SONGSTERR

-Requirements Specification-

2016310936 우승민

Contents

1. Preface

1.1 Objective

본 Preface에서는 이 문서의 예상 독자를 설정하며, 문서의 전반적인 내용의 세부사항들에 대한 간략한 개요를 제공한다.

1.2 Readership

독자층에 따라 서비스를 이용할 유저들은 User Requirements, 서비스를 개발할 개발자들은 System Requirements 두 가지로 문서를 나누어 제시한다.

A. User Requirements

본 시스템의 사용자를 독자로 하여, 독자층이 해당 문서를 쉽게 이해하는 것을 목표로 기술한다. 따라서 전문용어 사용을 자제하고 자연어로 작성을 기본으로 하며, 또한 도표, 이미지 등의 시각자료를 활용한다.

B. System Requirements

본 시스템을 개발할 개발자들을 독자로 하여, 실재 개발 과정에서 참고 사항으로 쓰이는 것을 목표로 기술한다. 따라서 시스템의 기능과 제약사항 등을 최대한 상세하게 기술하며, 개발자들에게 익숙한 용어들을 사용하여 기술한다.

1.3 Documents Contents

A. Introduction

본 시스템을 개발하게 된 배경과 목표에 대하여 설명한다. 또한 시스템의 대략적인 구조와 기능에 대해 서술하며, 해당 시스템이 개발되고 그에 따라 기대되어지는 기대효과를 설명한다.

B. Glossary

본 문서에서 사용되어질 기술용어들에 대한 정의들을 제시한다. 배경지식이 부족한 독자들이 본 문서를 읽는데 불편함이 없도록 하는 것을 목표로 하며, 가능한 상세하게 설명하는 것을 목표로 한다.

C. User Requirement Definition

본 시스템의 기능적, 비기능적 요소들을 실제 사용자의 입장에서 설명한다. 독자들이 이해하기 쉽게 전문용어 사용을 자제하고 자연어를 사용하고 다양한 시각자료를 이용하여 서술한다.

D. System Architecture

본 시스템에 대한 전체적인 구조를 서술하고 각 sub-system이 어떠한 역할을 가지고 있는지 설명한다.

E. System Requirements Specification

본 시스템의 기능적, 비기능적 요소들을 개발자의 입장에서 설명한다. 개발단계에서 사용되어질 문서이므로 따라서 User Requirement Definition의 내용을 바탕으로 더 상세하고 구체적으로 설명한다.

F. System Models

시스템의 각 Components들의 관계 및 시스템의 동작 환경과의 관계 들을 다이어그램과

같은 시각적인 모델로 나타낸다.

G. System Evolution

추후에 고객들의 requirements과 여러 환경 변화로 인해 변경될 수 있는 기능들을 제시한다.

H. Appendices

본문에서 생략된 참고자료 등을 기술한다.

I. Index

본 문서에서 사용된 그림, 표, 다이어그램 등의 색인을 기술한다.

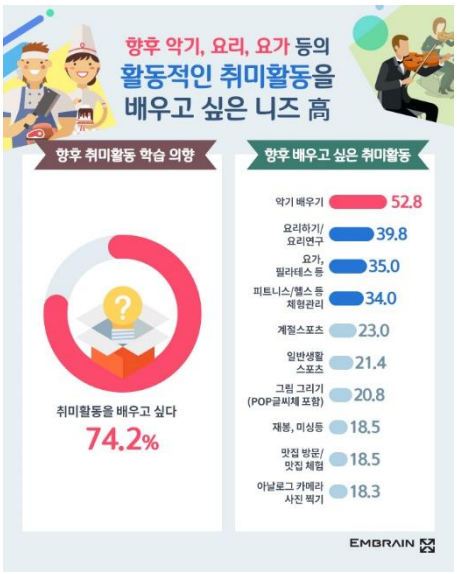
J. Reference

본 문서에서 인용된 자료의 출처를 기록한다.

2. Introduction

해당 챕터에서는 본 시스템이 개발하게 된 배경(Needs)과, 전반적인 시스템의 기능에 대해 Overview한다. 그리고 해당 시스템이 가져오는 기대효과를 제시한다.

2.1 Needs



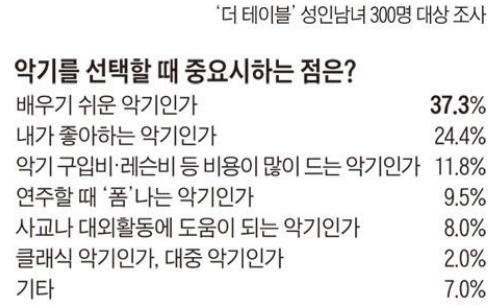
많은 현대인들은 새로운 취미생활을 배우는 것을 꿈꾸고 있다. 향후 배우고 싶은 취미생활을 통계하였을 때 악기 배우기는 그 중 1위를 차지했다. 이러한 마음을 가지고 있음에도 그들이 시작하지 않는 이유가 무엇일까?

Figure 1. 현대인이 원하는 취미생활 순위

막상 시작하고 싶어도 상황이 여의치 않기 때문이다. 아래 통계자료는 악기연주를 배우지 않은 이유에 대한 것이다. 1위는 당연하게도 흥미가 없기 때문이라 무시하고 흥미가 있어도 배우지 않은 사람들인 2위부터 보면, 비용문제와 배울 환경이 부족하다는 것이 가장 큰 이유이다.

Figure 2. 악기 연주를 배우지 않는 이유

이러한 상황에서 사람들이 가장 시작하기 좋은 악기는 단연코 기타라고 할 수 있다. 이유로는 첫째, 비용이 타악기에 비해 많이 낮다. 입문용 기타는 10만원 내외에 책정되어있고, 심지어 오래 사용이 가능하기 때문에 중고로 구하면 5만원에도 충분히 구할 수 있다. 둘째, 배우기 쉽다. 기타를 처음 시작할 때는 자주쓰이는 기본코드만 배워도 충분히 노래 하나를 완주할 수 있어서 금방 성과를 볼 수 있다. 이러한 성과는 더 배우고 싶다는 생각을 이끌어 내어 충분히 취미생활로 만들 수 있을 것이다.


'더 테이블' 성인남녀 300명 대상 조사

악기를 선택할 때 중요시하는 점은?

배우기 쉬운 악기인가	**37.3%**
내가 좋아하는 악기인가	24.4%
악기 구입비·레슨비 등 비용이 많이 드는 악기인가	11.8%
연주할 때 '폼'나는 악기인가	9.5%
사교나 대외활동에 도움이 되는 악기인가	8.0%
클래식 악기인가, 대중 악기인가	2.0%
기타	7.0%

Figure 3. 악기선택 고려사항 순위

그러나 무작정 기타를 시작하기에는 아직 부족한 점이 있다. 바로 배울 수 있는 환경이다. 주변에 배우고 연습할 수 있는 방법이 아무것도 없다면, 시작부터 난관에 부딪쳐 금방 흥미를 잃게 될 수있다. 본 시스템은 그러한 점을 해결하기 위해 만들어졌다.

2.2 System Overview

'Songsterr'은 다양한 장르의 음악을 기타 뿐만이 아닌 여러 악기의 악보를 가지고 있고 악보를 노래와 함께 재생하여 유저들이 음악에 맞추어 악기 연습을 할 수 있는 시스템이다. 기본적으로는 유저들이 원하는 노래의 악보를 검색할 수 있고, 특별히 찾는 음악이 없으면 이용한 유저 수에 따라 순위를 매겨논 음악을 선택할 수 있다. 이 중 원하는 악보들을 자신만의 favorite 에 등록하여 나중에 찾지 않아도 바로 볼 수 있다. 이 기능만 보면 굳이 본 시스템을 사용하지 않아도 될 것 같지만, 이것은 가장 기본적인 기능만을 설명한 것이다

'Songsterr'의 가장 좋은 점은 저가 연습하기 무척 좋다는 것이다. 단순히 노래의 악보만을 보여주는 것이 아니라 노래와 함께 실행시키면 노래에 맞추어 악보가 진행되어 유저들이 들으면서 연주, 연습하기 무척 좋다. 또한 자신의 노래 숙련도에 따라 어려우면 속도를 낮추거나 구간 반복을 하는 기능이 있고, 기타를 튜닝할 수 없는 상황이라면 반대로 노래의 음정 을 조절할 수 있다.

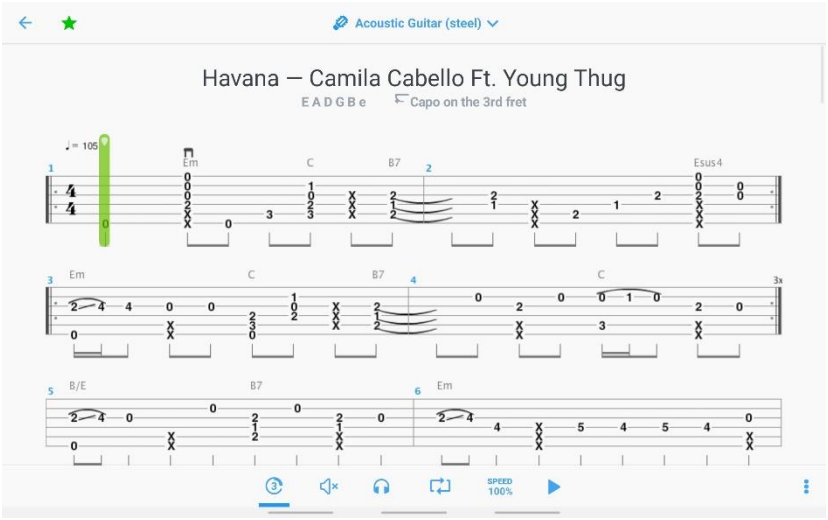


Figure 4. Songsterr 내 악보

2.3 Expected Benefits

A. 유저들이 취미생활을 가짐

'Songsterr'을 사용하는 유저들이 기타를 포함한 'Sonsgterr' 내에서 제공하는 악기들을 쉽게 배울 수 있어 하나의 취미생활을 가질 수 있다. 취미생활을 가지게 되면 유저들은 자신의 스트레스 관리 및 삶의 활력에 도움이 될 것이다.

B. 비용 절감

악기를 배우기 위해 학원을 다니거나, 원하는 악보를 일일이 구하면 꽤 많은 비용이 든다. 하지만 'Songsterr'를 사용하면 굳이 학원을 다닐 필요가 없게 되고, 악보도 이 시스템 내의 악보는 어느 것이든 추가비용 없이 사용이 가능하므로 많은 비용을 절감할 수 있다.

3. Glossary

이 문서에서 사용되는 기술적 용어들을 정의하여 독자들의 이해를 돕는 목적이다.

Term	Description
cache	사용자가 사용한 기록을 저장하여 같은 내용을 다시 볼 때 빠르게 볼 수 있게 해 줌
Data base	사용자들의 개인정보(아이디, 비밀번호 등)을 저장하는 거대한 저장소
FIFO	시간에 따라 저장된 목록을 수정하는 방식으로 가장 먼저 입력된 값을 먼저 삭제한다.

Table 1. Glossary

4. User Requirements Definition

4.1 Functional Requirements

A. Sign in

이 기능은 사용자가 본 시스템을 사용하기 위해서 반드시 처음으로 수행해야할 기능적 요구사항이다. 시스템의 데이터 베이스에 사용자 별로 unique한 아이디를 기준으로 개인정보를 저장한다. 저장된 개인정보는 사용자가 시스템을 접속할 때의 인증수단과 시스템 사용할 때의 설정 등을 저장한다. 따라서 사용자는 시스템 이용 시 처음에 회원가입(sign up)을 통해 각자의 unique한 아이디와 비밀번호를 등록해야 한다. 등록이 되면 만들어진 아이디와 비밀번호를 통해 시스템에 로그인(Sign in)하여 서비스를 이용할 수 있다.

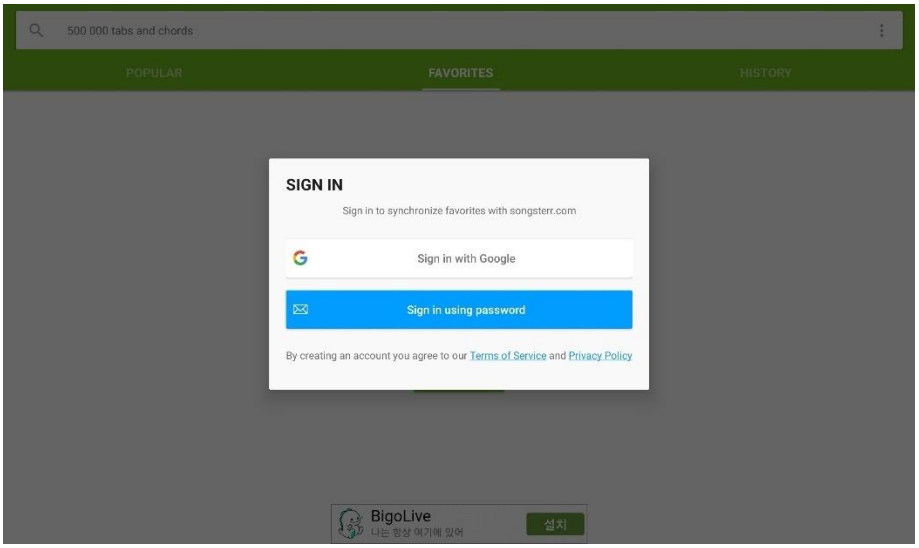


Figure 5. Sign in

B. Search

Search 는 사용자가 연주하기 원하는 노래의 제목이나 음악가의 이름이 있을 때 문자열을 검색하여 일치하는 악보들을 사용자에게 보여주는 기능이다. 자신이 원하는 것만이 아닌 해당 문자열이 포함된 노래와 음악가의 목록을 모두 나열한다.

Figure 6. Search

C. Favorite / History / Popular

사용자가 찾아본 악보 중 마음에 드는 악보에 표시를 하여 따로 목록을 만드는 기능이다. Favorite 에 저장된 목록은 cache 에 저장되어 오프라인에서도 검색하지 않고 볼 수 있다.

History는 Favorite과 다르게 사용자가 지정한 악보들이 아닌 최근에 본 악보를 cache에 저장한다. FIFO(선입선출) 방식을 사용하여 가장 본 지 오래된 악보가 목록에서 지워진다.

Popular는 사용자들에게 수집한 정보를 바탕으로 가장 많이 사용된 악보목록을 보여주는 기능이다. 상위 500위까지 볼 수 있다.

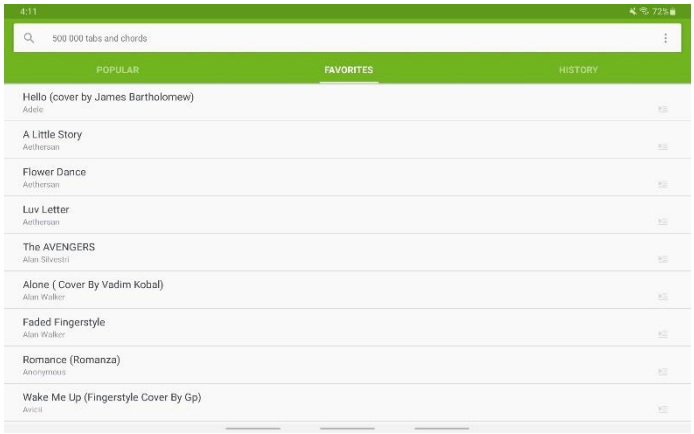


Figure 7. Favorite

D. Play 및 추가기능 (Change playback speed / Turn loop mode / Retune / latency compensation)

이 기능들은 본 시스템의 주요 기능이라고 할 수 있다. Play 는 해당 악보의 노래를 재생하여 박자에 맞추어 악보를 진행해 사용자가 듣고 보고 따라하면서 연주할 수 있게 해주는 기능이다. Change playback speed 는 사용자가 노래를 자신의 원하는 속도에 맞추어 속도를 조절하게 해주는 기능이다. Turn loop mode 는 사용자가 원하는 구간을 반복재생 해주는 기능이다. Retune 은 해당 곡의 음정을 조절해주는 기능이다. Latency compensation 은 노래와 악보의 속도가 조금 차이가 날 때 사용자가 조절할 수 있게 하는 기능이다.

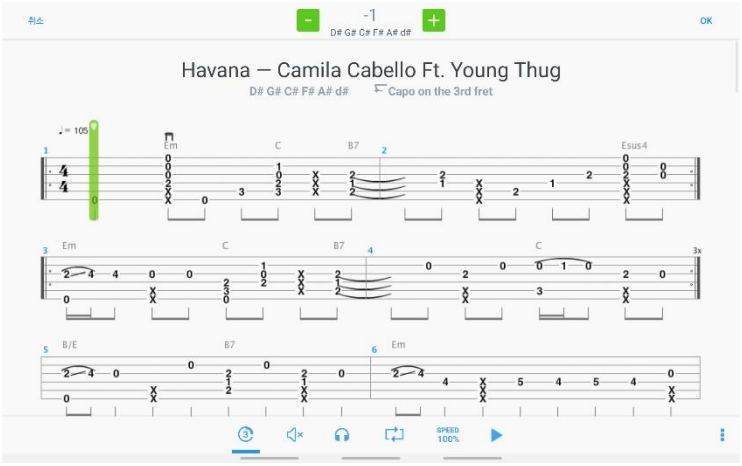


Figure 8. Return이 적용된 악보

4.2. Non-functional Requirements

A. Product requirements

A1. Usability requirements

본 시스템의 주 목적은 사람들이 기타 연주를 배우기에 쉬운 환경을 제공하는 것이다. 따라서 사용자가 시스템을 쉽게 사용하고 보고 따라할 수 있어야한다.

A2. Performance requirement

본 시스템을 사용하는 이유 중 하나가 원하는 악보를 찾는 소요시간과 구입 비용을 절감하기 위해서이다. 따라서 많은 악보를 저장하여 사용자가 원하는 악보를 대부분 소지해야 performance 측면에 문제가 없을 것이다.

A3. Security

본 시스템 자체에는 특별히 중요한 개인정보는 없다고 할 수 있다. 하지만 본 시스템은 구글 계정을 연동하여 사용할 수 있기 때문에 보안에 주의하여 사용자의 개인정보가 외부로 유출되는 경우가 없어야 한다.

B. Organization requirements

B1. Environmental requirement

본 서비스는 모바일 디바이스에서 제공되며, 모든 운영체제에서 문제없이 작동되어야 한다. 또한 데이터 및 와이파이 등 인터넷에 접속이 되지 않을 때도 기존에 cache 에 저장된 악보로 이용할 수 있어야 한다.

B2. delivery requirement

본 서비스는 모바일 디바이스에서 제공되어지므로, 사용자들이 쉽게 찾아 다운받을 수 있게 모바일 마켓에서 배포되어야 한다. 안드로이드의 경우 google play 에서 배포되고 애플의 경우 app store 에서 배포된다.

C. External requirements

C1. Regulatory requirement

사용자의 개인정보를 사용하기 때문에 사용자에게 동의를 구하여야 한다. 또한 제공자와 사용자 외에 해당 정보에 접근할 수 없어야 한다.

C2. Ethical requirement

악보에 대한 저작권이 문제가 될 수 있다. 따라서 악보를 제작한 작곡가 혹은 편곡자와 악보 사용에 대한 동의 및 계약을 잘 수행하여야 한다.

C3. Safety/security requirement

개인정보보호법에 따라 사용자의 개인정보를 외부/내부 적으로 안전하게 저장하고, 보호해야한다.

5. System Architecture

이번 문서에서는 시스템의 architecture에 대해 서술한다. 시스템의 전체적인 구조 및 sub-system의 구성 및 관계에 대해 설명한다.

5.1 Fronted Architecture

사용자가 직접 보고 사용하는 영역으로 Sign in, Search, Favorite/History/Popular, Play(추가기능 포함)로 크게 4개의 시스템으로 전체적인 시스템이 구성된다. User가 Fronted에 요청한 내용을 Backend(Server)로 넘겨 필요한 data를 Database에서 넘겨 받습니다.

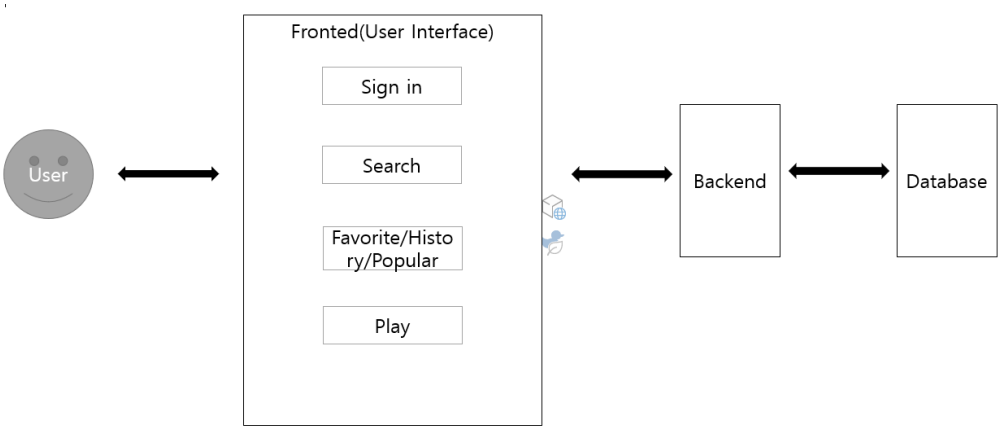


Diagram 1. Fronted Architecture

5.2 Backend

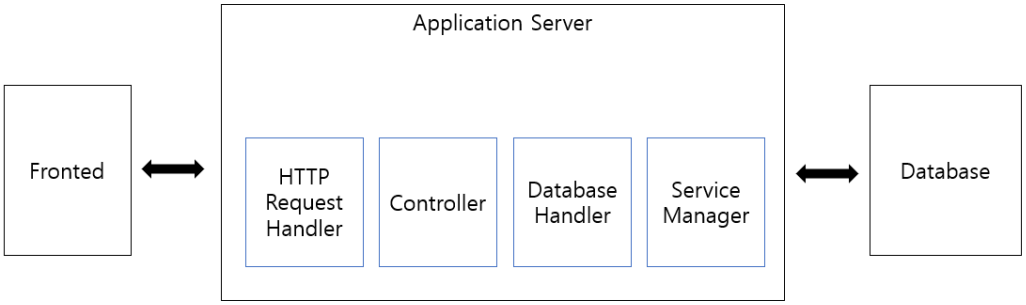


Diagram 2. Backend Architecture

5.3 Search System

사용자가 원하는 곡의 제목이나 음악가의 이름을 입력하여 악보를 찾는 system이다. User가 server에 keyword를 입력하면 server에서는 Database에 해당 keyword가 포함된 악보가 있는지 확인하고 있으면 user에게 목록을 보여주고 없으면 빈 화면을 보여준다.

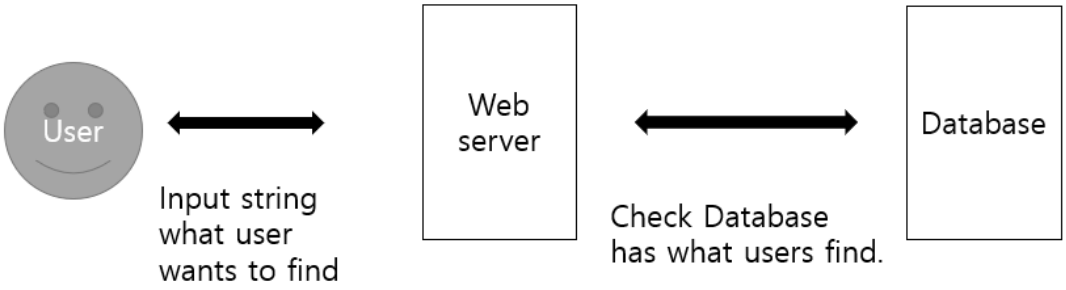


Diagram 3. Search System architecture

5.4 Favorite / History System

사용자가 search system을 통해 찾은 악보의 데이터를 database에 저장하여 따로 목록을 만드는 system이다. Favorite의 경우는 사용자가 표시를 한 악보를 따로 모아두고 History는 최근 본 목록을 FIFO 방식으로 저장한다. 해당 system은 모바일 디바이스의 cache에도 악보를 저장하여 오프라인 상에서도 볼 수 있게 만들었다.

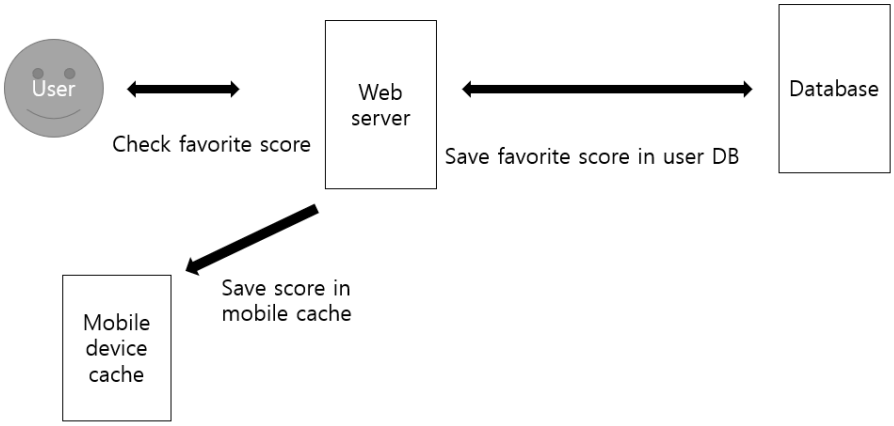


Diagram 4. Favorite System architecture

5.5 Popular System

Search system을 통해서 user들이 본 악보들의 정보를 database에서 가장 많이 본 악보를 순위별로 보여주는 System이다. 특별히 찾는 곡이 없는 user들이 인기있는 음악이 무엇인지 보고 연주해 볼 수 있게 해준다.

6. System Requirements Specification

6.1 Functional Requirements

A. Sign in

이름	**Login Function**
기능	본 시스템을 사용자만의 환경으로 사용하게 해준다.
입력	Google 계정 혹은 시스템 내의 아이디 및 비밀번호
조건	본 시스템 내의 아이디 비밀번호를 새로 만들거나 구글 계정 access에 동의 해야함
출력	아이디 및 비밀번호가 일치하면 본 시스템의 메인 화면에 들어감
처리	사용자의 구글 계정이 시스템 DB에 등록되어 있지 않으면 동의를 구한 후 DB에 저장함

Table 2. Sign in

B. Search

이름	**Search function**
기능	사용자의 검색어와 일치하는 악보를 보여준다.
입력	Keyword(노래 제목 혹은 artist의 이름)
조건	-
출력	입력 keyword가 포함된 악보들을 나열함
처리	DB에 keyword와 일치하는 악보들이 있으면 모두 보여주지만 없으면 빈 화면을 보여준다.

Table 3. Search

C. Favorite

이름	**Favorite function**
기능	사용자가 좋아하는 악보들의 list 생성
입력	각 악보 내 '★' 체크
조건	-
출력	Favorite page에 '★'가 체크된 악보들 나열
처리	사용자의 DB에 등록된 '★' 모양의 악보들을 favorite page에 나열함 없을 경우에는 빈 화면을 보여줌

Table 4. Favorite

D. History

이름	**History function**
기능	사용자가 최근 열람 한 악보들의 list 생성
입력	기존에 사용자가 악보를 열람한 DB기록
조건	-
출력	최근 본 악보들을 나열 최근일수록 위에 표시
처리	사용자의 DB 기록을 통해 FIFO방식으로 최근 본 악보들을 나열한다. 20개 이상 넘어갈 경우 오래된 악보를 list에서 지운다.

Table 5. History

E. Popular

이름	**Popular function**
기능	사용자들의 열람 횟수가 높은 악보들의 list 생성
입력	DB에 저장된 모든 사용자들의 열람 기록
조건	-
출력	열람 횟수가 높은 top 500을 순위에 따라 나열함
처리	모든 사용자의 DB 기록을 통해 통계를 내어 열람 횟수가 높은 악보들을 나열한다.

Table 6. Popular

F. Play

이름	**Play function**
기능	악보를 노래와 함께 재생한다.
입력	Play 버튼 클릭
조건	-
출력	노래에 맞추어 악보에 bar가 이동하여 현재 연주해야 하는 부분을 보여줌
처리	현재 사용자가 설정한 환경에 따라 실행된다.

Table 7. Play

G. Change playback speed

이름	**Change playback speed function**
기능	노래의 속도를 조절한다.
입력	등록되어진 배속들 중에 사용자가 선택
조건	-
출력	-
처리	노래의 속도를 악보와 함께 조절한다.

Table 8. Change playback speed

H. Turn loop mode

이름	**Turn loop mode function**
기능	특정 구간을 반복한다.
입력	사용자가 원하는 구간을 입력함
조건	-
출력	-
처리	사용자가 play를 종료하거나 mode를 off할 때까지 입력한 구간을 무한 반복한다.

Table 9. Turn loop mode

I. Retune

이름	**Retune function**
기능	노래의 음정을 조절한다.
입력	사용자가 원하는 음정을 입력함.
조건	특정 줄만 조절하지 못하고 전체를 한번에 조절함
출력	-
처리	노래의 음정을 조절한다.

Table 10. Retune

J. latency compensation

이름	**Latency compensation function**
기능	노래와 악보의 속도를 조정한다.
입력	사용자가 0.1초 단위로 변경시킬 초를 입력
조건	악보의 속도를 +- 0.1초 단위로 움직임.
출력	-
처리	악보의 속도를 조절한다.

Table 11. latency compensation

6.2. Non-functional Requirements

A. Product requirements

A1. Usability requirements

본 시스템의 주 목적은 사람들이 기타 연주를 배우기에 쉬운 환경을 제공하는 것이다. 따라서 사용자가 시스템을 쉽게 사용하고 보고 따라할 수 있어야한다.

A2. Performance requirement

본 시스템을 사용하는 이유 중 하나가 원하는 악보를 찾는 소요시간과 구입 비용을 절감하기 위해서이다. 따라서 많은 악보를 저장하여 사용자가 원하는 악보를 대부분 소지해야 performance 측면에 문제가 없을 것이다.

A3. Security

본 시스템 자체에는 특별히 중요한 개인정보는 없다고 할 수 있다. 하지만 본 시스템은 구글 계정을 연동하여 사용할 수 있기 때문에 보안에 주의하여 사용자의 개인정보가 외부로 유출되는 경우가 없어야 한다.

B. Organization requirements

B1. Environmental requirement

본 서비스는 모바일 디바이스에서 제공되며, 모든 운영체제에서 문제없이 작동되어야 한다. 또한 데이터 및 와이파이 등 인터넷에 접속이 되지 않을 때도 기존에 cache 에 저장된 악보로 이용할 수 있어야 한다.

B2. delivery requirement

본 서비스는 모바일 디바이스에서 제공되어지므로, 사용자들이 쉽게 찾아 다운받을 수 있게 모바일 마켓에서 배포되어야 한다. 안드로이드의 경우 google play 에서 배포되고 애플의 경우 app store 에서 배포된다.

C. External requirements

C1. Regulatory requirement

사용자의 개인정보를 사용하기 때문에 사용자에게 동의를 구하여야 한다. 또한 제공자와 사용자 외에 해당 정보에 접근할 수 없어야 한다.

C2. Ethical requirement

악보에 대한 저작권이 문제가 될 수 있다. 따라서 악보를 제작한 작곡가 혹은 편곡자와 악보 사용에 대한 동의 및 계약을 잘 수행하여야 한다.

C3. Safety/security requirement

개인정보보호법에 따라 사용자의 개인정보를 외부/내부 적으로 안전하게 저장하고, 보호해야한다.

7. System Models

System Models에서는 시스템과 시스템 컴포넌트, 시스템 환경 간의 관계를 여러 Diagram을 통해 보여준다.

7.1 Context model

Process Diagram

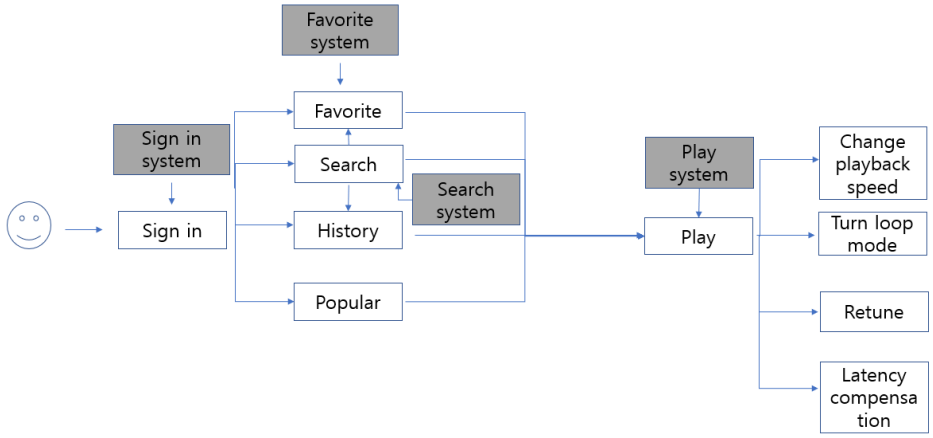


Diagram 5. Overall Process Diagram

7.2 Interaction model

A. Use Case Diagram

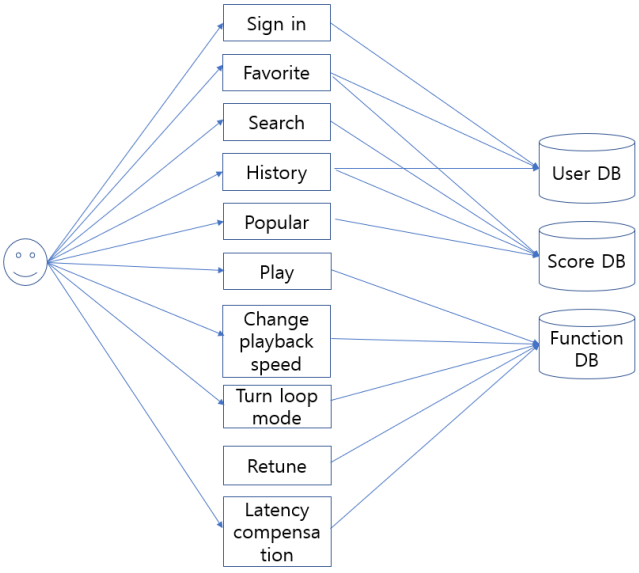


Diagram 6. Use case Diagram

B. Tabular Description of Use case Diagram

1) sign in

Use case	**Sign in**
Actor	User, database(user)
Description	User가 입력한 ID, 비밀번호가 database에 등록된 정보와 일치하는지 확인
Trigger	User가 ID, 비밀번호를 입력
Success response	ID, 비밀번호가 등록된 것과 같으면 로그인에 성공하며 user 정보를 반환한다.
Failure response	ID, 비밀번호가 등록된 것과 다르면 틀리다는 경고 메시지를 알림.

Table 12. sign in

2) search

Use case	**Search**
Actor	User, database(user, score)
Description	User가 원하는 악보를 keyword를 입력하여 확인
Trigger	User가 keyword(음악가 이름 / 노래 제목) 입력
Success response	입력한 keyword와 일치하는 악보가 있으면 화면에 list로 보여줌
Failure response	입력한 keyword와 일치하는 악보가 없으면 빈화면을 보여줌

Table 13. search

3) favorite

Use case	**Favorite**
Actor	User, database(score)
Description	User가 search를 통해 찾은 악보 중 원하는 악보를 나열
Trigger	User들이 악보에 '★' 표시
Success response	favorite page에 '★'가 표시된 악보들 나열
Failure response	-

Table 14. favorite

4) history

Use case	**History**
Actor	User, database(user, score)
Description	User가 최근 열람한 악보를 나열
Trigger	User가 search를 통해 악보들을 열람함
Success response	History page에 최근 열람한 악보들을 나열
Failure response	-

Table 15. history

5) popular

Use case	**Popular**
Actor	User, database(score)
Description	가장 조회수가 높은 악보들을 순위별로 나열
Trigger	User들이 기존에 악보를 열람
Success response	popular page에 조회수가 높은 악보들을 순위별로 나열
Failure response	-

Table 16. popular

6) play

Use case	**Play**
Actor	User, database(function)
Description	악보를 노래에 맞추어 실행
Trigger	User가 play 버튼을 클릭
Success response	노래에 맞추어 악보의 bar가 이동하여 현재 연주해야 하는 부분을 보여줌
Failure response	-

Table 17. play

7) change playback speed

Use case	**Change playback speed**
Actor	User, database(function)
Description	악보와 노래의 속도를 조절
Trigger	User가 원하는 속도를 클릭
Success response	노래와 악보 이동속도를 바꾼 상태로 재생
Failure response	-

Table 18. change playback speed

8) turn loop mode

Use case	**Turn loop mode**
Actor	User, database(function)
Description	악보의 일정 구간을 반복 재생함.
Trigger	user가 원하는 구간을 입력함
Success response	User가 설정한 악보의 일정 구간을 반복 재생함.
Failure response	-

Table 19. turn loop mode

9) retune

Use case	**Retune**
Actor	User, database(function)
Description	노래의 음정을 조절한다.
Trigger	user가 원하는 음정을 입력함.
Success response	노래의 음정을 바뀐 상태로 재생함.
Failure response	-

Table 20. retune

10) latency compensation

Use case	**latency compensation**
Actor	User, database(function)
Description	노래와 악보의 이동속도를 맞춘다.
Trigger	User가 0.1초 단위로 입력함
Success response	User가 설정한 속도에 노래와 악보가 맞춰진다.
Failure response	-

Table 21. latency compensation

8. System Evolution

본 시스템이 개발과정을 거치고 user 들에게 배포되었을 때 모든 개발단계가 끝났다고 할 수 없다. 왜냐면 고객들이 사용하면서 더 원하는 요구사항이 생기는 것은 물론이고 미래에 모바일 디바이스를 포함하여 본 시스템을 둘러싼 환경이 어떠한 변화과정이 생길지 모르기 때문이다. 따라서 이번 챕터에서는 그러한 변화사항을 미리 예상하고, 어떻게 대응할지에 대해 서술한다.

8.1. Score modifying

현재 등록된 악보들은 사용자들이 보고 노래를 들으면서 따라하여 연습할 수 있게 만들었다. 하지만 무척 연주하고 싶지만 악보의 난이도가 너무 높거나, 중간에 오히려 없으면 좋을 것 같은 구간이 있을 수 있다. 따라서 그러한 점을 해결하기 위해 사용자가 악보를 자신만의 style대로 수정이 가능하게 만들면 더 만족성이 높아질 것이라 기대된다.

8.2. Playing external score

현재 'Songsterr' 애플리케이션은 DB에 등록된 악보만을 재생할 수 있다. 하지만 사용자가 원하는 악보가 모두 애플리케이션에 등록되어 있다고 보장할 수 없다. 따라서 기존에 등록된 악보만이 아닌 애플리케이션 외부의 악보(gp 파일)도 재생할 수 있으면, 타 애플리케이션 필요 없게 되어 'Songsterr'의 사용량이 늘어날 것이라 기대된다.

8.3. Category

사용자들이 선호하는 음악의 장르, 연주실력 등 각자 찾는 음악의 부류가 다를 것이다. 따라서 'Songsterr'에 category 를 만들어 사용자가 특별히 찾는 노래가 없을 경우에도 사용자가 지정한 category 별로 악보를 보여주면 쉽게 선별할 수 있을 것이다.

9. Appendices

A. Hardware Requirements

본 시스템은 스마트폰과 같은 모바일 기기에서 사용하도록 제작되었다. 따라서 이용하기 위해서는 Android나 Ios 운영체제의 기기가 필요하다. 또한 설치하기 위해서 모바일 기기의 기본 마켓인 google play나 app store가 있어야 한다. 초기 이용시에는 악보를 서버에서 다운을 받아야 하기 때문에 모바일 네트워크나 와이파이 연결이 필수로 있어야한다. 하지만 이후에는 사용했던 악보들이 기기에도 저장되어 네트워크 없이도 이용할 수 있다.

B. Database Requirements

본 시스템은 RDBMS 데이터베이스 구조를 활용해 데이터를 저장한다.

10. Index

10.1 Tables

10.2 Figures

10.3 Diagrams