

**Design**

**Specification**

2018/11/11

TEAM #6

2018318979 컴퓨터공학과 Apoorv Rastogi

2014311021 컴퓨터공학과 김명준

2012314413 수학과 김상우

2015314758 컴퓨터공학과 윤현철

2016311438 컴퓨터공학과 이지현

2016314078 컴퓨터공학과 이창훈

2013314099 컴퓨터공학과 홍기환

목차

[1. Preface 3](#_Toc529744007)

[1.1. Objective 3](#_Toc529744008)

[1.2. Readers 3](#_Toc529744009)

[1.3. Document Structure 3](#_Toc529744010)

[A. Preface 3](#_Toc529744011)

[B. Introduction 3](#_Toc529744012)

[C. System Architecture 3](#_Toc529744013)

[D. User Management System 4](#_Toc529744014)

[E. Seller Exclusive System 4](#_Toc529744015)

[F. Picture Search System 4](#_Toc529744016)

[G. Filter Search System 4](#_Toc529744017)

[H. Recommendation System 5](#_Toc529744018)

[I. Trend System 5](#_Toc529744019)

[J. Community System 5](#_Toc529744020)

[K. Protocol Design 5](#_Toc529744021)

[L. Database Design 5](#_Toc529744022)

[M. Testing Plan 6](#_Toc529744023)

[N. Development Environment 6](#_Toc529744024)

[O. Develop Plan 6](#_Toc529744025)

[P. Index 6](#_Toc529744026)

[1.4. Version of the Document 6](#_Toc529744027)

[A. Version Format 6](#_Toc529744028)

[B. Version Management Policy 6](#_Toc529744029)

[C. Version Update History 7](#_Toc529744030)

[2. Introduction 8](#_Toc529744031)

[2.1. Objective 8](#_Toc529744032)

[2.2. Applied Diagram 8](#_Toc529744033)

[A. UML 8](#_Toc529744034)

[B. Package Diagram 10](#_Toc529744035)

[C. Deployment Diagram 11](#_Toc529744036)

[D. Class Diagram 12](#_Toc529744037)

[E. State Diagram 12](#_Toc529744038)

[F. Sequence Diagram 13](#_Toc529744039)

[G. ER Diagram 14](#_Toc529744040)

[2.3. Applied Tool 15](#_Toc529744041)

[A. Flow Chart Maker & Online Diagram Software 15](#_Toc529744042)

[B. Amazon Web Services(AWS) 15](#_Toc529744043)

[2.4. Project Scope 16](#_Toc529744044)

[3. System Architecture 19](#_Toc529744045)

[3.1. Objective 19](#_Toc529744046)

[3.2. System Organization 19](#_Toc529744047)

[A. Context Model 19](#_Toc529744048)

[B. User/Seller System 20](#_Toc529744049)

[C. Picture Search System 22](#_Toc529744050)

[C.1. Picture Search 22](#_Toc529744051)

[D. Filter Search System 22](#_Toc529744052)

[D.1. Filter Search 23](#_Toc529744053)

[E. Recommendation 23](#_Toc529744054)

[3.3. Package Diagram 25](#_Toc529744055)

[3.4. Deployment Diagram 25](#_Toc529744056)

[4. User Management System 26](#_Toc529744057)

[4.1. Objective 26](#_Toc529744058)

[4.2. Class Diagram 26](#_Toc529744059)

[A. DB Handler 26](#_Toc529744060)

[B. Account 26](#_Toc529744061)

[4.3. Sequence Diagram 27](#_Toc529744062)

[A. Sign Up using SNS(Naver/Facebook) 27](#_Toc529744063)

[B. Log In with using SNS 28](#_Toc529744064)

[C. Sign Up without SNS 29](#_Toc529744065)

[D. Log In without SNS 30](#_Toc529744066)

[4.4. State Diagram 31](#_Toc529744067)

[A. Sign Up using SNS 31](#_Toc529744068)

[B. Sign Up without SNS 31](#_Toc529744069)

[5. Seller Exclusive System 32](#_Toc529744070)

[5.1. Objectives 32](#_Toc529744071)

[5.2. Class Diagram 32](#_Toc529744072)

[A. DB Handler 32](#_Toc529744073)

[B. Account 32](#_Toc529744074)

[C. Product 33](#_Toc529744075)

[5.3. Sequence Diagram 34](#_Toc529744076)

[A. Product Management 34](#_Toc529744077)

[B. User Purchase System 35](#_Toc529744078)

[5.4. State Diagram 35](#_Toc529744079)

[A. Product Management System 35](#_Toc529744080)

[B. User Purchase System 36](#_Toc529744081)

[6. Picture Search System 37](#_Toc529744082)

[6.1. Objectives 37](#_Toc529744083)

[6.2. Class Diagram 37](#_Toc529744084)

[A. DB Handler 37](#_Toc529744085)

[B. Picture 38](#_Toc529744086)

[C. Search\_Engine 38](#_Toc529744087)

[D. Products\_list 39](#_Toc529744088)

[6.3. Sequence Diagram 39](#_Toc529744089)

[A. User – System Interface 40](#_Toc529744090)

[B. System Interface – Search Engine 40](#_Toc529744091)

[C. Search Engine – Product DB 40](#_Toc529744092)

[D. System Interface – Product DB 40](#_Toc529744093)

[7. Filter Search System 41](#_Toc529744094)

[7.1. Objectives 41](#_Toc529744095)

[7.2. Class Diagram 41](#_Toc529744096)

[A. DB Handler 41](#_Toc529744097)

[B. Product 41](#_Toc529744098)

[7.3. Sequence Diagram 42](#_Toc529744099)

[7.4. State Diagram 43](#_Toc529744100)

[8. Recommendation System 44](#_Toc529744101)

[8.1. Objectives 44](#_Toc529744102)

[8.2. Class Diagram 44](#_Toc529744103)

[A. DB Handler 44](#_Toc529744104)

[B. Product 44](#_Toc529744105)

[C. User 45](#_Toc529744106)

[8.3. Sequence Diagram 46](#_Toc529744107)

[8.4. State Diagram 46](#_Toc529744108)

[9. Trend System 47](#_Toc529744109)

[9.1. Objectives 47](#_Toc529744110)

[9.2. Class Diagram 47](#_Toc529744111)

[A. DB Handler 47](#_Toc529744112)

[B. Account 47](#_Toc529744113)

[C. Trend 48](#_Toc529744114)

[D. Picture Search 48](#_Toc529744115)

[E. Filter Search 49](#_Toc529744116)

[F. Recommendation 49](#_Toc529744117)

[9.3. Sequence Diagram 49](#_Toc529744118)

[A. View 49](#_Toc529744119)

[B. Write 50](#_Toc529744120)

[C. Search 50](#_Toc529744121)

[D. Edit 51](#_Toc529744122)

[9.4. State Diagram 51](#_Toc529744123)

[A. View 51](#_Toc529744124)

[B. Write 52](#_Toc529744125)

[C. Search 52](#_Toc529744126)

[D. Edit 52](#_Toc529744127)

[10. Community System 53](#_Toc529744128)

[10.1. Objectives 53](#_Toc529744129)

[10.2. Class Diagram 53](#_Toc529744130)

[A. DB Handler 53](#_Toc529744131)

[B. Account 53](#_Toc529744132)

[C. Community 54](#_Toc529744133)

[10.3. Sequence Diagram 55](#_Toc529744134)

[A. View 55](#_Toc529744135)

[B. Write 55](#_Toc529744136)

[C. Search 56](#_Toc529744137)

[D. Edit 56](#_Toc529744138)

[10.4. State Diagram 57](#_Toc529744139)

[A. View 57](#_Toc529744140)

[B. Write 57](#_Toc529744141)

[C. Search 57](#_Toc529744143)

[D. Edit 58](#_Toc529744144)

[10.5. Search Filter 58](#_Toc529744145)

[11. Protocol Design 59](#_Toc529744146)

[11.1. Objectives 59](#_Toc529744147)

[11.2. JSON 59](#_Toc529744148)

[11.3. Protocol Description 59](#_Toc529744149)

[A. Overview 59](#_Toc529744150)

[B. Login Protocol 60](#_Toc529744151)

[C. Registration Protocol 60](#_Toc529744152)

[D. Community Post Edit Protocol 60](#_Toc529744153)

[E. Community Search Protocol 61](#_Toc529744154)

[F. Community View Protocol 61](#_Toc529744155)

[G. Product Post Edit Protocol 62](#_Toc529744156)

[a. Response 62](#_Toc529744157)

[H. Product View Protocol 63](#_Toc529744158)

[a. Response 63](#_Toc529744159)

[I. Product Search Protocol 63](#_Toc529744160)

[J. Picture Upload Protocol 64](#_Toc529744161)

[12. Database Design 65](#_Toc529744162)

[12.1. Objectives 65](#_Toc529744163)

[12.2. ER Diagram 65](#_Toc529744164)

[A. Entity 66](#_Toc529744165)

[B. Entity 67](#_Toc529744166)

[12.3. Relational Schema 69](#_Toc529744167)

[A. Customer 69](#_Toc529744168)

[B. Seller 70](#_Toc529744169)

[C. Product 70](#_Toc529744170)

[D. Community\_notice 71](#_Toc529744171)

[E. Community\_news 71](#_Toc529744172)

[F. Community\_Free 71](#_Toc529744173)

[G. Community\_Comment\_Notice 72](#_Toc529744174)

[H. Community\_Comment\_News 72](#_Toc529744175)

[I. Community\_Comment\_Free 72](#_Toc529744176)

[12.4. Normalization 73](#_Toc529744177)

[12.5. SQL DDL 74](#_Toc529744178)

[A. Member 74](#_Toc529744179)

[B. Product 74](#_Toc529744180)

[C. Community Notice 75](#_Toc529744181)

[D. Community News 75](#_Toc529744182)

[E. Community Free 76](#_Toc529744183)

[F. Community Comment Notice 76](#_Toc529744184)

[G. Community Comment Notice 77](#_Toc529744185)

[H. Community Comment Free 77](#_Toc529744186)

[12. Testing Plan 78](#_Toc529744187)

[12.1. Objectives 78](#_Toc529744188)

[12.2. Testing Policy 78](#_Toc529744189)

[12.3. Test Case 78](#_Toc529744190)

[A. User Management System 78](#_Toc529744191)

[B. Seller Exclusive System 79](#_Toc529744192)

[C. Picture Search System 80](#_Toc529744193)

[D. Filter Search System 80](#_Toc529744194)

[E. Recommendation System 81](#_Toc529744195)

[F. Trend System 81](#_Toc529744196)

[G. Community System 82](#_Toc529744197)

[13. Development Environment 84](#_Toc529744198)

[13.1. Objectives 84](#_Toc529744199)

[13.2. Programming Language & IDE 84](#_Toc529744200)

[A. Programming Language 84](#_Toc529744201)

[B. IDE 85](#_Toc529744202)

[13.3. Coding Rule 85](#_Toc529744203)

[13.4. Version Management Tool 86](#_Toc529744204)

[14. Develop Plan 87](#_Toc529744205)

[14.1. Objectives 87](#_Toc529744206)

[14.2. Gantt Chart 87](#_Toc529744207)

[15. Index 88](#_Toc529744208)

1. **Preface**

1.1. Objective

Preface는 본 문서의 예상되는 독자들과 구조를 소개한다. 구조를 소개할 때는 각 목차의 목적을 서술한다. 또한, 각 버전 관리 정책, 버전 변경 기록, 그리고 문서의 변경사항들을 설명한다.

1.2. Readers

본 설계 명세서는 독자는 다음과 같다. 시스템을 직접 개발하는 소프트웨어 엔지니어, 시스템을 설계하는 아키텍처와 개발에 참여하는 모든 구성원을 독자로 정의한다. 만약, 시스템을 개발할 때 외주 업체를 이용한다면 해당 엄체에서 개발에 관련되는 모든 구성원 역시 독자에 포함한다. 즉, 본 문서의 독자는 본 문서에서 소개하는 시스템의 개발 및 유지 보수에 관련된 모든 구성원이다.

1.3. Document Structure

본 설계 명세서는 총 15개의 부분으로 구성되어 있다. Preface, Introduction, System Architecture, User Management System, Seller Exclusive System, Picture Search System, Filter Search System, Recommendation System, Trend System, Community System, Protocol Design, Database Design, Testing Plan, Development Environment, Develop Plan, Index로 구성된다.

A. Preface

본 문서의 예상되는 독자들과 전반적인 구조, 그리고 각 부분의 역할에 대해 설명한다. 또한, 각 버전 관리 정책, 버전 변경 기록, 그리고 문서의 변경사항들을 설명한다.

B. Introduction

본 시스템을 설계할 때 사용하는 모든 종류의 Diagram 및 Tool에 관해 서술한다.

C. System Architecture

System Architecture에서는 우리 팀에서 개발하고자 하는 시스템에 대해 전반적으로 서술한다. 전체적인 구조를 설명하고 시스템을 Block diagram으로 나타내어, 각각의 관계와 실제 어떻게 사용되는지를 Package diagram과 Deployment diagram을 사용하여 설명한다. 각 시스템에 대한 modular decomposition은 4~10장에 걸쳐 설명한다.

D. User Management System

쉽게 사용자가 사용하며 회원 가입과 로그인 기능을 통해 고객이 서비스를 이용하는 도중 발생하는 데이터 처리 및 고객 관리 시스템의 설계를 설명한다. Class diagram, Sequence diagram, State diagram을 통해 User management system의 구조를 나타낸다.

E. Seller Exclusive System

판매자와 관련된 시스템 설계를 설명한다. Seller Exclusive System은 크게 3개의 하위 시스템으로 나뉜다. 이는 각각 Product, Group purchasing, Link sending으로 나뉘어 Product에서는 제품의 serial number, type, name, stock으로 DB에 업로드한다. Group purchasing에서는 구매하고자 하는 고객들의 CID, Product name을 불러오도록 한다. 마지막으로 Link에서는 고객이 구매를 하고자 할 때, 판매자의 구매 경로 즉, link을 제공해 주도록 한다.

F. Picture Search System

Picture Search System이 제공하는 기능들을 설명한다. Picture Search System은 ‘FaReDy’의 주요 기능 중 하나인 ‘사진은 이용한 옷 검색 서비스’를 제공한다. 옷 검색 서비스를 제공하기 위해 본 시스템은 Interface, Picture Convertor, Feature Extractor, Search Engine, DB handler로 5개희 subsystem으로 구성된다. Class diagram, Sequence diagram, State diagram을 통해 Picture Search System의 구조를 표현하고 설명한다.

G. Filter Search System

검색을 할 때, 어떠한 검색 필터 및 키워드를 이용하느냐에 따라 구매자와의 상호 작용이 달라지기 때문에, 그것을 case로 설정하여 Class diagram, Sequence diagram과 State diagram을 통해 Search System의 구조와 구매자 간의 상호 작용을 표현하고 설명한다.

H. Recommendation System

이용자의 서비스 이용기록을 기반으로 개인에게 맞춤형 추천을 제공하는 시스템이다. CSP에 저장된 이용자의 검색기록을 이용하여 아이템 리스트 형태로 제공하는 형태와 자주 찾는 유형의 옷을 제공하는 형태로 나뉘어지기 때문에 Class diagram, Sequence diagram과 State diagram을 통해 각 케이스의 구조와 사용자와의 상호 작용을 표현하고 설명한다.

I. Trend System

시스템을 이용하는 사용자가 Trend 서비스를 제공받는 도중에 발생하는 데이터에 대한 처리와 데이터의 수정, 변경에 대한 설계를 설명한다. Class diagram, Sequence diagram과 State Diagram을 통해 Trend System의 구조를 표현한다.

J. Community System

실제 사용자가 사용하며 커뮤니티 게시판을 이용하는 도중에 발생하는 데이터를 처리하고 수정 사항이 발생하였을 때, 해당되는 데이터를 수정, 변경하며 도식화하는 커뮤니티 시스템의 설계를 설명한다. Class diagram, Sequence diagram과 State diagram을 통해 Community System의 구조를 표현하고 설명한다.

K. Protocol Design

Protocol Design에서는 subsystem들이 상호작용하는 프로토콜에 대해 서술한다. 프로토콜의 기본 형식은 JSON을 기본으로 하며 통신하는 메시지의 형식과 용도, 의미를 설명한다.

L. Database Design

Database Design은 요구사항 명세서에서 기술한 데이터베이스 요구사항을 바탕으로 수정사항을 수정하여 다시 요구사항을 작성하였다. 요구사항을 바탕으로 ER diagram을 작성하고, 이를 이용하여 Relational Schema를 작성하고 Normalization을 통해 redundancy와 anomaly를 제거한 후 마지막으로 SQL DDL을 작성한다.

M. Testing Plan

시스템이 의도한 방향으로 실행되고 시스템 내부의 결함을 찾기 위해 testing한다. 이를 위해 설계단계에 미리 계획한다. 이 때, Testing Plan에서는 Testing Policy와 여러 Test Case에 대해 기술한다.

N. Development Environment

Development Environment에서는 개발자의 환경에 대해 설명한다. 사용한 프로그래밍 언어와 IDE에 대해 서술한다.

O. Develop Plan

Develop Plan에서는 개발 계획에 대해 서술한다. 이 때, Gantt chart를 이용하여 개발 계획과 실제 계발 흐름에 대해 서술하고자 한다.

P. Index

Index에서는 본 문서의 표, 다이어그램 및 사진들의 인덱스를 표시한다. 이를 통해 원하는 정보가 문서의 몇 페이지에 있는지를 알 수 있다.

1.4. Version of the Document

본 요구사항 명세서의 각 버전 관리 정책, 버전 변경 기록, 그리고 문서의 변경사항들을 설명한다.

A. Version Format

본 요구사항 명세서의 버전 번호는 major.minor로 구성되며, 문서의 버전은 0.1부터 시작한다.

B. Version Management Policy

본 요구사항 명세서를 수정할 때마다 버전을 업데이트한다. 새로운 파트를 추가하거나 문서의 구성이 예전에 비해 큰 변화가 있을 경우 major number가 변경된다. 이미 완성된 파트를 변경할 때에는 minor number를 변경한다.

C. Version Update History

|  |  |  |
| --- | --- | --- |
| Version | Modified Date | Explanation |
| 0.1 | **2018.10.30** | 설계명세서 15개의 장에 대한 문서 목차 작성.  Preface, Introduction 초안 작성 및 완료 |
| 1.0 | **2018.11.02** | System Architecture, User Management System, Seller Exclusive System 작성 및 완료 |
| 2.0 | **2018.11.05** | Picture Search System, Filter Search System, Recommendation System, Trend System, Community System 초안 작성 |
| 2.1 | **2018.11.08** | Picture Search System, Filter Search System, Recommendation System, Trend System, Community System 수정사항 반영 |
| 3.0 | **2018.11.09** | Protocol Design, Database Design 초안 작성 |
| 4.0 | **2018.11.10** | Protocol Design, Database Design 수정사항 반영  Testing Plan, Development Environment, Develop Plan 작성 및 완료 |
| 4.1 | **2018.11.11** | 부분 수정 및 완료 |

Table 1. Version Update History

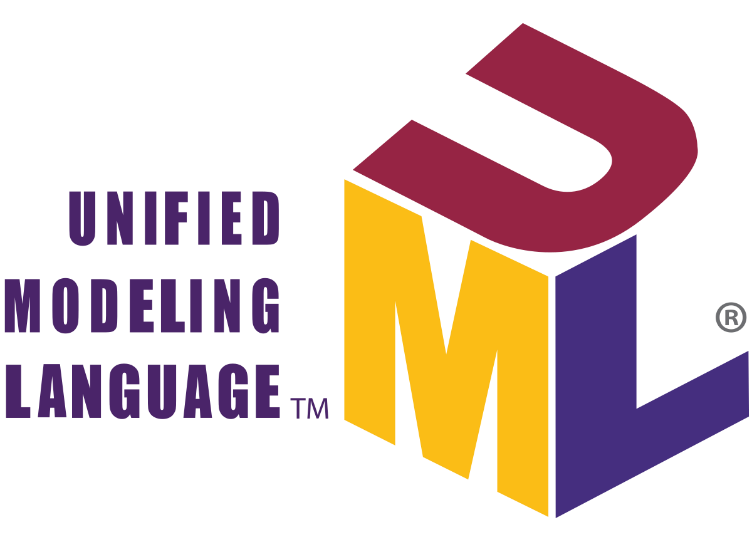
2. Introduction

2.1. Objective

Introduction에서는 본 문서에서 시스템 설계 시 사용하는 모든 종류의 다이어그램 및 툴에 관해 서술한다.

2.2. Applied Diagram

A. UML



Picture 1. UML Logo

United Modeling Language(이하, UML)은 객체지향 소프트웨어 설계를 위해 사용되던 여러 종류의 다이어그램들을 통합하여 만든 모델링 표기법으로, 현재 객체지향 시스템 개발 분야에서 가장 우수한 모델링 언어로 인식되고 있다. 그 이유는 UML이 개발자가 개발하려고 하는 소프트웨어를 구현하기에 앞서서 표준화되고 일반화되는 방식으로 설계되어 소프트웨어에 관련된 모든 사람들과의 상호 소통을 해줄 수 있는 메커니즘이 되기 때문이다.

UML이 있기 전, 시스템 개발 초창기에는 ‘운이 좋으면 성공하고, 그렇지 않으면 망한다.’라는 말이 있을 정도로, 개발자들이 개발하려고 하는 소프트웨어를 자신들만의 이해와 논리를 바탕으로 시스템을 구축하고 구현하였다. 이에 따른 결과로 개발된 시스템이 의뢰인의 입장에서 만족스럽지 않거나 의뢰인의 의도와 완전히 다른 시스템이 개발되는 경우가 부지기수였다. 이러한 결과는 시스템 의뢰인과 개발자 사이의 의사소통의 부재에 의한 것이었다.

오늘날에는 경쟁의 시대이다. 시스템 개발 초창기 때처럼 개발자가 의뢰인의 요구 조건을 임의로 이해하여 개발하는 것이 아닌, 시스템 구축 이전에 의뢰인과 개발자 사이의 충분한 의사소통을 통해 개발하려는 시스템의 기능과 동작을 확실하고 치밀하게 계획해야 한다. 또한 현재는 하나의 시스템을 개발하기 위해 수행하는 일이 다른 많은 팀이 요구된다. 여러 팀들 사이에서의 충분한 소통 및 시스템 설계 과정의 표준화 또한 치밀한 시스템 개발을 위해 필요로 되는 사항이다. 시스템 개발에서 분석가, 의뢰인, 개발자가 서로 오해 없이 이해할 수 있고, 시스템 설계를 일반적으로 표현할 수 있는 수단이 필요하다. UML은 바로 이러한 표준화 수단을 제공해준다.

UML은 개발할 시스템의 유형에 관계없이 모든 시스템에 대해 적용될 수 있다. 또한 시스템 개발 방법론, 개발에 사용되는 프로그래밍 언어, CASE 도구에 관계없이 적용될 수 있는 일반적인 표현 방법이기 때문에 UML을 이용하여 시스템 설계를 하였다.

위에서 소개한 UML은 객체 지향 설계를 위한 표준 언어로서, 소프트웨어 개발의 산출물을 시각화, 상세화, 구축, 문서화하는 특징을 가지고 있다. 이 4가지 특징에 대해 자세하게 설명한다.

1) UML은 시각화, 가시화 언어이다.

UML은 소프트웨어 개념 모델을 시각적인 그래픽 형태로 작성하며, 그 표기법에 있어서는 각 심볼에 대한 명확한 정의가 존재한다. 따라서 개발자들 사이에 오해 없는 원활한 의사소통이 이루어질 수 있으며, 여러 가지 시각적인 다이어그램을 통해 특정 관점으로 시스템을 표현할 수 있다.

2) UML은 상세화, 명세화 언어이다.

UML은 시스템 개발 과정인 분석, 설계, 구현 단계의 각 과정에서 필요한 모델을 정확하고 명백하고 완전하게 표현할 수 있는 수단이다. UML을 통해 시스템 설계의 복잡성을 명확하게 기술한다.

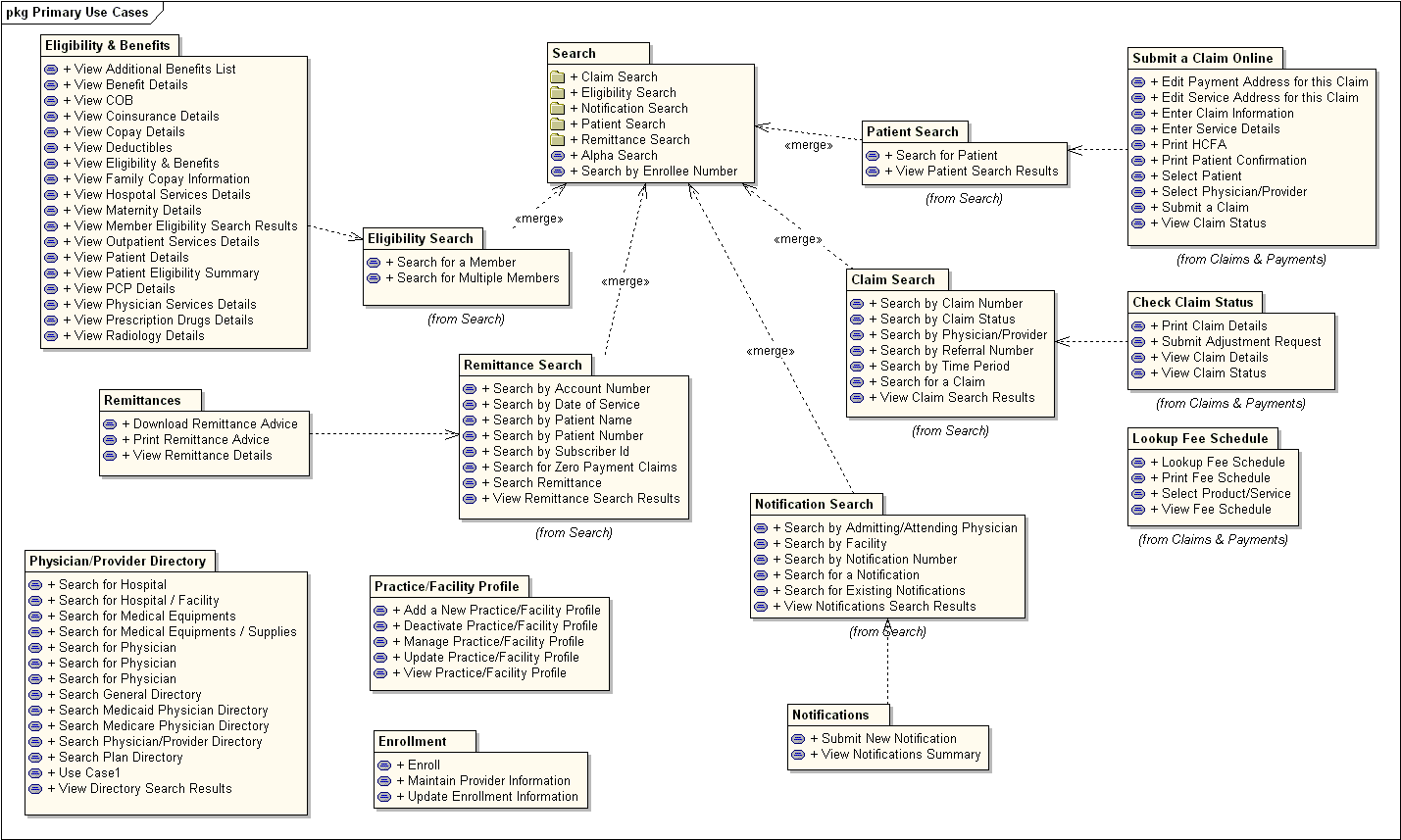
3) UML은 구축 언어이다.

UML은 C++, Visual Basic, Java 등과 같은 다양한 프로그래밍 언어로 표현할 수 있다. 따라서 UML로 상세화 된 설계 모델은 프로그래밍 소스 코드로 변환하여 구현이 가능하다. 또한 이미 구현되어 있는 소스 코드를 UML로 역변환하는 데 이용할 수 있어서 분석하는 역공학(Reverse Engineering)이 가능하다.

4) UML은 문서화 언어이다.

UML은 System Architecture와 이에 대한 모든 상세한 내역들을 문서로 작성한다. 이를 통해 요구사항 명세서, 프로세스, 소스 코드, Prototype, Release 등을 수행할 수 있다. 또한 요구사항을 표현하고 시스템을 테스트하는 언어도 제공한다.

B. Package Diagram

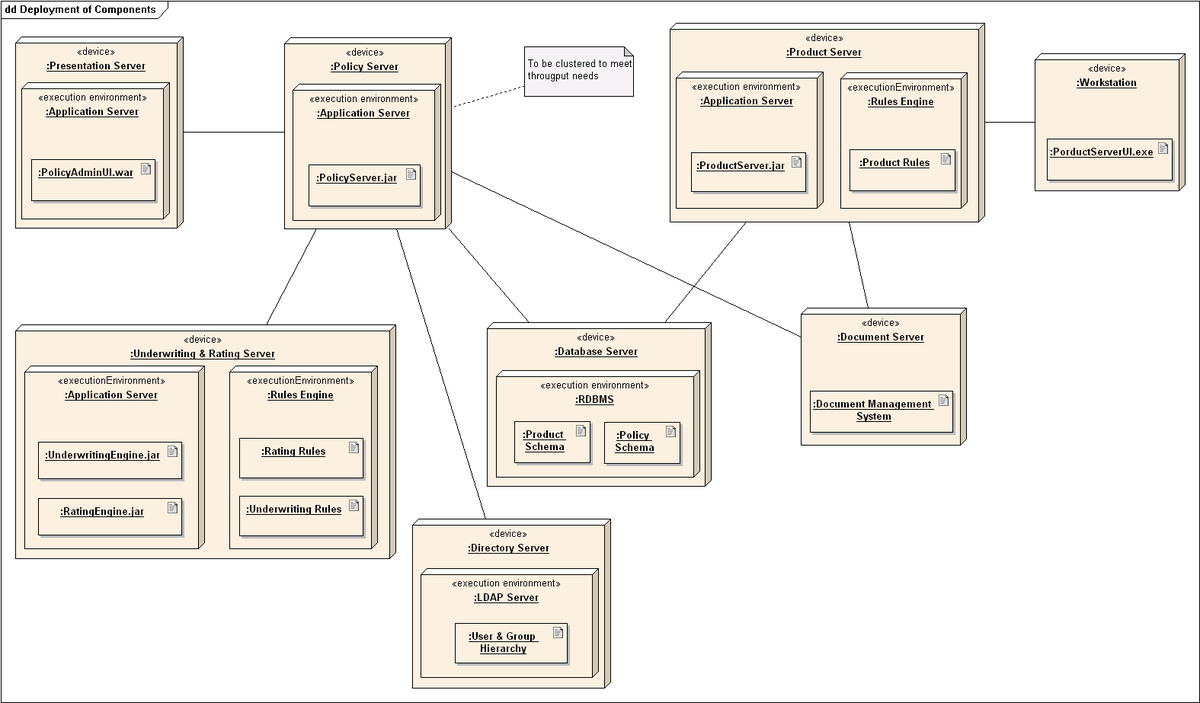


Picture 2. Package Diagram Example

복잡한 시스템을 이해하는 방법은 추상적인 개념들을 하나의 그룹으로 묶어서 이해하는 것이다. 이렇게 만든 그룹은 클래스와 같은 추상 개념으로 많은 인스턴스들을 가지고, 오로지 시스템을 이해하기 위한 목적으로만 존재한다. UML에서 시스템을 이해하기 위한 목적으로 추상적인 개념들을 모은 하나의 그룹을 패키지(Package)라고 한다. 패키지는 요소들의 그룹으로 조직하기 위한 범용 메커니즘으로 모델의 요소들을 조직하고 이해할 수 있도록 해준다. 패키지에 안에 담기는 것은 클래스에만 국한되는 것은 아니며, Use Case, Activity Diagram 등도 담을 수 있고, 다른 패키지도 담을 수 있다. 패키지를 구성할 때에는 여러 사람이 동의할 수 있는 형태로 구성되어야 하며, 패키지의 구성과 이름 체계는 개발자들이 쉽게 이해하고 사용할 수 있어야 한다.

패키지 내부의 모든 클래스들은 개념적, 기능적, 변화적, 관리적 측면에서 유사한 면을 가진다. 하나의 패키지 내부의 클래스들은 밀접한 관련성을 가지며, 다른 패키지의 클래스들과는 약한 의존관계가 있다. 이와 같이 패키지 다이어그램은 패키지 삽입 및 패키지 확장을 포함하여 모델 요소들을 그룹화(패키지화)함으로써 조직 및 요소의 독립성을 묘사한다. 또한 패키지 다이어그램은 요소들간의 관계를 시각화하여 제공한다.

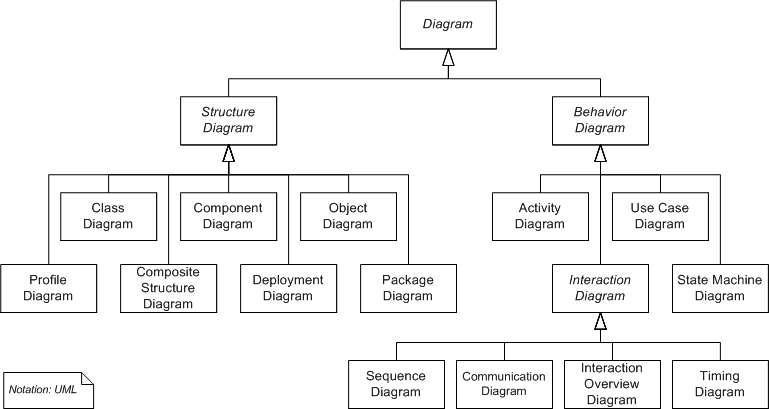
C. Deployment Diagram



Picture 3. Deployment Diagram Example

Deployment Diagram은 네트워크, 하드웨어 또는 소프트웨어들을 실행 파일 수준의 컴포넌트들과 함께 표현한 다이어그램이다. 이들은 시스템을 구성하는 하드웨어간의 연결 관계를 표현하고, 하드웨어 자원에 대한 소프트웨어 컴포넌트의 배치 상태를 표현한 다이어그램이다. 즉, 시스템이 실행되는 환경인 노드와 그 노드에 배치된 컴포넌트의 구성을 나타내는 다이어그램이다. 노드는 처리 능력을 가진 장치를 의미하며, Deployment Diagram에서 직육면체로 표기한다. Deployment Diagram은 시스템의 설계 단계의 마지막에 작성되며, 이는 모든 설계가 마무리되어야 소프트웨어의 컴포넌트가 정의되고, 시스템의 하드웨어 사양도 확정되기 때문이다. 배치 다이어그램은 소프트웨어 시스템이 배치, 실행될 하드웨어의 자원들을 정의하고, 소프트웨어의 컴포넌트가 어떤 하드웨어 자원에 탑재되어 실행될지를 정의하는 목적을 가지고 있다.

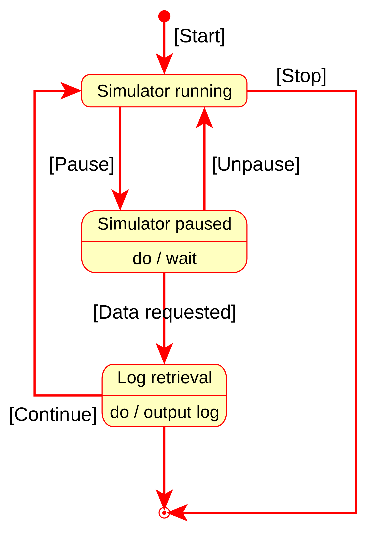
D. Class Diagram



Picture 4. Class Diagram Example

Class Diagram은 객체지향 설계에서 가장 널리 사용되는 다이어그램으로, 시스템의 정적인 상태인 논리적인 구조(클래스)를 표현하는 다이어그램이다. 클래스(Class)란 객체지향 프로그래밍에서 속성(Attribute)과 행위(Method, Member Function)를 갖는 하나의 객체 단위이다. 시스템의 클래스와 클래스 Attribute, 클래스들의 관계를 나타낼 수 있기 때문에 객체지향 시스템의 상속 등의 관계를 명확하게 표현할 수 있다. 또한 코드 생성의 직접적인 원인이 되기 때문에 프로그래밍 개념과 같은 의미의 표현을 통해 도식화한다.

E. State Diagram

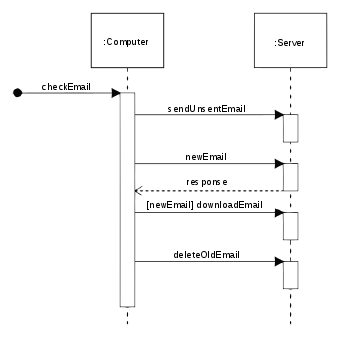


Picture 5. State Diagram Example

State Diagram은 객체지향 모델에서 클래스의 인스턴스 사건(Event)에 의거한 시스템의 전체적인 작동을 상세히 기술하는 UML의 다이어그램이다. State Diagram은 상태의 변화에 의한 동작 또는 하나의 상태에서 다른 상태로 변화되게 하는 사건의 주어진 시간 동안의 상태를 나타낸다. 이러한 표기는 실제 구현에서 UI를 정의하는 데 도움을 준다.

상태 다이어그램은 어떤 이벤트에 대한 반응적인(Reactive) 특성을 가지는 객체에 대해 기술하기 위해 이용되며, 객체가 갖는 여러 가지 상태를 표현한다. 객체는 기존 상태에 따라 동일한 메시지에 대해서 다른 행동을 보이기 때문에, 상태 다이어그램을 통해 시스템 내의 객체가 가질 수 있는 상태가 어떤 것이 있는지에 대해, 또 각 상태를 나타낼 때 특정 이벤트에 대해 어떤 반응을 보일지에 대해 기술한다.

F. Sequence Diagram



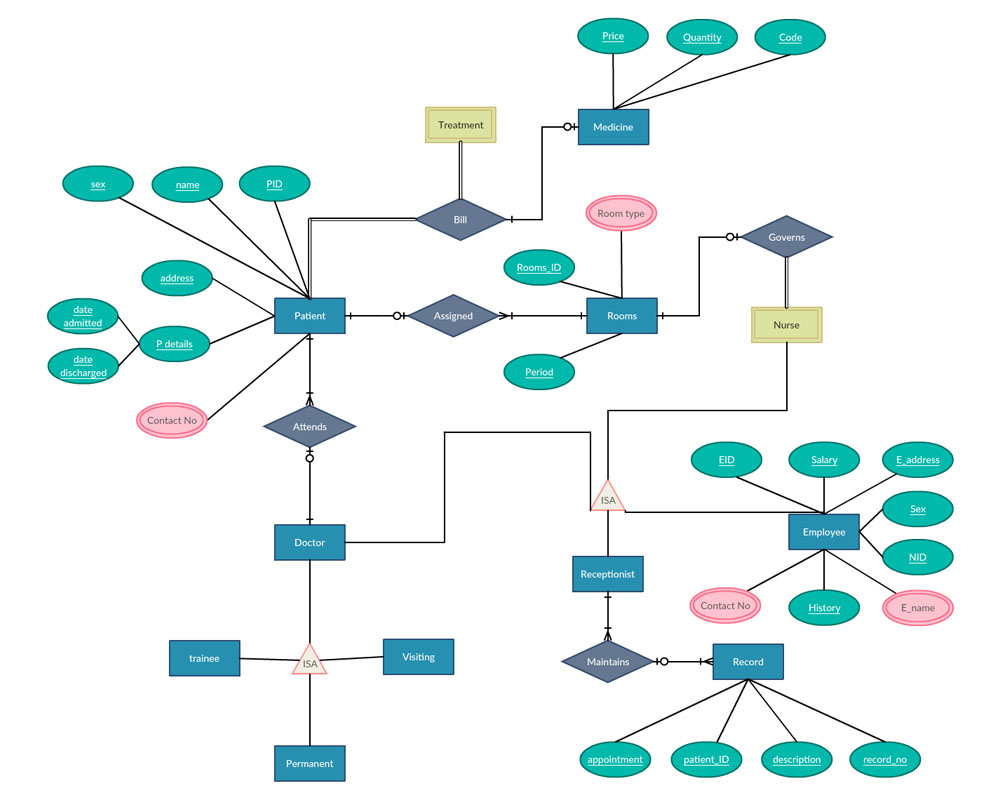
Picture 6. Sequence Diagram Example

Sequence Diagram은 시스템 내에서의 각 컴포넌트들이 주고받는 메시지의 흐름을 시간 순차적으로 표현하는 상호작용 다이어그램이다. Sequence Diagram은 Use-Case Diagram과 자주 같이 사용되는데, 이는 각 Use-Case를 상세히 표현하는데 Sequence Diagram을 사용하는 것이 유리하기 때문이다. 각 컴포넌트들의 상호작용 명확히 보이기 때문에 각 컴포넌트간의 관계와 각 컴포넌트들이 갖고 있는 속성, 행동들을 더욱 명확히 할 수 있다. 따라서 Sequence Diagram은 시스템 내부의 동적인 움직임을 표현해주는 다이어그램이기 때문에 속성 및 함수로 이루어진 Class Diagram을 동적 프로그래밍으로 설계하는 중요한 과정이다.

Sequence Diagram에서는 객체의 오퍼레이션과 속성을 상세히 정의한다. 이는 객체간 상호작용을 정의하는 과정에서 객체들이 가져야 하는 오퍼레이션과 속성을 구체적으로 정의할 필요가 있기 때문이다. 객체는 다른 객체가 의뢰하는 일을 처리하는 객체의 책임으로서, 객체의 책임은 오퍼레이션으로 정의되어야 하며, 이 행위를 위해 필요한 객체의 속성도 정의되어야 한다.

Sequence Diagram은 상호작용 다이어그램의 일종으로, 시스템의 동적 측면을 모델링 하기 위한 용도로 사용되며, 객체들과 그 사이의 교류 및 메시지를 보여주는 역할을 한다. Sequence Diagram은 Actor, 객체, 메시지, 회귀 메시지, 제어 블록 등으로 구성된다.

G. ER Diagram

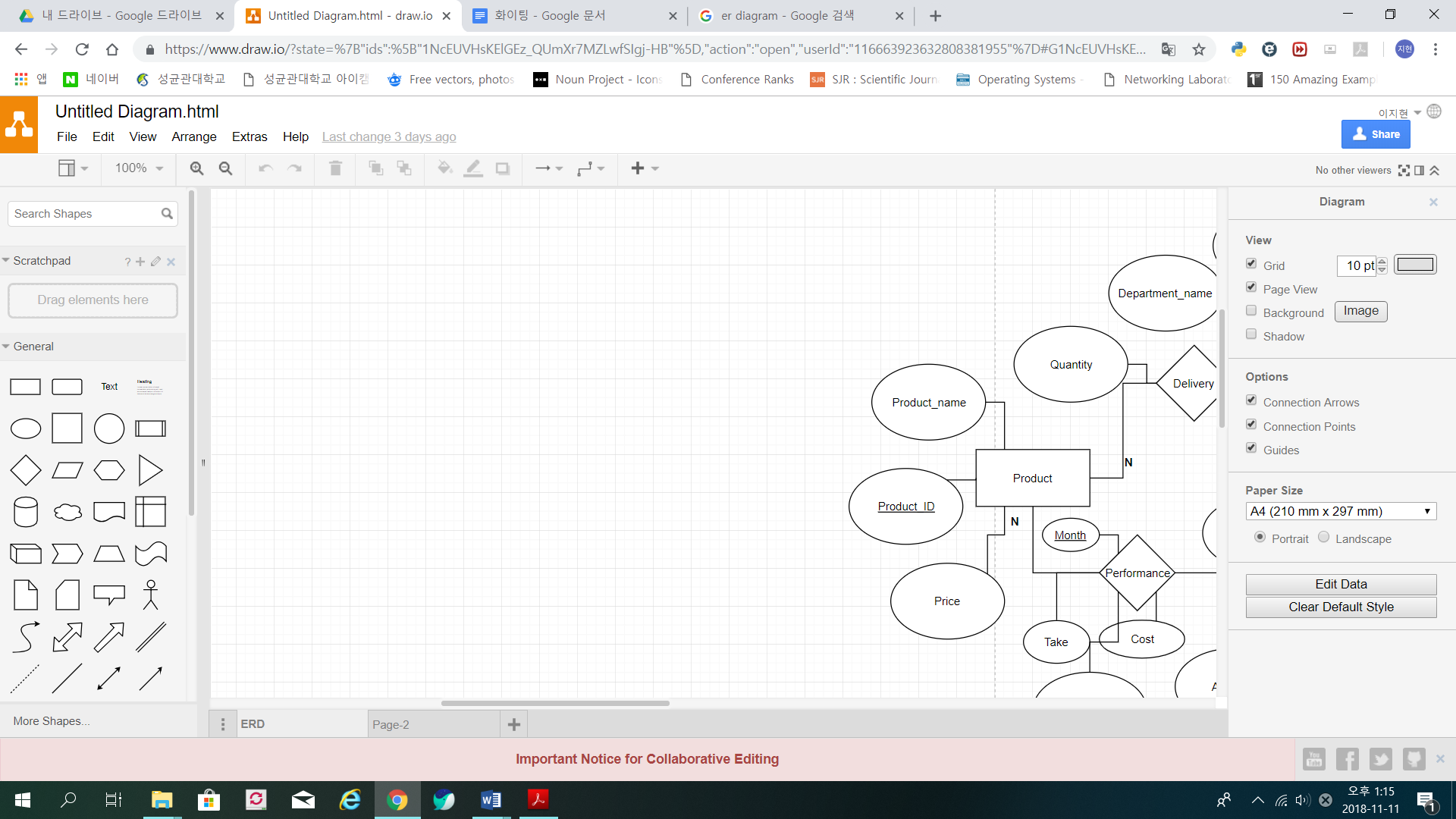


Picture 7. ER Diagram Example

ER Diagram은 데이터베이스에서 각 개체들의 관계를 표현하기 위해 사용하는 다이어그램으로 UML에 포함되는 다이어그램은 아니다. 데이터의 저장 공간인 데이터베이스에 저장되는 데이터의 구조 및 그에 수반한 제약 조건들은 다양한 기법에 의해 정의될 수 있다. 그 기법 중 하나가 개체-관계 모델링(Entity-Relationship Modelling)이며, ERM 프로세스의 산출물이 ER Diagram(Entity-Relationship Diagram)이다. ER Diagram에서 개체(Entity)란 현실 세계의 객체로서 유형 또는 무형의 정보를 대상으로 서로 구별할 수 있는 것이며, 관계(Relationship)란 두 개 이상의 개체 사이에 연관성을 기술한 것이다. 따라서 ER Diagram은 조직의 정보 자원(Resource)을 전반적으로 계획하는데 있어서 필수적이며 유용한 도구이다.

2.3. Applied Tool

A. Flow Chart Maker & Online Diagram Software



Picture 8. Flow Chart Maker & Online Diagram Software

본 문서에 첨부되어 있는 ER Diagram을 제외한 모든 시스템 설계도와 diagram의 작성은 ‘Flow Chart Maker & Online Diagram Software’ tool을 사용하였다. ‘Flow Chart Maker & Online Diagram Software’ tool은 특별한 가입 없이 이용할 수 있는 소프트웨어로 온라인 상으로 지원하기 때문에 이용에 편리하다. 또한 다양한 diagram 및 flow chart를 그리는 데 최적화되어 있고, 현 시스템 개발 문서를 작성하고 공유하는 데 사용되었던 구글 드라이브를 지원하기 때문에 진행 과정을 저장 및 공유에 편리하다.

B. Amazon Web Services(AWS)



Picture 9. Amazon Web Service(AWS) Logo

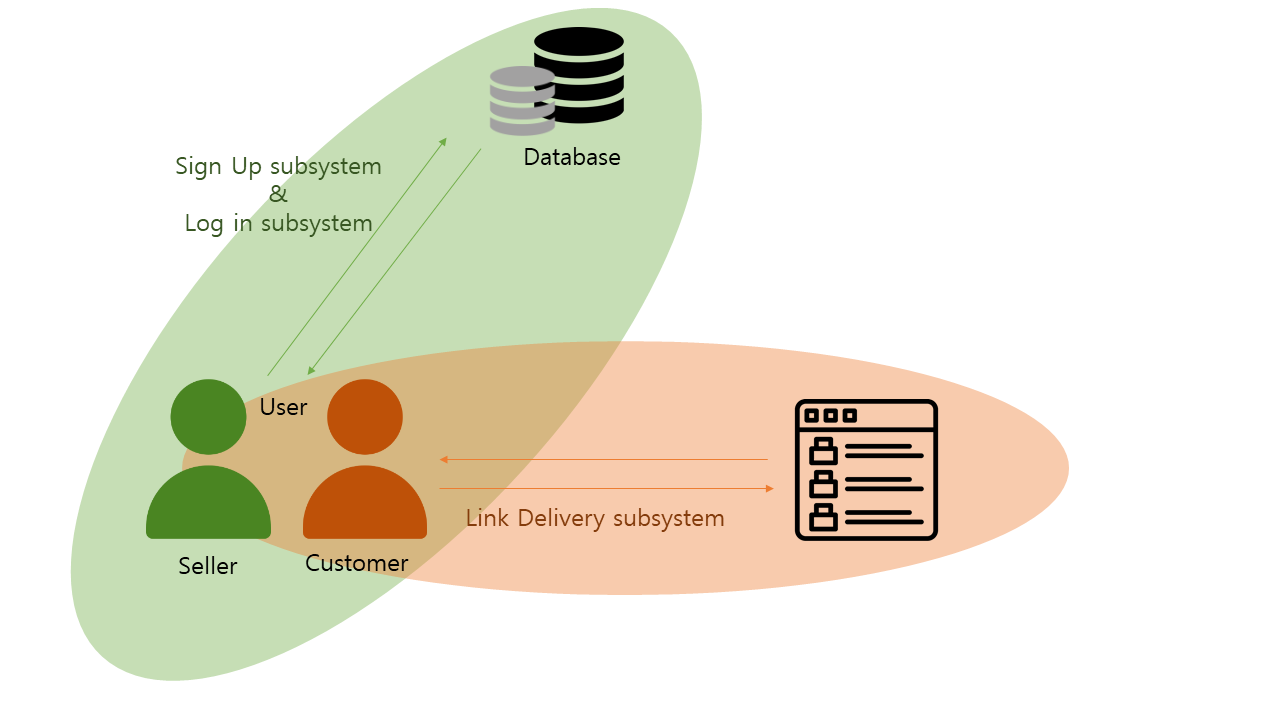
Amazon web service(AWS)는 대표적인 Cloud Service 중 하나이다. AWS는 Java, PHP, Node.js등의 다양한 플랫폼을 제공하고 있으며, 비교적 저렴한 가격에 웹 서버 구축을 지원하고 있다. 또한 추가적으로 컴퓨팅 파워를 증가시킬 수 있어 차후 유지보수에도 유리하다.

2.4. Project Scope

‘FaReDy’ 시스템은 기존의 존재하던 온라인 쇼핑 시스템, 이미지 검색 시스템과 커뮤니티 시스템을 기반으로, 추가적인 몇 가지 기능을 제공하여 사용자의 편리를 돕는 시스템이다. ‘FaReDy’ 시스템은 크게 7개의 시스템으로 구성되어 있다.

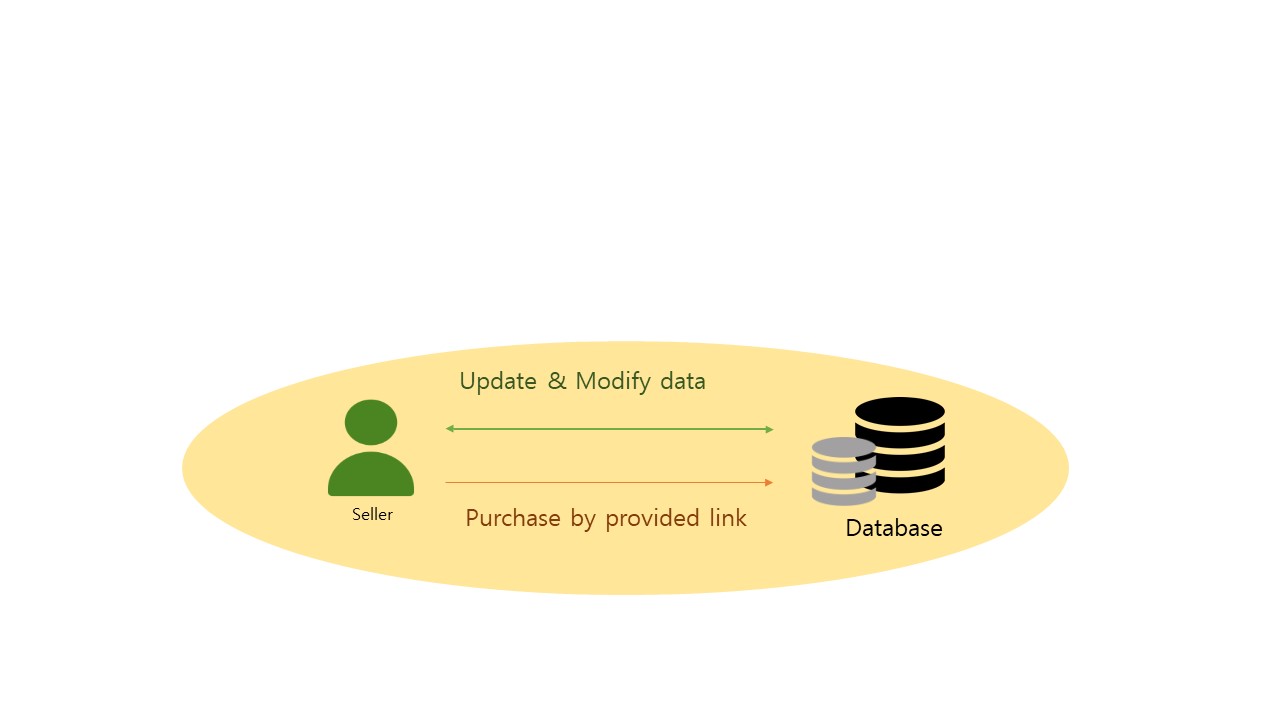
먼저 기존의 온라인 쇼핑 시스템에서 제공하는 기능을 기반으로 설계된 시스템으로는 User Management System과 Seller Exclusive System, 그리고 Filter Search System, Community System이 있다.

User Management System의 경우, 사용자의 이용과 관련된 시스템으로 다시 3개의 하위 시스템으로 나뉜다. 사용자의 회원가입을 관리하는 Sign Up subsystem과 로그인 기능을 제공하는 Log In subsystem, 그리고 구매자에게 product 쇼핑 link를 제공하는 Link Delivery subsystem으로 각 하위 시스템의 구조를 같다.

****

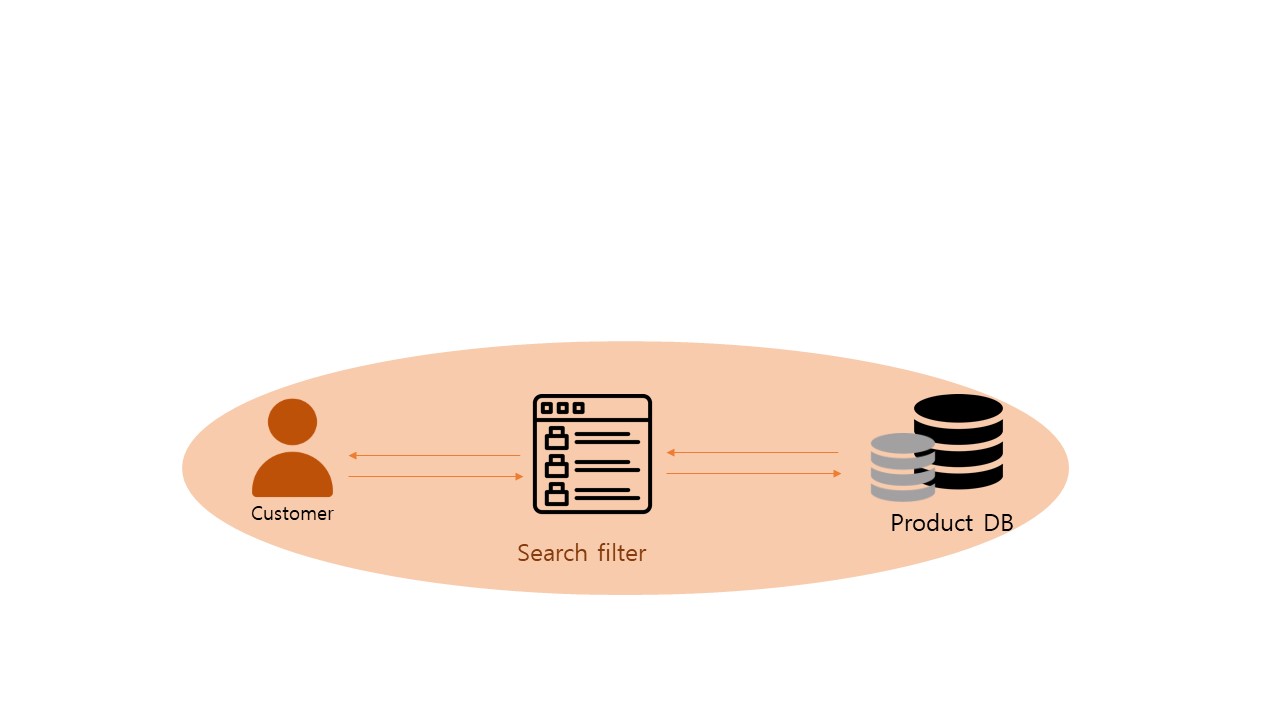
Picture 10. User Management System

Seller Exclusive System의 경우, 따로 서브시스템은 존재하지 않고 Data를 수정하거나 업데이트 할 수 있고 상품에 대한 link를 제공해야 한다.



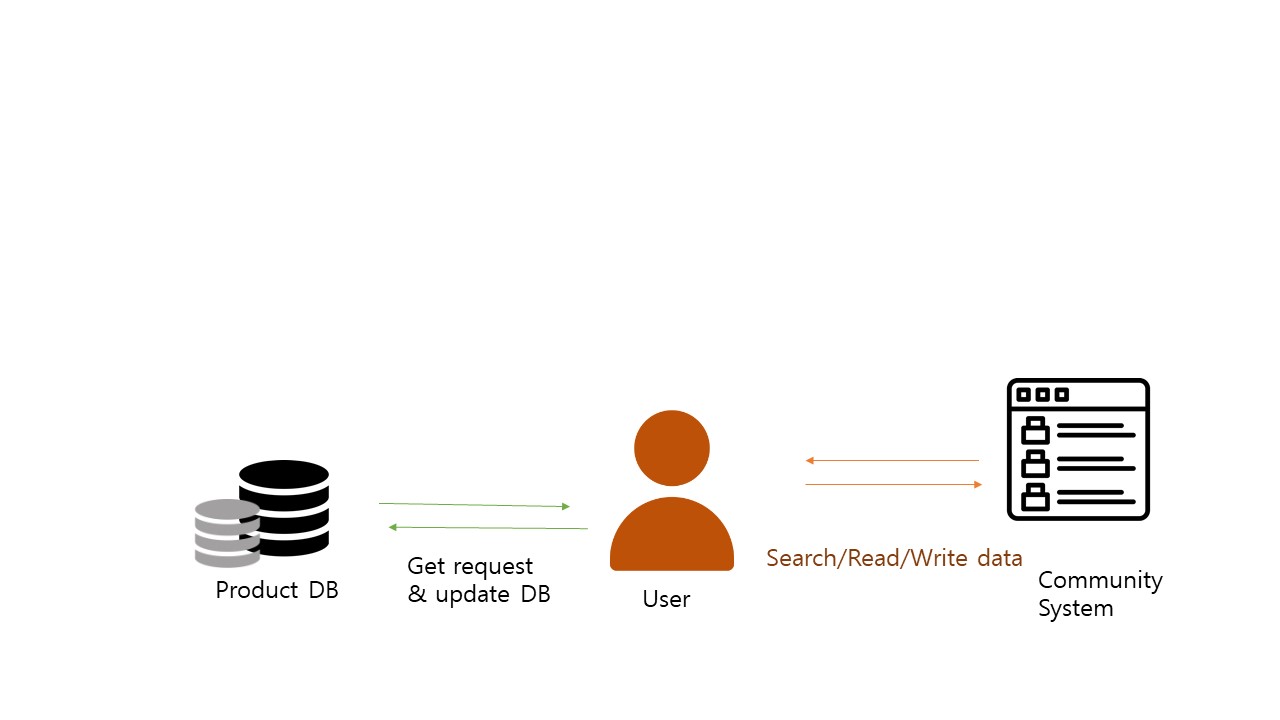
Picture 11. Seller Exclusive System의 구조

Filter Search System의 경우, customer가 다양한 검색 필터를 이용하여 제품을 검색하는 기능에 관련한 시스템으로 시스템의 구조는 다음과 같다.



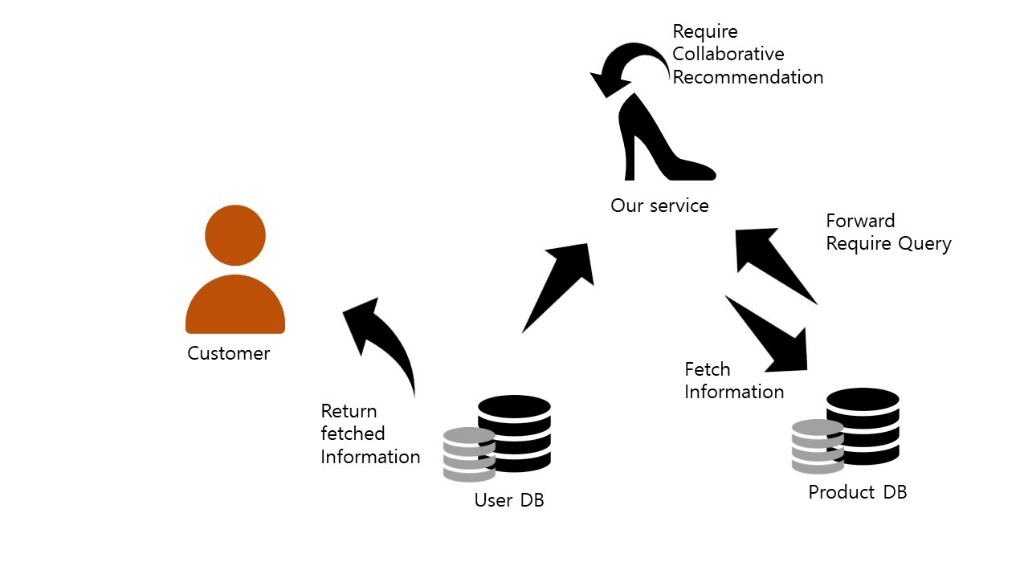
Picture 12. Filter Search System의 구조

Community System은 시스템 사용자(customer, seller(의 커뮤니티 사용과 관련된 시스템으로 시스템의 구조는 다음과 같다.



Picture 13. Community System의 구조

Recommendation System의 경우, Picture Search System에서 추출한 feature들을 종합해서 각 customer별로 추구하는 옷들을 추천해 준다. 시스템의 구조는 다음과 같다. Trend System은 Recommendation System의 결과를 종합하여 알려주기 때문에 시스템의 구조는 아래와 같다고 볼 수 있다.



Picture 14. Recommendation System & Trend System의 구조

3. System Architecture

3.1. Objective

System Architecture에서는 우리 시스템이 어떻게 구성되는지를 개략적으로 보일 것이다. System Organization에서는 우리의 시스템과 외부 환경이 어떻게 관계하는지를 Context model과 Block diagram, Activity diagram, Package diagram과 Deployment diagram들을 통하여 제시하고 있다. 그러나 이 장은 전체 시스템에 대해 개괄하는 것이며, 각 서브시스템들의 자세한 명세는 이후 4~10장에 걸쳐 설명한다.

3.2. System Organization

A. Context Model

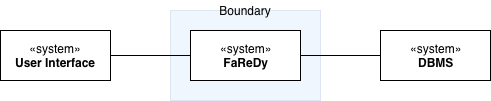


Diagram 1. Context Model

우리의 시스템은 다음과 같은 boundary를 갖는다. 고객과 판매자는 User Interface를 통해 우리의 시스템에 질의를 하거나 서비스를 제공받는다. 우리의 시스템은 고객과 판매자의 질의를 처리하기 위해 DBMS와 연결하여 데이터를 관리한다.

User Interface는 기존에 존재하는 Web browser를 이용하며 DBMS는 MySQL을 이용한다. 따라서 우리의 시스템의 구현은 명세된 FaReDy의 system들로 제한된다.

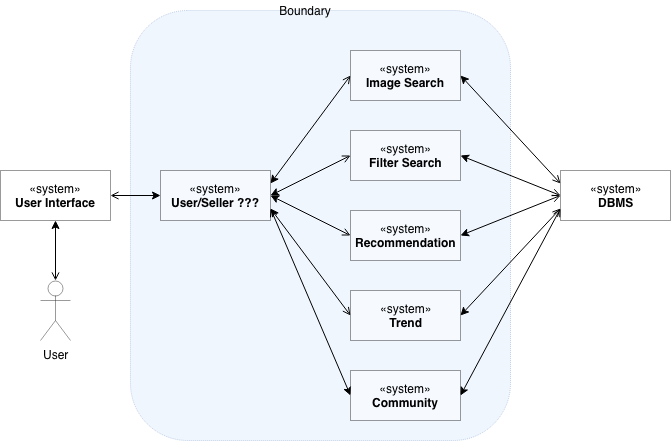


Diagram 2. Block Diagram

‘FaReDy’ 시스템은 Client-Server Model을 이용하고 있다. Client는 User와, User가 사용하는 User Interface를 의미한다. Server는 Boundary내부의 FaReDy 시스템과 외부의 DBMS를 포함하여 구성된다.

그림 2에서 보이는 바와 같이 FaReDy는 6 종의 system 구성을 통해 서비스를 제공한다.

B. User/Seller System

User/Seller System은 FaReDy를 사용하는 사용자들의 계정 관리 등을 제공한다. 뿐만 아니라 FaReDy가 제공하는 서비스를 user interface로 배포하기 위한 Web server 역할을 한다.

**B.1. Sign Up**

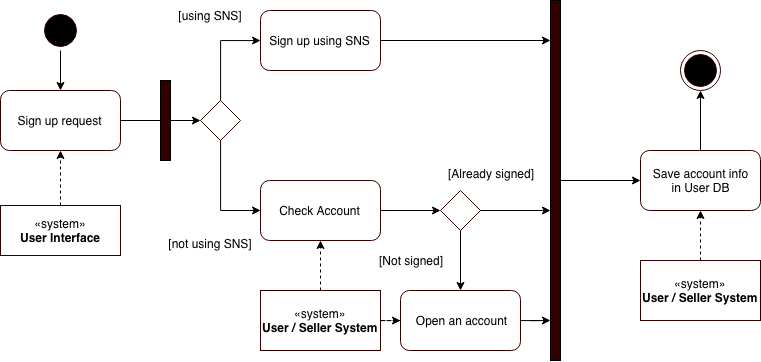
****

Diagram 3. Activity Diagram - Sign up

**B.2. Log In**

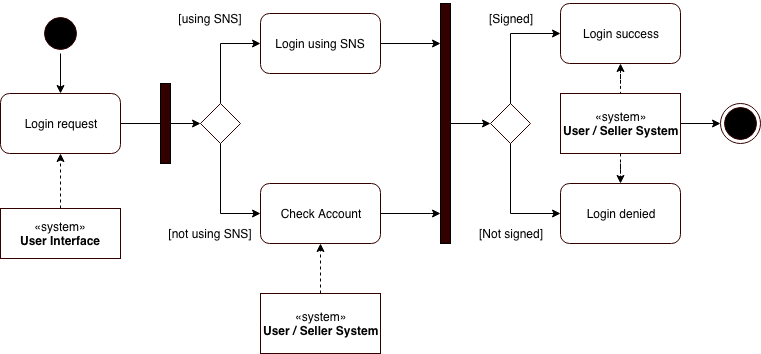
****

Diagram 4. Activity Diagram - Log in

**B.3. Product Management**

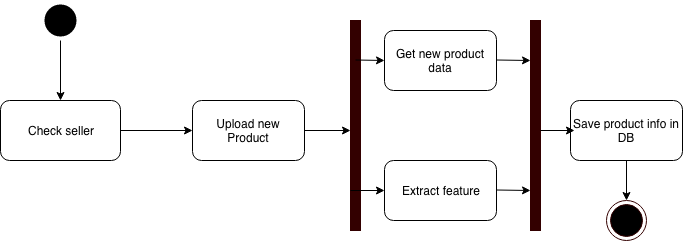
****

Diagram 5. Activity Diagram - Product Management

C. Picture Search System

Picture Search System은 사진을 기반으로 옷을 검색해주는 역할을 한다. Customer는 자신의 마음에 드는 옷이 찍힌 사진을 user interface를 통해 ‘FaReDy’로 사진을 업로드하며, 업로드된 사진을 바탕으로 ‘FaReDy’의 Picture Search System은 사진으로부터 옷의 feature들을 추출하고, 이와 유사한 옷을 검색하여 customer에게 제공한다.

C.1. Picture Search

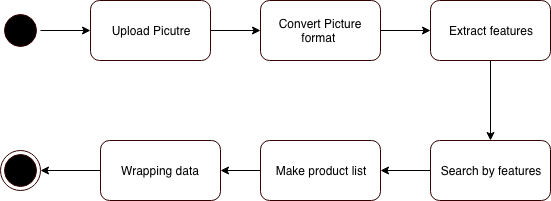


Diagram 6. Activity diagram - Picture Search

D. Filter Search System

Filter Search System은 customer가 자신이 원하는 옵션(feature)을 바탕으로 옷을 검색하는 시스템이다. Customer는 자신이 찾고자 하는 옷의 옵션(feature)을 직접 선택하여 FaReDy에게 검색을 요청할 수 있다. Filter Search System은 제안된 옵션을 바탕으로 옷을 검색하고 customer에게 검색 결과를 제공한다.

D.1. Filter Search

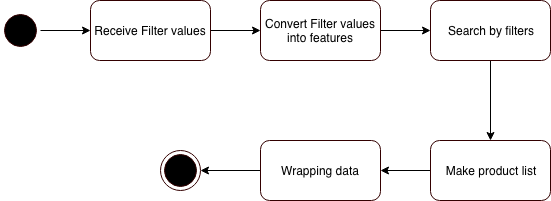


Diagram 7. Activity diagram - Filter Search

E. Recommendation

Recommendation 시스템은 customer의 평소 Search 기록을 바탕으로 개인화된 추천을 제공하는 시스템이다.

**E.1. Serving Recommendation**

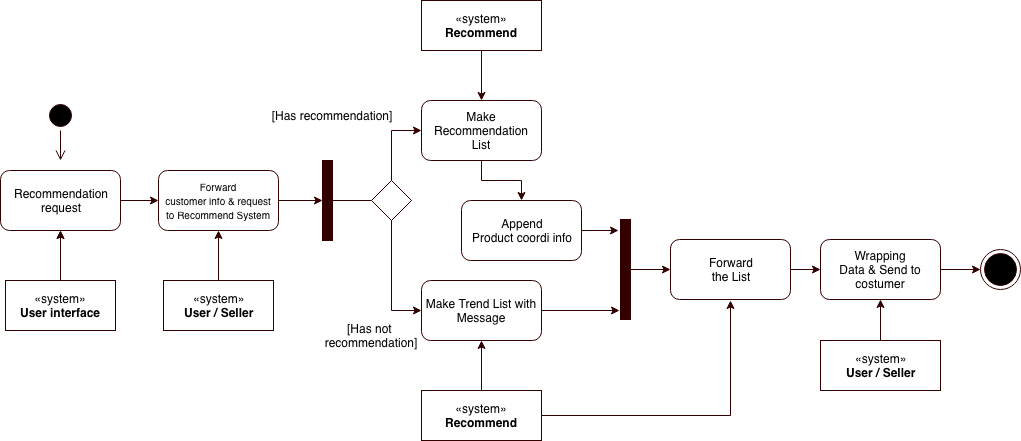
****

Diagram 8. Activity diagram - Serving Recommendation

**E.2. Batch Processing**

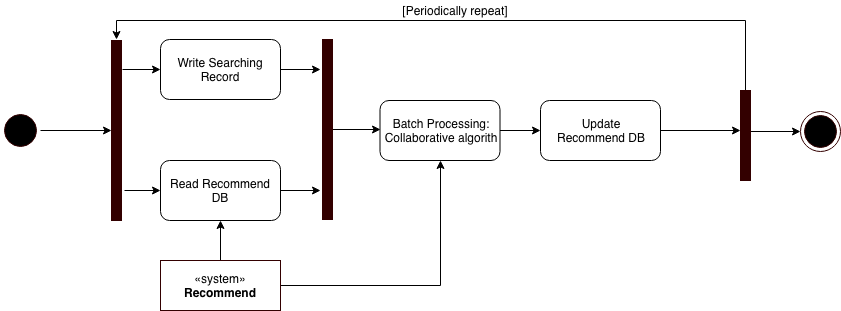
****

Diagram 9. Activity diagram - Batch Processing

3.3. Package Diagram

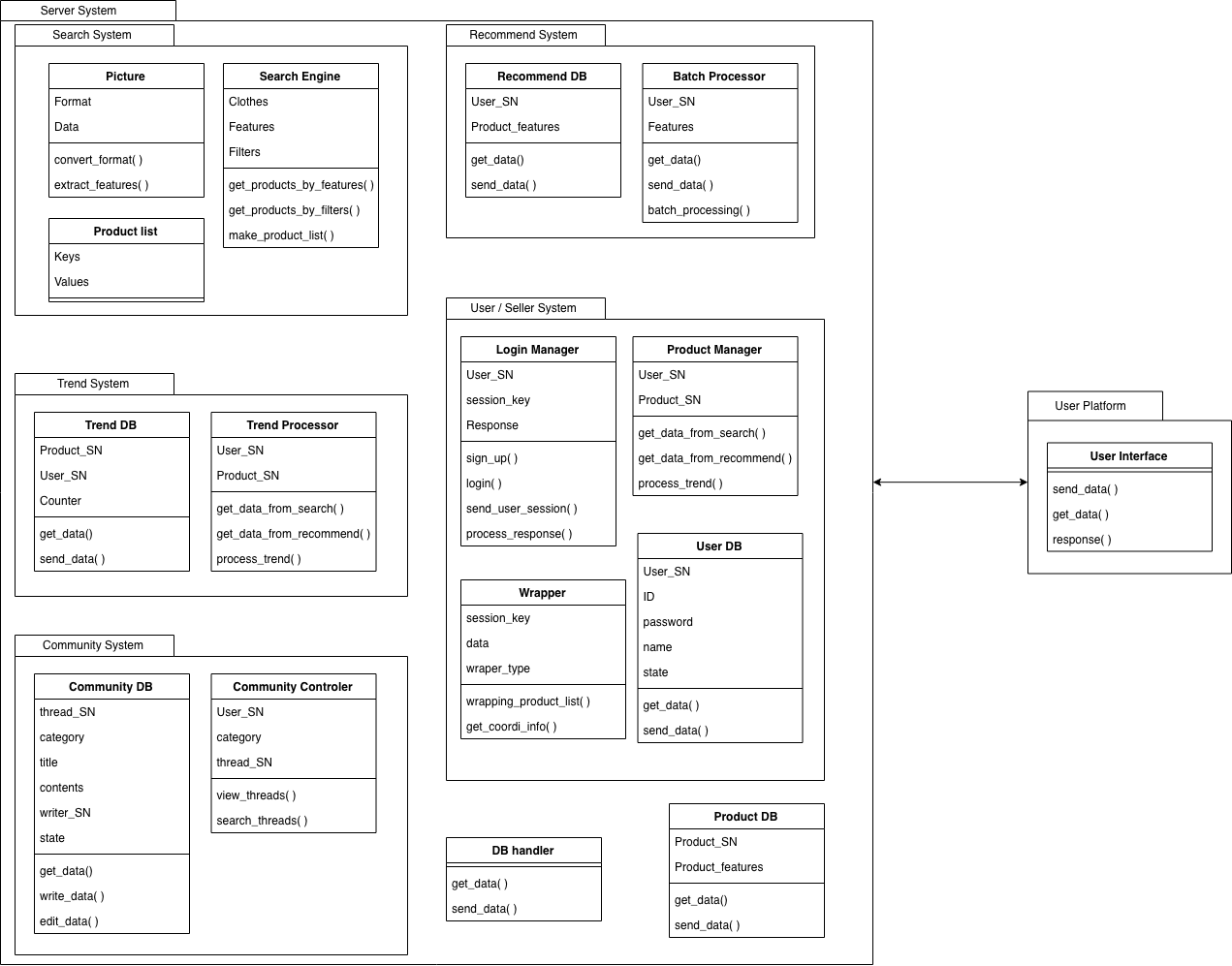


Diagram 10. Package Diagram

3.4. Deployment Diagram

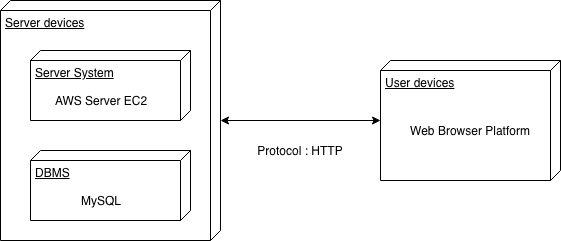


Diagram 11. Deployment Diagram

4. User Management System

4.1. Objective

쉽게 사용자가 사용하며 회원 가입과 로그인 기능을 통해 고객이 서비스를 이용하는 도중 발생하는 데이터 처리 및 고객 관리 시스템의 설계를 설명한다. Class diagram, Sequence diagram, State diagram을 통해 User management system의 구조를 나타낸다.

4.2. Class Diagram

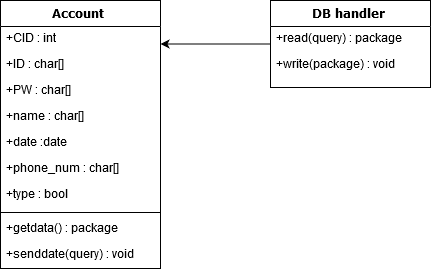
****

Diagram 12. Class Diagram - User Management System

1. DB Handler
2. **Attributes**
3. 해당사항 없음.
4. **Methods**
5. +package Read(query): DB에서 요청된 data를 읽어옴.
6. +void Write(getData): DB에서 data 저장.
7. Account
8. **Attributes**
9. +CIN: 고객 고유번호 값
10. +ID: Account ID 값
11. +PW: 비밀번호
12. +name: 고객 이름
13. +type: buyer/customer 구분
14. +data: 가입 날짜
15. +phone\_num: 고객 번화번호 정보
16. **Methods**
17. +package getData(): DB에서 data 수신
18. +void sendData(query): DB로 data 전송

4.3. Sequence Diagram

A. Sign Up using SNS(Naver/Facebook)

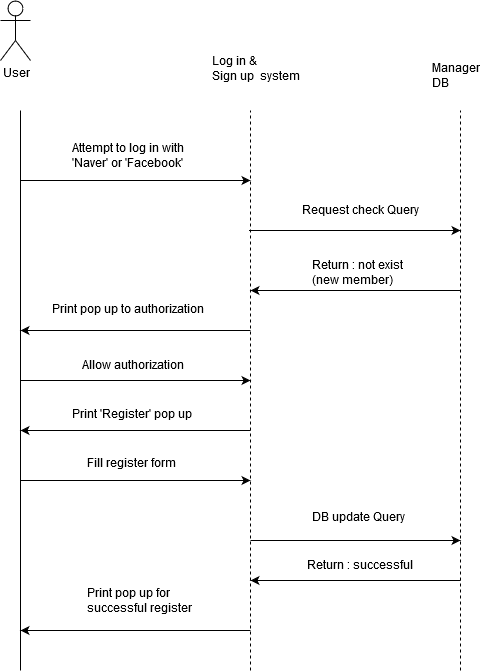


Diagram 13. Sequence Diagram - Sign Up using SNS(Naver/Facebook)

B. Log In with using SNS

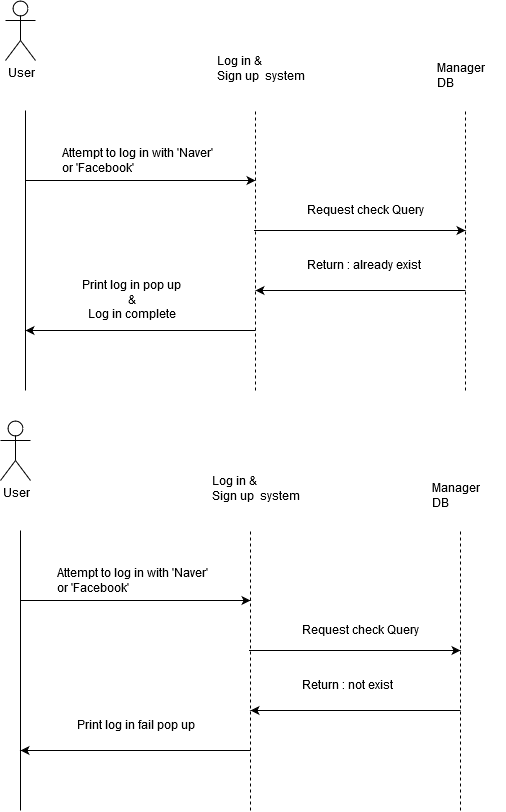


Diagram 14. Sequence Diagram - Log In with using SNS

1. Sign Up without SNS

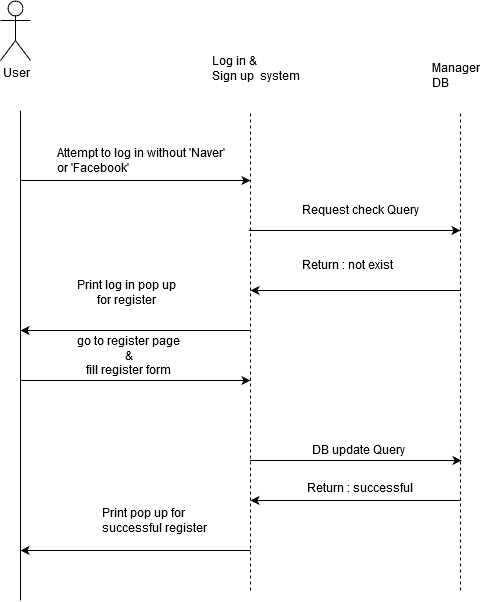


Diagram 15. Sequence Diagram - Sign Up without SNS

1. Log In without SNS

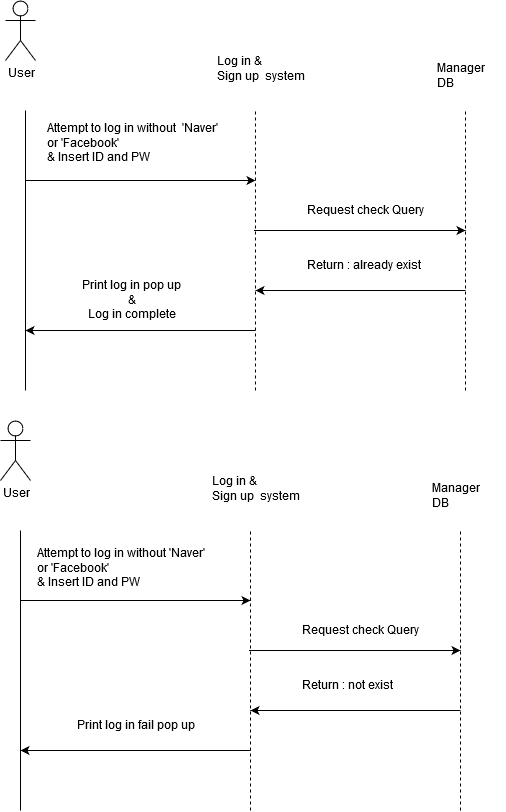


Diagram 16. Sequence Diagram - Log In without SNS

4.4. State Diagram

A. Sign Up using SNS

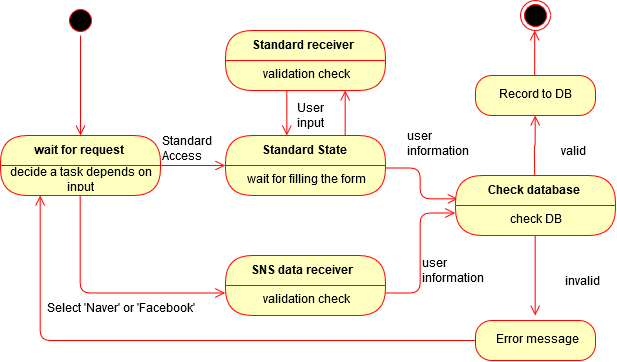


Diagram 17. State Diagram - Sign Up using SNS

B. Sign Up without SNS

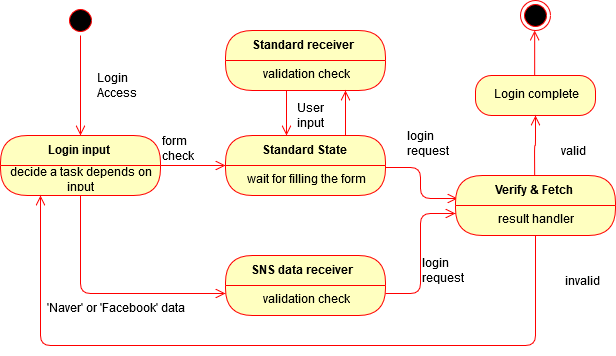


Diagram 18. State Diagram - Sign Up without SNS

5. Seller Exclusive System

5.1. Objectives

판매자와 관련된 시스템 설계를 설명한다. Seller Exclusive System은 크게 3개의 하위 시스템으로 나뉜다. 이는 각각 Product, Group purchasing, Link sending으로 나뉘어 Product에서는 제품의 serial number, type, name, stock으로 DB에 업로드한다. Group purchasing에서는 구매하고자 하는 고객들의 CID, Product name을 불러오도록 한다. 마지막으로 Link에서는 고객이 구매를 하고자 할 때, 판매자의 구매 경로 즉, link을 제공해 주도록 한다.

5.2. Class Diagram

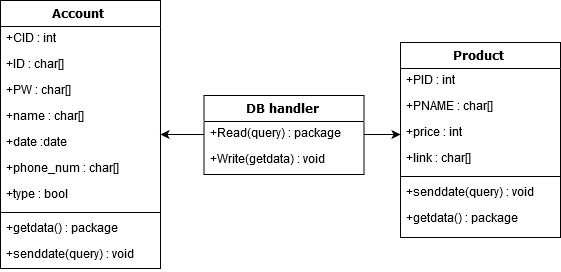


Diagram 19. Class Diagram - Seller Exclusive System

1. DB Handler
2. **Attributes**
3. 해당사항 없음.
4. **Methods**
5. Package Read(query) DB에서 요청된 data 읽어옴.
6. void Write(getData) DB에 data 저장.
7. Account
8. **Attributes**
9. +CIN: 고객 고유번호 값
10. +ID: Account ID 값
11. +PW: 비밀번호
12. +name: 고객 이름
13. +type: buyer/customer 구분
14. +data: 가입날짜
15. +phone\_num: 고객 전화번호 정보
16. **Methods**
17. +package getData(): DB에서 data 수신
18. +void sendData(query): DB에서 data 전송
19. Product
20. **Attributes**
21. +PIN: product 고유번호 값
22. +PNAME: product 이름
23. +price: product 제품 판매가
24. +link: product 구매 링크
25. **Methods**
26. +getData(): DB에서 data 수신
27. +sendData(): DB로 data 전공
28. +updateData(): data업데이트

5.3. Sequence Diagram

A. Product Management

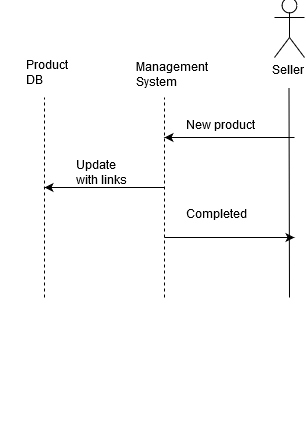


Diagram 20. Sequence Diagram - Product Management

B. User Purchase System

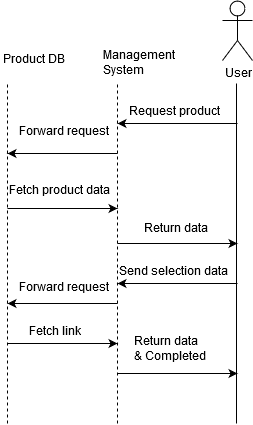


Diagram 21. Sequence Diagram - User Purchase System

5.4. State Diagram

A. Product Management System

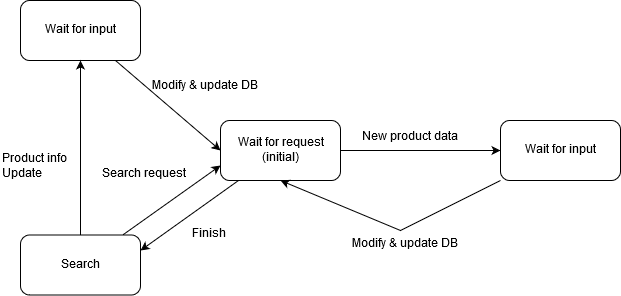


Diagram 22. State Diagram - Product Management System

B. User Purchase System



Diagram 23. State Diagram - User Purchase System

6. Picture Search System

6.1. Objectives

Picture Search System이 제공하는 기능들을 설명한다. Picture Search System은 ‘FaReDy’의 주요 기능 중 하나인 ‘사진은 이용한 옷 검색 서비스’를 제공한다. 옷 검색 서비스를 제공하기 위해 본 시스템은 Interface, Picture Convertor, Feature Extractor, Search Engine, DB handler로 5개희 subsystem으로 구성된다. Class diagram, Sequence diagram, State diagram을 통해 Picture Search System의 구조를 표현하고 설명한다.

6.2. Class Diagram

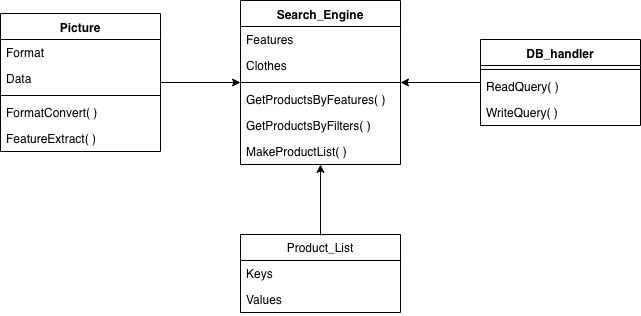


Diagram 24. Class Diagram - Picture Search System

1. DB Handler

DBMS와 시스템 사이의 인터페이스 기능을 제공하는 클래스이다. Search\_Engine으로부터 ReadQuery()를 요청받으며, 그 결과값으로 product의 상품 정보와 feature들을 반환한다.

1. **Attributes**
2. 해당사항 없음
3. **Methods**
4. +ReadQuery( ) : Product DB에서 조건에 맞는 data를 불러온다.
5. +WriteQuery( ) : 해당 사항 없음. 해당 시스템은 DB에 데이터를 기록하지 않는다.
6. Picture

Customer가 제공한 사진을 관리하는 클래스이다. System interface로부터 전달받은 원본 사진의 정보를 시스템이 이용할 수 있도록 가공하는 역할을 한다.

1. **Attributes**
2. +Format: 사진의 포맷
3. +Data: 사진 데이터
4. **Methods**
5. +FormatConvert( ): 사진의 포맷을 시스템이 처리 할 수 있는 형태로 변환한다. Customer는 여러 종류의 포맷으로 사진을 업로드할 수 있다. 따라서 시스템이 features를 추출할 수 있는 형태로 변환하는 과정이 필요하다.
6. +FeatureExtract( ): 정규화된 사진 데이터로부터 옷의 특징들을 추출한다.
7. Search\_Engine

Picture 클래스에서 추출한 features 를 이용하여 DB\_handler에 질의를 요청하고, DB\_handler로부터 얻은 product 들의 정보들을 key-value set 형식의 리스트로 조합하여 System interface로 전달한다.

1. **Attributes**
2. +Features: Picture 클래스에서 추출된 특징들. 옷 검색에 이용된다.
3. +Clothes: DB\_handler로부터 얻어진 결과물. 옷 검색에 이용된다.
4. **Methods**
5. GetProductByFeatures(): Picture에서 추출한 feature들을 기준으로 DB에 질의한다.
6. GetProductByFilters(): 해당 사항 없음.
7. MakeProductList(): DB로부터 제공받은 자료들을 바탕으로 Key-value 셋일 조합하여 interface로 product\_list를 전달한다.
8. Products\_list

Products\_list는 Key-value set 형태로 제공된다. Key는 product\_list 내에서 각 product를 구분하는 인덱스역할을 하며, value는 각 product의 정보 자체를 담고 있다.

1. **Attributes**
2. +Keys: List 내에서 각각의 product를 구분하는 식별자를 역할을 한다.
3. +Values: List 가 담고있는 product dml
4. **Methods**
5. 해당 사항 없음

6.3. Sequence Diagram

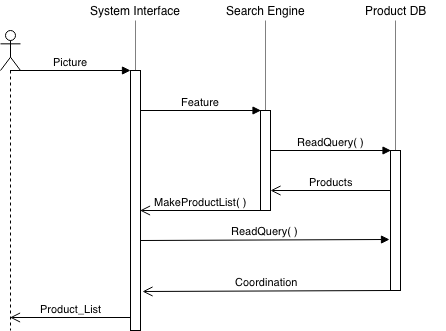


Diagram 25. Sequence Diagram - Picture Search System

1. User – System Interface

User interface를 통해 서버로 업로드 된 사진은 System interface를 통해 Picture Search System으로 전달된다.

Picture Search System 내부의 검색 작업이 끝나고 난 뒤, 결과물로 Product\_list를 제공받는다.

1. System Interface – Search Engine

Interface가 전달받은 picutre의 포멧을 정규화하고, 자체적으로 feature를 추출한다. 추출의 결과물을 search engine으로 전달한다.

Search Engine은 검색 결과물을 product\_list로 만들어 interface에 전달한다. 전달되는 형식은 json 타입의 key-value 형태로 전달한다. 검색결과가 존재하지 않는 경우, 예외 처리를 수행한다.

1. Search Engine – Product DB

Search engine은 interface로부터 전달받은 feature들을 검색조건으로 사용하여 DB에 옷 검색을 질의한다.

DB는 질의에 따른 검색결과를 search engine으로 전달한다.

1. System Interface – Product DB

Interface는 search engine에게 받은 product\_list를 바탕으로, product에 등록된 coordination들을 DB에 질의한다. DB는 결과물을 interface에 전달하고, interface는 DB에서 전달받은 결과를 product\_list에 첨부하여 customer에게 전달한다.

7. Filter Search System

7.1. Objectives

검색을 할 때, 어떠한 검색 필터 및 키워드를 이용하느냐에 따라 구매자와의 상호 작용이 달라지기 때문에, 그것을 case로 설정하여 Class diagram, Sequence diagram과 State diagram을 통해 Search System의 구조와 구매자 간의 상호 작용을 표현하고 설명한다.

7.2. Class Diagram

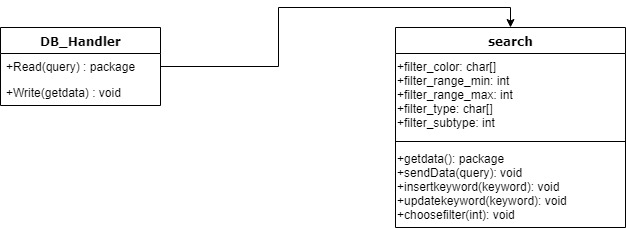


Diagram 26. Class Diagrma - Filter Search System

1. DB Handler
2. **Attributes**
3. 해당사항 없음.
4. **Methods**
5. +package Read(query): 해당되는 DB에서 원하는 data를 읽어옴.
6. +void Write(getData): 해당되는 DB에 data를 저장함.
7. Product
8. **Attributes**
9. +filter\_color: filter 색 정보 값
10. +filiter\_range\_min: filiter 가격 최솟값 정보
11. +filter\_range\_max: filter 가격 최댓값 정보
12. +filter\_type: filter 카테고리 타입 정보
13. +filter\_subtype: filter 카테고리 소 타입 정보
14. **Methods**
15. +package getData(): DB로부터 data를 받음.
16. +void sendData(query): DB로 data를 보냄.
17. +void insertKeyword(keyword): keyword를 입력 받는다.
18. +void updateKeyword(keyword): keyword를 수정한다.
19. +void chooseFilter(int): filter를 선택한다.

7.3. Sequence Diagram

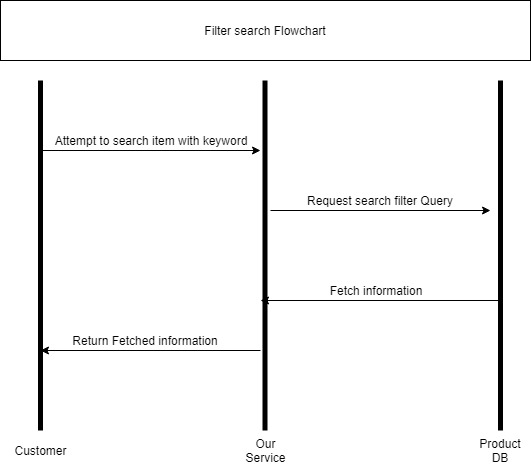


Diagram 27. Sequence Diagram - Filter Search System

7.4. State Diagram

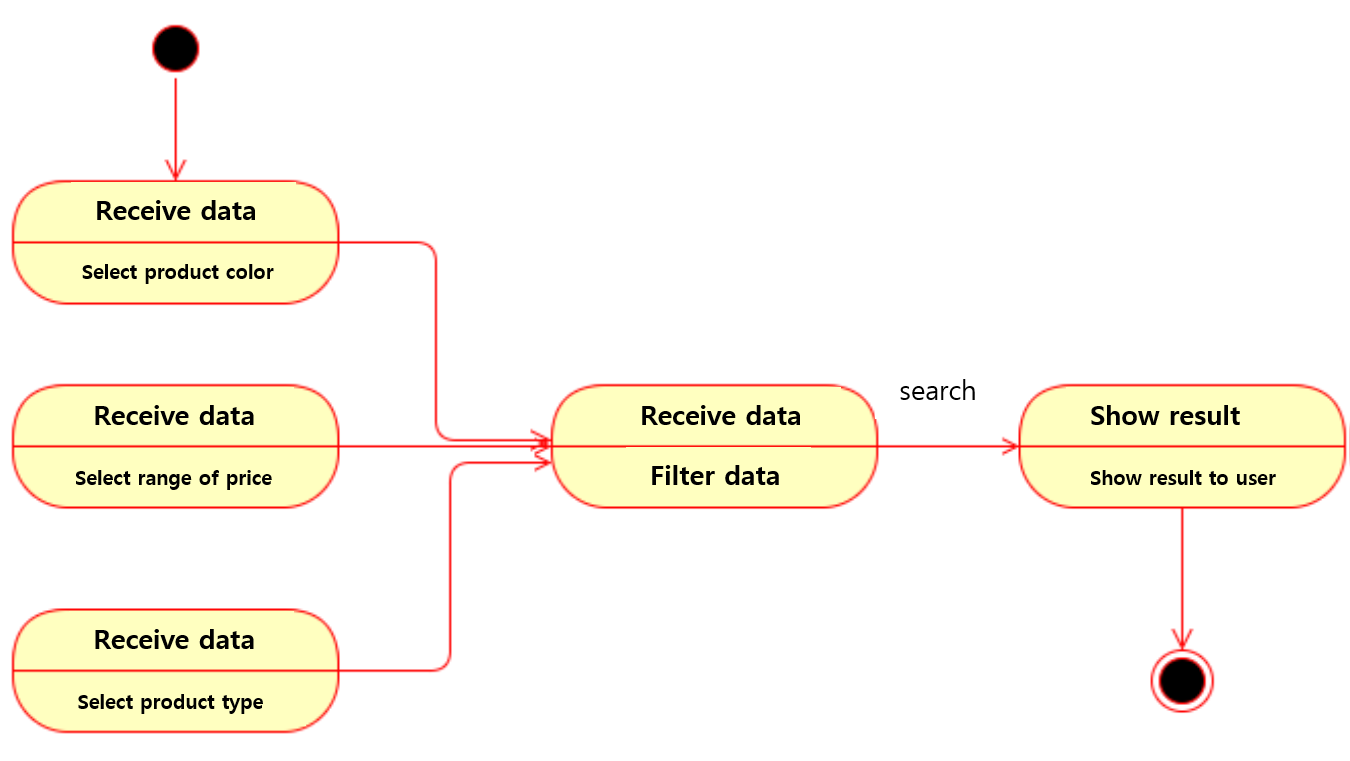


Diagram 28. State Diagram - Filter Search System

8. Recommendation System

8.1. Objectives

이용자의 서비스 이용기록을 기반으로 개인에게 맞춤형 추천을 제공하는 시스템이다. CSP에 저장된 이용자의 검색기록을 이용하여 아이템 리스트 형태로 제공하는 형태와 자주 찾는 유형의 옷을 제공하는 형태로 나뉘어지기 때문에 Class diagram, Sequence diagram과 State Diagram을 통해 각 케이스의 구조와 사용자와의 상호 작용을 표현하고 설명한다.

8.2. Class Diagram

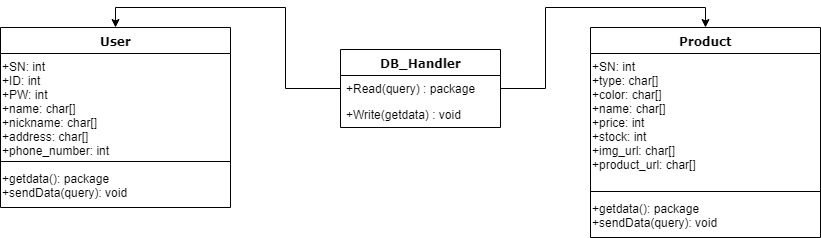


Diagram 29. Class Diagram - Recommendation System

1. DB Handler
2. **Attributes**
3. 해당사항 없음.
4. **Methods**
5. +package Read(query): 해당되는 DB에서 원하는 data를 읽어옴.
6. +void Write(getData): 해당되는 DB에 data를 저장.
7. Product
8. **Attributes**
9. +SN: product 고유번호 값
10. +type: product 옷 종류 정보
11. +color: product 색상 정보
12. +name: product 이름 정보
13. +price: product 가격 정보
14. +stock: product 재고 정보
15. +img\_url: product 이미지 정보
16. +product\_url: product 링크 정보
17. **Methods**
18. +package getData(): DB로 data를 받음.
19. +void sendData(query): DB로 data를 보냄.
20. User
21. **Attributes**
22. +SN: user 고유번호 값
23. +ID: user ID 정보
24. +PW: user PW 정보
25. +name: user 이름 정보
26. +nickname: user 닉네임 정보
27. +address: user 주소 정보
28. +phone\_number: user 전화번호 정보
29. **Methods**
30. +package getData(): DB로부터 data를 받음.
31. +void sendData(query): DB로부터 data를 보냄.

8.3. Sequence Diagram

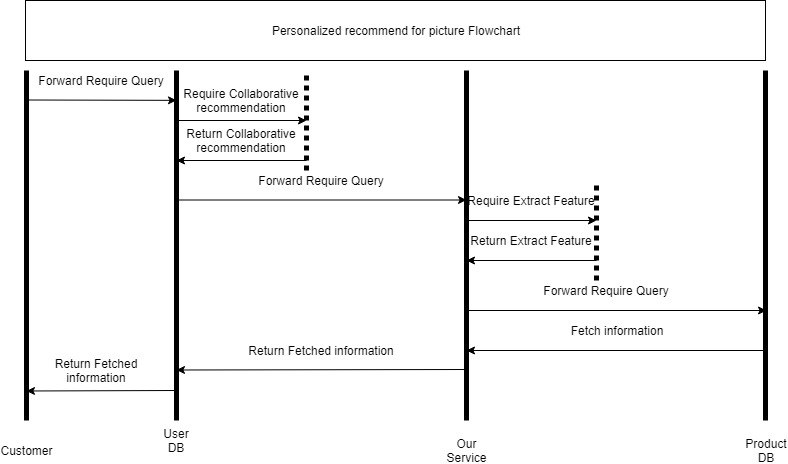


Diagram 30. Sequence Diagram - Recommendation System

8.4. State Diagram

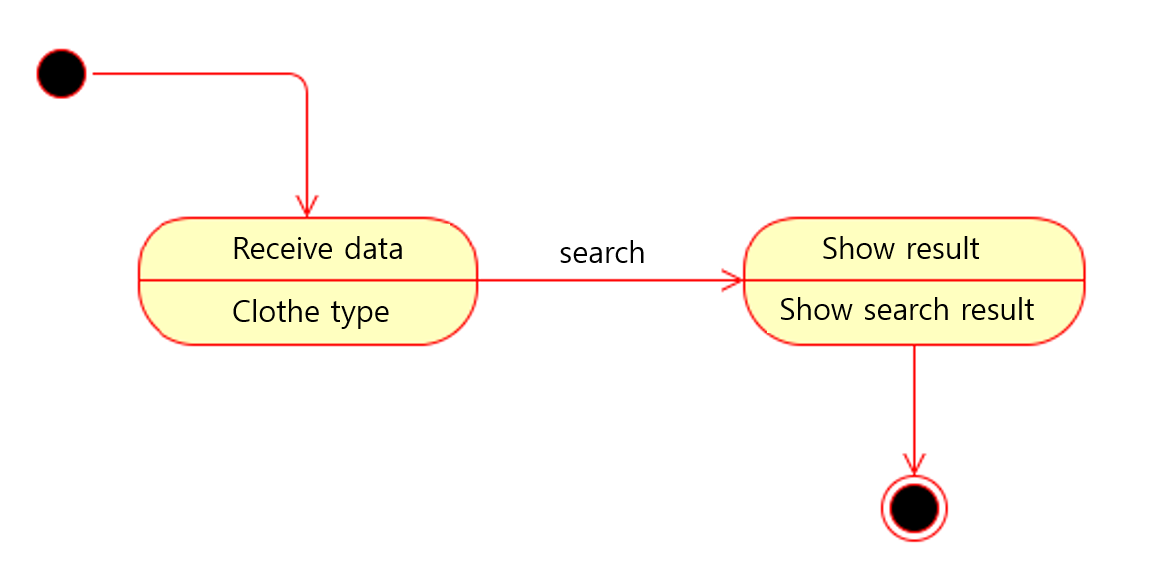


Diagram 31. State Diagram - Recommendation System

9. Trend System

9.1. Objectives

시스템을 이용하는 사용자가 Trend 서비스를 제공받는 도중에 발생하는 데이터에 대한 처리와 데이터의 수정, 변경에 대한 설계를 설명한다. Class diagram, Sequence diagram과 State Diagram을 통해 Trend System의 구조를 표현한다.

9.2. Class Diagram

스크린샷이(가) 표시된 사진

자동 생성된 설명

Diagram 32. Class Diagram - Trend System

1. DB Handler
2. **Attributes**
3. 해당사항 없음.
4. **Methods**
5. +package Read(query): 해당되는 DB에서 원하는 data를 읽어옴.
6. +void Write(getData): 해당되는 DB에 data를 저장.
7. Account
8. **Attributes**
9. +SN: account 고유번호 값
10. +ID: account ID 정보
11. +PW: account 비밀번호 정보
12. +name: account 이름 정보
13. +nickname: account 닉네임 정보
14. +address: account 주소 정보
15. +type: buyer, seller 정보
16. +phone\_number: account 전화번호 정보
17. +corp\_num: account 사업자 등록번호 정보
18. +date: account 가입 날짜 정보
19. +img\_url: account 이미지의 주소 정보
20. **Methods**
21. +package getData(): DB로부터 data를 받음.
22. +void sendData(query): DB로부터 data를 보낸다.
23. Trend
24. **Attributes**
25. 해당사항 없음.
26. **Methods**
27. +void search(query): 검색.
28. +void Write(char[], char[]): data생성.
29. +void Edit(int, char[], char[]): 해당 data를 수정.
30. +package view(int): DB로 해당 data를 받음.
31. Picture Search
32. **Attributes**
33. 해당사항 없음.
34. **Methods**
35. +void sendData(query): DB로 data를 보냄.
36. Filter Search
37. **Attributes**
38. 해당사항 없음.
39. **Methods**
40. +void sendData(query): DB로 data를 보냄.
41. Recommendation
42. **Attributes**
43. 해당사항 없음.
44. **Methods**
45. +void sendData(query): DB로 data를 보냄.

9.3. Sequence Diagram

A. View

스크린샷이(가) 표시된 사진

자동 생성된 설명

Diagram 33. Sequence Diagram - View

B. Write

스크린샷이(가) 표시된 사진

자동 생성된 설명

Diagram 34. Sequence Diagram - Write

C. Search

스크린샷이(가) 표시된 사진

자동 생성된 설명

Diagram 35. Sequence Diagram - Search

1. Edit

스크린샷이(가) 표시된 사진

자동 생성된 설명

Diagram 36. Sequence Diagram - Edit

9.4. State Diagram

A. View

스크린샷이(가) 표시된 사진

자동 생성된 설명

Diagram 37. State Diagram - View

B. Write

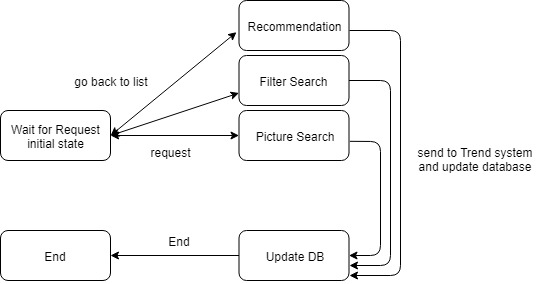


Diagram 38. State Diagram - Write

1. Search

스크린샷이(가) 표시된 사진

자동 생성된 설명

Diagram 39. State Diagram - Search

1. Edit

스크린샷이(가) 표시된 사진

자동 생성된 설명

Diagram 40. State Diagram - Edit

10. Community System

10.1. Objectives

본 시스템을 이용하는 사용자가 Community 서비스를 제공받으면서 생기는 데이터에 대한 입력, 수정에 대한 설계를 설명한다. Class diagram, Sequence diagram과 State Diagram을 통해 Community System의 구조를 표현하고 설명한다.

10.2. Class Diagram

스크린샷이(가) 표시된 사진

자동 생성된 설명

Diagram 41. Class Diagram - Community System

1. DB Handler
2. **Attributes**
3. 해당사항 없음.
4. **Methods**
5. +package Read(query): 해당하는 DB에서 원하는 data를 읽어옴.
6. +void Write(getData): 해당되는 DB에 data를 저장.
7. Account
8. **Attributes**
9. +SN: account 고유번호 값
10. +ID: account ID 정보
11. +PW: account 비밀번호 정보
12. +name: account 이름 정보
13. +nickname: account 닉네임 정보
14. +address: account 주소 정보
15. +type: buyer, seller 정보
16. +phone\_number: account 전화번호 정보
17. +corp\_num: account 사업자 등록번호 정보
18. +data: account 가입 날짜 정보
19. +img\_url: account 이미지 주소 정보
20. **Methods**
21. +package getData(): DB로부터 data를 받음.
22. +void sendData(query): DB로 data를 보냄.
23. Community
24. **Attributes**
25. 해당사항 없음
26. **Methods**
27. +void search(query): 검색한다.
28. +void Write(char[], char[]): data를 생성한다.
29. +void Edit(int, char[], char[]): 해당 data를 수정한다.
30. +package view(int): DB로 해당 data를 받는다.

10.3. Sequence Diagram

A. View

스크린샷이(가) 표시된 사진

자동 생성된 설명

Diagram 42. Sequence Diagram - View

B. Write

스크린샷이(가) 표시된 사진

자동 생성된 설명

Diagram 43. Sequence Diagram - Write

C. Search

스크린샷이(가) 표시된 사진

자동 생성된 설명

Diagram 44. Sequence Diagram - Search

1. Edit

스크린샷이(가) 표시된 사진

자동 생성된 설명

Diagram 45. Sequence Diagram - Edit

10.4. State Diagram

A. View

스크린샷이(가) 표시된 사진

자동 생성된 설명

Diagram 46. State Diagram - View

B. Write

스크린샷이(가) 표시된 사진

자동 생성된 설명

Diagram 47. State Diagram - Write

C. Search

스크린샷이(가) 표시된 사진

자동 생성된 설명

Diagram 48. State Diagram - Search

D. Edit

스크린샷이(가) 표시된 사진

자동 생성된 설명

Diagram 49. State Diagram - Edit

10.5. Search Filter

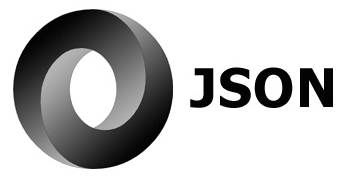
사용자가 게시판에서 원하는 글을 수월하게 찾기 위해 시스템이 제공하는 서비스다. 검색 창 왼쪽에 drop bar형식으로 제공된다. 필터는 제목, 내용, 작성자로 검색이 가능하도록 제공되며 세가지 필터는 동시에 적용 불가능하다. 기본 필터는 제목이다. 사용자는 게시판에서 검색을 할 경우 검색 창 왼쪽에 필터를 설정하고 검색 keyword를 입력해 검색할 수 있다.

11. Protocol Design

11.1. Objectives

Protocol Design에서는 subsystem들이 상호작용하는 프로토콜에 대해 서술한다. 프로토콜의 기본 형식은 JSON을 기본으로 하며 통신하는 메시지의 형식과 용도, 의미를 설명한다.

11.2. JSON



Picture 15. JSON Logo

JSON은 JavaScript Object Notation의 약어로써 XML과 더불어 대표적인 데이터 교환 방식이다. 자바스크립트에 기반하여 만들어진 데이터 표현형식이지만 프로그래밍언어나 플랫폼에 독립적인 특성을 갖고 있어, 다양한 언어에서 JSON을 활용할 수 있다. 또한 기능이 적어 파싱(Parsing)이 빠르다는 장점을 가지고 있다. 자료의 종류에 큰 제한이 없으며, 컴퓨터 프로그램의 변수 값을 표현하는 데 적합하다. Attribute와 value의 쌍으로 표현되며, 거의 대부분의 자료형을 사용할 수 있다.

11.3. Protocol Description

A. Overview

HTTP 통신에서 client와 server 사이에서 전송되는 메시지의 형태를 용도별로 정의한다. Client에서의 요청(request) 메시지와 server에서의 응답(response) 메시지로 구분한다.

B. Login Protocol

1. **Request**



Table 2. Login Protocol - Request

1. **Response**



Table 3. Login Protocol - Response

C. Registration Protocol

1. **Request**



Table 4. Registration Protocol - Request

1. **Response**



Table 5. Registration Protocol - Response

D. Community Post Edit Protocol

1. **Request**

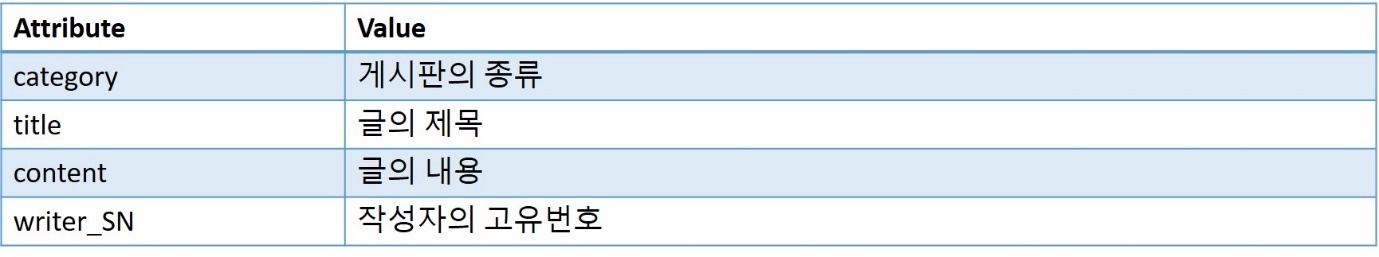


Table 6. Community Post Edit Protocol - Request

1. **Response**



Table 7. Community Post Edit Protocol - Response

E. Community Search Protocol

1. **Request**

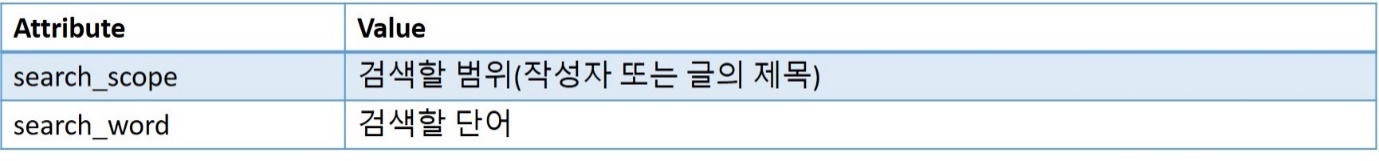


Table 8. Community Search Protocol - Request

1. **Response**



Table 9. Community Search Protocol - Request

F. Community View Protocol

1. **Request**



Table 10. Community View Protocol - Request

1. **Response**



Table 11. Community View Protocol - Response

G. Product Post Edit Protocol

1. **Request**



Table 12. Product Post Edit Protocol - Request

1. Response

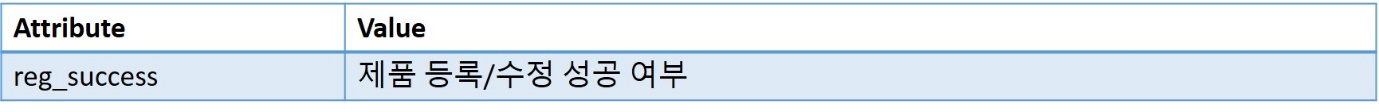


Table 13. Product Post Edit Protocol - Response

H. Product View Protocol

1. **Request**



Table 14. Product View Protocol - Request

1. Response



Table 15. Product View Protocol - Response

I. Product Search Protocol

1. **Request**

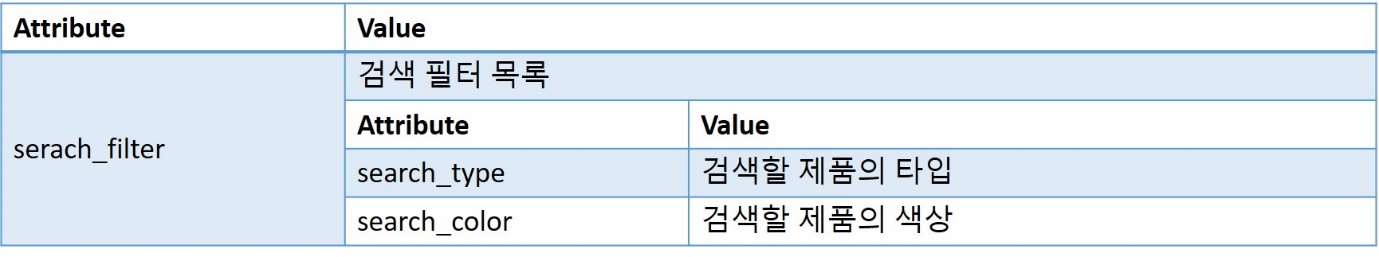


Table 16. Product Search Protocol - Request

1. **Response**

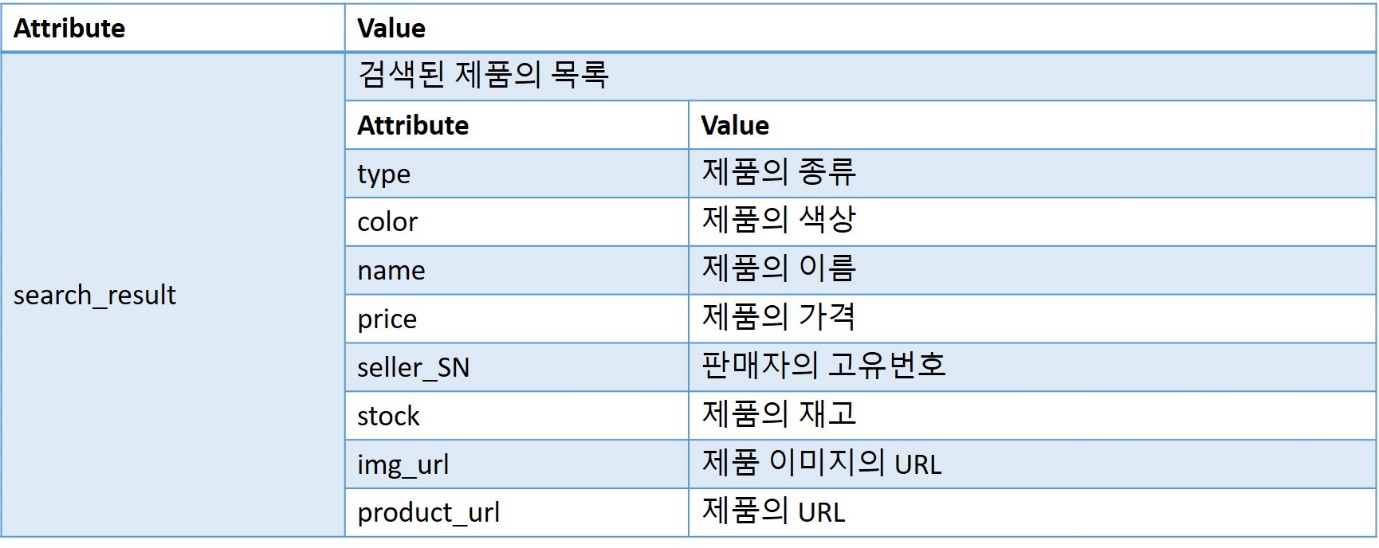


Table 17. Product Search Protocol - Response

J. Picture Upload Protocol

1. **Request**



Table 18. Picture Upload Protocol - Request

1. **Response**

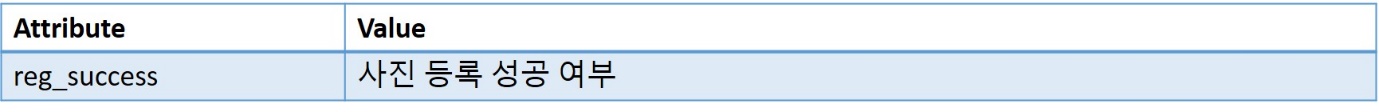


Table 19. Picture Upload Protocol - Response

12. Database Design

12.1. Objectives

Database Design은 요구사항 명세서에서 기술한 데이터베이스 요구사항을 바탕으로 수정사항을 수정하여 다시 요구사항을 작성하였다. 요구사항을 바탕으로 ER diagram을 작성하고, 이를 이용하여 Relational Schema를 작성하고 Normalization을 통해 redundancy와 anomaly를 제거한 후 마지막으로 SQL DDL을 작성한다.

12.2. ER Diagram

개체는 분리된 물체 하나를 표현한다. 개체는 사각형으로 표현되며, 관계는 다이아몬드로 표현된다. 개체나 관계는 특성을 가질 수 있으며, 이 특성들은 관계 집합에 실선으로 연결된 타원형으로 표현한다.

관계는 두개 이상의 개체들의 연관 관계를 표현한다. 모든 개체는 고유하게 식별되는 특성 집합을 가지고 있어야 하며, 최소한의 고유 식별 특성 집합은 개체의 기본 키(Primary Key)라 불린다.

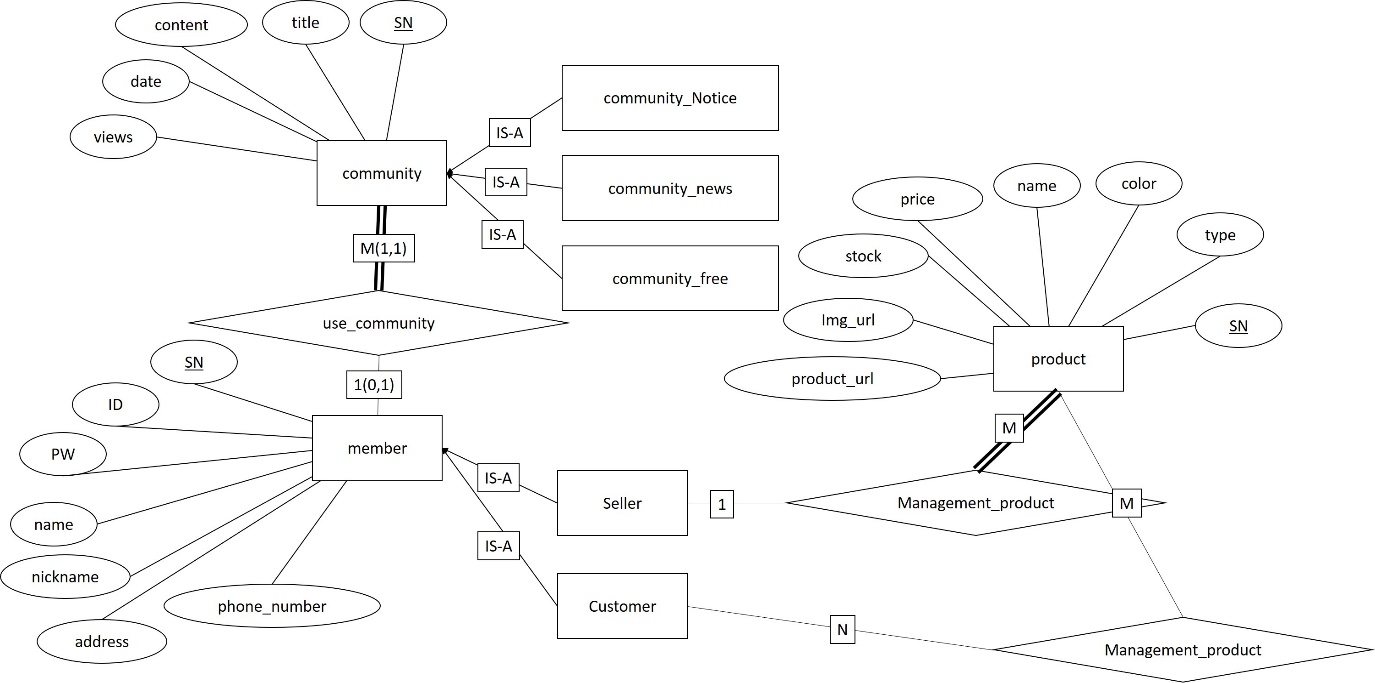


Diagram 50. ER Diagram

1. Entity

**A.1. Member**

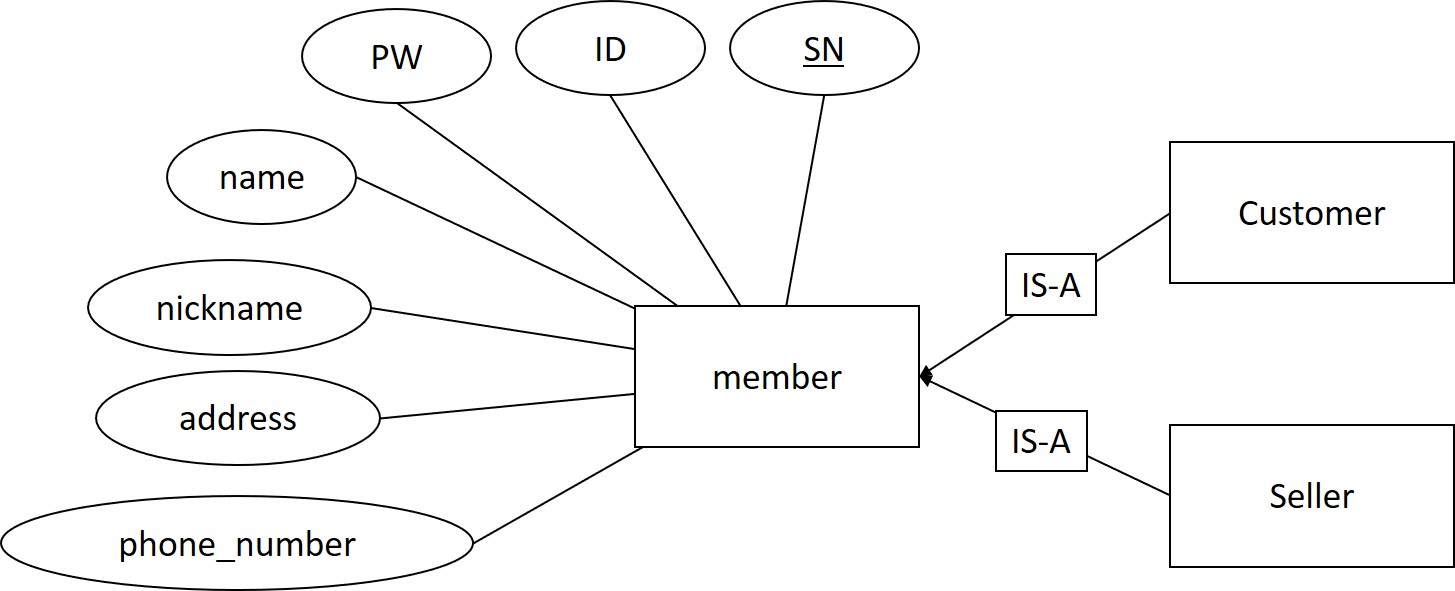


Diagram 51. Member

Member는 회원 정보를 나타낸다. Customer, Seller와 IS-A 관계를 맺고 있다. Serial number, ID, Password, name, nickname, address, phone number의 속성을 갖고 있다. Key는 Serial number이다.

**A.2. Product**

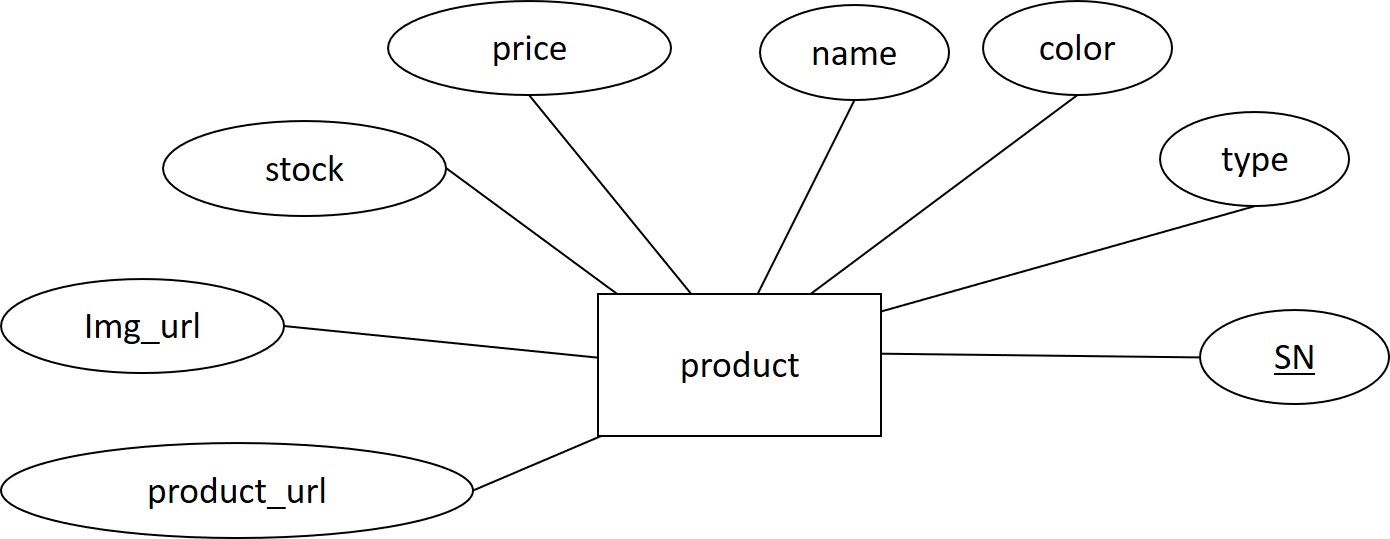


Diagram 52. Product

Product는 제품 정보를 나타낸다. Serial number, type, color, name, price, stock, img\_url, product\_url의 속성을 갖고 있다.

Key는 Serial number이다.

**A.3. Community**

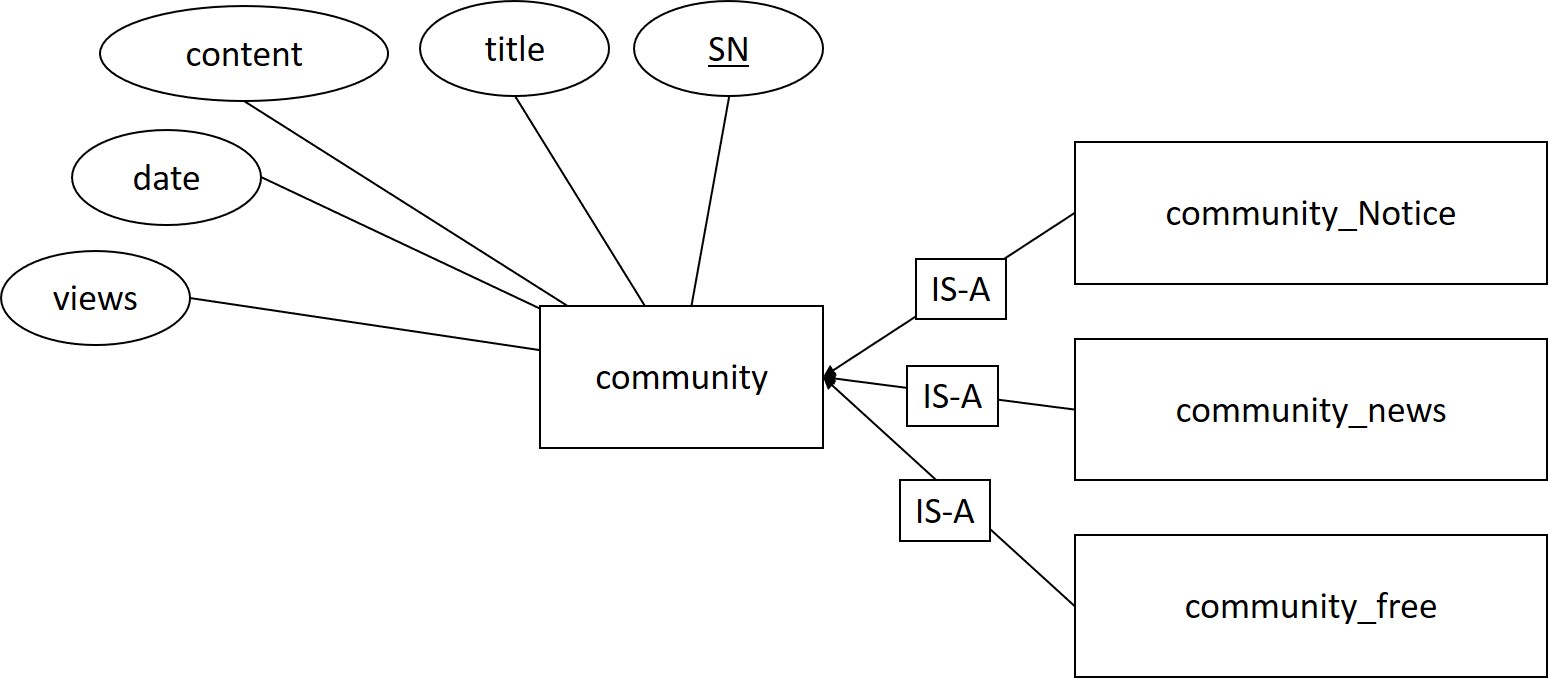


Diagram 53. Community

Community는 커뮤니티 게시판 정보 관리를 나타낸다. Community Notice, News, Free와 IS-A 관계를 맺고 있다. Serial number, title, content, date, views의 속성을 갖고 있다. Key는 Serial number이다.

1. Entity

**B.1. Management Product**

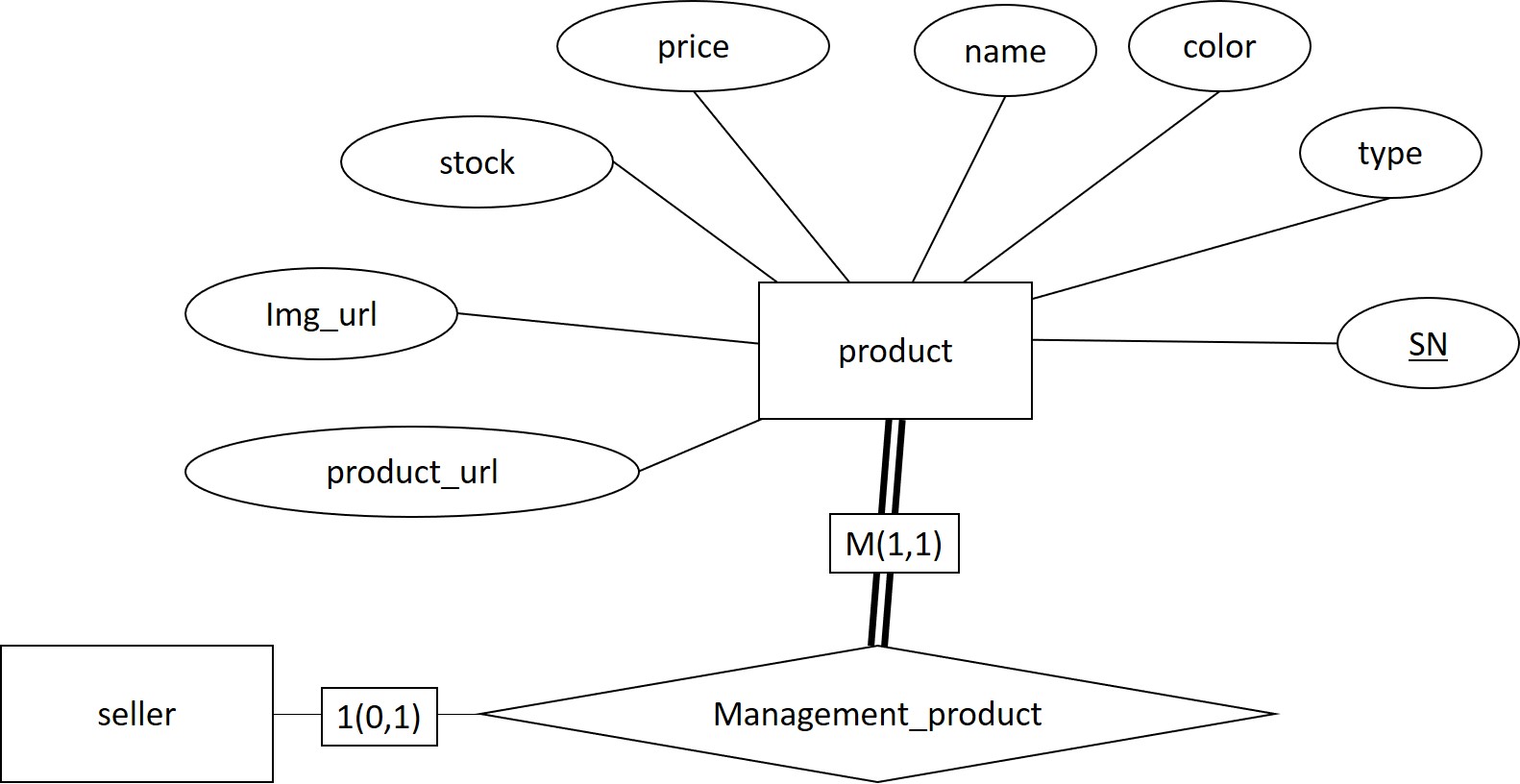


Diagram 54. Management Product

Management Product Relationship은 판매자(Seller)가 제품을 관리하는 관계이다. 1명의 판매자가 여러 개의 제품을 관리할 수 있으며, 모든 등록된 제품은 관리하는 판매자가 1명이다.

**B.2. Recommend**

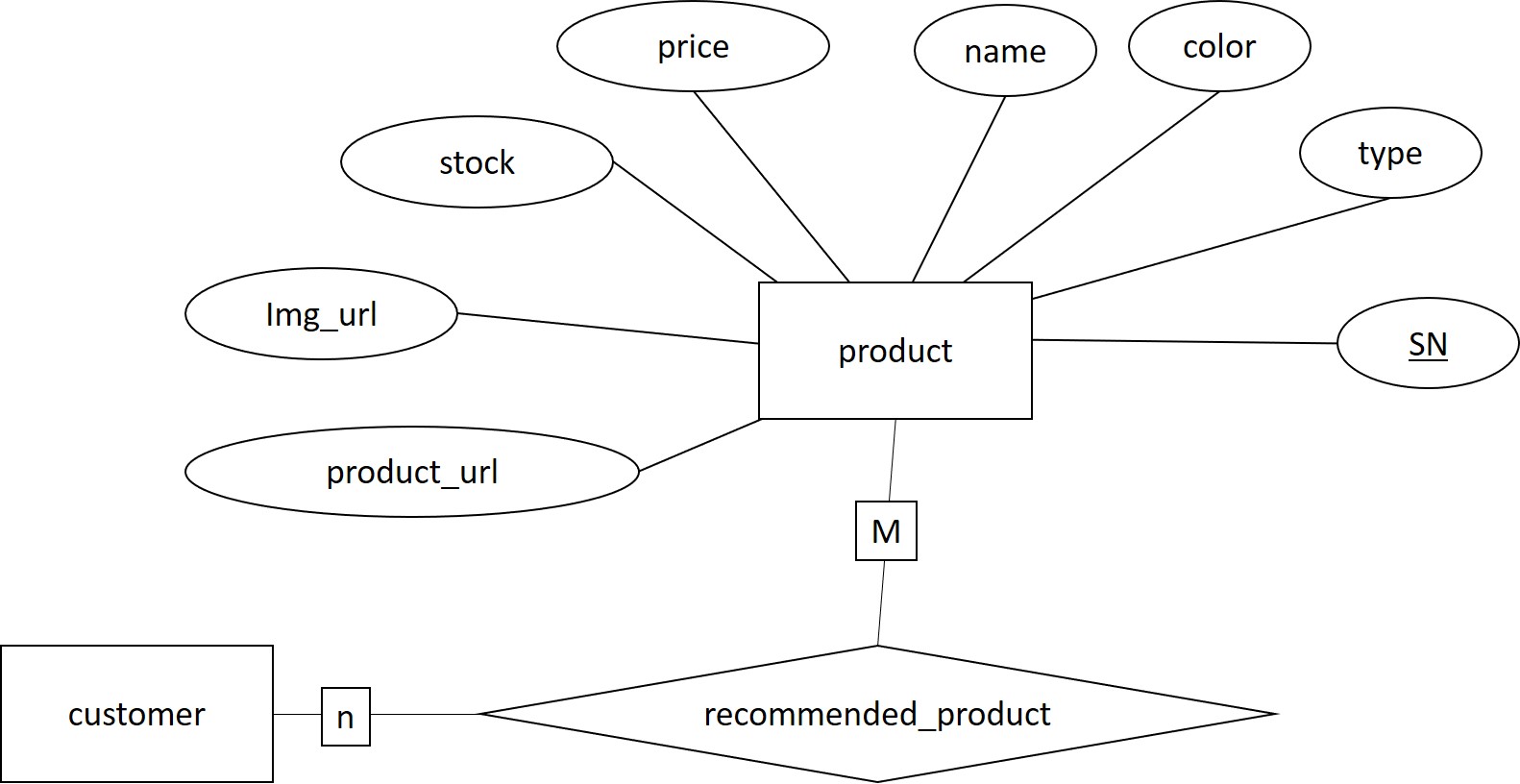


Diagram 55. Recommend

Management Product Relationship은 구매자(customer)가 제품을 추천받는 관계이다. 1명의 구매자가 여러 개의 제품을 추천받을 수 있다. 모든 제품은 여러 고객들에게 추천될 수 있다.

**B.3. User Community**

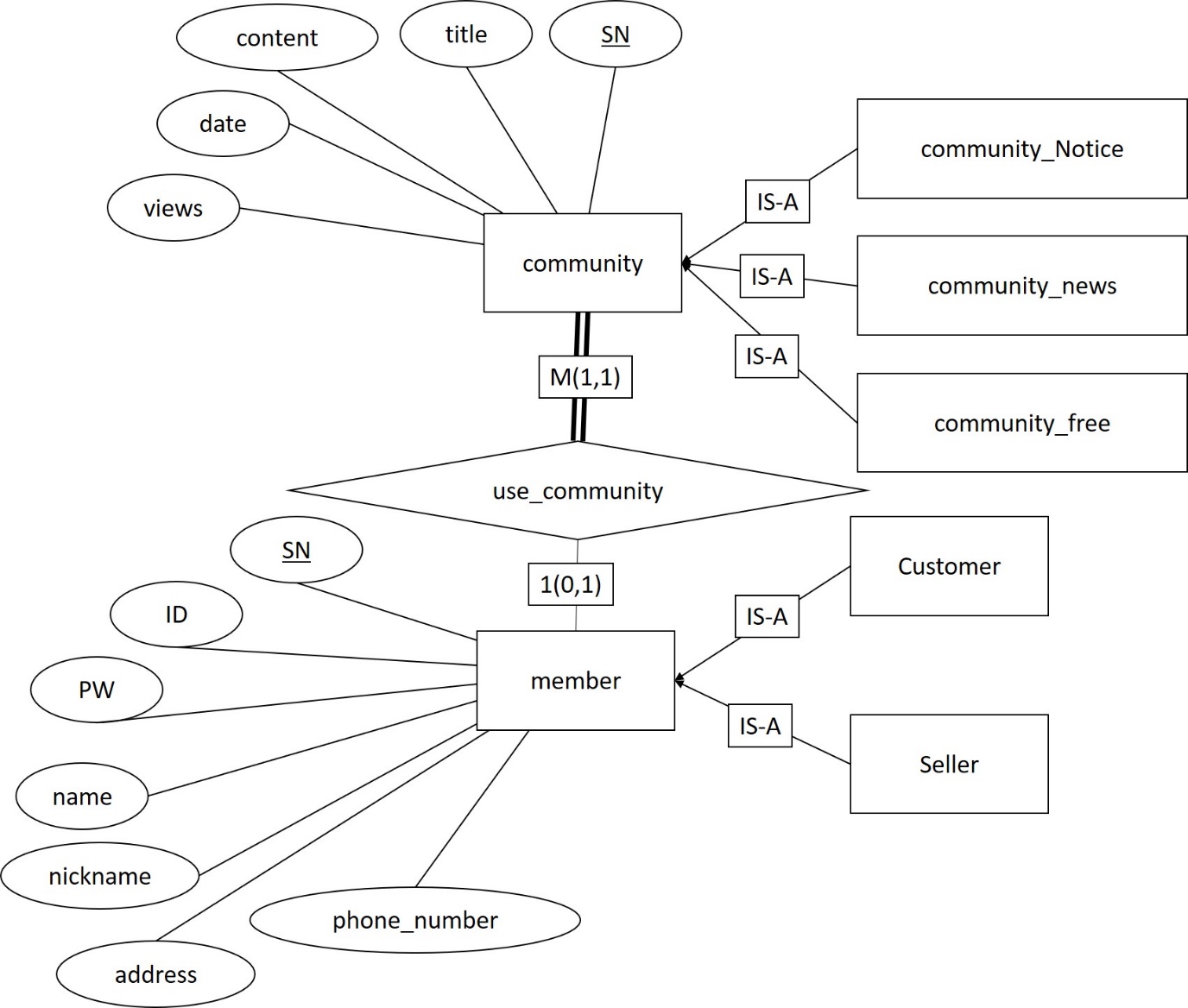


Diagram 56. User Community

Use Community Relationship은 사용자(member)가 커뮤니티를 이용하는 관계이다. 1명의 사용자 여러 개의 커뮤니티 글을 이용할 수 있으며, 모든 등록된 커뮤니티 글은 1명의 사용자에 의해 작성된다.

12.3. Relational Schema

A. Customer

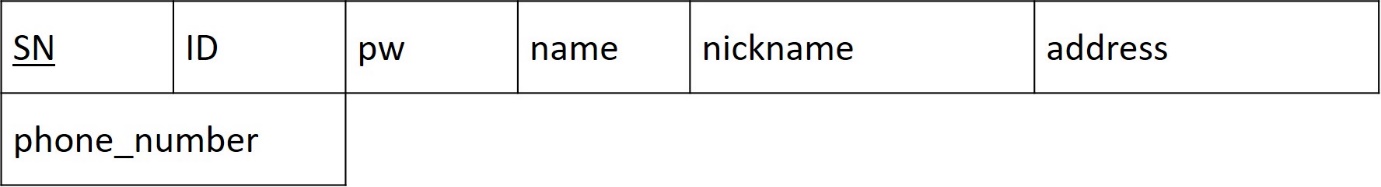


Table 20. Customer

Primary Key (PK): SN

Foreign Key (FK): 없음.

FUNCT DEP (FD):

SN → {ID, Password, name, nickname, address, phone number}

ID → {SN, Password, name, nickname, address, phone number}

nickname→ {SN, ID, Password, name, address, phone number}

Description: Entity Buyer에 관한 테이블이다. 모든 속성에 NULL을 허용하지 않는다.

B. Seller

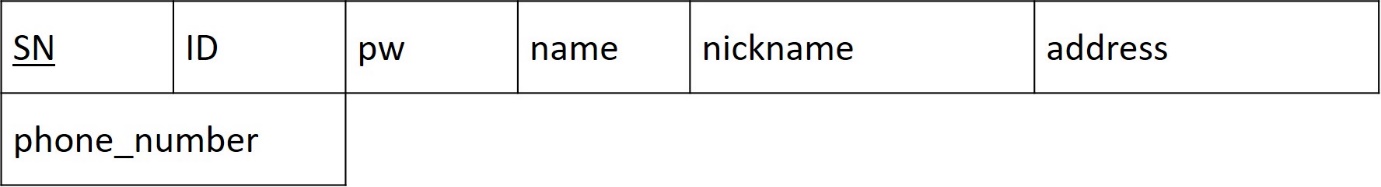


Table 21. Seller

Primary Key (PK): SN

Foreign Key (FK): 없음.

FUNCT DEP (FD):

SN → {ID, Password, name, nickname, address, phone number}

ID → {SN, Password, name, nickname, address, phone number}

nickname→ {SN, ID, Password, name, address, phone number}

Description: Entity Buyer에 관한 테이블이다. 모든 속성에 NULL을 허용하지 않는다.

1. Product

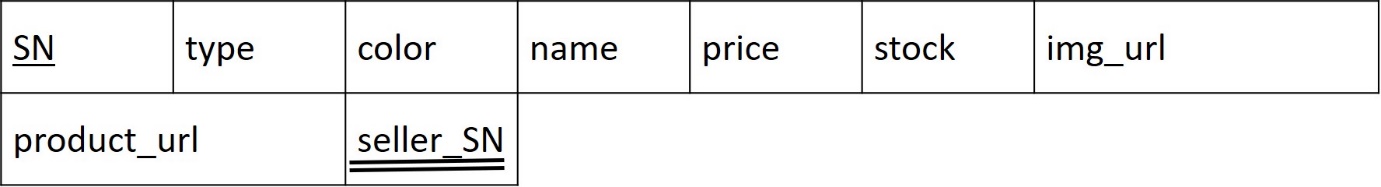


Table 22. Product

Primary Key (PK): SN

Foreign Key (FK): seller\_SN.

FUNCT DEP (FD):

SN → {type, color, name, price, stock, img\_url, product\_url, seller\_SN}

Description: Entity Product에 관한 테이블이다. 모든 속성에 NULL을 허용하지 않으며, Seller의 PK를 FK로 가져온다.

1. Community\_notice



Table 23. Community\_notice

Primary Key (PK): SN

Foreign Key (FK): writer\_SN.

FUNCT DEP (FD):

SN → {title, content, date, views, writer\_SN}

Description: Entity Notice Community에 관한 테이블이다. 모든 속성에 NULL을 허용하지 않으며, member의 PK를 가져온다.

1. Community\_news



Table 24. Community\_news

Primary Key (PK): SN

Foreign Key (FK): writer\_SN.

FUNCT DEP (FD):

SN → {title, content, date, views, writer\_SN}

Description: Entity Notice Community에 관한 테이블이다. 모든 속성에 NULL을 허용하지 않으며, member의 PK를 가져온다.

1. Community\_Free



Table 25. Community\_Free

Primary Key (PK): SN

Foreign Key (FK): writer\_SN.

FUNCT DEP (FD):

SN → {title, content, date, views, writer\_SN}

Description: Entity Notice Community에 관한 테이블이다. 모든 속성에 NULL을 허용하지 않으며, member의 PK를 가져온다.

1. Community\_Comment\_Notice

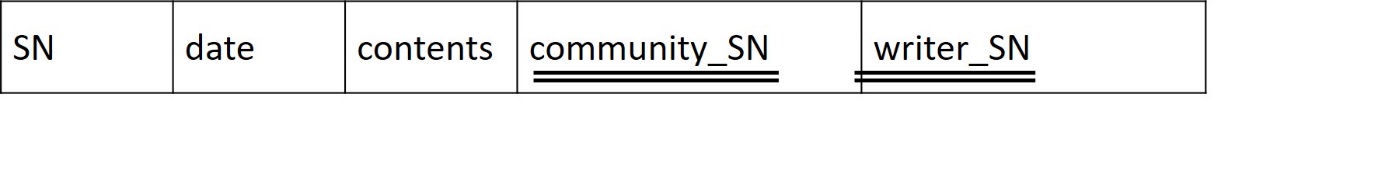


Table 26. Community\_Comment\_Notice

Primary Key (PK): SN

Foreign Key (FK): writer\_SN.

FUNCT DEP (FD):

SN → {data, contents, community\_SN, writer\_SN}

Description: Entity Notice Community Comment에 관한 테이블이다. 모든 속성에 NULL을 허용하지 않으며, community와 writer의 PK를 FK로 가져온다.

1. Community\_Comment\_News

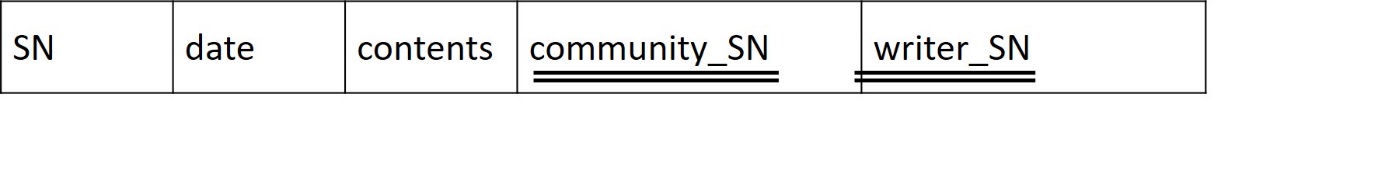


Table 27. Community\_Comment\_News

Primary Key (PK): SN

Foreign Key (FK): writer\_SN.

FUNCT DEP (FD):

SN → {data, contents, community\_SN, writer\_SN}

Description: Entity Notice Community Comment에 관한 테이블이다. 모든 속성에 NULL을 허용하지 않으며, community와 writer의 PK를 FK로 가져온다.

1. Community\_Comment\_Free

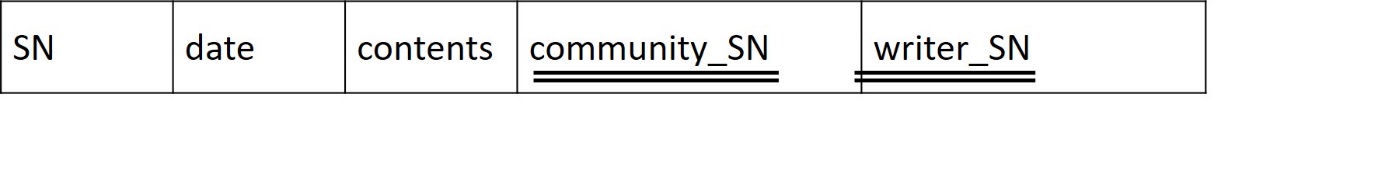


Table 28. Community\_Comment\_Free

Primary Key (PK): SN

Foreign Key (FK): writer\_SN.

FUNCT DEP (FD):

SN → {data, contents, community\_SN, writer\_SN}

Description: Entity Notice Community Comment에 관한 테이블이다. 모든 속성에 NULL을 허용하지 않으며, community와 writer의 PK를 FK로 가져온다.

12.4. Normalization

정규화란 나쁜 테이블을 더 좋은 작은 테이블로 분해하는 것을 말한다. 나쁜 테이블은 데이터베이스에 이상을 가져온다.

1) Redundancy: 필요 이상의 정보가 여러 번 반복된다.

2) Insert Anomaly: 새로운 tuple을 삽입할 수 없다.

3) Delete Anomaly: 어떤 정보를 삭제할 때, 원치 않은 정보도 삭제된다.

4) Update Anomaly: 정보를 업데이트 할 때, 여러 번의 업데이트가 필요하다.

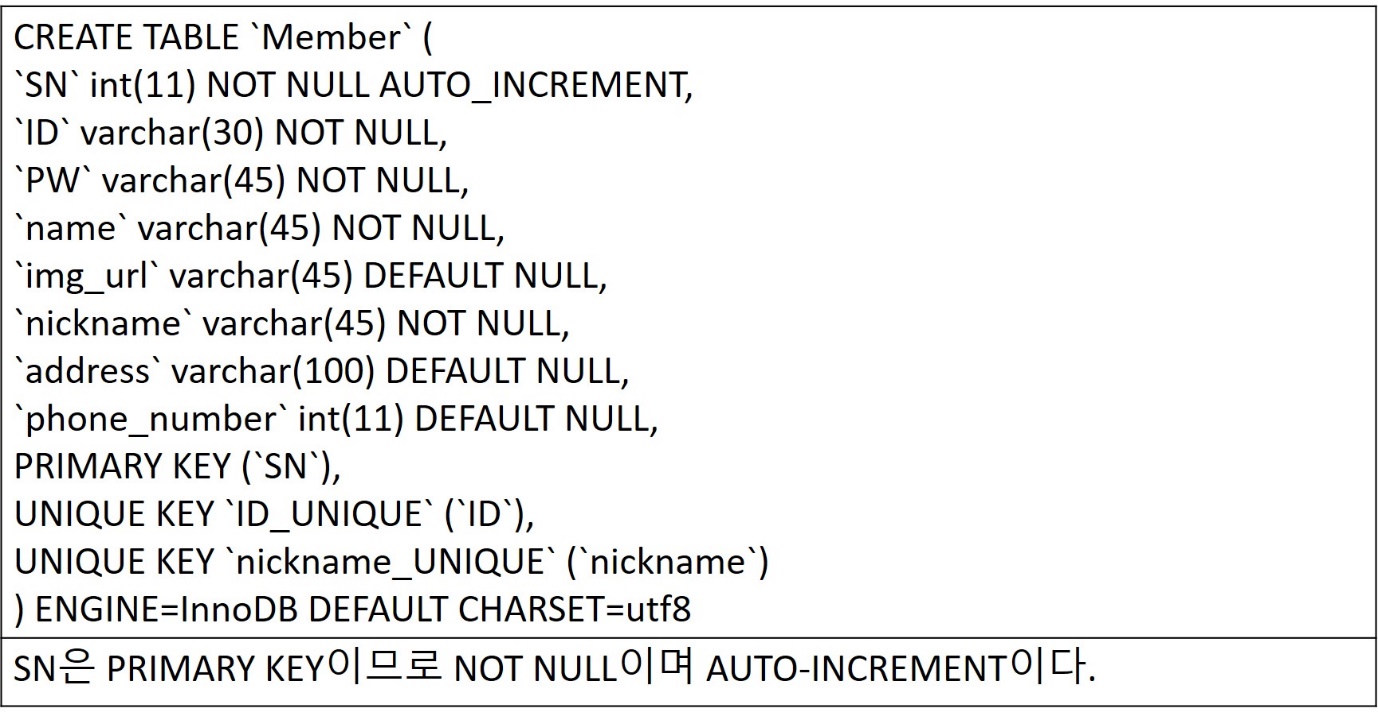
이러한 문제를 해결하기 위해 정규화를 실시한다. 정규화는 제1정규화부터 제6정규화까지 있지만 대다수의 경우 BCNF까지의 정규형을 고려한다. 따라서 본 시스템에서도 BCNF까지의 정규형을 적용하였다.

BCNF를 만족하는지 확인하기 위해서 모든 Functional Dependency X→A에 대하여 X의 Super Key 여부를 확인한다.

앞 절에서의 Relational Schema 중에 BCNF를 만족하지 않는 경우가 없기 때문에, 특별히 정규화 과정이 진행되지는 않았다.

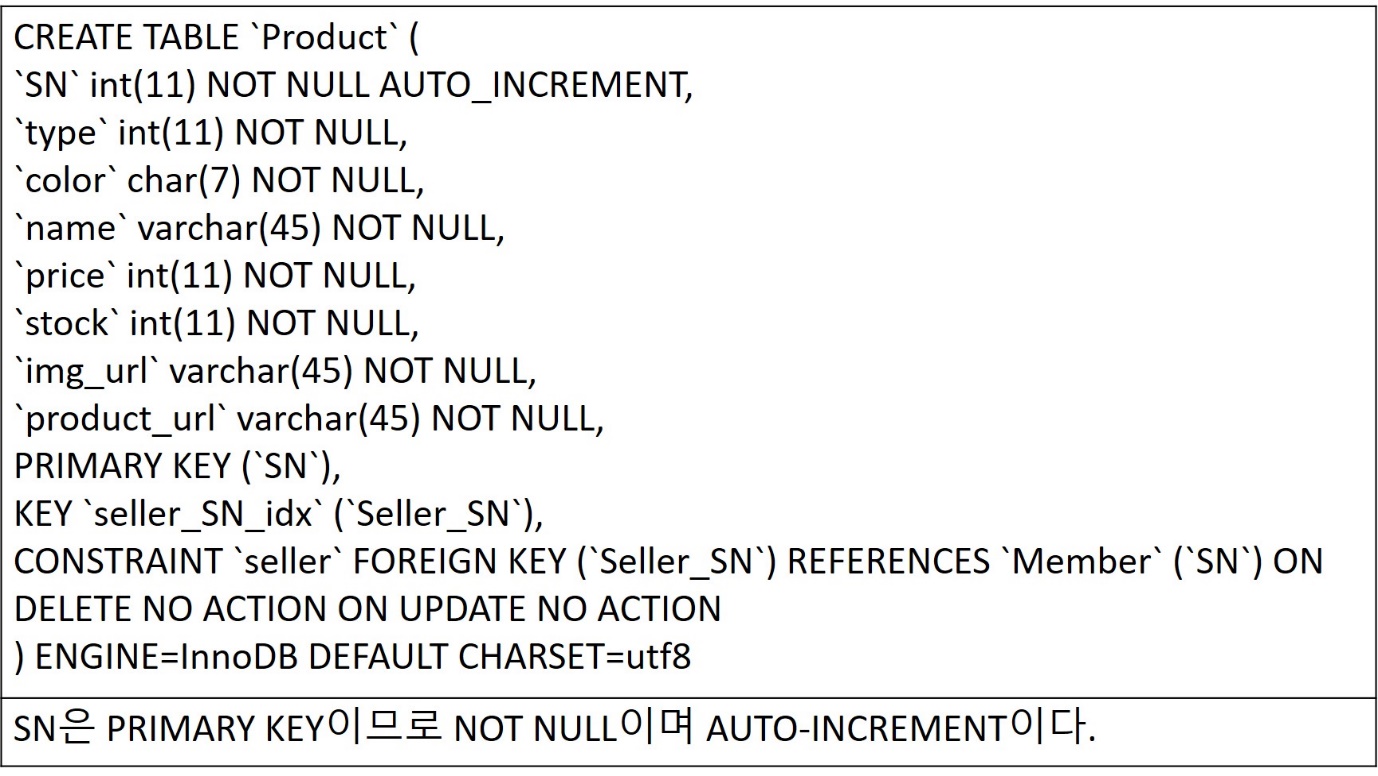
12.5. SQL DDL

A. Member



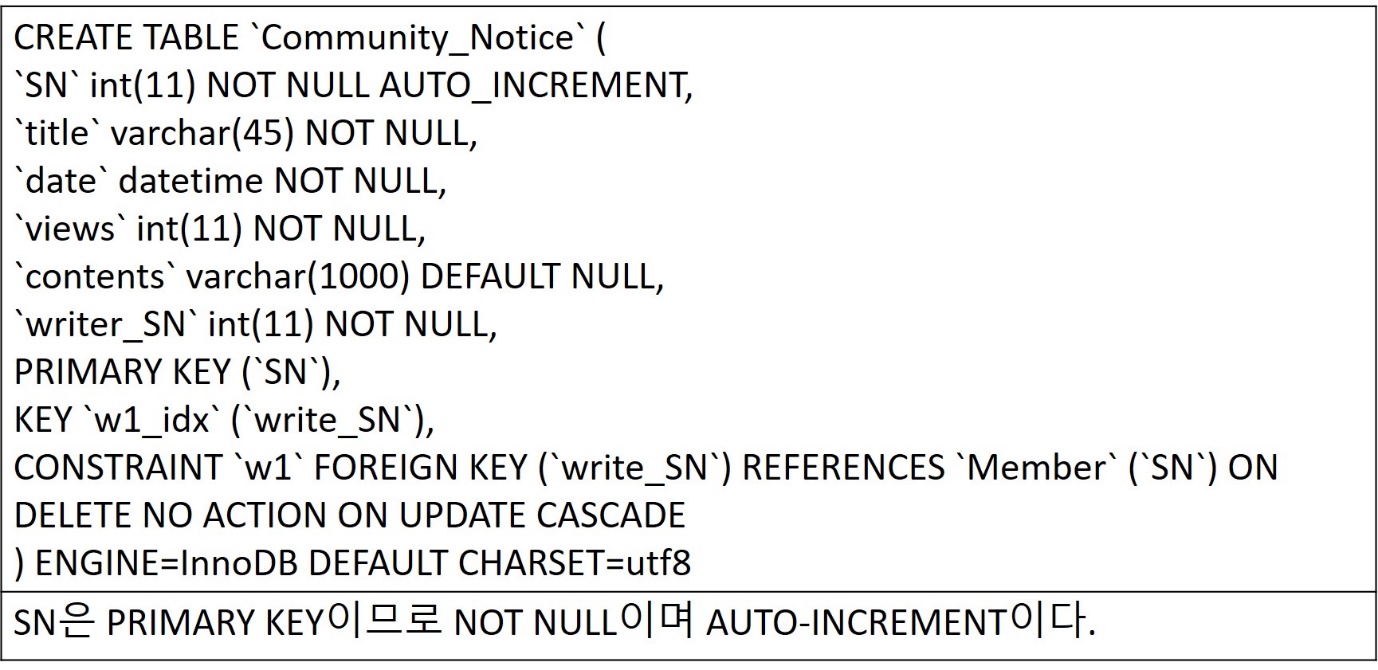
SQL DDL 1. Member

B. Product



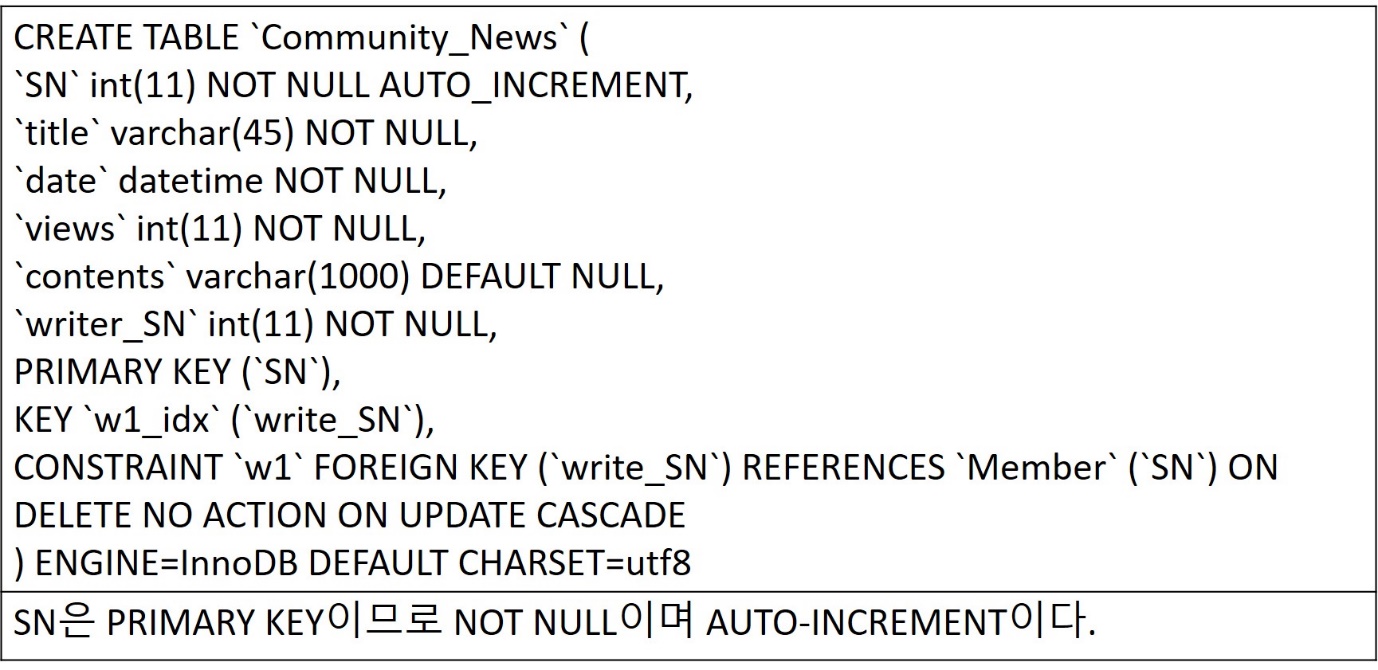
SQL DDL 2. Member

C. Community Notice



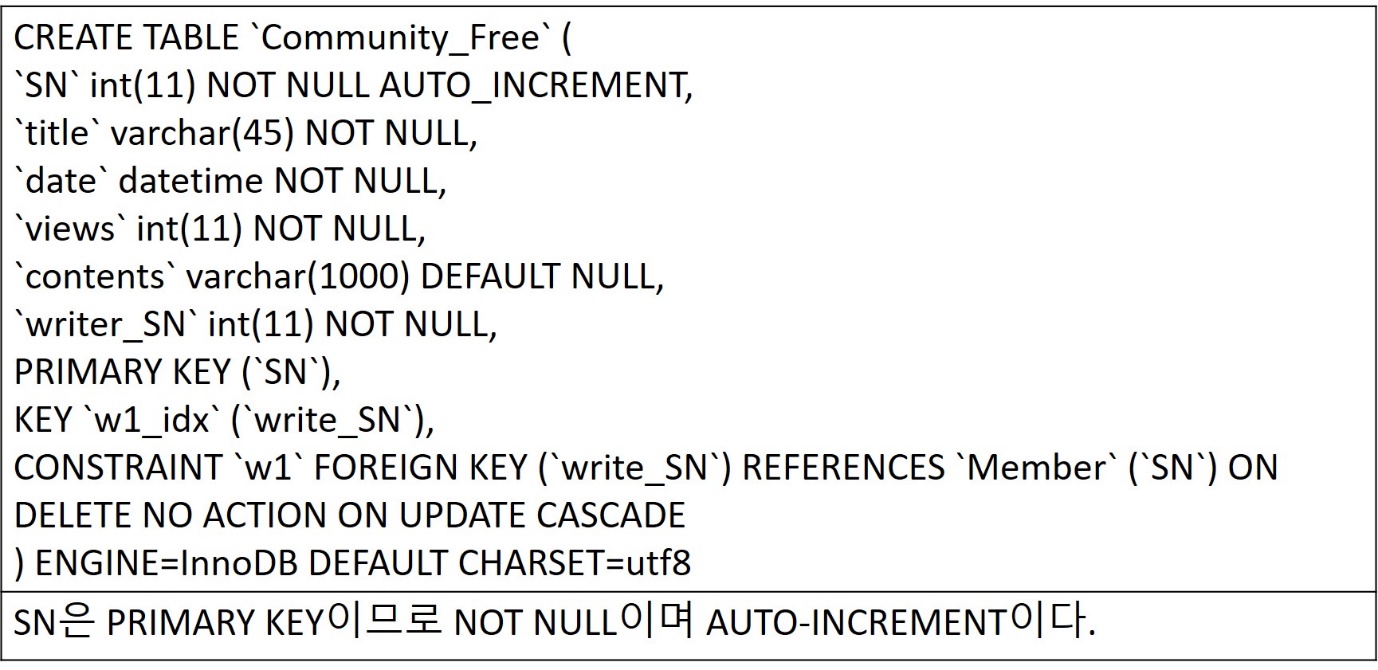
SQL DDL 3. Community Notice

D. Community News



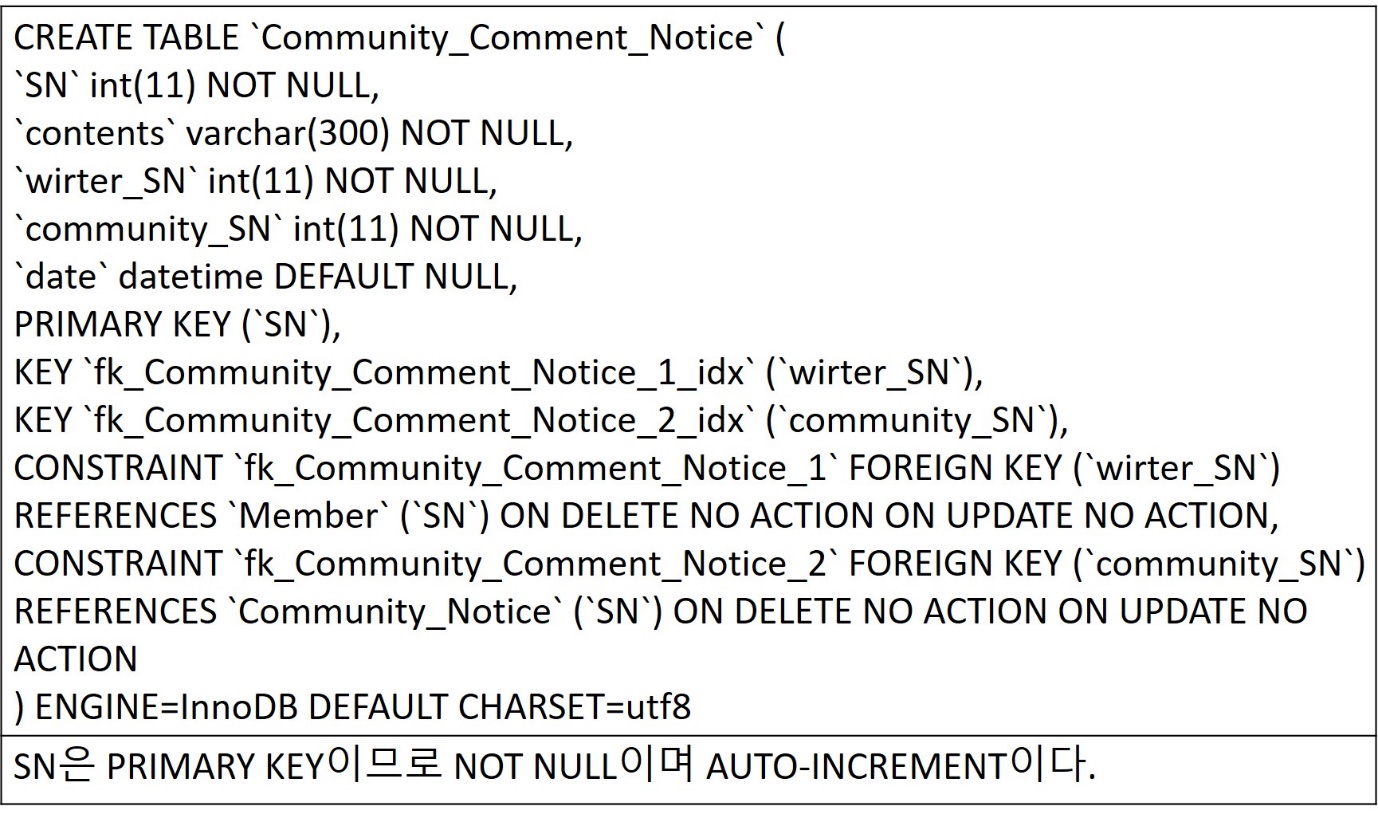
SQL DDL 4. Community News

1. Community Free



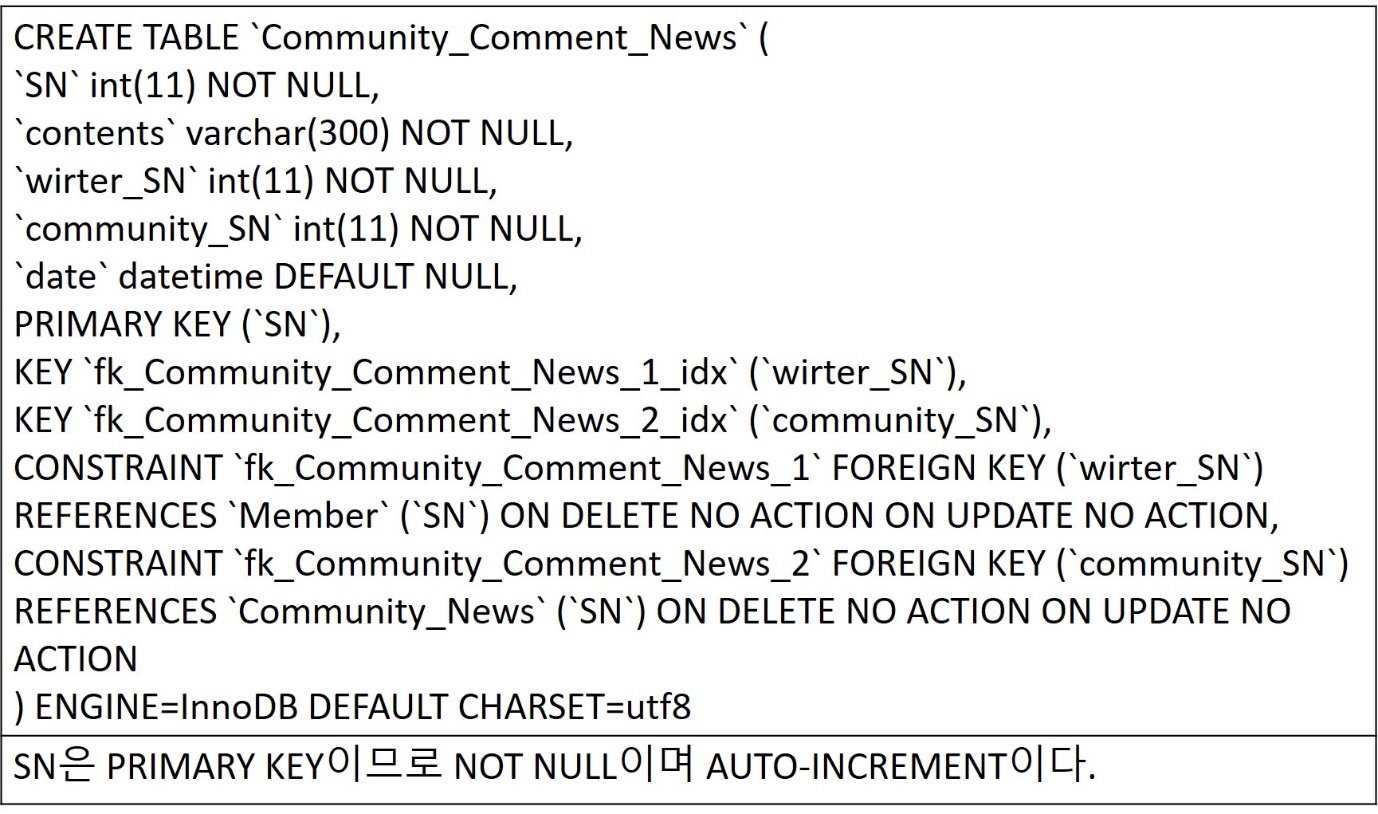
SQL DDL 5. Community Free

1. Community Comment Notice



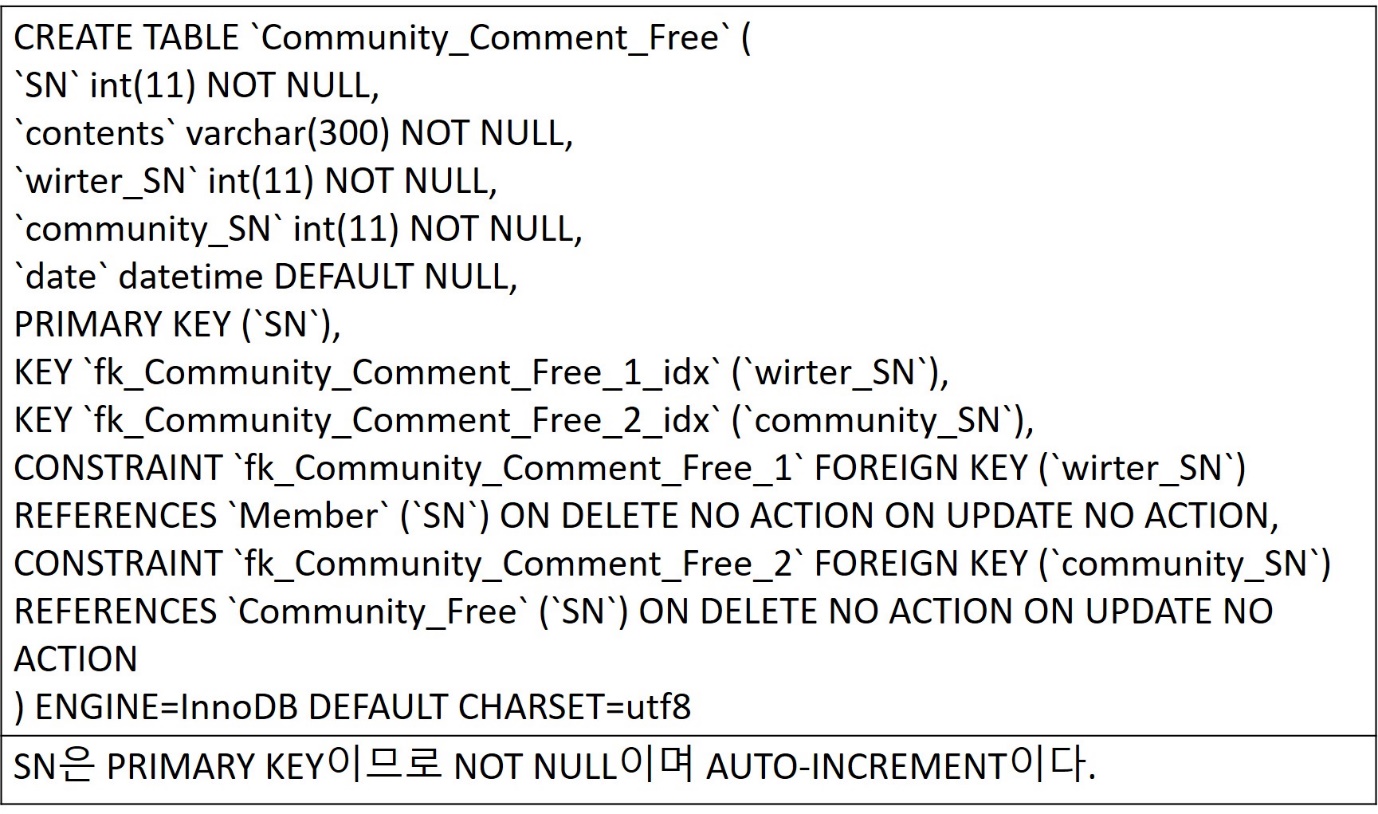
SQL DDL 6. Community Comment Notice

1. Community Comment Notice



SQL DDL 7. Community Comment Notice

1. Community Comment Free



SQL DDL 8. Community Comment Free

12. Testing Plan

12.1. Objectives

시스템이 의도한 방향으로 실행되고 시스템 내부의 결함을 찾기 위해 testing한다. 이를 위해 설계단계에 미리 계획한다. 이 때, Testing Plan에서는 Testing Policy와 여러 Test Case에 대해 기술한다.

12.2. Testing Policy

‘FaReDy’ 시스템의 개발에서는 크게 3 단계로 나누어 testing을 진행한다. Developing Testing, Release Testing, User Testing으로 나눠지며, Developing Testing은 다시 Component Testing, Integrating Testing, System Testing, Acceptance Testing의 4 단계로 나눠진다.

**1) Developing Testing**

개발 과정에서 수행되는 testing으로 Component Testing은 각 component(unit) 단위로 개발이 진행되면, 각 요소들이 개발한 후에 제대로 작동하는지 확인하는 testing이다. Integrating Testing은 Component Testing 후 각 요소들을 하나씩 점진적으로 합치면서 하는 testing이다. System Testing은 모든 하위 시스템을 하나로 합친 후, 그 시스템이 잘 동작하는지 testing하는 것이다. Acceptance Testing은 사용자의 정보를 이용하여 시스템에 대한 사용자의 요구사항을 testing하는 것이다.

**2) Release Testing**

사용자에게 출시하기 전에 최종 시스템을 testing하는 것이다. 요구사항 명세서에서 작성되었던 요구사항이 제대로 반영되었는지를 확인한다.

**3) User Testing**

사용자가 사용자의 환경에서 시스템을 testing하는 것이다.

12.3. Test Case

A. User Management System

**A.1. Log in and Sign up using SNS(Naver / Facebook)**

1) User: SNS(Naver / Facebook)을 이용하여 로그인을 시도한다.

2) 시스템 동작: Account DB에 저장되어 있는 데이터와 로그인하려고 하는 User의 데이터가 일치하는지 확인한다.

2-1) (로그인 성공)

시스템 알림: “Faredy에 오신 것을 환영합니다.”

시스템 동작: 로그인한다.

2-2) (로그인 실패)

시스템 알림: “Faredy회원이 아니군요, 회원가입을 해주세요.”

시스템 동작: SNS(Naver / Facebook)을 통한 회원가입 절차로 넘어간다. 정보 제공 동의에 대한 요청을 한 후, 동의가 이루어지면 SNS에서 제공받은 아이디가 회원가입 양식에 맞춰 자동으로 입력되어 있어 사용자는 회원가입 확인 버튼만 누르면 가입이 완료된다.

**A.2. Log in and Sign up without SNS(Naver / Facebook).**

1) User: Faredy 시스템을 이용하기 위해 기존에 가입했던 정보(ID, PW)를 입력하여, 로그인을 시도한다.

2) 시스템 동작: Account DB에 저장되어 있는 데이터와 로그인하려고 하는 User의 데이터가 일치하는지 확인한다.

2-1) (로그인 성공)

시스템 알림: “Faredy에 오신 것을 환영합니다.”

시스템 동작: 로그인한다.

2-2) (로그인 실패)

시스템 알림: “ID와 PW가 틀렸습니다. 올바른 ID와 PW를 입력해주세요.”

시스템 동작: 로그인 화면으로 돌아간다.

B. Seller Exclusive System

**B.1. Product Management**

1) Seller: 제품 등록 버튼을 선택한다.

2) 시스템 동작: 등록하고자 하는 제품의 타입을 요청한다.

3) Seller: 제품의 타입을 선택한다.

4) 시스템 동작: 보드의 색상, 제품명, 가격, 물량, 제품 이미지의 정보를 요청한다.

**B.2. User Purchase**

1) User: 구매하고자 하는 제품에 대해 검색한다.

2) 시스템 동작: 해당되는 제품에 대한 Seller의 정보를 보여준다.

3) User: 구매하고자 하는 제품에 대해 구매 선택 버튼을 클릭한다.

4) 시스템 알림: “해당 사이트로 이동하겠습니다. 즐거운 쇼핑 되세요.”

시스템 동작: 링크를 통해 해당되는 제품에 대한 구매 사이트로 화면을 변경한다.

C. Picture Search System

**C.1. User가 Picture을 업로드하여 서비스를 제공받으려 한다.**

1) User: Picture을 업로드 하려 한다.

2) 시스템 동작: 해당되는 사진이 추출가능한 양식인지 확인한다.

3-1) 추출가능

시스템 알림: “해당 사진은 이용 가능합니다.”

시스템 동작: 해당되는 사진을 분석하여 키워드를 추출한다.

3-1) 추출불가능

시스템 알림: “해당 사진은 이용 불가능합니다.”

시스템 동작: 사진을 추출하기 위한 양식을 보여주고 다시 사진을 입력 받는다.

4) 시스템 동작: 키워드를 추출하여 Product DB에 있는지 확인한다.

5-1) 키워드에 만족하는 제품이 없을 경우

시스템 알림: “사진과 관련된 제품이 없습니다. 다른 사진을 업로드 해주세요.”

시스템 동작: 사진 업로드 창으로 이동하여 사진을 입력 받는다.

5-2) 키워드에 만족하는 제품이 있는 경우

시스템 알림: “사진과 관련된 제품을 보여주겠습니다.”

시스템 동작: 키워드에 해당되는 제품을 Product DB에서 가져와 리스트를 출력한다.

D. Filter Search System

**D.1. User가 키워드를 통해 제품을 필터링 하려고 한다.**

1) User: 키워드를 입력하고 필터링을 하려고 한다.

2) 시스템 동작: 키워드에 해당하는 제품이 Product DB에 있는지 확인한다.

3-1) 키워드에 만족하는 제품이 없을 경우

시스템 알림: “키워드에 해당하는 제품이 없습니다. 다시 입력해주세요.”

시스템 동작: 키워드 입력창으로 이동하여 키워드를 다시 입력 받는다.

3-2) 키워드에 만족하는 제품이 있는 경우

시스템 알림: “키워드에 해당되는 제품을 보여드리겠습니다.”

시스템 동작: 키워드에 해당되는 제품을 Product DB에서 가져와 리스트를 출력한다.

E. Recommendation System

**E.1. User가 Recommendation 서비스를 제공받으려 한다**.

1) User: Recommendation 서비스를 제공받으려 한다.

2) 시스템 동작: Recommendation 서비스를 제공받기 위한 조건을 만족하는지 확인한다.

3-1) 조건 만족

시스템 알림: “Recommendation 서비스 화면으로 이동하겠습니다.”

시스템 동작: User의 정보를 Recommendation DB에서 가져와 리스트를 출력한다.

3-1) 조건 불만족

시스템 알림: “Recommendation 서비스를 제공받기 위한 정보가 부족합니다. 정보를 입력해주세요.”

시스템 동작: User의 정보를 입력 받기 위한 화면을 출력한다.

F. Trend System

**F.1. User가 Trend 정보를 확인하려고 한다.**

1) User: Category를 선택하여 Trend 정보를 확인한다.

2) 시스템 동작: Category에 해당하는 데이터가 Trend DB에 존재하는지 확인한다.

3-1) 데이터가 있는 경우

시스템 알림: “최신 트렌드 정보입니다.”

시스템 동작: 조건에 맞는 리스트를 출력한다.

3-2) 데이터가 없는 경우

시스템 알림: “정보가 부족합니다. 다른 카테고리를 선택해주세요.”

시스템 동작: 카테고리 선택 창으로 이동한다.

**F.2. User가 Trend 정보를 받고 구매하려고 한다.**

1) User: Trend 리스트에서 마음에 드는 상품에 대해 구매하려 한다.

2) 시스템 동작: 해당되는 제품의 재고가 있는지 확인하고 해당 제품에 대한 구매의사가 있기 때문에 Trend DB에 해당 제품의 관심도 대한 정보를 업데이트한다.

3-1) 재고가 없는 경우

시스템 알림: “재고가 없습니다. 다른 제품을 구매해주세요.”

시스템 동작: 바로 전의 선택 창으로 이동한다.

3-2) 재고가 있는 경우

시스템 알림: “해당 사이트로 이동하겠습니다. 즐거운 쇼핑 되세요.”

시스템 동작: Seller 사이트로 이동하여 해당 제품에 대한 구매창을 보여준다.

G. Community System

**G.1. User가 ‘Notice’ 게시판에 글을 남기려고 시도한다.**

1) User: 글 작성을 시도한다.

2) 시스템 동작: ‘Notice’ 게시판에 글을 남길 수 있는 권한이 있는지 확인한다.

3-1) 권한이 없는 경우

시스템 알림: “이 게시판에는 글을 남길 수 없습니다.”

시스템 동작: Community 기본 화면으로 돌아간다.

3-2) 권한이 있는 경우

시스템 동작: Write 버튼을 선택하면, 작성 페이지로 연결된다.

4-1) Write

시스템 알림: “게시글 작성이 완료되었습니다.”

시스템 동작: 글이 작성된다.

4-2) Cancel

시스템 동작: 글 목록 페이지로 돌아간다.

**G.2. User가 일반 게시판에 글을 남기려고 시도한다.**

1) User: 글 작성을 시도한다.

2) 시스템 동작: Write 버튼을 선택하면, 작성 페이지로 연결된다.

2-1) Write

시스템 알림: “게시글 작성이 완료되었습니다.”

시스템 동작: 글이 작성된다.

2-2) Cancel

시스템 동작: 글 목록 페이지로 돌아간다.

**G.3. User가 search filter를 이용하려고 한다.**

1) User: 검색 필터를 선택하고 키워드를 검색한다.

2) 시스템 동작: Search 버튼을 선택하면, 해당되는 글 목록이 출력된다.

13. Development Environment

13.1. Objectives

Development Environment에서는 개발자의 환경에 대해 설명한다. 사용한 프로그래밍 언어와 IDE에 대해 서술한다.

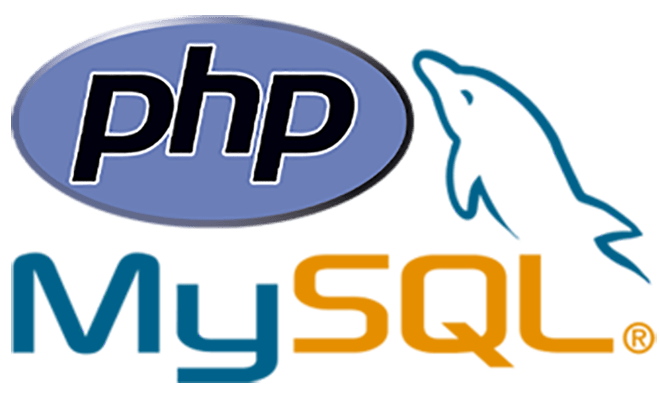
13.2. Programming Language & IDE

A. Programming Language



Picture 16. node.js Logo

서버 개발 언어로는 Node.js를 사용한다. Node.js는 JavaScript기반의 네트워크 어플리케이션 개발용 오픈소스 플랫폼이다. 일반적인 Java Script와는 달리 서버에서 바로 동작하며, 자체 HTTP 서버 라이브러리를 포함하고 있어 아파치 등의 별도의 소프트웨어 없이 서버를 동작시킬 수 있다. Node.js는 JavaScript기반으로 별도의 컴파일 과정을 필요로 하지 않기 때문에, 웹 브라우저에서 바로 결과를 확인할 수 있다. 따라서 ‘FaReDy’ 시스템에 적합한 실시간 처리 웹 서버 구현에 용이하다.



Picture 17. php & MySQL Logo

데이터베이스 관리를 위해 php와 MySQL을 이용한다.



Picture 18. Bootstrap Logo

부트스트랩(Bootstrap)은 웹사이트를 쉽게 만들 수 있게 도와주는 HTML, CSS, JS 프레임워크이다. 하나의 CSS 로 휴대폰, 태블릿, 데스크탑까지 다양한 기기에서 작동한다. 다양한 기능을 제공하여 사용자가 쉽게 웹사이트를 제작, 유지, 보수할 수 있도록 도와준다.

B. IDE

개발 과정에서 사용할 통합개발환경(IDE)가 없기 때문에 기술하지 않는다.

13.3. Coding Rule

‘FaReDy’ 시스템에서 사용한 Coding Rule은 다음과 같다.

1) 반응형 웹 페이지를 구축하는 시스템이기 때문에 jade 파일과 그 파일을 대응되는 js 파일을 작성한다.

2) 서버에 바로 작업하는 경우 버전 관리가 어렵기 때문에, 바로 서버에 작업하는 것이 아니라 GitHub를 이용하여 코드를 관리한다.

3) GitHub를 이용하는 경우, 두 명 이상이 동시에 같은 파일은 작업하는 경우가 없도록 한다.

13.4. Version Management Tool



Picture 19. GitHub Logo

효율적인 개발과 코드 관리, 공유를 위해 GitHub를 이용한다. GitHub는 전세계적으로 많이 사용되는 웹 기반의 호스팅 서비스로 신뢰도가 높은 서비스이다. 또한 오픈 소스가 많이 등록되어 있어, 이를 이용하는 데 편리하고 유용한 서비스를 제공한다.

14. Develop Plan

14.1. Objectives

Develop Plan에서는 개발 계획에 대해 서술한다. 이 때, Gantt chart를 이용하여 개발 계획과 실제 계발 흐름에 대해 서술하고자 한다.

14.2. Gantt Chart

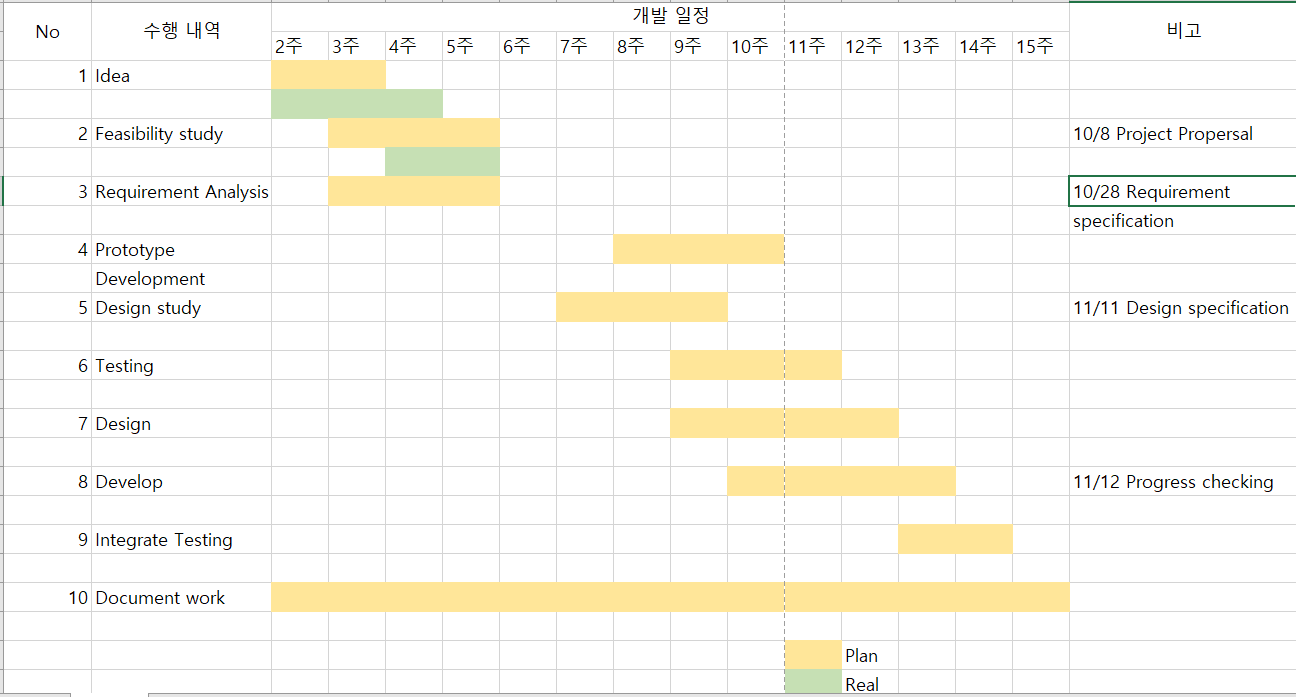


Table 29. Gantt Chart

15. Index

[Picture 1. UML Logo 15](#_Toc529744209)

[Picture 2. Package Diagram Example 17](#_Toc529744210)

[Picture 3. Deployment Diagram Example 18](#_Toc529744211)

[Picture 4. Class Diagram Example 19](#_Toc529744212)

[Picture 5. State Diagram Example 19](#_Toc529744213)

[Picture 6. Sequence Diagram Example 20](#_Toc529744214)

[Picture 7. ER Diagram Example 21](#_Toc529744215)

[Picture 8. Flow Chart Maker & Online Diagram Software 22](#_Toc529744216)

[Picture 9. Amazon Web Service(AWS) Logo 22](#_Toc529744217)

[Picture 10. User Management System 23](#_Toc529744218)

[Picture 11. Seller Exclusive System의 구조 24](#_Toc529744219)

[Picture 12. Filter Search System의 구조 24](#_Toc529744220)

[Picture 13. Community System의 구조 24](#_Toc529744221)

[Picture 14. Recommendation System & Trend System의 구조 25](#_Toc529744222)

[Picture 15. JSON Logo 66](#_Toc529744223)

[Picture 16. node.js Logo 91](#_Toc529744224)

[Picture 17. php & MySQL Logo 91](#_Toc529744225)

[Picture 18. Bootstrap Logo 92](#_Toc529744226)

[Picture 19. GitHub Logo 93](#_Toc529744227)

[Diagram 1. Context Model 26](#_Toc529744228)

[Diagram 2. Block Diagram 27](#_Toc529744229)

[Diagram 3. Activity Diagram - Sign up 28](#_Toc529744230)

[Diagram 4. Activity Diagram - Log in 28](#_Toc529744231)

[Diagram 5. Activity Diagram - Product Management 29](#_Toc529744232)

[Diagram 6. Activity diagram - Picture Search 29](#_Toc529744233)

[Diagram 7. Activity diagram - Filter Search 30](#_Toc529744234)

[Diagram 8. Activity diagram - Serving Recommendation 30](#_Toc529744235)

[Diagram 9. Activity diagram - Batch Processing 31](#_Toc529744236)

[Diagram 10. Package Diagram 32](#_Toc529744237)

[Diagram 11. Deployment Diagram 32](#_Toc529744238)

[Diagram 12. Class Diagram - User Management System 33](#_Toc529744239)

[Diagram 13. Sequence Diagram - Sign Up using SNS(Naver/Facebook) 34](#_Toc529744240)

[Diagram 14. Sequence Diagram - Log In with using SNS 35](#_Toc529744241)

[Diagram 15. Sequence Diagram - Sign Up without SNS 36](#_Toc529744242)

[Diagram 16. Sequence Diagram - Log In without SNS 37](#_Toc529744243)

[Diagram 17. State Diagram - Sign Up using SNS 38](#_Toc529744244)

[Diagram 18. State Diagram - Sign Up without SNS 38](#_Toc529744245)

[Diagram 19. Class Diagram - Seller Exclusive System 39](#_Toc529744246)

[Diagram 20. Sequence Diagram - Product Management 41](#_Toc529744247)

[Diagram 21. Sequence Diagram - User Purchase System 42](#_Toc529744248)

[Diagram 22. State Diagram - Product Management System 42](#_Toc529744249)

[Diagram 23. State Diagram - User Purchase System 43](#_Toc529744250)

[Diagram 24. Class Diagram - Picture Search System 44](#_Toc529744251)

[Diagram 25. Sequence Diagram - Picture Search System 46](#_Toc529744252)

[Diagram 26. Class Diagrma - Filter Search System 48](#_Toc529744253)

[Diagram 27. Sequence Diagram - Filter Search System 49](#_Toc529744254)

[Diagram 28. State Diagram - Filter Search System 50](#_Toc529744255)

[Diagram 29. Class Diagram - Recommendation System 51](#_Toc529744256)

[Diagram 30. Sequence Diagram - Recommendation System 53](#_Toc529744257)

[Diagram 31. State Diagram - Recommendation System 53](#_Toc529744258)

[Diagram 32. Class Diagram - Trend System 54](#_Toc529744259)

[Diagram 33. Sequence Diagram - View 56](#_Toc529744260)

[Diagram 34. Sequence Diagram - Write 57](#_Toc529744261)

[Diagram 35. Sequence Diagram - Search 57](#_Toc529744262)

[Diagram 36. Sequence Diagram - Edit 58](#_Toc529744263)

[Diagram 37. State Diagram - View 58](#_Toc529744264)

[Diagram 38. State Diagram - Write 59](#_Toc529744265)

[Diagram 39. State Diagram - Search 59](#_Toc529744266)

[Diagram 40. State Diagram - Edit 59](#_Toc529744267)

[Diagram 41. Class Diagram - Community System 60](#_Toc529744268)

[Diagram 42. Sequence Diagram - View 62](#_Toc529744269)

[Diagram 43. Sequence Diagram - Write 62](#_Toc529744270)

[Diagram 44. Sequence Diagram - Search 63](#_Toc529744271)

[Diagram 45. Sequence Diagram - Edit 63](#_Toc529744272)

[Diagram 46. State Diagram - View 64](#_Toc529744273)

[Diagram 47. State Diagram - Write 64](#_Toc529744274)

[Diagram 48. State Diagram - Search 64](#_Toc529744275)

[Diagram 49. State Diagram - Edit 65](#_Toc529744276)

[Diagram 50. ER Diagram 72](#_Toc529744277)

[Diagram 51. Member 73](#_Toc529744278)

[Diagram 52. Product 73](#_Toc529744279)

[Diagram 53. Community 74](#_Toc529744280)

[Diagram 54. Management Product 74](#_Toc529744281)

[Diagram 55. Recommend 75](#_Toc529744282)

[Diagram 56. User Community 76](#_Toc529744283)

[SQL DDL 1. Member 81](#_Toc529744284)

[SQL DDL 2. Member 81](#_Toc529744285)

[SQL DDL 3. Community Notice 82](#_Toc529744286)

[SQL DDL 4. Community News 82](#_Toc529744287)

[SQL DDL 5. Community Free 83](#_Toc529744288)

[SQL DDL 6. Community Comment Notice 83](#_Toc529744289)

[SQL DDL 7. Community Comment Notice 84](#_Toc529744290)

[SQL DDL 8. Community Comment Free 84](#_Toc529744291)

[Table 1. Version Update History 14](#_Toc529744292)

[Table 2. Login Protocol - Request 67](#_Toc529744293)

[Table 3. Login Protocol - Response 67](#_Toc529744294)

[Table 4. Registration Protocol - Request 67](#_Toc529744295)

[Table 5. Registration Protocol - Response 67](#_Toc529744296)

[Table 6. Community Post Edit Protocol - Request 67](#_Toc529744297)

[Table 7. Community Post Edit Protocol - Response 68](#_Toc529744298)

[Table 8. Community Search Protocol - Request 68](#_Toc529744299)

[Table 9. Community Search Protocol - Request 68](#_Toc529744300)

[Table 10. Community View Protocol - Request 68](#_Toc529744301)

[Table 11. Community View Protocol - Response 69](#_Toc529744302)

[Table 12. Product Post Edit Protocol - Request 69](#_Toc529744303)

[Table 13. Product Post Edit Protocol - Response 69](#_Toc529744304)

[Table 14. Product View Protocol - Request 70](#_Toc529744305)

[Table 15. Product View Protocol - Response 70](#_Toc529744306)

[Table 16. Product Search Protocol - Request 70](#_Toc529744307)

[Table 17. Product Search Protocol - Response 71](#_Toc529744308)

[Table 18. Picture Upload Protocol - Request 71](#_Toc529744309)

[Table 19. Picture Upload Protocol - Response 71](#_Toc529744310)

[Table 20. Customer 76](#_Toc529744311)

[Table 21. Seller 77](#_Toc529744312)

[Table 22. Product 77](#_Toc529744313)

[Table 23. Community\_notice 78](#_Toc529744314)

[Table 24. Community\_news 78](#_Toc529744315)

[Table 25. Community\_Free 78](#_Toc529744316)

[Table 26. Community\_Comment\_Notice 79](#_Toc529744317)

[Table 27. Community\_Comment\_News 79](#_Toc529744318)

[Table 28. Community\_Comment\_Free 79](#_Toc529744319)

[Table 29. Gantt Chart 94](#_Toc529744320)