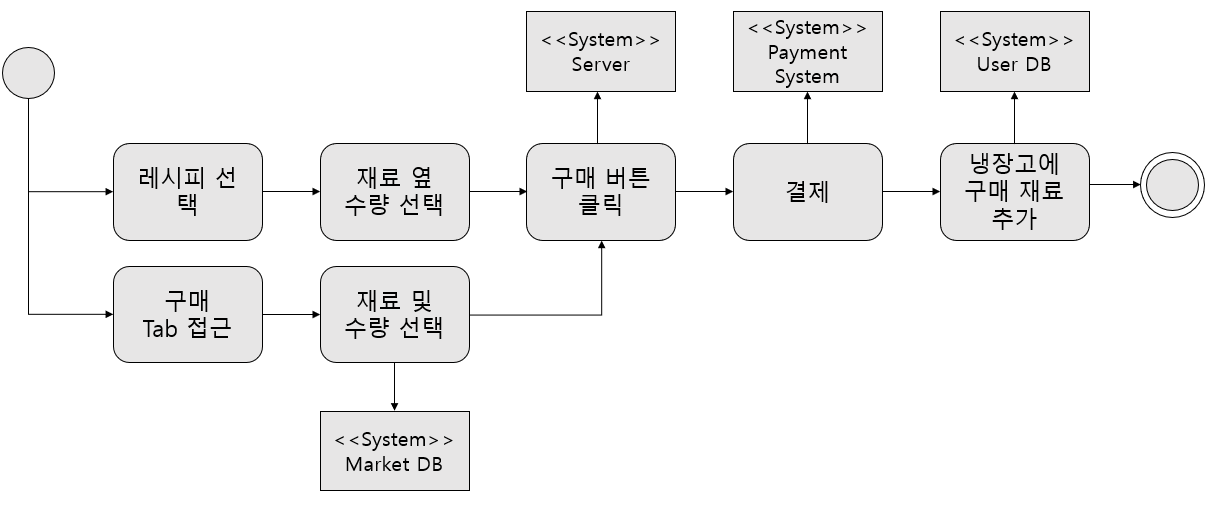
**diagram 10: 냉장고 재료 추가**

**C. 냉장고 재료 삭제 Process**



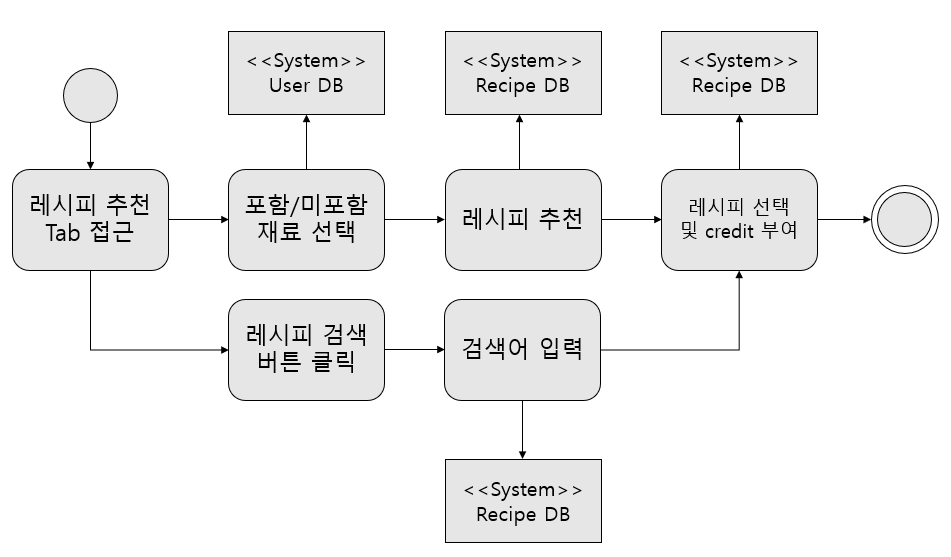
**diagram 11: 냉장고 재료 삭제**

**D. 재료 구매 Process**



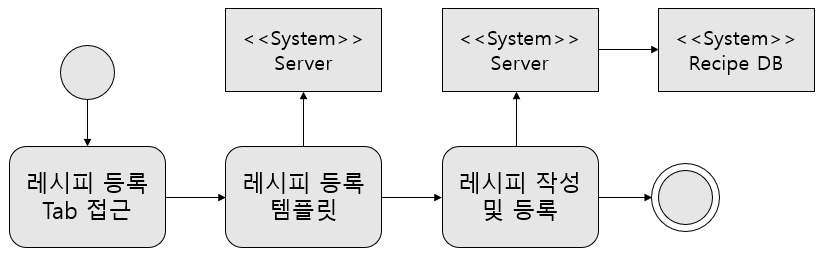
**diagram 12: 재료 구매**

**E. 레시피 추천 및 검색 Process**



**diagram 13: 레시피 추천 및 검색**

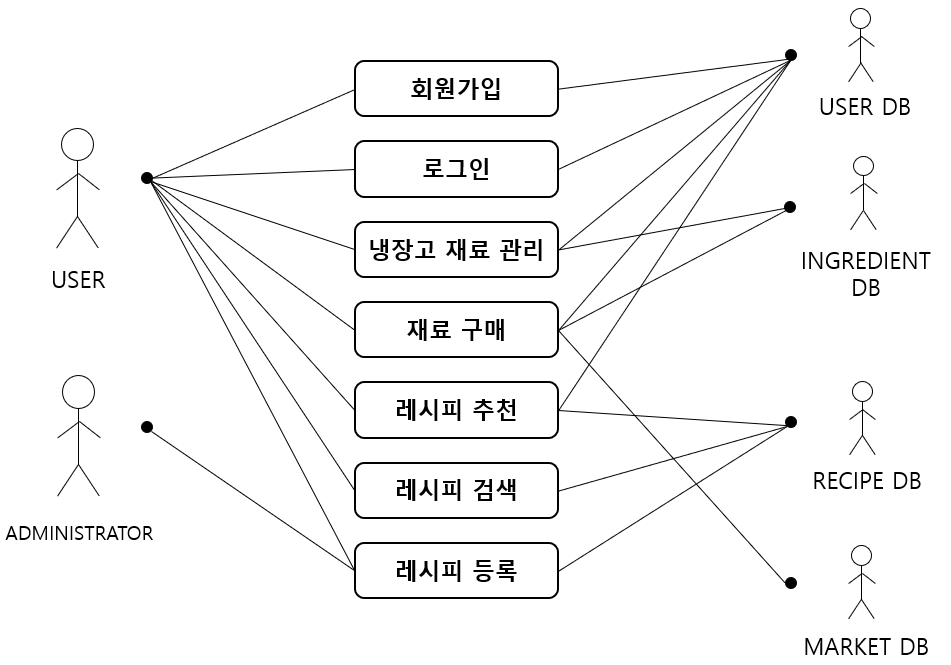
**F. 레시피 등록 Process**



**diagram 14: 레시피 등록**

**7.3. Interaction Models**

**7.3.1. Use case models**



**diagram 15: Use case**

**7.3.2. Tabular description for each use case**

**A. 회원가입**

|  |  |
| --- | --- |
| SUB SYSTEM |  |
| USE CASE | 회원가입 |
| ACTOR | User, User DB |
| DESCRIPTION | - User가 냉모밀 서비스에 회원가입 하기 위해 서비스에서 요청한 개인정보를 입력하고 회원가입 요청을 보낸다. |
| STIMULUS | - User가 회원가입 버튼을 누르고 정보 입력창이 뜰 때까지 기다린다.  - User가 회원가입 페이지에 ID, 비밀번호, 닉네임을 다 입력하면 서버에 회원가입 request를 보낸다. |
| RESPONSE | - 정보가 서버로 제대로 보내지면 User DB에 update 하는 쿼리를 실행한다. |

**table 11: 회원가입**

**B. 로그인**

|  |  |
| --- | --- |
| SUB SYSTEM |  |
| USE CASE | 로그인 |
| ACTOR | User, USER DB |
| DESCRIPTION | - 회원가입한 본인 계정으로 로그인 한다. |
| STIMULUS | - User가 ID와 비밀번호를 입력하면 User DB에 등록된 정보와 일치하는지 확인한다. |
| RESPONSE | - User DB에 있는 회원정보와 일치한다면 User는 로그인을 마치고 본인 DB에 접근할 수 있게 된다. |

**table 12: 로그인**

**C. 냉장고 재료 관리**

|  |  |
| --- | --- |
| SUB SYSTEM |  |
| USE CASE | 냉장고 재료 관리 |
| ACTOR | User, USER DB, INGREDIENT DB |
| DESCRIPTION | - User가 가지고 있는 냉장고 재료를 추가 또는 삭제한다. |
| STIMULUS | (추가)  - 추가를 원하는 경우엔 User가 본인의 냉장고 화면에서 추가 버튼을 눌러서 Ingredient DB에서 추가 가능한 재료 목록을 받는다.  - 추가 가능한 재료들이 뜨면 그 중에서 추가하고 싶은 재료들을 찾아 선택하고 확인 버튼을 누른다.  (삭제)  - 삭제를 원하는 경우엔 본인의 냉장고 화면에서 삭제를 원하는 재료를 누른 후, 삭제 버튼을 누른다. |
| RESPONSE | (추가)  - User가 본인의 냉장고 화면에서 추가 버튼을 누르면 Ingredient DB에 있는 재료들을 보여준다.  - User가 추가할 재료들을 선택하고 확인 버튼을 누르면 User DB에 사용자가 가지고 있는 재료 목록을 update 한다.  (삭제)  - User가 본인의 냉장고 화면에서 삭제하려는 재료를 누르면 삭제를 원하냐는 팝업을 띄우고 삭제 버튼을 누르면 User DB에서 해당 재료를 delete 한다. |

**table 13: 냉장고 재료 관리**

**D. 재료 구매**

|  |  |
| --- | --- |
| SUB SYSTEM |  |
| USE CASE | 재료 구매 |
| ACTOR | User, USER DB, Market DB, Ingredient DB |
| DESCRIPTION | - 원하는 재료를 구매한다. |
| STIMULUS | - User는 레시피를 보다가 구매를 원하는 재료가 있으면 수량을 설정한 후 구매 버튼을 누른다.  - User는 구매 창으로 가서 원하는 재료와 수량을 설정한 후 구매 버튼을 누른다.  - 결제 창이 뜨면 결제 정보(카드번호, 배송지 주소)를 입력한다. |
| RESPONSE | - User가 구매 창으로 가면 판매중인 재료의 목록을 Ingredient DB와 Market DB를 바탕으로 띄워준다.  - User가 구매 재료와 수량을 선택하고 구매 버튼을 누르면 결제 시스템에 연결하여 결제를 진행한다.  - 결제까지 완료되면 User가 구매한 재료를 해당 User 냉장고에 추가해준다. |

**table 14: 재료 구매**

**E. 레시피 추천**

|  |  |
| --- | --- |
| SUB SYSTEM |  |
| USE CASE | 레시피 추천 |
| ACTOR | User, USER DB, Recipe DB |
| DESCRIPTION | - User가 가지고 있는 재료를 기반으로 레시피 추천을 받는다. |
| STIMULUS | - User가 본인의 냉장고에 들어있는 재료 중에 ‘사용하고 싶은 재료’와 ‘사용하지 않을 재료’를 선택한 후, 레시피 추천 버튼을 누른다.  - 추천 레시피가 뜨면 원하는 레시피를 눌러 상세정보를 확인한다.  - 해당 레시피가 마음에 들면 ‘좋아요’ 버튼을 눌러서 레시피에 credit을 준다. |
| RESPONSE | - Recipe DB에서 User가 표시한 ‘사용하고 싶은 재료’는 포함하고 ‘사용하지 않을 재료’는 포함하지 않는 레시피들을 선별하여 보여준다.  - 이때 동일 메뉴에 대해서는 ‘좋아요’가 많은 레시피를 우선으로 보여준다.  - 추천 레시피 목록 중 User가 클릭한 레시피의 상세정보를 보여준다.  - User가 레시피의 ‘좋아요’ 버튼을 누르면 해당 레시피의 credit에 1을 더한다. |

**table 15: 레시피 추천**

**F. 레시피 검색**

|  |  |
| --- | --- |
| SUB SYSTEM |  |
| USE CASE | 레시피 검색 |
| ACTOR | User, Recipe DB |
| DESCRIPTION | - 원하는 레시피를 검색한다. |
| STIMULUS | - 레시피 검색창에 레시피를 원하는 메뉴의 이름을 입력하여 검색한다. |
| RESPONSE | - Recipe DB에서 해당 메뉴의 이름을 포함하는 레시피들을 보여준다.  - 이때 레시피는 credit이 높은 순서대로 정렬된다. |

**table 16: 레시피 검색**

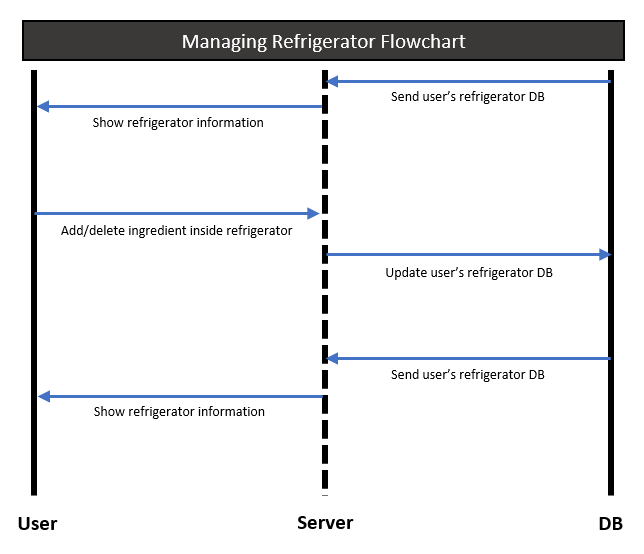
**G. 레시피 등록**

|  |  |
| --- | --- |
| SUB SYSTEM |  |
| USE CASE | 레시피 등록 |
| ACTOR | User, Recipe DB |
| DESCRIPTION | - User가 자신의 레시피를 등록한다. |
| STIMULUS | - User가 레시피 등록 버튼을 누르고 레시피 등록 템플릿이 뜰때까지 기다린다.  - User는 레시피 등록 템플릿에 맞게 재료, 요리 순서, 사진 등을 작성하고 등록 버튼을 누른다. |
| RESPONSE | - User가 레시피 등록 버튼을 누르면 레시피 등록 템플릿을 띄워준다.  - User가 템플릿에 필요한 모든 정보들을 입력하고 등록 버튼을 누르면 해당 정보들을 Server로 전송한다.  - Server에 쌓인 새로운 레시피 목록을 관리자가 검수 작업을 하고 통과된 레시피들만 DB에 저장한다.  - 위에 검수 작업이란, 레시피의 퀄리티가 제공하기에 충분한지, 완전히 중복되는 레시피가 없는지 확인하고 DB에서 관리할 때 쓰이는 카테고리를 메뉴에 맞게 설정해주는 작업을 말한다. |

**table 17: 레시피 등록**

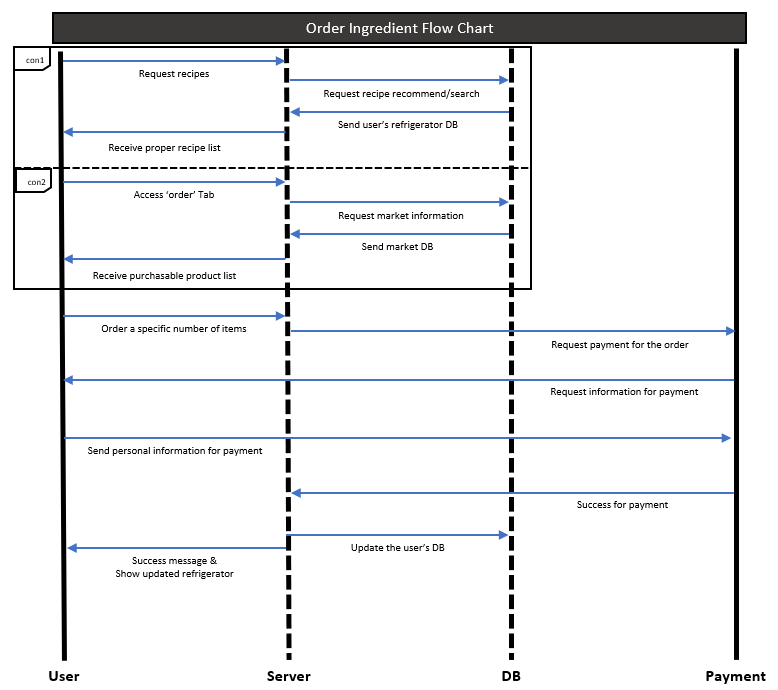
**7.3.3. Sequence Diagram**

**A. Sequence Diagram of Managing Refrigerator**



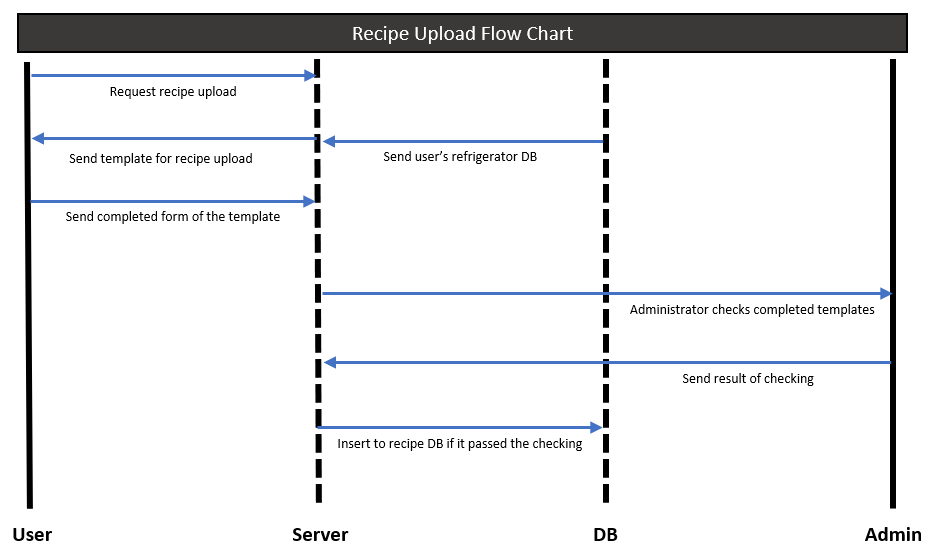
**diagram 16: Sequence Diagram of Managing Refrigerator**

**B. Sequence Diagram of Order Ingredient**



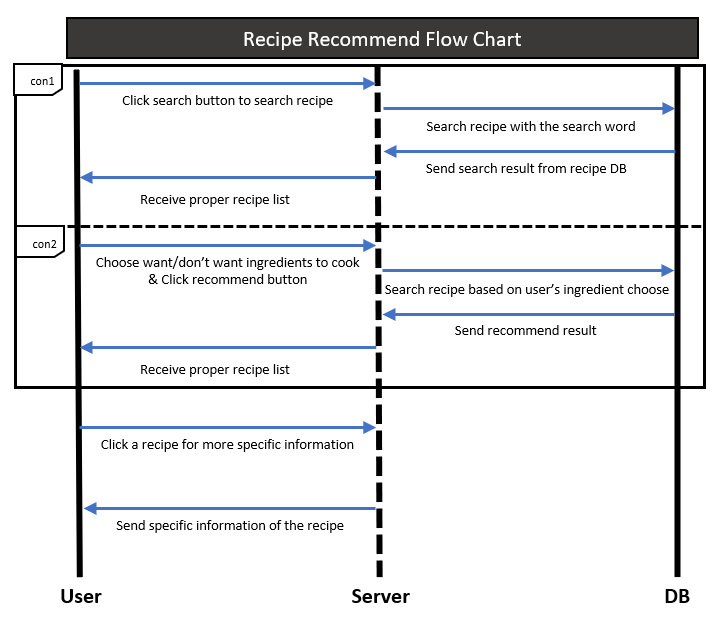
**diagram 17: Sequence Diagram of Order Ingredient**

**C. Sequence Diagram of Recipe Upload**



**diagram 18: Sequence Diagram of Recipe Upload**

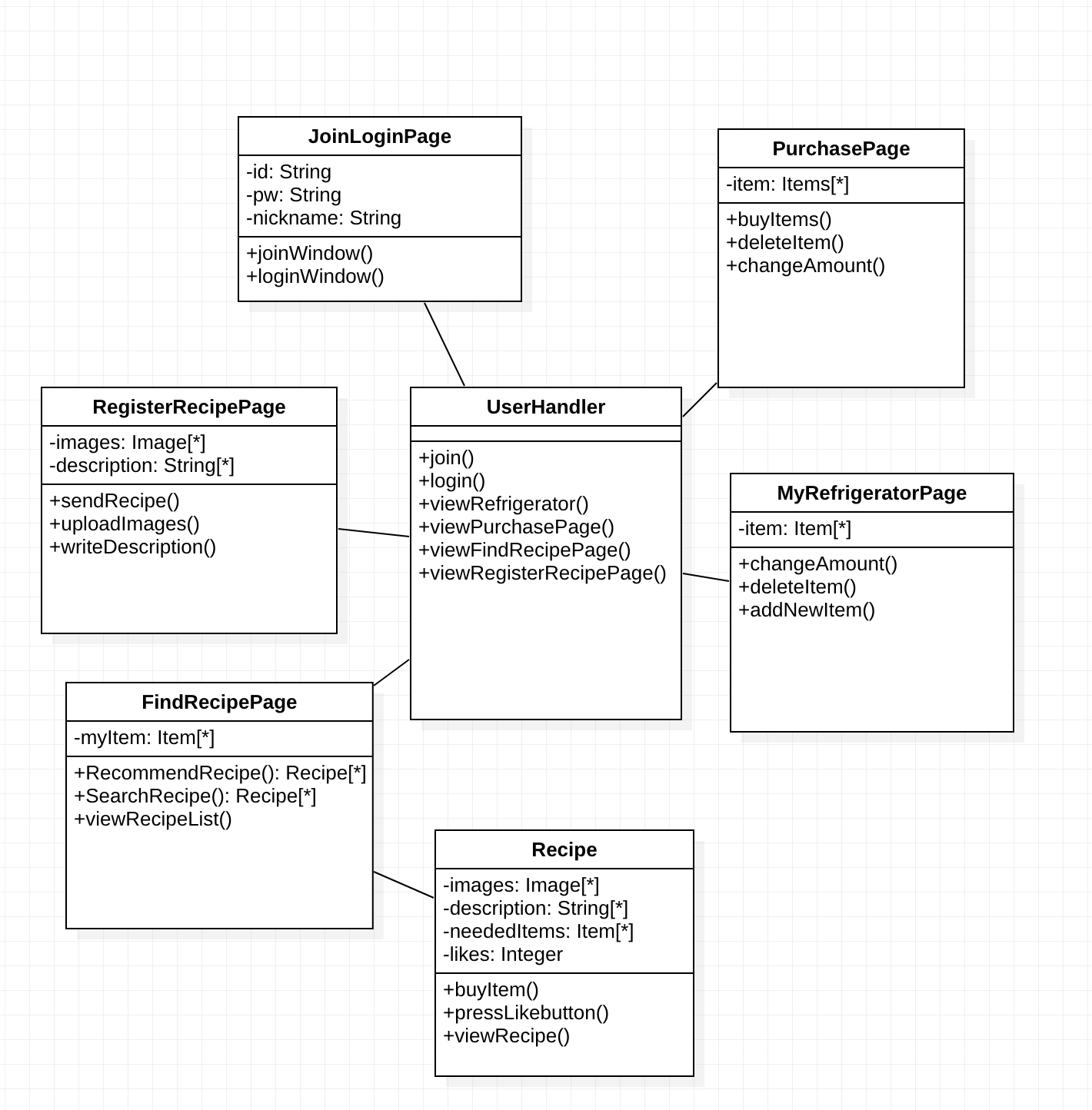
**D. Sequence Diagram of Recipe Recommend**



**diagram 19: Sequence Diagram of Recipe Recommend**

**7.4. Structural Models**

**7.4.1. User System Class Diagram**

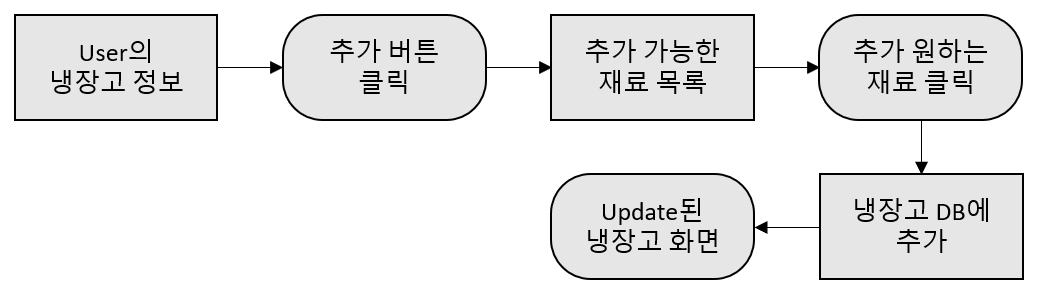


**diagram 20: User System Class**

**7.5. Behavioral Models**

**7.5.1. Data-Driven Modeling**

**A. Activity Model of 냉장고 재료 추가**



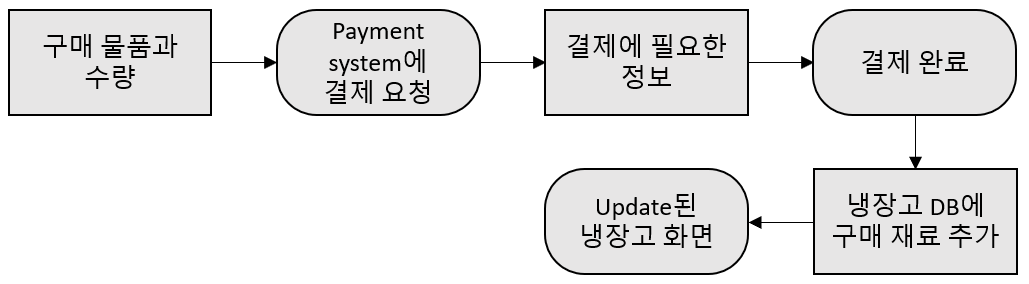
**diagram 21: Activity Model of 냉장고 재료 추가**

**B. Activity Model of 냉장고 재료 삭제**



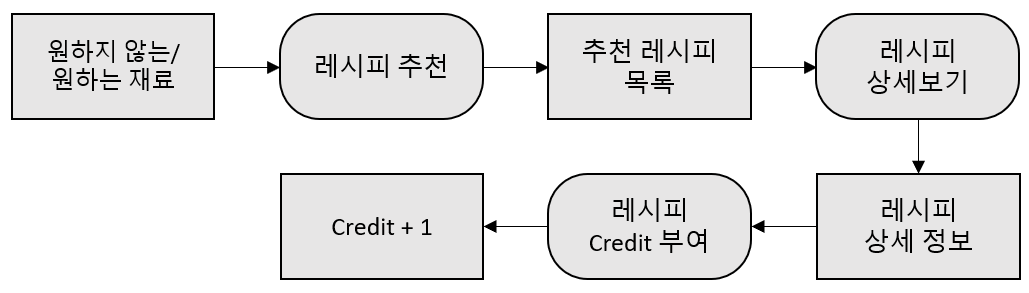
**diagram 22: Activity Model of 냉장고 재료 삭제**

**C. Activity Model of 재료 구매**



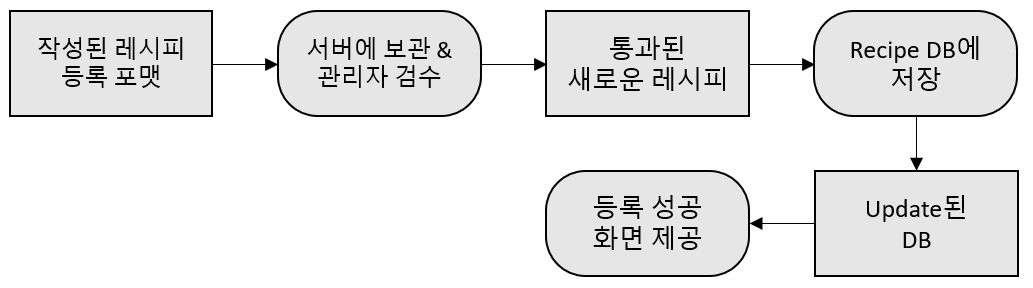
**diagram 23: Activity Model of 재료 구매**

**D. Activity Model of 레시피 추천**



**diagram 24: Activity Model of 레시피 추천**

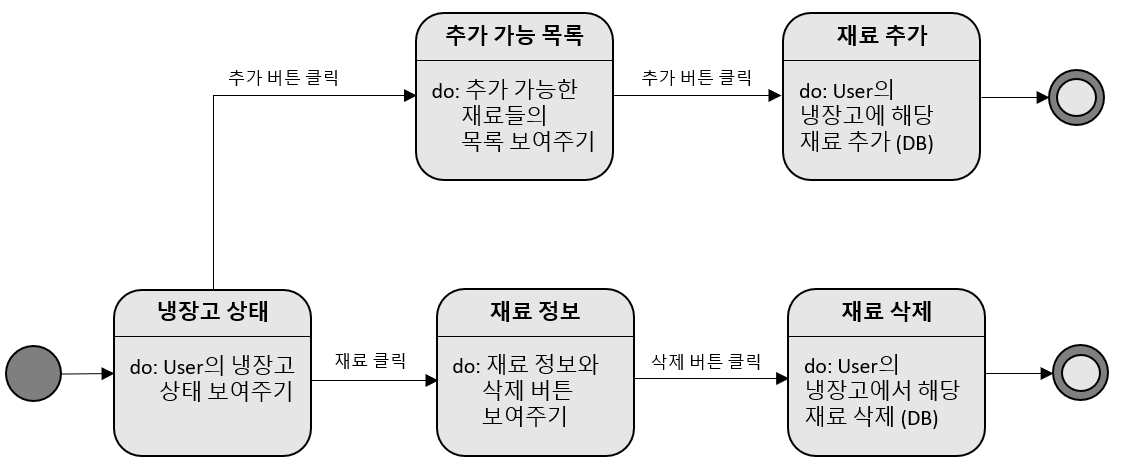
**E. Activity Model of 레시피 등록**



**diagram 25: Activity Model of 레시피 등록**

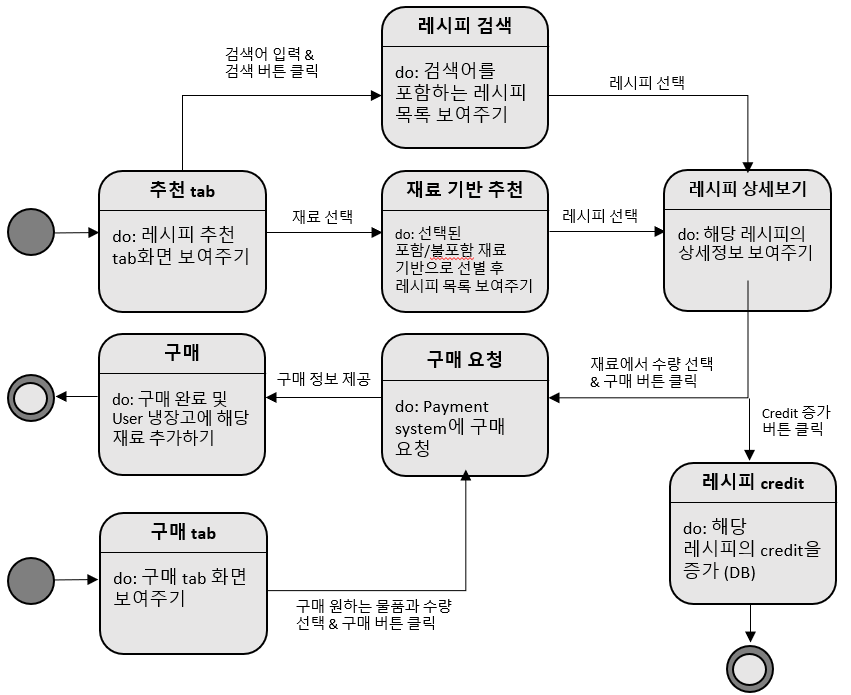
**7.5.2. Event-Driven Modeling**

**A. Event-Driven 냉장고 재료 관리**



**diagram 26: Event-Driven Diagram 냉장고 재료 관리**

**B. Event-Driven 레시피 추천 및 재료 구매**



**diagram 27: Event-Driven Diagram 레시피 추천/검색 & 재료 구매**

**C. Event-Driven 레시피 등록**



**diagram 28: Event-Driven Diagram 레시피 등록**

**8. System Evolutions**

**8.1. Objective**

System Evolution에서는 시스템이 세우고 있는 주요한 가정들에 대해 설명한다. 시스템에 일어날 수 있는 예상되는 변화들 (하드웨어의 발전, 사용자의 Needs 변화 등에 대해 설명한다.

**8.2. Limitation and Assumption**

오늘날 온라인 시장은 점점 더 그 크기가 커지고 있다. 오프라인 마켓들의 매출이 온라인 시장의 매출에 따라 잡히고, 오프라인 매장을 줄이는 일도 생겨난다. 결국 일부 생필품과 식자재들은 온라인 시장에 완전히 잡아 먹힐 것으로 전망된다. 이러한 상황은 대한민국의 배송 서비스와도 그 관계를 갖고 있다. 만약 배송 서비스가 일주일이나 걸리는 사회라면 직접 나가서 필요한 것들은 사오는 것이 훨씬 경제적일 것이다. 하지만 대한민국에서는 도시 기준으로는 길면 3일, 서울(경기) 기준으로는 2일이면 배송이 가능하고, 심지어 최근에는 당일배송, 새벽배송을 타이틀로 내세워서 경쟁에 돌입하고 있다. 이렇듯 과잉 경쟁에 빠진 배송 업체들의 문제도 분명 존재하지만, 결국 대한민국에서는 배송 서비스가 우리나라의 인터넷 서비스만큼이나 빨라졌다고 볼 수 있다. 우리는 이러한 배송 시스템을 활용해서 이용자가 만들고 싶은 음식이 있다면, 우리의 어플을 활용해서 레시피를 찾고 재료를 주문할 수 있기를 원한다.

하지만 우리가 목표로 하는 서비스는 원하는 재료를 당일배송을 통해서 원하는 레시피를 직접 이용하여 음식을 만들어 먹을 수 있게 하는 것이다. 그런데 아직까지 재료를 당일배송으로 전지역에 서비스하는 업체는 존재하지 않는다. ssg마켓의 경우에는 수도권 내에서는 훌륭한 서비스를 보이지만, 지역사회에서는 다른 마트 서비스를 이용해야 하는 상황이다. 그렇기 때문에 우리의 어플로 한 업체와 연결하여 이러한 서비스를 이뤄낼 수 없는 상황이다.

**8.3. Evolutions of Hardware**

**스마트 냉장고**

우리가 만들고자 하는 시스템에서는 이용자가 직접 본인의 냉장고 재료를 추가/제거해야 하는 번거로움을 가지고 있다. 하지만 어플의 이용자가 스마트 냉장고를 사용하고 있다면 이러한 번거로움을 줄일 수 있을 것이다. 스마트 냉장고는 냉장고 내부에 설치된 특수 카메라를 통해서 이용자가 냉장고 문을 닫을 때마다 냉장고 안의 상황을 업데이트 해준다. 이러한 스마트 냉장고의 정보를 우리 어플에 연동할 수 있다면 한층 더 유용한 어플이 될 수 있을 것이다.

**8.4. Evolutions of User Requirement**

**이용자 보상 시스템**

다양한 레시피를 보유하는 것은 우리가 만들고자 하는 어플에게는 큰 강점이 될 것이다. 관리자가 레시피를 업데이트 하는 것에는 한계가 있기 때문에 결국 이용자들의 적극적인 참여가 필요하다. 이러한 참여를 이끌어 내기 위한 것이 이용자 보상 시스템이다. 이용자가 레시피를 올리고, 이용자의 레시피가 일정 이상의 좋은 평가를 받았을 때, 이용자에게 포인트를 지급한다. 이러한 포인트는 차후 우리 어플의 광고 수익을 배분 받는데 사용할 수 있다.

포인트와는 별개로 이용자들에게 활동 이력에 따라서 등급을 매겨서 이용자의 활동을 장려할 수도 있다.

**8.5. Evolutions of Environment**

**동영상/스트리밍**

정보 전달 매체는 그림에서 말로. 말에서 글로. 글에서 영상으로 다양한 형태를 취하고 있다. 인터넷 트래픽의 절반이상이 유튜브나 넷플릭스와 같은 동영상 스트리밍 사이트에 사용되는 것을 보면 과히 동영상의 시대라 해도 지나치지 않는 것 같다. 이러한 상황 속에서 결국 정보 전달은 잘 만들어진 동영상이 인기를 끌 수밖에 없다. 이러한 상황에 맞추어서 우리 레시피 어플도 훌륭한 재료기반 추천을 바탕으로 동영상으로 레시피를 전달하는 것이 더욱 좋은 방향이 될 수 있다.

**9. Appendices**

**9.1. Database Requirements**

본 시스템은 Mysql을 활용한 관계형 데이터베이스를 사용하여 데이터를 저장 및 관리한다. 데이터베이스 내의 엔티티들은 아래와 같다. 세부적인 데이터베이스 설계는 Design Architecture Specification에서 명시한다.

1. User DB

ID, password, 닉네임, 가지고 있는 재료

1. Ingredient DB

본 서비스에서 사용 가능한 모든 재료들. 이 DB에 없는 재료들은 냉장고에 추가하거나 구매할 수 없다.

1. Recipe DB

메뉴 이름, 레시피 주소, 레시피 작성자 ID

\* 레시피의 전체적인 내용은 서버에 저장해두고 주소 값만 DB에 저장한다.

1. Market DB

제품 이름, 가격

**10. Index**

**10.1 Tables**

table 1: Glossary………………………….………………………….………………………….………………………….10

table 2 : 회원가입 Functional Requirement…………………………………………………………………..19

table 3: 로그인 Functional Requirement……………………………………………………………………….19

table 4 : 냉장고 재료 Functional Requirement……………………………………………………………..20

table 5 : 냉장고 재료 제거 Functional Requirement…………………………………………………….20

table 6: 레시피 추천 Functional Requirement………………………………………………………………20

table 7: 레시피 검색 Functional Requirement………………………………………………………………20

table 8: 레시피 등록 신청 Functional Requirement………………………………………………………21

table 9: 레시피 등록 Functional Requirement………………………………………………………………21

table 10: 재료 구매 Functional Requirement………………………………………………………………..21

table 11: 회원가입………………………………………………………………………………………………………….32

table 12: 로그인……………………………………………………………………………………………………………..33

table 13: 냉장고 재료 관리…………………………………………………………………………………………….33

table 14: 재료 구매………………………………………………………………………………………………………..33

table 15: 레시피 추천…………………………………………………………………………………………………….34

table 16: 레시피 검색…………………………………………………………………………………………………….34

table 17: 레시피 등록…………………………………………………………………………………………………….35

**10.2 Diagrams**

diagram1: Frontend Architecture…………………………………………………………………….….………...15

diagram 2: Backend Architecture………………………………………………………………………………….16

diagram 3: Recipe Registration System………………………………………………………………………..16

diagram 4: Recipe Recommendation System……………………………………………………………….17

diagram 5: Recipe Searching System……………………………………………………………………………17

diagram 6: Refrigerator Management System………………………………………………………………17

diagram 7: Products Purchasing System………………………………………………………………………18

diagram 8: Context Diagram…………………………………………………………………………………………28

diagram 9: Overall Process……………………………………………………………………………………………29

diagram 10: 냉장고 재료 추가………………………………………………………………………………………30

diagram 11: 냉장고 재료 삭제………………………………………………………………………………………30

diagram 12: 재료 구매…………………………………………………………………………………………………..30

diagram 13: 레시피 추천 및 검색………………………………………………………………………………….31

diagram 14: 레시피 등록………………………………………………………………………………………………..31

diagram 15: Use case…………………………………………………………………………………………………….32

diagram 16: Sequence Diagram of Managing Refrigerator…………………………………………36

diagram 17: Sequence Diagram of Order Ingredient…………………………………………………..37

diagram 18: Sequence Diagram of Recipe Upload………………………………………………………38

diagram 19: Sequence Diagram of Recipe Recommend……………………………………………..39

diagram 20: User System Class……………………………………………………………………………………..40

diagram 21: Activity Model of 냉장고 재료 추가……………………………………………………….…41

diagram 22: Activity Model of 냉장고 재료 삭제………………………………………………………….41

diagram 23: Activity Model of 재료 구매………………………………………………………………………41

diagram 24: Activity Model of 레시피 추천…………………………………………………………………..41

diagram 25: Activity Model of 레시피 등록………………………………………………………………….42

diagram 26: Event-Driven Diagram 냉장고 재료 관리…………………………………………………..42

diagram 27: Event-Driven Diagram 레시피 추천/검색 & 재료 구매……………………………..43

diagram 28: Event-Driven Diagram 레시피 등록…………………………………………………………..44