버전 1.0

2019.10.27

Requirement Specification

팀9

가웅현

조현우

송선민

차승호

목차

[1. Preface 5](#_Toc23116069)

[1.1 Readers 5](#_Toc23116070)

[A. User Requirement Readers 5](#_Toc23116071)

[B. System Requirement Readers 5](#_Toc23116072)

[1.2 Document Structure 5](#_Toc23116073)

[A. Preface 6](#_Toc23116074)

[B. Introduction 6](#_Toc23116075)

[C. Glossary 6](#_Toc23116076)

[D. User Requirements Definition 6](#_Toc23116077)

[E. System Architecture 6](#_Toc23116078)

[F. System Requirements Specification 7](#_Toc23116079)

[G. System Models 7](#_Toc23116080)

[H. System Evolution 7](#_Toc23116081)

[I. Appendices 7](#_Toc23116082)

[J. Index 8](#_Toc23116083)

[1.3 Version Update History 8](#_Toc23116084)

[2. Introduction 9](#_Toc23116085)

[2.1 Needs 9](#_Toc23116086)

[2.2 Eat Now 12](#_Toc23116087)

[2.3 Expected Effect 12](#_Toc23116088)

[A. User 12](#_Toc23116089)

[B. Creator 13](#_Toc23116090)

[C. Commercial 13](#_Toc23116091)

[3. Glossary 13](#_Toc23116092)

[3.1 Term Definitions 13](#_Toc23116093)

[4. User Requirements Definition 14](#_Toc23116094)

[4.1 Functional Requirements 14](#_Toc23116095)

[A. 회원가입 14](#_Toc23116096)

[B. 로그인 15](#_Toc23116097)

[C. 검색 15](#_Toc23116098)

[D. 비디오 상세정보 15](#_Toc23116099)

[E. 추천 영상 16](#_Toc23116100)

[F. 랭킹 16](#_Toc23116101)

[G. 마이 페이지 16](#_Toc23116102)

[4.2 Non-Functional Requirements 16](#_Toc23116103)

[A. Product Requirement 16](#_Toc23116104)

[B. Organization Requirement 17](#_Toc23116105)

[C. External Requirement 18](#_Toc23116106)

[5. System Architecture 19](#_Toc23116107)

[5.1 Objective 19](#_Toc23116108)

[5.2 Front end architecture 19](#_Toc23116109)

[5.3 User space architecture 20](#_Toc23116110)

[5.4 Back end Architecture 21](#_Toc23116111)

[5.5 Sub-system Architecture 22](#_Toc23116112)

[A. muckbang specification 22](#_Toc23116113)

[B. recommend system 23](#_Toc23116114)

[6. System Requirement Specification 24](#_Toc23116115)

[6.1 Front end Functional Requirements 24](#_Toc23116116)

[A. Sign up 24](#_Toc23116117)

[B. Sign in 26](#_Toc23116118)

[C. Category 27](#_Toc23116119)

[D. Recommend video 27](#_Toc23116120)

[E. Video ranking 28](#_Toc23116121)

[F. My page 28](#_Toc23116122)

[G. Information of video 29](#_Toc23116123)

[H. Review 30](#_Toc23116124)

[I. Connect to Delivery/Market system 30](#_Toc23116125)

[6.2 Backend Functional Requirement 31](#_Toc23116126)

[A. Register user for sign up 31](#_Toc23116127)

[B. Confirm user for sign in 32](#_Toc23116128)

[C. Get video information from video streaming site 32](#_Toc23116129)

[D. Recommended video settings based on user experience. 33](#_Toc23116130)

[6.2 Non-Functional Requirements 33](#_Toc23116131)

[A. Product Requirement 33](#_Toc23116132)

[B. Organization Requirement 34](#_Toc23116133)

[C. External Requirement 35](#_Toc23116134)

[6.3 Scenario 35](#_Toc23116135)

[A. Sign up and sign in scenario 35](#_Toc23116136)

[B. Searching Videos Scenario 36](#_Toc23116137)

[C. Video details Scenario 37](#_Toc23116138)

[D. My page Scenario 38](#_Toc23116139)

[7. System Models 39](#_Toc23116140)

[7.1 Context Models 39](#_Toc23116141)

[A. Context Model 39](#_Toc23116142)

[B. Process Diagram 40](#_Toc23116143)

[7.2 Interaction Models 41](#_Toc23116144)

[A. Use case Diagram 41](#_Toc23116145)

[B. Tubular Description for each Use case 41](#_Toc23116146)

[7.3 Behavioral Models 45](#_Toc23116147)

[A. Database Modeling 45](#_Toc23116148)

[8. System Evolutions 46](#_Toc23116149)

[8.1 Limitation and Assumption 46](#_Toc23116150)

[8.2 Evolutions of Requirement 47](#_Toc23116151)

[A. 커뮤니티 47](#_Toc23116152)

[B. AI를 이용한 자동화 48](#_Toc23116153)

[8.3 Evolutions of Environment 48](#_Toc23116154)

[A. Scope of Service 48](#_Toc23116155)

[9. Appendices 49](#_Toc23116156)

[9.1 Hardware Requirements 49](#_Toc23116157)

[9.2 Database Requirements 49](#_Toc23116158)

[9.2 User-System Requirements 49](#_Toc23116159)

[9.3 Development Process 49](#_Toc23116160)

[10. Index 49](#_Toc23116161)

[10.1 Figure Index 49](#_Toc23116162)

[10.2 Table Index 49](#_Toc23116163)

[10.3 Diagram Index 50](#_Toc23116164)

[10.4 Reference 51](#_Toc23116165)

# 1. Preface

본 문서의 예상되는 독자들과 전반적인 구조, 그리고 각 부분의 역할에 대해 설명한다. 또한 버전기록도 작성한다.

## 1.1 Readers

본 문서는 먹방에 관한 시스템 요구사항 명세서로서 크게 user와 system의 관점으로 나눌 수 있다.

### A. User Requirement Readers

User requirement의 예상 독자는 본 시스템의 사용자이다. 유저의 관점에서 요구사항, 제약사항, 제공되는 기능들을 자연어나 쉽게 이해할 만한 다이어그램으로 설명한다. 최대한 전문용어를 자제하고 알기 쉽게 설명한다. Functional requirement는 서비스의 기능요소들을 하나하나 기록하였고 non-functional requirement는 성능, 보안 등 시스템 전체에서 나타나는 요구사항들을 정리하였다.

### B. System Requirement Readers

System Requirement의 예상 독자는 개발자나 개발에 관여하는 이해관계자들이 될 수 있다. 시스템의 기능과 제약사항을 구조적으로 표현함으로써 실제 개발 과정에서 참고가 될 수 있도록 하며 경우에 따라서는 디자인 명세서에서 작성할 부분도 포함하고 있다.

## 1.2 Document Structure

본 요구사항 명세서는 총 10개의 부분으로 구성된다. Preface, Introduction, Glossary, User Requirements Definition, System Architecture, System Requirements Specification, System Models, System Evolution, Revenue Model, Appendices, Index로 구성된다. 각 장의 역할과 전반적인 서술 내용은 다음과 같다.

### A. Preface

본 문서의 예상되는 독자들과 전반적인 구조, 그리고 각 부분의 역할에 대해 설명한다. 또한 버전기록도 작성한다.

### B. Introduction

본 시스템의 background와 needs를 설명하고 해당 시스템이 가질 수 있는 기능들을 설명한다. 또한 본 시스템의 목적과 기대효과를 설명한다. 기대효과는 사용자, 크리에이터, 음식관련 회사들의 측면에서 설명한다.

### C. Glossary

본 요구사항 명세서에 등장하는 용어들에 대해 정의한다. 일반인이 읽더라도 문서를 이해할 수 있도록 가능한 모든 용어에 대해 간단하고 이해하기 쉽도록 설명한다. 범위는 기본적인 용어부터 개발 수준에서 사용되는 용어까지 다양하다.

### D. User Requirements Definition

본 요구사항 명세서에서 제시한 사용자에게 제공되는 서비스에 대하여 설명한다. Non-functional system requirements도 설명한다. 자연어를 사용하여 시스템의 잠재적인 사용자라면 모두 이해할 수 있게 한다.

### E. System Architecture

본 요구사항 명세서에 제시한 목표 시스템의 Architecture에 대한 개요를 보여준다. 시스템의 Architecture를 각 기능별로 설명한다. Front end와 back end로 나누어서 작성하였으며 유저의 요구사항들을 sub-system으로 그룹화 하였다. 설명은 자연어와 다이어그램을 사용하여 구조적이고 이해하기 쉽게 설명하였다.

### F. System Requirements Specification

System architecture에서 설명한 sub-system에서 2가지 분류, Functional requirements와 non-functional requirements로 나누어 구체적으로 설명하였다. 또한, 각 기능에 따른 시나리오와 함께 설명하여 각 기능의 흐름을 보다 간략적으로 확인할 수 있다.

### G. System Models

System, System component, 그리고 System environment사이의 관계를 설명한다. 유저와 시스템의 상호작용에 관한 부분과 시스템의 각 컴포넌트와의 상호작용을 구체적으로 tabular description을 이용해 구체적으로 기술하고 모델링 하였다. 작성한 모델의 종류로는 use case diagram과 behavioral model등이 있다.

### H. System Evolution

System에 기저하여 있는 주요한 가정들에 대해 설명한다. 또한, System에 일어날 수 있는 예상되는 변화들에 대해 설명한다. 이 항목은 차후에 발생할 시스템의 설계 변경에 있을 Risk를 피할 수 있도록 도와줄 수 있기 때문에 시스템 설계를 담당하는 매니저에 도움이 될 것이다. 이를 통해, 변화에 잘 적응하는 시스템을 개발할 수 있다. 또한 차후에 요구될 수 있는 부분들도 작성하여 발전가능성을 모색한다.

### I. Appendices

개발되는 시스템에 대한 더 자세하고 구체적인 정보들을 제공한다. 앞서 항목에 다루지 않은 Hardware, database, 시스템 사용에 적합한 혹은 최소의 설정 등을 구체적으로 설명한다.

### J. Index

본 문서에 포함된 그림, 테이블, 다이어그램, 관련자료 등 인덱스가 포함된다.

## 1.3 Version Update History

|  |  |  |  |
| --- | --- | --- | --- |
| 버전 | 수정일자 | 변경사항 | |
| 1.0 | 2019.10.27 | 최초 작성 |
|  |  |  |
|  |  |  |
|  |  |  |

# 2. Introduction

본 시스템의 background와 needs를 설명하고 해당 시스템이 가질 수 있는 기능들을 설명한다. 또한 본 시스템의 목적과 기대효과를 설명한다. 기대효과는 사용자, 크리에이터, 음식관련 회사들의 측면에서 설명한다.

## 2.1 Needs

먹방은 이미 하나의 주류문화로 자리잡은 지 오래다. 유튜브를 예로 들면 이미 유튜버들의 상당수는 먹방 유튜버이며 2019년 미디어잡의 유튜브 소비현황 및 선호도 조사결과를 보면 먹방이 유튜브에서 가장 관심있게 시청하는 분야 1위를 달성했다.

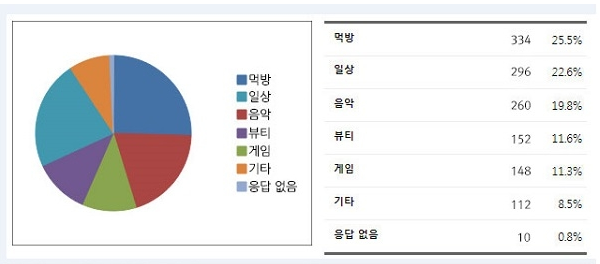


Figure . 유튜브 소비현황 설문조사 결과

먹방의 인기는 무엇일까? 먹방은 식(食)의 개념을 충실히 하면서도 다양한 컨텐츠와 결합해왔다. 요리, 투어, 이색, 정보제공, 예능과 결합해 끊임없이 성장하고 있으며 해외에서도 화제가 되고 있다. 또한 먹는다는 것은 남녀노소 일반적인 주제이기 때문에 시청자의 폭이 다양하다. 마지막으로 먹방은 소비자의 다양한 욕구를 충족시켜 준다. 먹는다는 것에 대한 대리만족, 호기심을 충족시키고, 레시피를 따라해 볼 수 있다. 또한 먹어보고 싶은 음식에 대한 정보를 획득할 수 있다. 여기 트렌드 모니터가 분석한 먹방의 시청이유에 대한 통계자료가 있다.



Figure . 먹방 및 쿡방의 시청이유

다양한 레시피와 음식정보를 얻을 수 있다는 것은 다른 말로 하면 그 레시피와 요리정보를 더욱 검색하고 소비로 이어질 수 있다는 뜻이기도 하다. 한 인터넷 쇼핑몰은 유튜브에서 뜬 음식과 매출사이의 상관관계를 조사했다.



Figure . 유튜브에서 뜬 이색음식과 온라인 쇼핑몰 매출의 상관관계

Figure 3처럼 먹방의 영향은 소비로 이어진다는 것을 알 수 있다. 하지만 소비자는 자신이 원하는 레시피나 음식정보를 찾기 위해서는 여러 사이트를 돌아다니며 그것에 대한 정보를 찾아다녀야 한다. 이러한 정보를 제공할 수 있는 기능은 소비자의 편리를 위해서도, 상술적으로도 중요하다. 하지만 유용한 정보를 같이 제공하는 현재 있는 비디오 스트리밍 사이트들은 단순한 동영상과 댓글정도만 제공하고 있고 따라서 크리에이터들은 영상정보나 영상에 직접 정보를 제공해주어야 하는 번거로움이 따른다.

## 2.2 Eat Now



Figure . Eat Now logo

이 시스템의 목표는 소비자와 크리에이터, 그 음식과 관련된 판매업체들에게 도움을 주기 위해 양질의 정보를 편리하고 보기 쉽도록 제공하는 것이다. 여기서 양질의 정보란 레시피나 음식의 가격, 구매경로와 같이 소비자가 필요로 할 수 있는 정보를 말한다.

또 다른 목표로는 먹방이라는 주제 하에 다양한 스트리밍 사이트들을 한 데로 통합하는 것이다. 유튜브 이외에도 트위치, 아프리카TV, 카카오TV 등 다양한 스트리밍 사이트가 활성화되어 있으며 이것을 통합하고 먹방이라는 주제에 관심이 있는 소비자에게 먹방에 관한 내용만 전문적으로 제공한다면 소비자의 욕구 충족에도 도움이 될 것이다.

이 시스템의 이름은 Eat Now로 하고 비디오 검색, 비디오 인기랭킹과 같은 기본적인 스트리밍 사이트의 기능과 마이페이지, 추천영상과 같은 개인화 기능, 특히 유저가 관심있어 할 만한 먹방에 특화된 정보를 제공한다.

## 2.3 Expected Effect

### A. User

유저는 어떤 음식인지에 대한 상세한 정보와 레시피를 쉽게 접할 수 있고 구매 경로나 음식점 정보, 배달 어플리케이션 링크를 제공받아 더욱 더 편리하게 먹방을 경험할 수 있다. 또한 다양한 스트리밍 사이트를 결합시켜 풍부한 볼거리를 제공받을 수 있고 추천 영상이나 인기 순위같은 기술은 유저의 동영상 선택을 도와준다. 따라서 기존에 먹방을 즐겨보는 사람들에게는 더욱 편리한 시스템이 될 것이다.

### B. Creator

크리에이터는 자신을 더욱 더 인터넷에 홍보할 수 있고 다양한 정보를 지원받을 수 있기 때문에 불편함도 덜어주고 영상의 퀄리티도 높아진다. 또한 다른 먹방도 보면서 참고할 수 있다.

### C. Commercial

배달회사, 음식점, 인터넷 쇼핑몰 등 먹방 컨텐츠와 관련된 모든 회사는 상업적인 측면에서 도움이 될 것이다. Eat Now는 소비자의 검색을 유도하고 직접 링크를 제공하기 때문에 광고의 효과가 증대될 것이고 매출의 증가로 이어질 것이다. 또한 유입경로를 알 수 있기 때문에 그것을 분석하고 앞으로의 매출 전략을 세울 때도 도움이 될 수 있다.

# 3. Glossary

본 요구사항 명세서에 등장하는 용어들에 대해 정의한다. 일반인이 읽더라도 문서를 이해할 수 있도록 가능한 모든 용어에 대해 간단하고 이해하기 쉽도록 설명한다. 범위는 기본적인 용어부터 개발 수준에서 사용되는 용어까지 다양하다.

## 3.1 Term Definitions

|  |  |
| --- | --- |
| 용어 | 정의 |
| 유저 | 본 시스템을 이용하는 사용자를 의미한다. 크리에이터도 유저에 포함될 수 있다. |
| 크리에이터 | 본 시스템에서 동영상의 원작자를 의미한다. 즉 먹방 스트리머가 된다. |
| 관리자 | 본 시스템을 관리하는 개발자, 운영자를 의미한다. |
| 먹방 | 먹방이란 ‘먹다’와 ‘방송’의 합성어로서 본 시스템의 주제가 된다. 여기서는 주로 비디오 스트리밍 사이트의 분야를 통칭한다. |
| 인터페이스 | 유저와 서버가 연결을 주고받는 접점이다. |
| Front end |  |
| Back end |  |
| DB |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

# 4. User Requirements Definition

본 요구사항 명세서에서 제시한 사용자에게 제공되는 서비스에 대하여 설명한다. Non-functional system requirements도 설명한다. 자연어를 사용하여 시스템의 잠재적인 사용자라면 모두 이해할 수 있게 한다.

## 4.1 Functional Requirements

### A. 회원가입

고객에게 회원가입을 요청한다. 기본적으로 로그인 정보가 사용되는 곳은 추천영상, 마이페이지이다. 회원을 구분할 수 있는 정도의 정보만 요구하고 최대한 간단하게 만든다. 또한 SNS를 연동하여 로그인할 수도 있다. SNS는 카카오톡, 네이버를 이용한다.

### B. 로그인

맨 처음 화면은 로그인 창으로 아이디와 비밀번호만으로 로그인할 수 있다. 최초 한번만 로그인하면 앱을 삭제하거나 하지 않는 이상 그대로 유지가 되어 번거롭게 로그인을 계속 할 필요 없이 처음화면이 바로 넘어갈 수 있게 한다. 추천영상이나 시청기록과 같은 부분은 유저마다 다르게 제공되어야 한다.

### C. 검색

유저는 메인 화면이나 동영상을 보는 도중 언제든지 다른 영상을 검색할 수 있다. 데이터베이스에서 검색된 키워드가 들어가는 영상, 또는 그와 연관된 동영상을 찾아 가장 관련성이 높은 순서대로 보여준다. 만약 일치하는 결과가 없다면 E와 비슷하게 사용자가 좋아할 만한 동영상을 추천해준다.

### D. 비디오 상세정보

유저가 시청중인 동영상과 관련된 정보를 동영상과 함께 확인이 가능하도록 한다. 관련된 정보는 다음과 같다.

1) 크리에이터가 먹는 음식들을 이미지와 함께 나열한다.

2) 각 음식들의 이름, 가격, 음식에 대한 한 줄 설명을 보여준다.

3) 각 음식들의 레시피를 정리하여 보여준다.

4) 배달 가능한 음식이라면 링크를 주어 배달 어플리케이션에 직접 접근할 수 있도록 한다.

5) 특정한 음식점에서 구매한 음식은 음식점에 대한 정보를 보여준다.

6) 인터넷 쇼핑몰에서의 음식이름을 검색한 결과를 보여주는 링크를 준다.

7) 영상에 관한 정보(크리에이터에 대한 상세한 설명, 비디오 업로드 일자)을 보여준다.

각 정보는 선택적으로 제공하며 체계적인 구조로 정보를 제공한다.

### E. 추천 영상

유저가 지금까지 시청한 기록을 바탕으로 유저가 좋아할 만한 비디오를 제공한다. 시청기록을 삭제하지 않는 한 시청기록은 계속 유지되어 데이터베이스에 저장된다. 추천영상은 유저의 행동(시청기록이나 검색 키워드)에 의해 더 정밀한 추천을 제공할 수 있어야 한다.

### F. 랭킹

전체 사용자들을 바탕으로 현재 가장 인기가 있는 동영상들을 순위대로 알려준다. 랭킹은 24시간 내 조회수 급상승 동영상, 업로드 날짜 등을 토대로 하며 하루마다 갱신된다.

### G. 마이 페이지

유저는 비밀번호 변경과 같은 계정사항 설정, 시청기록 조회, 나중에 볼 동영상의 기능을 이용할 수 있다. 나중에 볼 동영상은 동영상을 직접 저장하는 것은 아니고 링크만 저장하여 검색하지 않고도 찾을 수 있게 한다. 시청기록이나 검색기록과 같은 사항은 삭제할 수 있고 이는 추천기능에도 영향을 미친다. 또한 시스템의 개선점이나 QnA 등을 할 수 있는 기능도 만든다. 따라서 시스템 관리자와 쉽게 소통할 수 있도록 한다.

## 4.2 Non-Functional Requirements

### A. Product Requirement

1) Performance

유저의 모든 요청들은 서버내에서 즉각적으로 이루어져 유저에게 알맞은 형태로 제공되어야 한다. 동영상에 끊김이 없도록 하고 각 통신은 최소한의 정보만을 전달하도록 한다. 모든 자원은 서버에 저장되고 관리된다.

2) Dependability

신뢰성을 극대화하기 위해 각 동작을 간단하게 한다. 유저는 최대한 간단한 연산만을 수행하여 최대한 failure가 없도록 서버요청 전에 문제가 발생한다면 강제종료를 시키고 서버에게 오류내용을 보낼 수 있도록 한다. 만약 서버가 문제가 발생하기 전 발생했다면 원래상태로 되돌리고 오류 내용을 기록한다.

3) Usability

로그인만 한다면 다른 기기에서도 동일한 기능을 제공하도록 한다. 원활한 접속이 가능하고 기기마다 맞는 규격의 레이아웃을 제공해야 한다. 유저가 학습하지 않아도 직관적으로 이해할 수 있는 UI, 서비스를 제공하고 어플리케이션을 처음 깔았을 때 어떤 들이 있는지 살펴볼 수 있는 서비스를 제공해 누구나 서비스의 기능에 대해 이해할 수 있게 한다.

4) Security

유저의 계정정보, 특히 ID와 비밀번호나 시청기록은 안전하게 저장하여 보호하고 관리자만 접속가능해야 한다. 추천기능 같이 유저마다 다른 결과를 제공해야하는 요청은 매번 서버에서 올바른 유저인지 확인할 수 있어야 한다.

### B. Organization Requirement

1) Development

개발인원이 4명으로 고정되어 있기 때문에 개발 사전에 역할을 나누어 parallel development를 하도록 한다. 또한 waterfall model의 방식을 적용하여 순차적으로 개발하고 문서관리를 철저히 하지만 시스템의 규모가 작기 때문에 변경사항은 가능한 한 받아들이고 역할을 분담하더라도 역할은 유동적으로 변경할 수 있다. Front end와 back end는 최대한 독립적으로 구성한다. 또한 서버나 다른 하드웨어에 의존적이지 않도록 코드를 작성한다.

2) Operational

서버는 상용 클라우드 서버를 이용하여 배포하고 사용자가 늘어난다면 서버 성능을 확장하는 쪽으로 한다. 또한 서버의 변경에도 대응할 수 있도록 코드를 작성한다. 또한 고객과의 소통(오류사항 보고, 서비스 피드백)을 지원할 수 있어야 한다.

### C. External Requirement

1) Regulatory

유저 관련한 정보들의 저장과 이용에 대한 정보를 정확히 명시하고 동의를 받아야 한다. 또한 SNS 계정을 이용할 시에도 적용된다. 유저에 관한 데이터를 수집하고 그를 이용하는 기능을 추가할 때는 운영정책을 위반하고 있지 않는지 항상 확인하도록 한다. 또한 이 서비스가 저작권이나 비디오의 원 사이트의 정책을 위반하고 있지 않는지 살펴봐야한다.

2) Safety, Security

유저식별에 최소한의 정보만 제공하고 로그인하지 않아도 서비스를 이용할 수 있게 한다. 안드로이드, http, 서버의 모든 활동에서 유저의 개인정보가 잘 보호되고 있는지 확인하고 노출되지 않도록 최선을 다한다. 서버와의 통신에서 무단으로 회원정보를 읽거나 수정하는 것을 방지한다.

# 5. System Architecture

## 5.1 Objective

System Architecture에서는 목표 시스템의 Architecture에 대한 고수준에서의 개요를 보여준다. 또한 시스템 기능의 전체적 분포를 보여준다. 재사용 되는 구성요소들은 강조되어 있다.

## 5.2 Front end architecture

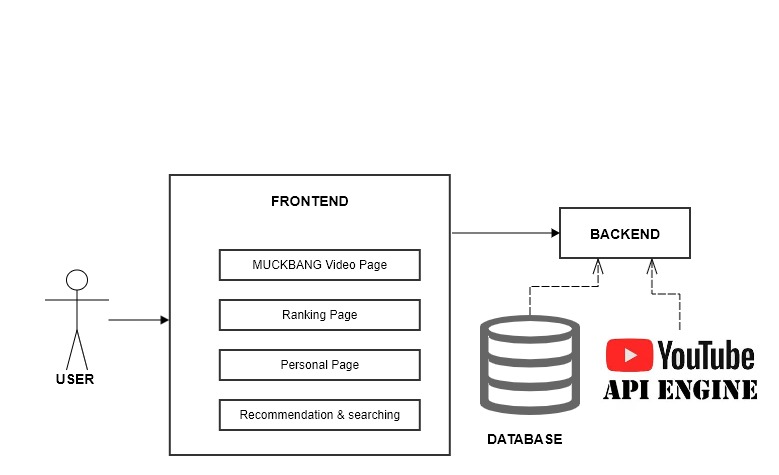


Diagram . Front end architecture

프론트 엔드 설계는 실제 사용자들이 마주치는 인터페이스에 대한 설계이자 client 즉 이번 프로젝트에 안드로이드 어플리케이션에 해당하는 부분이다. 어플리케이션은 크게 4개의 카테고리의 페이지를 가진다. 불필요한 동영상 없이 먹방 영상만 모여 있는 비디오 페이지, 먹방 영상의 순위를 알려주는 Ranking page, 추천과 검색을 해주는 Recommendation & searching page, 그리고 개개인이 구독하는 크리에이터와 시청한 동영상, 관심있는 카테고리를 알 수 있는 personal 페이지로 구성되었다.

## 5.3 User space architecture

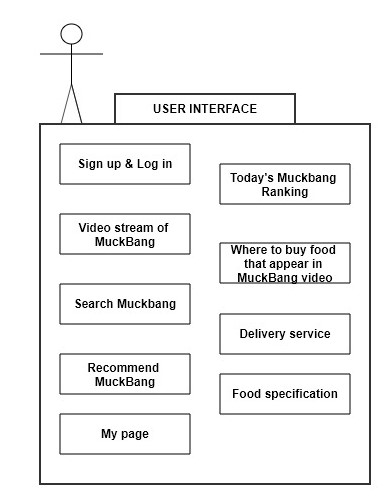


Diagram . User space architecture

5.2에서 user space 즉 어플리케이션의 항목을 다루었다면, 5.3 에서는 user interface 들이 구성하는 요소에 대하여 나열한다. 먼저 로그인과 회원가입을 하는 페이지가 있고, 메인으로 먹방 영상의 재생 목록이 뜬다. 영상을 시청함으로써 먹방 시청자들은 이색 음식이나 크리에이터들이 먹는 음식에 대한 정보, 배달 장소, 가격 등을 한눈에 볼 수 있게 된다. 또한 검색으로 영상을 찾을 경우에 해당 키워드와 검색된 영상들의 카테고리, 사용자의 기록 등으로 새로운 먹방 영상을 추천해준다.

## 5.4 Back end Architecture

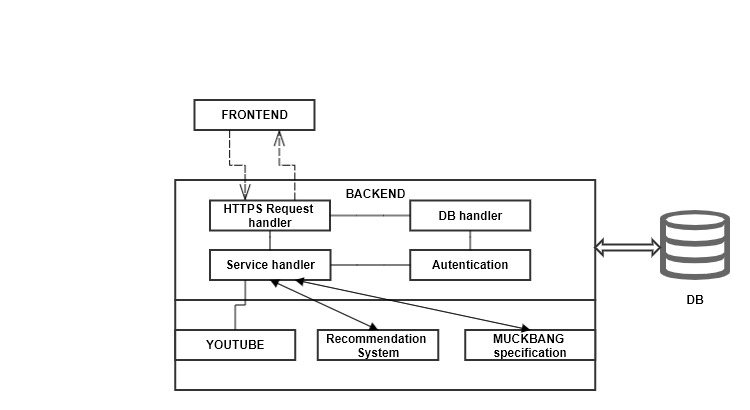


Diagram . Back end Architecture

back end architecture는 크게 4가지로 구성되어 있다. 먼저 HTTPS 형식으로request를 client(android application)으로 받는다. 그 뒤에 해당 request에 맞춰서 대응하는 응답을 해주는데 여기서 유투브 API를 이용하여 영상을 추천해주거나, 먹방에 등장하는 음식에 대하여 정보를 주거나 혹은 DB handler를 이용하여 기록된 사용자들의 정보들을 front end에 전달하게 된다. 결국 이 4가지는 서로 상호 작용하며 클라이언트에 필요한 정보들을 제공하고 DB에 저장하거나 꺼내 쓰게 된다.

## 5.5 Sub-system Architecture

### A. muckbang specification

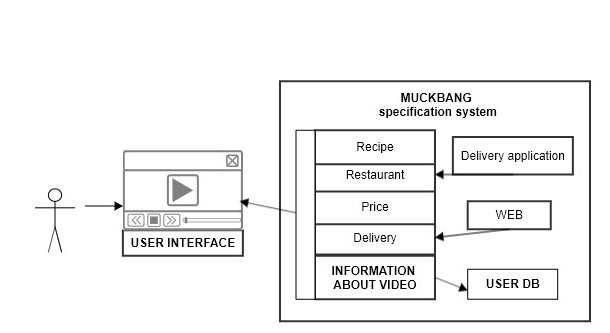


Diagram . 먹방 Specification overview

첫 번째 sub-system 은 MuckBang specification system이다. 가장 중요시 되는 요구 사항은 특정 영상에 대하여 그 영상에 태그, 제목, 카테고리 등을 이용하여 어떤 음식을 먹는지 파악하고 그 음식에 대한 정보를 WEB으로 가격 / 구입처 / 레시피 등을 알아 내어 제공해 주는 것이다. 또한 배달 가능 여부를 판단하여 배달이 가능 할 경우 기존의 배달 application 과 연결해주는 시스템이다.

### B. recommend system

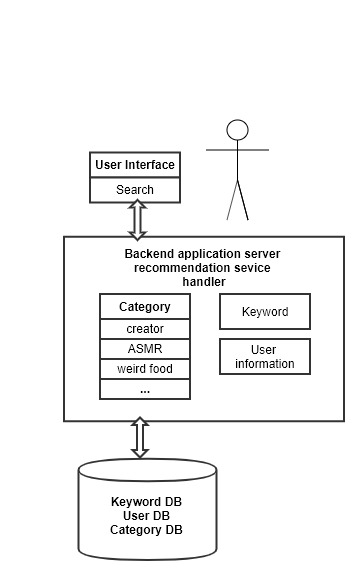


Diagram . Recommend System overview

두 번째 sub – system 은 Muckbang recommendation system이다. 먼저 사용자가 어떤 단어로 검색을 하게 되면 해당 검색어로 나오는 동영상들을 카테고리 별로 분류하여 DB에 저장한다. 또한 keyword 별로도 저장을 하여 사용자가 자주 검색하는 것들에 대해서 DB에 저장하도록 한다. 그 뒤에 DB에 저장된 사용자의 정보를 토대로 사용자에게 같은 키워드 , 비슷한, 연관된 카테고리의 먹방 동영상을 추천해 주도록 하는 시스템이다.

# 6. System Requirement Specification

System architecture에서 설명한 sub-system에서 2가지 분류, Functional requirements와 non-functional requirements로 나누어 구체적으로 설명하였다. 또한, 각 기능에 따른 시나리오와 함께 설명하여 각 기능의 흐름을 보다 간략적으로 확인할 수 있다.

## 6.1 Front end Functional Requirements

### A. Sign up

|  |  |  |
| --- | --- | --- |
| Name | Sign up | |
| Description | 유저의 정보를 입력 받고 아이디/이메일 중복 확인, 비밀번호 생성규칙 확인 후 서버에 저장한다. | |
| Inputs | ID | 사용자의 아이디 |
| Password | 사용자의 비밀번호 |
| E-mail | 사용자의 이메일 |
| Source (Input) | Sign up button | |
| Outputs | 아이디/이메일 중복 여부, 비밀번호 생성 규칙 여부에 의한 성공/실패 메시지 | |
| Destination (Output) | 성공/실패 여부 알려주는 toast message | |
| Action | 입력 받은 ID, Password와 e-mail을 이용하여 서버 데이터베이스를 바탕으로 중복여부, 규칙 준수 여부를 체크한다. 만약 기준을 충족하지 못했다면 실패 메시지를 반환하고, 충족했다면 성공 메시지의 반환과 함께 서버 데이터베이스에 저장한다. | |
| Requirements | ID, Password, E-mail | |
| Pre-condition | 회원가입 창 표시 | |
| Post-condition | 성공한다면, 메인 페이지로 이동 | |

Table . Sign up

### B. Sign in

|  |  |  |
| --- | --- | --- |
| Name | Sign in | |
| Description | ID, password를 입력 받고 유효하다면 서버에서 유저의 정보를 가져온다. | |
| Inputs | ID | 사용자의 아이디 |
| Password | 사용자의 비밀번호 |
| Source (Input) | Sign in button | |
| Outputs | 아이디, 비밀번호 의 유효성 여부에 의한 성공/실패 메시지 | |
| Destination (Output) | 성공/실패 여부 알려주는 toast message | |
| Action | 입력 받은 ID, Password 이용하여 서버 데이터베이스를 바탕으로 아이디와 패스워드의 유효성 여부를 체크한다. 만약 기준을 충족하지 못했다면 실패 메시지를 반환하고, 충족했다면 성공 메시지의 반환과 함께 유저 정보를 서버로부터 받아온다. | |
| Requirements | ID, Password | |
| Pre-condition | 로그인 창 표시 | |
| Post-condition | 성공한다면, 메인 페이지로 이동 | |

Table . Sign in

### C. Category

|  |  |  |
| --- | --- | --- |
| Name | Category | |
| Description | 서버에 저장 된 비디오의 분류를 바탕으로 유저에게 분류별로 비디오 전달 | |
| Inputs | 분류의 종류 | 한식, 중식 등 해당하는 분류에 따른 분류 ID |
| Source (Input) | 각 분류에 해당하는 button | |
| Outputs | 선택한 분류에 맞는 창 표시 | |
| Destination (Output) | 선택한 분류에 맞는 video listview | |
| Action | 카테고리 탭을 선택하면 존재하는 카테고리가 뜨고, 카테고리를 선택하면 해당 카테고리에 맞는 비디오들의 목록을 표시해주고, 비디오를 선택하면 해당 비디오의 정보가 뜬다. | |
| Requirements | 분류가 완료 된 비디오의 정보 | |

Table . Category

### D. Recommend video

|  |  |  |
| --- | --- | --- |
| Name | Recommend video | |
| Description | 유저의 정보를 바탕으로 추천 된 비디오의 목록을 보여준다 | |
| Inputs | 유저의 경험 | 유저가 설정한 관심사나 시청 기록 |
| Source (Input) | Recommend video tab click | |
| Outputs | 추천 비디오 목록 창 | |
| Destination (Output) | 추천 비디오의 listview | |
| Action | Recommend video tab을 누르면 서버에서 사용자 경험을 바탕으로 비디오를 분류하고 그 목록을 사용자 인터페이스에 표시한다. 표시 된 비디오를 클릭하면 비디오의 정보를 보여준다. | |
| Requirements | User의 시청 목록, 등록된 관심사 | |
| Pre-condition | 관심사의 설정이 필요함 | |

Table . Recommend Video

### E. Video ranking

|  |  |  |
| --- | --- | --- |
| Name | Video ranking | |
| Description | 시스템의 모든 비디오에 대한 시청횟수, 기간별 시청횟수를 토대로 비디오의 랭킹을 제공한다. | |
| Inputs | 비디오 ID | 비디오 식별 ID |
| 시청횟수 | 사용자의 비밀번호 |
| Source (Input) | Ranking tab | |
| Outputs | 기간(day, week, month, year)을 바탕으로 시청 횟수가 많은 순으로 정렬된 비디오 목록 | |
| Destination (Output) | 비디오 목록을 보여주는 listview | |
| Action | Ranking tab을 클릭하면 기간을 선택할 수 있는 부분이 있고 선택 된 기간을 바탕으로 시청 횟수가 많은 순으로 정렬 된 비디오 목록을 보여준다. 비디오를 클릭하면 비디오의 정보를 보여주는 창으로 이동한다. | |
| Requirements | Video id, video의 시청 횟수, | |

Table . Video ranking

### F. My page

|  |  |  |
| --- | --- | --- |
| Name | My page | |
| Description | 내 정보를 수정할 수 있고, 관심사를 편집하고 시청목록을 볼 수 있음 | |
| Inputs | ID | 사용자를 식별하기 위한 ID |
| Source (Input) | My page tab | |
| Outputs | 비밀번호 변경 버튼, 시청목록, 관심사 목록, 관심사 편집 버튼 | |
| Destination (Output) | My page에 Output의 요소들을 가지고 있는 layout | |
| Action | My page tab을 터치하면 시청목록, 관심사 목록, 아이디, 이메일 등의 사용자 정보를 표시하고 시청목록, 관심사목록, 비밀번호를 변경할 수 있는 버튼이 존재함. 각각의 버튼을 눌러 목록이나 정보를 변경할 수 있다. | |
| Requirements | ID, 시청목록, 관심사 목록 | |

Table . My page

### G. Information of video

|  |  |  |
| --- | --- | --- |
| Name | Information of video | |
| Description | 비디오의 정보를 표시한다. | |
| Inputs | Video ID | 사용자의 아이디 |
| Source (Input) | Video thumbnail click | |
| Outputs | 비디오의 전체적인 정보를 보여주는 창. 비디오 url을 바탕으로 하는 비디오 스트리밍, 비디오 제목, 비디오에 나오는 음식 정보 | |
| Destination (Output) | Output의 요소를 가지고 있는 layout | |
| Action | Video thumbnail의 클릭을 통해 해당 창으로 넘어오면, 상단에 video player가 있고 아래에는 video의 information이 있다. | |
| Requirements | Video ID, Video information | |

Table . Information of Video

### H. Review

|  |  |  |
| --- | --- | --- |
| Name | Review | |
| Description | 특정 비디오에 남겨진 리뷰를 읽거나 쓸 수 있음 | |
| Inputs | ID | 사용자의 아이디 |
| review | Review의 내용 |
| Source (Input) | Write review button, Read review button | |
| Outputs | Review 내용 반영, 등록된 review 목록 | |
| Destination (Output) | Review listview | |
| Action | 특정 비디오의 review button을 누르면 등록 된 review를 읽을 수 있고 write review 버튼으로 review를 입력할 수 있다. | |
| Requirements | User id, video id, 이미 등록된 review | |

Table . Review

### I. Connect to Delivery/Market system

|  |  |  |
| --- | --- | --- |
| Name | Connect to Delivery/Market system | |
| Description | 비디오 정보에서 등록된 음식에 대한 connection을 한다. | |
| Inputs | Food ID | 음식의 아이디 |
| Source (Input) | Delivery button, Market button | |
| Outputs | 배달 어플리케이션 연결, 인터넷 쇼핑몰 연결 | |
| Destination (Output) | 외부 어플리케이션(배달 어플리케이션, 인터넷 쇼핑몰) | |
| Action | 비디오 정보에서 음식 정보에 대한 delivery button과 market 버튼이 있고, 해당 버튼 클릭 시 각 어플리케이션이나 웹으로 연결 | |
| Requirements | Food id, Delivery or market connection | |

Table . Connect to delivery/market system

## 6.2 Backend Functional Requirement

### A. Register user for sign up

|  |  |  |
| --- | --- | --- |
| Name | Register user for sign up | |
| Description | Frontend의 sign up 에서 전송 받은 사용자의 정보들을 서버의 유에 저장한다. | |
| Inputs | ID | 사용자의 아이디 |
| Password | 사용자의 비밀번호 |
| E-mail | 사용자의 이메일 |
| Source (Input) | Sign up http request | |
| Outputs | 등록 여부 | |
| Destination (Output) | http response | |
| Action | 등록을 요청하는 http request가 도착하면 DB에 등록 후 정상적으로 완료된다면 완료되었다는 http response를 전송 | |

Table . Register user for sign up

### B. Confirm user for sign in

|  |  |  |
| --- | --- | --- |
| Name | Confirm user for sign in | |
| Description | ID/pw로 이루어진 로그인 요청에 대해 일치여부와 로그인 가능 여부를 체크하고 전송 | |
| Inputs | ID | 사용자의 아이디 |
| Password | 사용자의 비밀번호 |
| Source (Input) | Sign in http request | |
| Outputs | 일치 및 로그인 가능 여부 | |
| Destination (Output) | http response | |
| Action | 로그인을 요청하는 http request가 도착하면 DB와 비교 후 일치하고 로그인이 가능하다면 완료되었다는 http response를 전송 | |

Table . Confirm user for sign in

### C. Get video information from video streaming site

|  |  |  |
| --- | --- | --- |
| Name | Get video information from video streaming site | |
| Description | 스트리밍 사이트로부터 먹방, eating 등의 키워드로부터 검색 된 검색 결과를 바탕으로 음식 정보를 추출해낸다. | |
| Inputs | Keyword | 원하는 음식에 관한 keyword |
| Source (Input) | Server request | |
| Outputs | 비디오 DB | |
| Destination (Output) | Server DB | |
| Action | 스트리밍 사이트로부터 먹방, eating 등의 키워드로부터 검색 된 검색 결과를 바탕으로 비디오의 url과 제목, 썸네일 등을 가져오고, 영상의 정보를 크롤링 혹은 파싱 하여 음식 정보를 추출해낸다. | |

Table . Get video information from video streaming site

### D. Recommended video settings based on user experience.

|  |  |  |
| --- | --- | --- |
| Name | Recommended video settings based on user experience | |
| Description | 사용자의 사용 경험을 바탕으로 추천 영상 선정 | |
| Inputs | 사용자의 경험 | 사용자의 경험 및 관심사 |
| Source (Input) | http request | |
| Outputs | 추천 영상의 목록과 순위 | |
| Destination (Output) | http response | |
| Action | 사용자의 경험의 항목마다 개별적인 점수를 부여하여 개별적인 점수의 양상과 가장 일치하는 영상에 매칭시키고 정렬하여 response | |

Table . Get video information from video streaming site

## 6.2 Non-Functional Requirements

### A. Product Requirement

1) Accuracy

본 시스템의 핵심은 스트리밍 사이트에서 받아오는 영상 정보와 실제 사용자가 느끼는 정보의 일치가 첫 번째 우선사항이고 다음 우선사항은 사용자 경험 기반 영상 추천이 실제 사용자가 느끼는 것과 일치하는 것이 두 번째 목표이다. 이를 위해 정확한 크롤링, 파싱 기술이 필요하고, 외부적인 요인에 의한 정보의 불일치를 해결할 장치를 도입해야 한다. 또한 추천 영상의 정확도를 위해 사용자 경험을 판단할 수 있는 함수나 장치의 도입과 함께 정확도를 향상 할 수 있도록 지속적인 수정과 user test가 필요하다.

2) Performance

Application의 성능이나 Server의 응답속도를 고려해야한다. User가 application을 사용할 때 느린 반응속도를 보인다면 사용성에 있어서 매우 큰 약점이 될 것이다. 정해진 시나리오를 바탕으로 사용자의 다음 행동을 예측하고, 필요한 정보들은 background에서 미리 로딩하는 방법을 사용할 것이다. 또한 서버 성능을 위해 요청과 반응을 간결하게 하고, 필요 없는 정보는 배제하는 방법을 사용할 것이다.

3). Scalability/Encapsulation

즉각적인 오류 수정이나 신뢰성을 위해 기능의 각 부분을 encapsulate하고 이후에 있을 업데이트나 기능의 확장을 위해 확장성 있게 코드를 구성할 것이다.

4. Security

로그인이나 회원가입의 보안을 위해 카카오톡이나 페이스북 같은 로그인 API를 사용하는 방향으로 할 것이고, 개발 이후 평가를 통해 자체적으로 개발한 function을 사용할 수도 있다. 또한 리뷰의 익명성, 사용 경험의 보안을 위한 장치도 개발해야 한다.

### B. Organization Requirement

1. Operational

Application상에 데이터를 영구적으로 저장하는 것을 지양하고, 대부분의 데이터들은 server와의 통신을 통해 업데이트 된다. Prototype level에서는 개인 컴퓨터에 구현 된 서버를 사용하고, 실제 배포 단계에서는 외부 클라우드 서버를 사용 할 것이다.

2. Development

Waterfall model을 기반으로 이 모델의 핵심인 parallel development를 위해 server team, android app team을 나누고 전체적인 개발을 컨트롤 하는 team을 구성하여 운영 할 것이다. Android app team은 frontend와 backend로 나뉘어 개발한다. 코드의 공동 작업을 위해 git을 사용 할 것이다.

### C. External Requirement

1. Copyright

본 시스템은 외부 스트리밍 사이트에 있는 영상들을 받아오기 때문에 저작권에 대한 개념을 확립하여야 한다. 스트리밍 사이트의 운영방침을 확인하고 허용되는 범위 내에서 가능한 기능들을 구현 할 것이다.

## 6.3 Scenario

### A. Sign up and sign in scenario

1. initial assumption

사용자는 SNS 연계 등을 통해 계정을 만들고 등록된 id 로 로그인을 하게 된다.

2. Normal flow of events

사용자는 자신의 로그인 정보를 입력하고 로그인 버튼을 누른다. 그러면 어플리케이션에서 서버에 데이터베이스에 관한 https 요청을 보내고, DB에서 확인 후에 일치하면 응답을 주어 로그인에 성공하게 된다.

3. what go wrong

사용자가 가입했던 정보 대로 로그인 정보를 기입하지 않거나, DB에 저장하는 과정에서 오류가 생기면 DB에 없는 정보에 대하여 확인을 요청함으로 로그인에 실패하게 된다.

4.system state on completion

사용자는 가입과 로그인에 성공하여 user interface 에 접근하게 된다.

### B. Searching Videos Scenario

1. initial assumption

사용자는 검색하고자 하는 음식이나, 키워드, 먹방BJ 등의 이름으로 검색창에 검색을 하게 된다.

2. Normal flow of events

사용자가 특정 단어를 입력하면 마찬가지로 http request 형식으로 서버로 그 단어가 보내지고 youtube api를 통해 해당 특정 단어 검색의 결과에 대한 정보를 클라이언트로 보내주게 된다. 그렇게 되면 사용자는 해당 검색에 대한 검색 결과를 얻게 된다.

3. What can go wrong

사용자가 잘 못된 형식의 단어를 입력하거나 , 먹방에 관련된 단어 즉 , 어느 먹방 방송에도 해당하지 않는 키워드를 입력하게 되면 기존의 사용자의 관심있는 영상을 출력해주도록 한다.

4. System state on completion

사용자는 검색을 함으로서 해당 키워드에 대한 먹방들을 한눈에 모아서 볼 수 있게 된다.

### C. Video details Scenario

1. Initial Assumption

사용자는 특정 영상을 보기 위해 영상을 눌렀다. 사용자는 제목, 썸네일 등을 보고 들어왔지만 그 음식의 가격 구매처, 레시피 등은 알지 못하는 상태로 들어오게 된다.

2. Normal flow of events

사용자는 먹방을 보면서 음식에 대해 궁금증이 생긴다. 하지만 기존처럼 음식에 대한 궁금증을 다른 웹사이트나 핸드폰으로 검색할 필요 없이 동영상 밑에 나누어진 카테고리 별로 설명이 잘 되어있다. 예를 들어 사용자가 구매처가 궁금하여 누르면 구매할 수 있는 웹사이트로 바로 이동이 되고, 혹시 배달이 가능한 음식이었다면 배달 버튼을 누를 경우 사용자에게 특정 배달 어플리케이션이 설치되어 있을 경우 어플리케이션으로 연결을 해준다.

3.What can go wrong

어떤 음식이 새로 만들거나, 기존의 레시피가 없는 음식이라면 다른 검색을 통해서도 찾을 수 없음으로 설명란이 공란으로 비워 있을 수도 있다.

4. System state on completion

사용자가 취미로 본 먹방에 나오는 음식에 대하여 정보를 알 수 있고, 구매처 배달 등을 손쉽게 연결해줌으로 대리만족이 아닌 직접 체험할 수 있는 기회가 늘어난다.

### D. My page Scenario

1.Initial Assumption

사용자는 자기 정보에 대해 알아보기 위해 내 정보 탭을 누르게 된다.

2.Normal flow of events

먼저 내 정보 탭에는 사용자가 시청했던 동영상 들의 목록, 자주 검색한 키워드, 자주 검색한 키워드 들의 카테고리, 추천 영상 등이 되어 있다. 사용자는 내 정보 탭을 들어 감으로써 기존의 보던 먹방과 비슷한 카테고리의 새로운 영상을 보게 될 수 있게 되고, 자신이 재미있게 봤던 먹방이나 인상깊었던 영상을 손쉽게 다시보기 할 수 있다.

3.what can go wrong

말 그대로 사용자에 정보에 관한 것이기 때문에 사용자의 정보에 관한 것은 잘 못되기 힘들지만 사용자가 검색을 많이 하지 않았거나 영상을 많이 시청하지 않은 경우에 영상 추천의 결과물이 좋지 않을 수도 있다.

4. system state on completion

사용자는 내 정보 탭을 통해 손쉽게 자신이 봤던 영상들을 한눈에 볼 수 있고 새로운 영상을 추천 받게 된다.

# 7. System Models

System, System component, 그리고 System environment사이의 관계를 설명한다. 유저와 시스템의 상호작용에 관한 부분과 시스템의 각 컴포넌트와의 상호작용을 구체적으로 tabular description을 이용해 구체적으로 기술하고 모델링 하였다. 작성한 모델의 종류로는 use case diagram과 behavioral model등이 있다.

## 7.1 Context Models

### A. Context Model

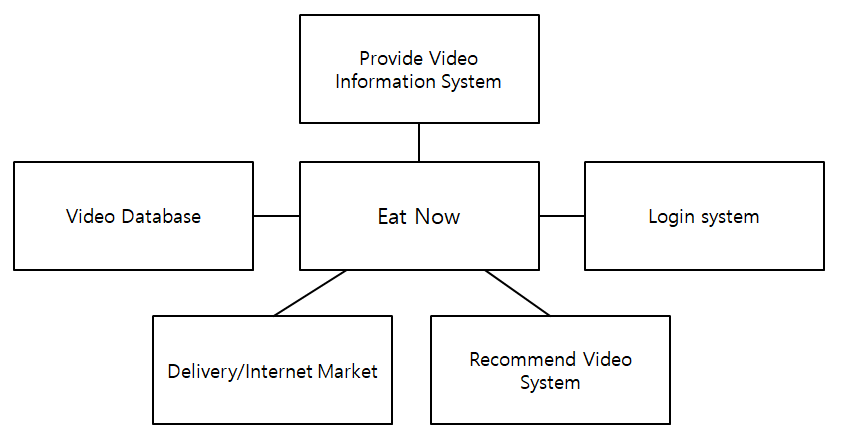


Diagram . System Context Model

### B. Process Diagram

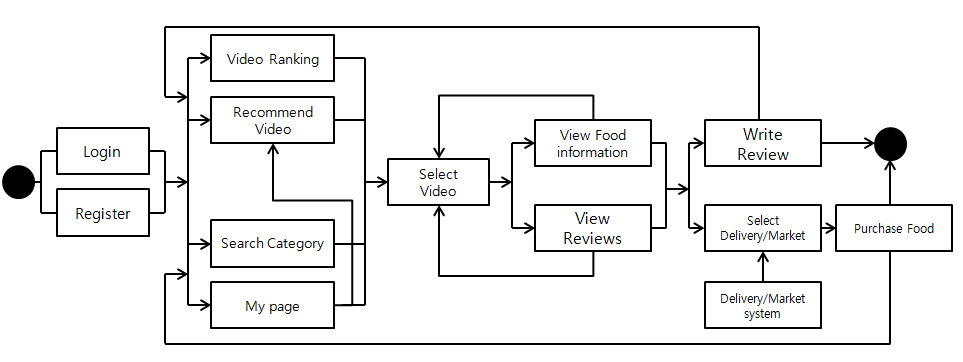


Diagram . System Process overview

## 7.2 Interaction Models

### A. Use case Diagram

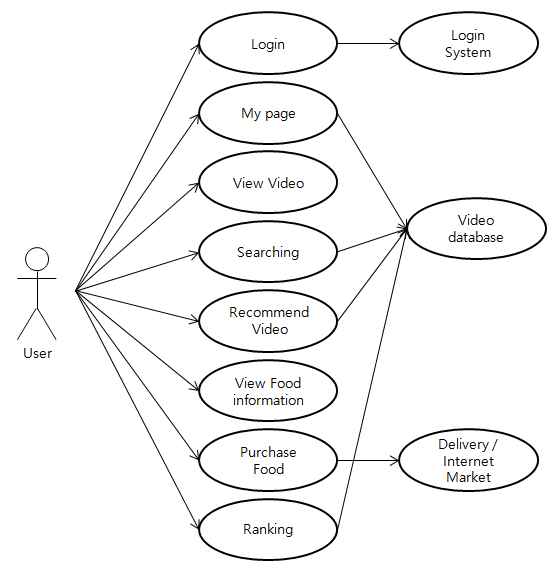


Diagram . Use case Diagram

### B. Tubular Description for each Use case

**1. Login**

|  |  |
| --- | --- |
| Use case | Login |
| Actor | User, Database(User) |
| Description | 접속한 사용자가 시스템에 등록되어 있는 유저 정보와 일치하는지 판단하는 과정이다. 등록이 이미 되어있다면 로그인을 하고 되어있지 않다면 회원가입을 권유한다. |
| Stimulus | 사용자가 ID와 PW를 입력한다. |
| Response | 입력한 정보가 데이터베이스에 존재하는 정보와 일치하는지 판단하고, 사용자 정보에 접속 권한을 부여한다. |

Table . Login tabular Description

**2. My page**

|  |  |
| --- | --- |
| Use case | My page |
| Actor | User, Database(User) |
| Description | 사용자가 이때까지 사용해왔던 DB를 확인할 수 있다. 동영상을 저장해놓거나, 봤던 동영상을 확인할 수 있다. |
| Stimulus | 사용자가 본 동영상 데이터를 저장해 놓는다. |
| Response | 사용자가 이용한 내역을 저장해 놓고 자신의 정보와 이전에 봤던 동영상, 저장해놓은 동영상 등이 보여진다. |

Table . My page tabular description

**3. View Video**

|  |  |
| --- | --- |
| Use case | View Video |
| Actor | User, Database(Video) |
| Description | 사용자가 데이터 베이스에 존재하는 동영상을 시청한다. |
| Stimulus | 검색, 추천, 저장 등 여러가지 경로를 통해서 얻은 동영상을 선택하여 시청한다. |
| Response | 동영상을 시청한 뒤에는 그 동영상이 사용자의 DB에 저장된다. |

Table . View video tabular description

**4. Searching**

|  |  |
| --- | --- |
| Use case | Searching |
| Actor | User, Database(Video) |
| Description | 사용자가 원하는 동영상을 DB에서 검색한다. |
| Stimulus | 원하는 동영상을 찾기 위해 DB에 키워드를 입력한다. |
| Response | 검색한 키워드에 맞게 동영상을 DB에서 찾은 후 출력한다. |

Table . Searching tabular description

**5. Recommend Video**

|  |  |
| --- | --- |
| Use case | Recommend Video |
| Actor | User, Database(Video), Database(User) |
| Description | 사용자가 본 동영상 내역을 토대로 사용자에게 여러가지 동영상을 추천한다. |
| Stimulus | 어떤 영상을 볼지 정하지 못할 때 영상추천 기능을 통해서 사용자가 관심있는 영상을 볼 수 있도록 한다. |
| Response | 사용자 DB에서 관심있는 내용을 가지고 온다. 그에 알맞은 동영상을 DB에서 찾아서 추천한다. |

Table . Recommend tabular description

**6. View Food Information**

|  |  |
| --- | --- |
| Use case | View Food Information |
| Actor | User, Database(Video), Database(Delivery) |
| Description | 사용자가 원하는 음식에 대한 정보를 제공한다. |
| Stimulus | 동영상을 시청하며 음식에 대한 정보가 함께 나와서 바로 확인 할 수 있도록 한다. |
| Response | 동영상의 DB를 토대로 음식에 대한 레시피, 배달 정보 등을 사용자에게 제공한다. |

Table . View Food Information tabular description

**7. Purchase Food**

|  |  |
| --- | --- |
| Use case | Purchase Food |
| Actor | User, Database(Video), Database(Delivery) |
| Description | View Food Information에서 확인한 음식 정보를 직접 먹어볼 수 있도록 음식을 구매할 수 있게 한다. |
| Stimulus | 동영상을 시청하며 원하는 음식을 바로 배달하거나, 음식 제료를 주문 할 수 있도록 한다. |
| Response | Delivery system과 Internet market system을 이용하여 동영상을 보면서, 바로 음식이나 재료를 구매할 수 있게 한다. |

Table . Purchase food tabular description

**8. Ranking**

|  |  |
| --- | --- |
| Use case | Ranking |
| Actor | User, Database(Video) |
| Description | DB에 저장되어 있는 영상중에서 가장 인기있는 영상 목록을 사용자에게 제공한다. |
| Stimulus | 사용자들에게 가장 인기 있는 동영상을 모아볼 수 있게 하여 인기 있는 음식이나 스트리머를 알 수 있도록 한다. |
| Response | 동영상 스트리밍 사이트에서 먹방 랭킹 DB를 가지고 와서 사용자들에게 제공한다. |

Table . Ranking tabular description

## 7.3 Behavioral Models

### A. Database Modeling

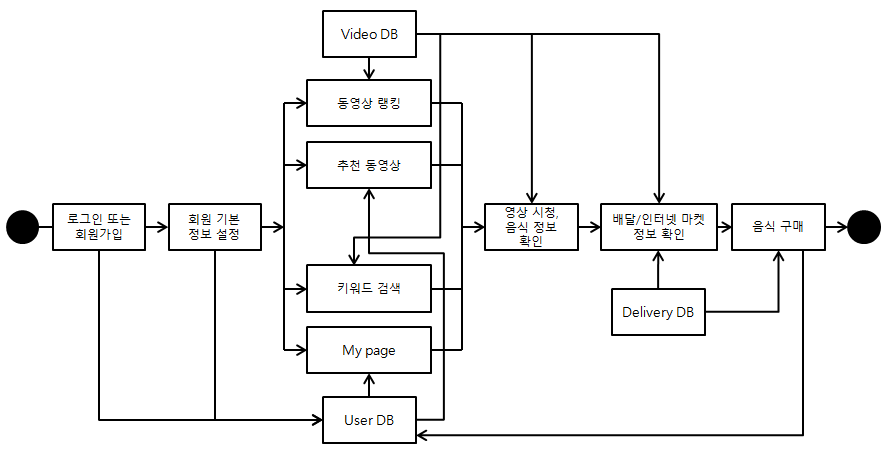


Diagram . Database Modeling

# 8. System Evolutions

System에 기저하여 있는 주요한 가정들에 대해 설명한다. 또한, System에 일어날 수 있는 예상되는 변화들에 대해 설명한다. 이 항목은 차후에 발생할 시스템의 설계 변경에 있을 Risk를 피할 수 있도록 도와줄 수 있기 때문에 시스템 설계를 담당하는 매니저에 도움이 될 것이다. 이를 통해, 변화에 잘 적응하는 시스템을 개발할 수 있다. 또한 차후에 요구될 수 있는 부분들도 작성하여 발전가능성을 모색한다.

## 8.1 Limitation and Assumption

1인미디어 시대가 전성기 시대를 맞았다. 그 중에서도 먹방은 이미 하나의 주류문화로 자리잡은 지 오래다. 유튜브를 예로 들면 이미 유튜버들의 상당수는 먹방 유튜버이며 먹방은 유튜브에서 가장 관심있게 시청하는 분야 1위를 달성했다. 먹방은 먹는다는 것과 방송의 합성어로서 먹방이라는 새로운 개념이 잡힌 것은 불과 몇 년이 채 안되었다. 먹방의 시작은 인터넷방송에서 시작되었다. 인터넷방송에서 인기를 끌며 유명세를 탄 것이 점점 화제가 되었고 뉴스에서도 이러한 현상을 많이 다루기 시작했다. 점점 많은 사람들이 관심을 가지며 사회적으로도 많은 영향을 끼쳤고 결국 TV프로그램에서도 많이 활용하는 컨텐츠가 되었다.

현재 먹방의 인기는 식을 줄 모르고 있지만 반대로 먹방에 지겨움을 토로하거나 뻔한 소재를 비판하는 사람들도 많고 먹방에서 다른 컨텐츠를 찾는 시청자들도 있다. 먹방과 결합했지만 먹방이 주 소재가 아닌 다른 포맷의 방송도 많이 나오고 있는 실정이다. 즉 현재 먹방은 꾸준히 사랑받고 있으나 미래에도 먹방의 인기가 꾸준하다고 단언할 수 없다.

Eat Now는 먹방 및 쿡방의 인기와 같이하는 시스템이다. 만약 Eat Now가 유저들에게 도움을 많이 주고 편리하여 사랑받더라도 만약 먹방의 인기가 식는다면 Eat Now의 활용가치도 떨어질 것이다. Eat Now는 어디까지나 먹방의 시청과 활용을 도와주는 시스템이기 때문에 먹방의 인기가 시스템에게 큰 영향을 끼칠 것이다.

## 8.2 Evolutions of Requirement

### A. 커뮤니티

이 요구사항은 개별 동영상에 대한 상세한 정보만 주었던 시스템의 특성에서 벗어나 커뮤니티를 형성해 유저들 간 소통이 가능하게 하고 먹방과 관련된 정보도 공유할 수 있게 한다. 커뮤니티를 형성한다는 것은 장단점이 있다. 일단 목적을 벗어날 가능성이 있고 커뮤니티를 활성화시키는 데에 유지비용이 많이 따른다. 따라서 유저의 수요를 더 조사하고 결정해야 한다. 구체적으로 기능은 다음과 같은 것들이 있을 수 있다.

1) 댓글

기존 스트리밍 사이트의 댓글을 읽을 수 있게, 또는 새로운 댓글 시스템을 만들어 사용할 수 있게 한다. 댓글이 가능한 것은 장단점이 있지만 기본적으로 유저들의 소통을 가능하게 하고 영상 시청에 재미도 부여할 것이다. 또한 댓글 기능이 활성화된다면 크리에이터의 참여도 요구할 수 있고 참고점이 될 수 있다. 하지만 이 기능은 처음 버전에 없어도 충분하다고 판단되어 evolution에 넣었다.

2) 읽을거리

관리자가 직접 먹방에 관한 다양한 읽을거리를 제공하고 정보도 제공할 수 있다. 예를 들어 이번 달 이색적인 먹방 주제들을 골라 정리해줄 수도 있고 좋은 레시피도 제공해 줄 수 있다. 읽을거리는 유저의 흥미를 돋울 수 있고 시스템 활성화에 도움이 되지만 실제적으로 유지보수에 한계가 있고 꼭 필요한 기능이 아니다.

3) 다양한 주제의 게시판

다양한 주제, 다양한 크리에이터에 대한 게시판을 만든다. 예를 들어 유명 크리에이터의 팬 사이트나 특정한 음식에 대한 게시판을 만들어 서로 주제가 같은 유저끼리 사진이나 동영상도 올리고 정보도 공유할 수 있게 한다. 게시판은 유저의 참여를 유도하고 먹방 문화 활성화에도 도움이 되지만 시스템의 목적에서 벗어날 가능성이 있고 유지비용도 많이 든다. 또한 먹방이라는 단일 주제로 큰 활성화를 할 수 있을지는 미지수다.

### B. AI를 이용한 자동화

기본적으로 추천 기능과 같은 경우는 AI를 통해 구현하지만 먹방에 대한 영상을 가져오거나 개개의 영상에 대한 정보는 완전히 자동화하는 것이 현실적으로 힘들다. 비디오 스트리밍 사이트는 먹방 이외에 다양한 분야의 영상을 가지고 있는데 어떤 동영상이 먹방 인지 구분할 수 있고 영상이 업로드 될 때 자동으로 Eat Now에 등록하는 것은 쉽지 않다 따라서 동영상을 업로드 하기 전 사람이 직접 검토하고 업로드 하도록 하고 AI의 기능을 나중에 도입할 수 있다.

## 8.3 Evolutions of Environment

### A. Scope of Service

Eat Now는 먹방과 음식에 대한 유용한 정보를 제공해주는 시스템으로서 먹방이나쿡방과 같은 분야 이외에는 철저히 내용을 배제하나 비슷한 소재를 다룬 컨텐츠는 제공할 수 있다. 실제로 100% 먹방만 방송하는 크리에이터는 거의 없으며 일상이나 예능 같은 소재와 결합한다. 따라서 어느 정도까지 범위를 설정해 두어야 할지를 잘 결정하고 확장가능성을 검토해 보아야 한다. 또한 Eat Now의 특성을 먹방뿐만 아니라 다른 영역에도 확장할 수 있다. 예를 들어 여행과 관련된 동영상이 있다면 영상에서 갔던 투어스팟들을 정리해 주고 레스토랑의 위치나 정보들을 제공할 수 있다. 많은 부분에서 비슷한 기능을 갖추고 있기 때문에 적용하기가 쉽다. 따라서 유저의 반응이 성공적이라면 이러한 확장가능성을 고려해 보는 것도 나쁘지 않다.

# 9. Appendices

개발되는 시스템에 대한 더 자세하고 구체적인 정보들을 제공한다. 앞서 항목에 다루지 않은 Hardware, database, 시스템 사용에 적합한 혹은 최소의 설정 등을 구체적으로 설명한다.

## 9.1 Hardware Requirements

## 9.2 Database Requirements

## 9.2 User-System Requirements

## 9.3 Development Process

# 10. Index

본 문서에 포함된 그림, 테이블, 다이어그램, 관련자료 등 인덱스가 포함된다.

## 10.1 Figure Index

[Figure 1. 유튜브 소비현황 설문조사 결과 9](#_Toc23114828)

[Figure 2. 먹방 및 쿡방의 시청이유 10](#_Toc23114829)

[Figure 3. 유튜브에서 뜬 이색음식과 온라인 쇼핑몰 매출의 상관관계 11](#_Toc23114830)

[Figure 4. Eat Now logo 12](#_Toc23114831)

## 10.2 Table Index

[Table 1. Sign up 25](#_Toc23114832)

[Table 2. Sign in 26](#_Toc23114833)

[Table 3. Category 27](#_Toc23114834)

[Table 4. Recommend Video 28](#_Toc23114835)

[Table 5. Video ranking 28](#_Toc23114836)

[Table 6. My page 29](#_Toc23114837)

[Table 7. Information of Video 30](#_Toc23114838)

[Table 8. Review 30](#_Toc23114839)

[Table 9. Connect to delivery/market system 31](#_Toc23114840)

[Table 10. Register user for sign up 32](#_Toc23114841)

[Table 11. Confirm user for sign in 32](#_Toc23114842)

[Table 12. Get video information from video streaming site 33](#_Toc23114843)

[Table 13. Get video information from video streaming site 33](#_Toc23114844)

[Table 14. Login tabular Description 42](#_Toc23114845)

[Table 15. My page tabular description 42](#_Toc23114846)

[Table 16. View video tabular description 43](#_Toc23114847)

[Table 17. Searching tabular description 43](#_Toc23114848)

[Table 18. Recommend tabular description 44](#_Toc23114849)

[Table 19. View Food Information tabular description 44](#_Toc23114850)

[Table 20. Purchase food tabular description 45](#_Toc23114851)

[Table 21. Ranking tabular description 45](#_Toc23114852)

## 10.3 Diagram Index

[Diagram 1. Front end architecture 20](#_Toc23114853)

[Diagram 2. User space architecture 21](#_Toc23114854)

[Diagram 3. Back end Architecture 22](#_Toc23114855)

[Diagram 4. 먹방 Specification overview 23](#_Toc23114856)

[Diagram 5. Recommend System overview 24](#_Toc23114857)

[Diagram 6. System Context Model 40](#_Toc23114858)

[Diagram 7. System Process overview 40](#_Toc23114859)

[Diagram 8. Use case Diagram 41](#_Toc23114860)

[Diagram 9. Database Modeling 46](#_Toc23114861)

## 10.4 Reference

<https://www.mediajob.co.kr/support/support.htm?ctg_idx=24&cmd=view&brd_idx=31136>

<https://www.trendmonitor.co.kr/tmweb/trend/allTrend/detail.do?bIdx=1329&code=0202&trendType=CKOREA>

<http://www.sisajournal-e.com/news/articleView.html?idxno=201488>