**스마트앱프로그래밍 Term Project 결과 보고서**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 분반 | 3분반 | | | |
| 작 품 명 | 걸그룹 멤버 리스트 프로그램 | | | |
| 개발기간 | 2023 년 12 월 11 일 ～ 2023 년 12 월 21 일 | | | |
| 지도교수 | 컴퓨터공학과 | | 오병우 | |
| 구 분 | 학년 | 학 번 | | 성 명 |
| 책임자(팀장) | 4 | 20161163 | | 조장호 |
| 팀원1 | 해당 사항 없음 |  | |  |
| 팀원2 | 해당 사항 없음 |  | |  |
| 팀원3 | 해당 사항 없음 |  | |  |
| 본인은 스마트앱프로그래밍 Term Project 결과 보고서를 첨부와 같이 제출합니다. 프로젝트를 진행할 때 부정한 방법을 사용했거나 보고서에 사실과 다른 내용을 적었을 때는 모든 처벌을 감수하겠습니다.  2023년 12월 21일  팀 장 조장호    컴퓨터공학심화프로그램 | | | | |

**결과 보고서**

**걸그룹 멤버 리스트 프로그램**

**1. 서론**

**개발 목표**

이 앱의 목적은 사용자가 걸그룹 멤버 정보를 볼 수 있는 편리한 인터페이스를 제공하는 것입니다. 사용자는 리스트를 스크롤하여 Kpop 걸그룹 멤버들의 정보를 확인할 수 있으며, 특정 멤버를 선택하면 해당 멤버의 상세 정보를 확인할 수 있습니다. 이를 통해 사용자는 좋아하는 걸그룹 멤버에 대한 자세한 정보를 얻을 수 있습니다.

**배경 설명**



K-pop은 전 세계적으로 큰 인기를 얻고 있으며, 많은 사람들이 걸그룹 멤버들에게 관심을 가지고 있습니다. 이러한 관심을 바탕으로, 이 앱은 사용자가 쉽게 걸그룹 멤버들의 정보를 탐색하고 확인할 수 있는 플랫폼을 제공합니다. 사용자들은 멤버의 프로필, 이미지, 데뷔 날짜 등과 같은 정보를 확인하여 걸그룹에 대한 더 많은 통찰을 얻을 수 있습니다.

**개발동기 또는 필요성**

사용자들은 걸그룹 멤버에 대한 정보를 원하고 있으며, 이를 쉽게 접근할 수 있는 앱이 필요합니다.

정보를 정리하고 시각적으로 표현하여 사용자가 원하는 정보를 빠르게 찾을 수 있도록 합니다.

모바일 앱의 편리한 UI/UX를 통해 사용자들이 걸그룹 멤버에 대한 더 많은 정보를 얻을 수 있습니다.

**개발하는 과제의 관련 기술현황**

**Jetpack Compose 및 Kotlin**: Jetpack Compose를 사용하여 Android 앱의 UI를 구축하며, Kotlin 언어를 통해 안정적이고 효율적인 개발을 진행합니다.

**MVVM 아키텍처**: MVVM 패턴을 활용하여 백엔드로부터 데이터를 관리하고 UI와 데이터의 분리를 유지하여 코드를 모듈화합니다.

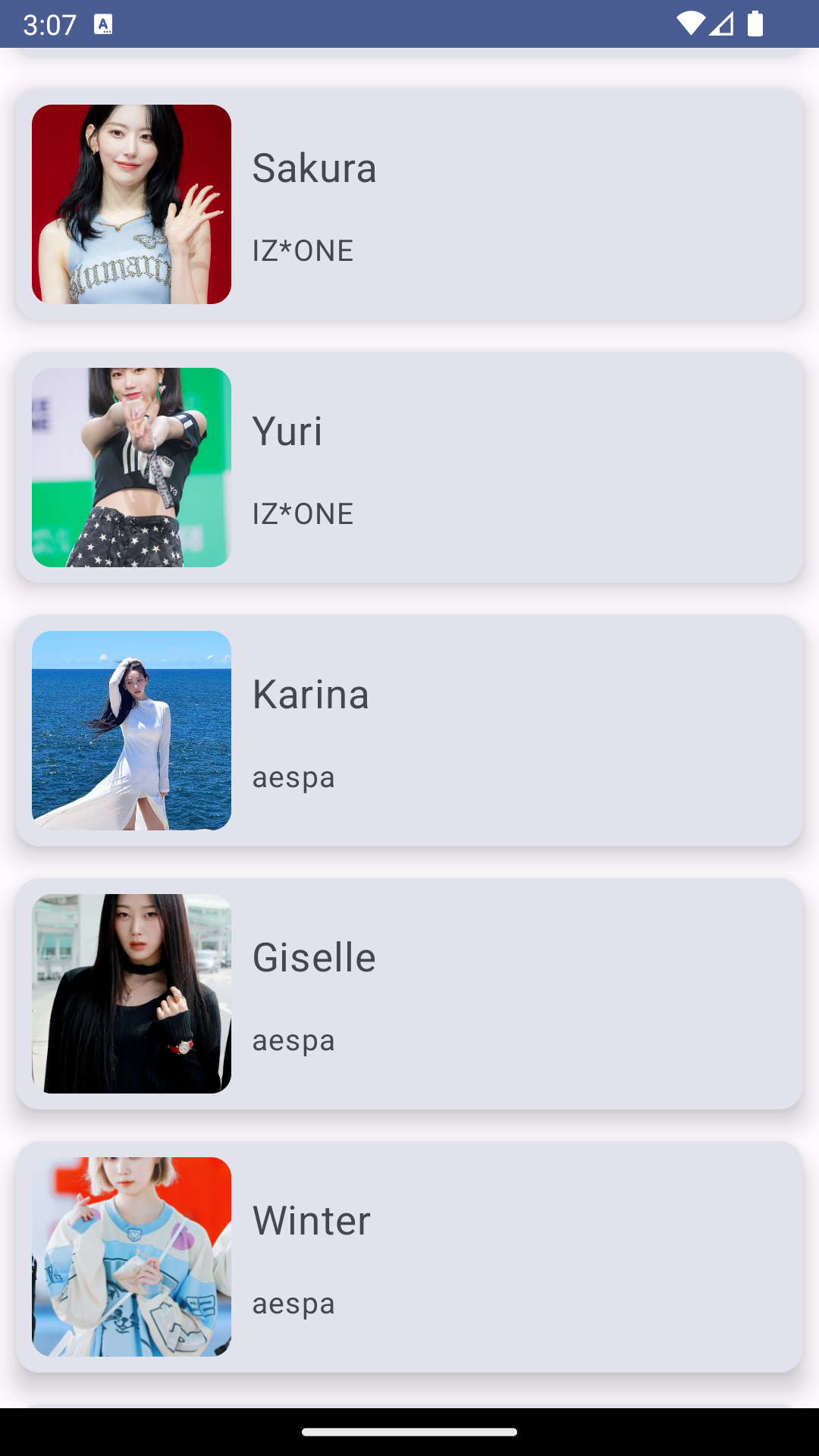
**리스트 및 상세 정보 출력**: LazyColumn을 사용하여 멤버 리스트를 출력하고, 아이템을 클릭하면 상세 정보를 보여줍니다.

관계형 데이터베이스 및 백엔드 서비스: 좋아하는 취미에 대한 관계형 데이터베이스를 구축하고, 해당 데이터를 JSON 형태로 제공하는 백엔드를 구축합니다.

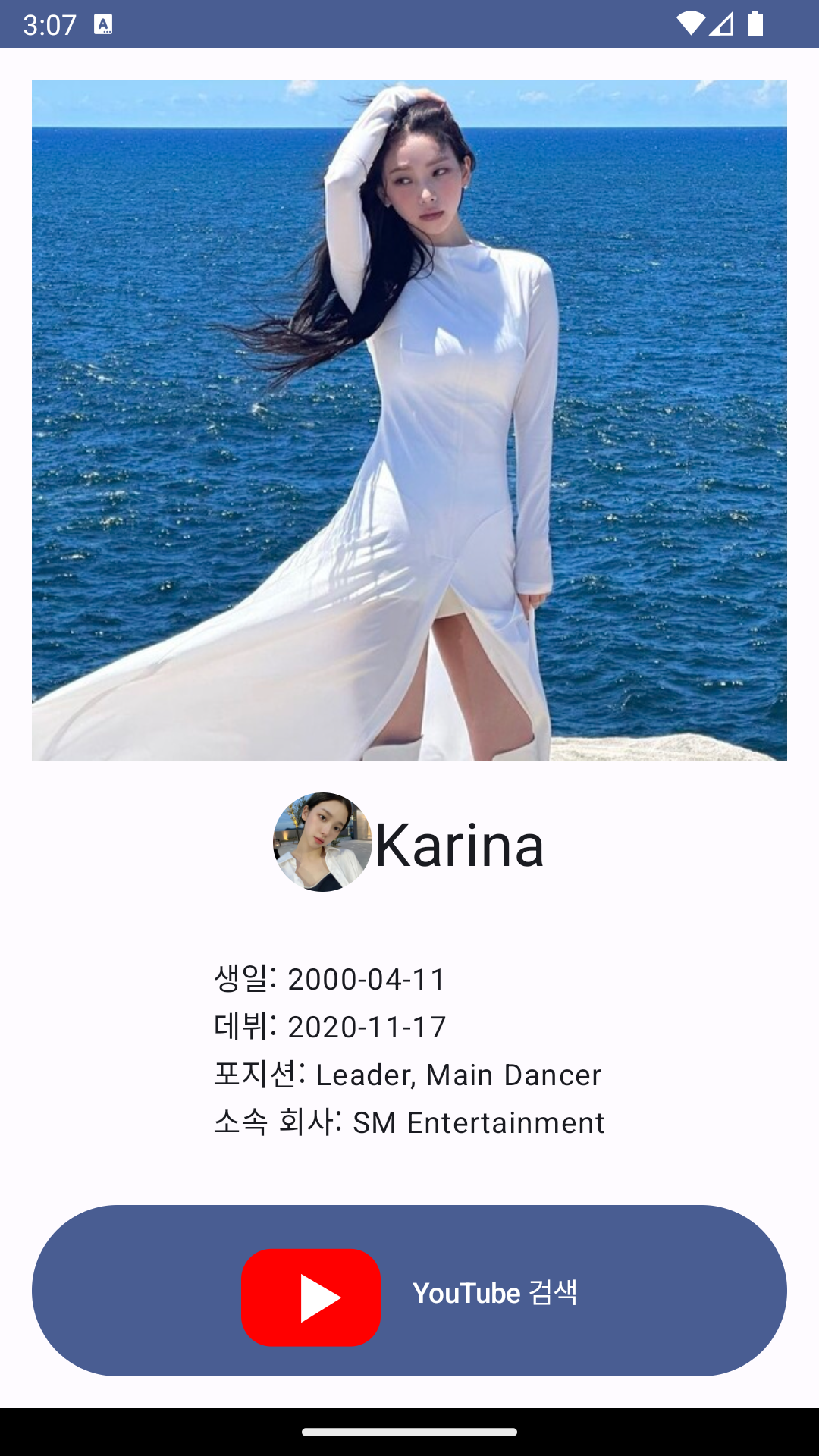
이미지 URL 및 이미지 출력: 데이터베이스에 이미지 URL을 저장하고, 앱에서 실제 이미지를 로드하여 출력합니다.

**2. 작품 개요**

**2.1 전체 구성도**



앱을 시작하면 사용자는 걸그룹 멤버들의 리스트를 스크롤하여 확인할 수 있습니다. LazyColumn을 사용하여 멤버들을 표시하고, 리스트 아이템을 클릭하면 해당 멤버의 상세 정보 화면으로 이동합니다.



사용자가 멤버를 클릭하면 해당 멤버의 상세 정보를 확인할 수 있습니다. 멤버의 그룹 이름, 생년월일, 데뷔 날짜, 소속 회사, 포지션 등과 함께 Full Body 및 Face 이미지를 볼 수 있습니다.

각 멤버의 프로필에는 YouTube에서 해당 멤버에 대한 검색 결과 링크가 포함되어 있어, 사용자는 해당 링크를 클릭하여 YouTube에서 더 많은 멤버 관련 컨텐츠를 검색할 수 있습니다.

**2.2 작품 설명**

걸그룹 멤버 리스트는 걸그룹 멤버에 대한 정보를 제공하는 Android 애플리케이션입니다. 이 앱은 사용자가 여러 걸그룹 멤버들의 프로필, 이미지, 데뷔 날짜 등을 확인할 수 있도록 합니다.

**주요 기능:**

**멤버 리스트 조회**: 앱을 시작하면 사용자는 걸그룹 멤버들의 리스트를 스크롤하여 확인할 수 있습니다. LazyColumn을 사용하여 멤버들을 표시하고, 리스트 아이템을 클릭하면 해당 멤버의 상세 정보 화면으로 이동합니다.

**멤버 상세 정보**: 사용자가 멤버를 클릭하면 해당 멤버의 상세 정보를 확인할 수 있습니다. 멤버의 그룹 이름, 생년월일, 데뷔 날짜, 소속 회사, 포지션 등과 함께 Full Body 및 Face 이미지를 볼 수 있습니다.

**외부 링크 연결**: 각 멤버의 프로필에는 YouTube에서 해당 멤버에 대한 검색 결과 링크가 포함되어 있어, 사용자는 해당 링크를 클릭하여 YouTube에서 더 많은 멤버 관련 컨텐츠를 검색할 수 있습니다.

**MVVM 아키텍처**: 앱은 MVVM 아키텍처를 따르며, 백엔드 서버에서 받은 데이터를 ViewModel을 통해 관리하고, Jetpack Compose로 UI를 렌더링합니다.

**3. 설계**

**3.1 Use Case**

**멤버 리스트 보기**

사용자: 앱 사용자

설명: 앱을 실행하면 사용자는 홈 화면에서 걸그룹 멤버 리스트를 확인할 수 있다.

동작:

사용자는 LazyColumn을 스크롤하여 여러 걸그룹 멤버의 이름과 이미지를 볼 수 있다.

사용자는 특정 걸그룹 멤버를 클릭하여 해당 멤버의 상세 정보를 볼 수 있다.

**멤버 상세 정보 보기**

사용자: 앱 사용자

설명: 사용자는 걸그룹 멤버의 상세 정보를 확인할 수 있다.

동작:

사용자는 리스트에서 특정 멤버를 클릭하여 해당 멤버의 상세 정보로 이동할 수 있다.

사용자는 해당 멤버의 그룹 이름, 생년월일, 데뷔 날짜, 소속 회사, 포지션 등과 함께 Full Body 및 Face 이미지를 확인할 수 있다.

사용자는 해당 멤버의 프로필에 연결된 YouTube 링크를 클릭하여 해당 멤버에 대한 더 많은 컨텐츠를 검색할 수 있다.

**외부 링크 연결하기**

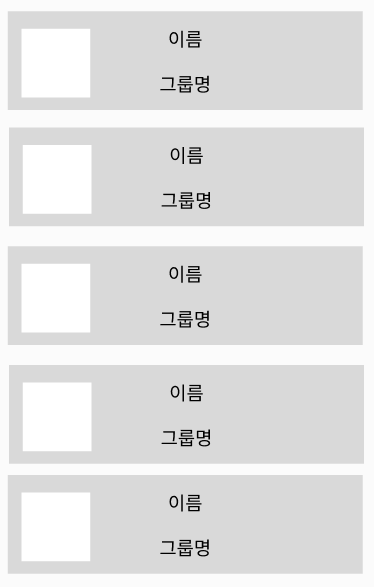
사용자: 앱 사용자

설명: 사용자는 해당 멤버 프로필에 연결된 YouTube 링크를 통해 외부 콘텐츠를 탐색할 수 있다.

동작:

사용자는 앱에서 특정 멤버의 프로필을 클릭하면 해당 멤버에 대한 YouTube 검색 결과 링크로 이동할 수 있다.

**3.2 UI Design**



앱을 시작하면 사용자는 걸그룹 멤버들의 리스트를 스크롤하여 확인할 수 있습니다. LazyColumn을 사용하여 멤버들을 표시하고, 리스트 아이템을 클릭하면 해당 멤버의 상세 정보 화면으로 이동합니다.

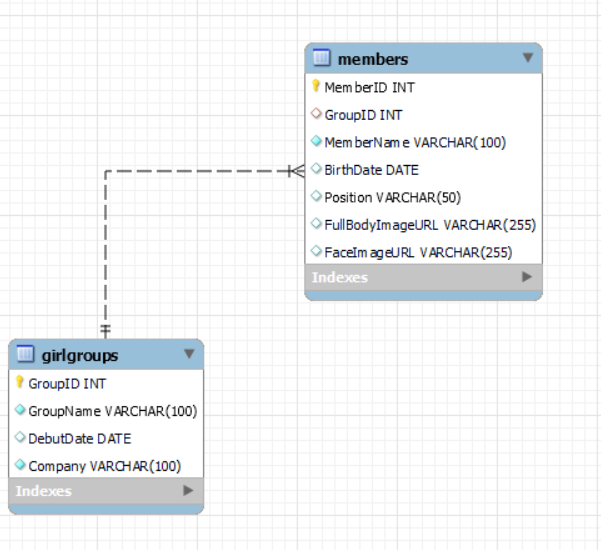


사용자가 멤버를 클릭하면 해당 멤버의 상세 정보를 확인할 수 있습니다. 멤버의 그룹 이름, 생년월일, 데뷔 날짜, 소속 회사, 포지션 등과 함께 Full Body 및 Face 이미지를 볼 수 있습니다.

각 멤버의 프로필에는 YouTube에서 해당 멤버에 대한 검색 결과 링크가 포함되어 있어, 사용자는 해당 링크를 클릭하여 YouTube에서 더 많은 멤버 관련 컨텐츠를 검색할 수 있습니다.

**3.3 DB Design**

**3.3.1 Conceptual Design**



**GirlGroups 테이블:**

* GroupID (PK): 그룹의 고유 식별자 (자동 증가)
* GroupName: 그룹 이름
* DebutDate: 데뷔 날짜
* Company: 소속사

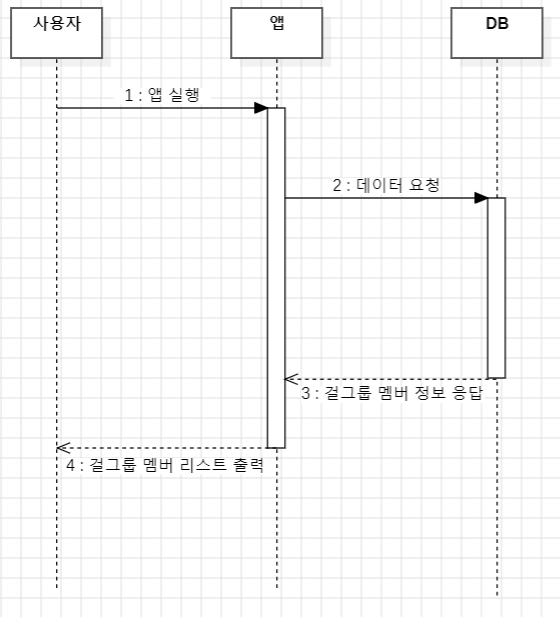
**Members 테이블:**

* MemberID (PK): 멤버의 고유 식별자 (자동 증가)
* GroupID (FK): 그룹을 식별하는 외래 키
* MemberName: 멤버 이름
* BirthDate: 생년월일
* Position: 멤버의 활동 포지션
* FullBodyImageURL: 멤버의 전신 이미지 URL
* FaceImageURL: 멤버의 얼굴 이미지 URL

**3.3.2 Logical Design**

1. 사용자가 앱을 통해 걸그룹 멤버 리스트를 확인하고 특정 멤버를 선택합니다.
2. 선택한 멤버에 대한 정보 요청이 앱 내부의 ViewModel로 전달됩니다.
3. ViewModel은 Repository를 통해 데이터베이스에서 해당 멤버의 정보를 가져옵니다.
4. ViewModel은 UI에 결과를 반환하여 사용자가 선택한 멤버에 대한 상세 정보를 표시합니다.

**3.4 Use Case별 Sequence Diagram**



**리스트 출력 시나리오:**

Actor: 사용자

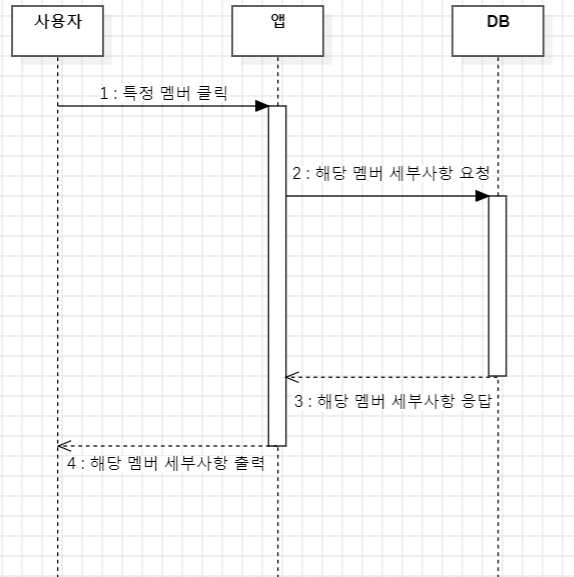
시나리오:

사용자가 앱을 실행합니다.

앱은 백엔드에 데이터 요청을 보냅니다.

백엔드는 데이터베이스에서 걸그룹 멤버 정보를 검색하여 응답합니다.

앱은 받은 정보를 바탕으로 리스트를 구성하여 사용자에게 출력합니다.



**아이템 클릭 시 세부 내용 출력 시나리오:**

Actor: 사용자

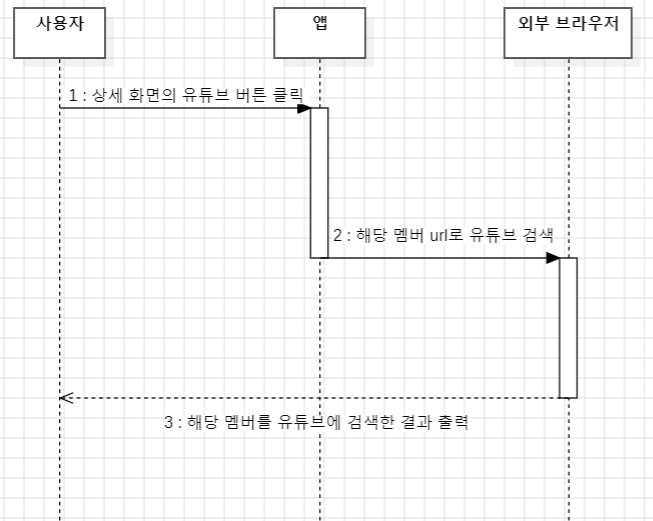
시나리오:

사용자가 리스트에서 특정 멤버를 선택합니다.

앱은 해당 멤버의 세부 정보를 요청하기 위해 백엔드에 요청을 보냅니다.

백엔드는 데이터베이스에서 해당 멤버의 세부 정보를 검색하여 응답합니다.

앱은 받은 정보를 바탕으로 해당 멤버의 세부 정보를 사용자에게 출력합니다.



**유튜브 버튼 클릭 시 세부 내용 출력 시나리오:**

Actor: 사용자

시나리오:

사용자가 세부 내용 화면에서 유튜브 버튼을 클릭합니다.

앱은 해당 멤버의 URL 정보를 바탕으로 유튜브 검색 페이지를 생성합니다.

외부 브라우저가 해당 멤버의 유튜브 검색 페이지 URL을 받아 해당 페이지를 엽니다.

**4. 구현**

**MainActivity.kt**

package kr.ac.kumoh.s20161163.myapplication

import android.os.Bundle

import androidx.activity.ComponentActivity

import androidx.activity.compose.setContent

import androidx.activity.viewModels

import androidx.compose.foundation.layout.fillMaxSize

import androidx.compose.material3.MaterialTheme

import androidx.compose.material3.Surface

import androidx.compose.runtime.Composable

import androidx.compose.runtime.getValue

import androidx.compose.runtime.livedata.observeAsState

import androidx.compose.ui.Modifier

import kr.ac.kumoh.s20161163.myapplication.ui.theme.MyApplicationTheme

// ComponentActivity를 상속받는 MainActivity 클래스 정의

class MainActivity : ComponentActivity() {

// KpopMemeberViewModel 인스턴스 생성

private val viewModel: KpopMemeberViewModel by viewModels()

override fun onCreate(savedInstanceState: Bundle?) {

super.onCreate(savedInstanceState)

// setContent 함수를 사용하여 화면을 설정

setContent {

// MainScreen 함수 호출하여 MainScreen 컴포저 호출

MainScreen(viewModel)

}

}

}

// Composable 함수 MainScreen 정의

@Composable

fun MainScreen(viewModel: KpopMemeberViewModel) {

// KpopMemeberViewModel의 memberList를 observeAsState로 가져와서 memberList에 할당

val memberList by viewModel.memberList.observeAsState(emptyList())

// MyApplicationTheme으로 감싸진 Surface

MyApplicationTheme {

Surface(

modifier = Modifier.fillMaxSize(),

color = MaterialTheme.colorScheme.background

) {

// MemberApp 함수 호출하여 MemberApp 컴포저 호출

MemberApp(memberList)

}

}

}

// Composable 함수 MemberApp 정의

@Composable

fun MemberApp(memberList: List<KpopMember>) {

// LazyColumn을 사용하여 멤버 리스트 출력

LazyColumn {

itemsIndexed(memberList) { index, member ->

// 리스트의 각 항목을 나타내는 Composable 함수 Member 호출

Member(member = member, index = index)

}

}

}

// Composable 함수 Member 정의

@Composable

fun Member(member: KpopMember, index: Int) {

// Card를 사용하여 각 멤버 정보를 나타냄

Card(

modifier = Modifier

.padding(8.dp)

.fillMaxWidth(),

elevation = 8.dp,

shape = RoundedCornerShape(8.dp)

) {

// Column을 사용하여 멤버 정보 레이아웃 구성

Column(

modifier = Modifier

.clickable { /\* 해당 멤버를 클릭했을 때의 동작 구현 \*/ }

.padding(16.dp)

) {

// 멤버의 이름과 그룹을 Text로 표시

Text(

text = "${member.MemberName} - ${member.GroupName}",

fontSize = 18.sp

)

// AsyncImage를 사용하여 멤버의 이미지 출력

AsyncImage(

data = member.ImageURL,

contentDescription = null,

contentScale = ContentScale.Crop,

modifier = Modifier

.size(120.dp)

.clip(RoundedCornerShape(8.dp))

)

}

}

}

**KpopViewModel.kt**

package kr.ac.kumoh.s20161163.myapplication

import android.util.Log

import androidx.lifecycle.LiveData

import androidx.lifecycle.MutableLiveData

import androidx.lifecycle.ViewModel

import androidx.lifecycle.viewModelScope

import kotlinx.coroutines.launch

import retrofit2.Retrofit

import retrofit2.converter.gson.GsonConverterFactory

class KpopMemberViewModel : ViewModel() {

// API 서버 URL

private val SERVER\_URL = "https://port-0-s23w10backend-3yl7k2blonrwyx5.sel5.cloudtype.app/"

// KpopApi 인터페이스를 사용하여 Retrofit 생성

private val kpopApi: KpopApi

// MutableLiveData를 사용하여 멤버 목록을 저장

private val \_memberList = MutableLiveData<List<KpopMember>>()

val memberList: LiveData<List<KpopMember>>

get() = \_memberList

init {

// Retrofit을 초기화하고 API 서비스를 생성

val retrofit = Retrofit.Builder()

.baseUrl(SERVER\_URL)

.addConverterFactory(GsonConverterFactory.create())

.build()

kpopApi = retrofit.create(KpopApi::class.java)

// fetchData 함수 호출하여 데이터 가져오기

fetchData()

}

// API를 통해 데이터를 가져오는 함수

private fun fetchData() {

viewModelScope.launch {

try {

// Retrofit을 사용하여 API를 호출하고 응답을 받아옴

val response = kpopApi.getMembers()

// 받아온 데이터를 MutableLiveData에 할당하여 UI 업데이트

\_memberList.value = response

} catch (e: Exception) {

// 오류 발생 시 로그에 기록

Log.e("fetchData()", e.toString())

}

}

}

}

**MemberScreen.kt**

package kr.ac.kumoh.s20161163.myapplication

import android.content.Intent

import android.net.Uri

import androidx.compose.foundation.Canvas

import androidx.compose.foundation.clickable

import androidx.compose.foundation.layout.Arrangement

import androidx.compose.foundation.layout.Column

import androidx.compose.foundation.layout.IntrinsicSize

import androidx.compose.foundation.layout.PaddingValues

import androidx.compose.foundation.layout.Row

import androidx.compose.foundation.layout.Spacer

import androidx.compose.foundation.layout.fillMaxSize

import androidx.compose.foundation.layout.fillMaxWidth

import androidx.compose.foundation.layout.height

import androidx.compose.foundation.layout.padding

import androidx.compose.foundation.layout.size

import androidx.compose.foundation.layout.width

import androidx.compose.foundation.lazy.LazyColumn

import androidx.compose.foundation.rememberScrollState

import androidx.compose.foundation.shape.CircleShape

import androidx.compose.foundation.shape.RoundedCornerShape

import androidx.compose.foundation.verticalScroll

import androidx.compose.material3.Button

import androidx.compose.material3.Card

import androidx.compose.material3.CardDefaults

import androidx.compose.material3.Text

import androidx.compose.runtime.Composable

import androidx.compose.ui.Alignment

import androidx.compose.ui.Modifier

import androidx.compose.ui.draw.clip

import androidx.compose.ui.geometry.CornerRadius

import androidx.compose.ui.geometry.Offset

import androidx.compose.ui.geometry.Size

import androidx.compose.ui.graphics.Color

import androidx.compose.ui.graphics.Path

import androidx.compose.ui.layout.ContentScale

import androidx.compose.ui.platform.LocalContext

import androidx.compose.ui.text.style.TextAlign

import androidx.compose.ui.unit.dp

import androidx.compose.ui.unit.sp

import androidx.core.content.ContextCompat.startActivity

import androidx.navigation.NavType

import androidx.navigation.compose.NavHost

import androidx.navigation.compose.composable

import androidx.navigation.compose.rememberNavController

import androidx.navigation.navArgument

import coil.compose.AsyncImage

enum class MemberScreen {

List, Detail

}

@Composable

fun MemberApp(memberList: List<KpopMember>) {

// NavController를 생성하여 Composable 내에서 목적지 간의 네비게이션을 처리합니다.

val navController = rememberNavController()

// NavHost를 사용하여 화면의 내비게이션을 관리합니다.

NavHost(

navController = navController,

startDestination = MemberScreen.List.name,

) {

// List 화면을 표시하는 Composable을 정의합니다.

composable(route = MemberScreen.List.name) {

// MemberList Composable을 사용하여 목록을 표시합니다.

MemberList(memberList) {

// List에서 Detail로 네비게이션하는 데 필요한 정보를 전달합니다.

// it은 index 값으로 "Detail/$index" 형태의 String이 됩니다.

navController.navigate(it)

}

}

// Detail 화면을 표시하는 Composable을 정의합니다.

composable(

route = MemberScreen.Detail.name + "/{index}",

arguments = listOf(navArgument("index") {

type = NavType.IntType

})

) {

val index = it.arguments?.getInt("index") ?: -1

if (index >= 0)

// memberEtc 함수를 호출하여 멤버의 상세 정보를 표시합니다.

memberEtc(memberList[index])

}

}

}

@Composable

fun MemberList(list: List<KpopMember>, onNavigateToDetail: (String) -> Unit) {

// LazyColumn을 사용하여 멤버 목록을 표시합니다.

LazyColumn(

verticalArrangement = Arrangement.spacedBy(16.dp),

contentPadding = PaddingValues(horizontal = 8.dp)

) {

items(list.size) {

// Member Composable을 사용하여 각각의 멤버를 표시합니다.

Member(it, list[it], onNavigateToDetail)

}

}

}

@Composable

fun Member(

index: Int,

member: KpopMember,

onNavigateToDetail: (String) -> Unit

) {

// 각 멤버를 Card로 감싸서 클릭 시 상세 화면으로 네비게이션합니다.

Card(

modifier = Modifier.clickable {

// 클릭 시 상세 화면으로 이동하기 위한 정보를 네비게이션합니다.

onNavigateToDetail(MemberScreen.Detail.name + "/$index")

}, elevation = CardDefaults.cardElevation(8.dp)

) {

Row(

modifier = Modifier

.fillMaxWidth()

.height(IntrinsicSize.Min)

.padding(8.dp)

) {

// 각 멤버의 이미지와 정보를 표시합니다.

AsyncImage(

model = member.FullBodyImageURL,

contentDescription = "노래 앨범 이미지",

contentScale = ContentScale.Crop,

modifier = Modifier

.size(100.dp)

.clip(RoundedCornerShape(percent = 10)),

)

Spacer(modifier = Modifier.width(10.dp))

Column(

modifier = Modifier.fillMaxSize(), verticalArrangement = Arrangement.SpaceEvenly

) {

// 멤버 이름과 소속 그룹을 표시하는 함수를 호출합니다.

memberName(member.MemberName)

memberGroupe(member.GroupName)

}

}

}

}

@Composable

fun memberName(name: String) {

// 멤버 이름을 표시하는 Text Composable입니다.

Text(name, fontSize = 20.sp)

}

@Composable

fun memberGroupe(groupeName: String) {

// 소속 그룹 이름을 표시하는 Text Composable입니다.

Text(groupeName, fontSize = 15.sp)

}

@Composable

fun memberEtc(member: KpopMember) {

val context = LocalContext.current

Column(

modifier = Modifier

.padding(16.dp)

.verticalScroll(rememberScrollState()),

horizontalAlignment = Alignment.CenterHorizontally

) {

// 소속 그룹 이름을 큰 글씨로 표시합니다.

Text(

member.GroupName,

fontSize = 40.sp,

textAlign = TextAlign.Center,

lineHeight = 45.sp

)

Spacer(modifier = Modifier.height(16.dp))

// 멤버의 전신 이미지를 표시합니다.

AsyncImage(

model = "${member.FullBodyImageURL}",

contentDescription = "멤버 이미지",

contentScale = ContentScale.Crop,

modifier = Modifier

.size(400.dp),

)

Spacer(modifier = Modifier.height(16.dp))

Row(

verticalAlignment = Alignment.CenterVertically

) {

// 멤버의 얼굴 이미지를 원형으로 표시합니다.

AsyncImage(

model = "${member.FaceImageURL}",

contentDescription = "가수 이미지",

contentScale = ContentScale.Crop,

modifier = Modifier

.size(50.dp)

.clip(CircleShape)

)

Text(member.MemberName, fontSize = 30.sp)

}

Spacer(modifier = Modifier.height(32.dp))

// 멤버의 생일, 데뷔일, 포지션, 소속 회사 정보를 텍스트로 표시합니다.

Text(

"생일: ${member.BirthDate.substring(0, 10)}\n데뷔: ${member.DebutDate.substring(0, 10)}\n포지션: ${member.Position}\n소속 회사: ${member.Company}",

fontSize = 15.sp,

)

Spacer(modifier = Modifier.height(32.dp))

// YouTube 검색을 위한 Button을 생성합니다.

Button(onClick = {

// 멤버의 이름을 검색 쿼리로 넣어 YouTube 검색을 수행합니다.

val intent = Intent(

Intent.ACTION\_VIEW,

Uri.parse("https://www.youtube.com/results?search\_query=${member.MemberName}")

)

startActivity(context, intent, null)

}) {

Row(

modifier = Modifier.fillMaxWidth(),

horizontalArrangement = Arrangement.Center,

verticalAlignment = Alignment.CenterVertically

) {

// YouTube 아이콘을 그리기 위한 Canvas입니다.

YoutubeIcon()

Spacer(modifier = Modifier.width(16.dp))

Text("YouTube 검색")

}

}

}

}

@Composable

fun YoutubeIcon() {

Canvas(

modifier = Modifier

.size(70.dp)

) {

// YouTube 아이콘을 그리기 위한 경로입니다.

val path = Path().apply {

moveTo(size.width \* .43f, size.height \* .38f)

lineTo(size.width \* .72f, size.height \* .55f)

lineTo(size.width \* .43f, size.height \* .73f)

close()

}

drawRoundRect(

color = Color.Red,

cornerRadius = CornerRadius(40f, 40f),

size = Size(size.width, size.height \* .70f),

topLeft = Offset(size.width.times(.0f), size.height.times(.20f))

)

drawPath(color = Color.White, path = path)

}

}

**KpopMember.kt**

package kr.ac.kumoh.s20161163.myapplication

data class KpopMember(

val MemberID : Int,

val GroupName: String,

val DebutDate: String,

val Company: String,

val MemberName: String,

val BirthDate: String,

val Position: String,

val FullBodyImageURL: String,

val FaceImageURL: String,

)

**KpopApi.kt**

package kr.ac.kumoh.s20161163.myapplication

import retrofit2.http.GET

interface KpopApi {

@GET("member")

suspend fun getMembers(): List<KpopMember>

}

**Database**

**-- 데이터베이스 생성**

DROP DATABASE IF EXISTS KpopGroups;

CREATE DATABASE KpopGroups;

USE KpopGroups;

**-- GirlGroups 테이블 생성**

CREATE TABLE IF NOT EXISTS GirlGroups (

GroupID INT AUTO\_INCREMENT PRIMARY KEY,

GroupName VARCHAR(100) NOT NULL,

DebutDate DATE,

Company VARCHAR(100) NOT NULL

) DEFAULT CHARACTER SET utf8mb4;

**Members 테이블 생성**

CREATE TABLE IF NOT EXISTS Members (

MemberID INT AUTO\_INCREMENT PRIMARY KEY,

GroupID INT,

MemberName VARCHAR(100) NOT NULL,

BirthDate DATE,

Position VARCHAR(50),

FullBodyImageURL VARCHAR(255), -- 전신 이미지 URL 열 추가

FaceImageURL VARCHAR(255), -- 얼굴 이미지 URL 열 추가

FOREIGN KEY (GroupID) REFERENCES GirlGroups(GroupID)

) DEFAULT CHARACTER SET utf8mb4;

**-- GirlGroups 테이블 데이터 추가**

INSERT INTO GirlGroups (GroupName, DebutDate, Company) VALUES

('TWICE', '2015-10-20', 'JYP Entertainment'),

('Red Velvet', '2014-08-01', 'SM Entertainment'),

('IZ\*ONE', '2018-10-29', 'Off the Record Entertainment'),

('aespa', '2020-11-17', 'SM Entertainment'),

('ITZY', '2021-04-21', 'YG Entertainment'),

('BLACKPINK', '2016-08-08', 'YG Entertainment'),

('NewJeans', '2022-08-08', 'Hybe Entertainment');

**-- Members 테이블 데이터 추가**

INSERT INTO Members (GroupID, MemberName, BirthDate, Position, FullBodyImageURL, FaceImageURL) VALUES

(1, 'Nayeon', '1995-09-22', 'Lead Vocalist', 'https://t1.daumcdn.net/cfile/tistory/99254C365CE3E05D1C', 'https://img.seoul.co.kr/img/upload/2023/09/19/SSC\_20230919115642\_O2.jpg'),

(1, 'Jeongyeon', '1996-11-01', 'Lead Vocalist', 'https://image.ytn.co.kr/general/jpg/2021/0819/202108191411566114\_d.jpg', 'https://www.tfmedia.co.kr/data/photos/20201044/art\_16041337464895\_8e5314.jpg'),

(1, 'Momo', '1996-11-09', 'Main Dancer', 'https://i.namu.wiki/i/nVcm3cRzaI7JXRe7FIldg1xsJ-S3ukGv9tda-ktlACnYjM4FfqyzjaRSQ0avMHmkU60jAKNRrmWmdciHl0gPnQ.webp', 'https://cdn.pinpointnews.co.kr/news/photo/202105/50687\_67463\_1733.jpg'),

(2, 'Irene', '1991-03-29', 'Leader, Main Rapper', 'https://thumb.mtstarnews.com/06/2023/10/2023101915003551361\_1.jpg/dims/optimize', 'https://sports.hankooki.com/news/photo/201905/img\_6416398\_0.jpg'),

(2, 'Seulgi', '1994-02-10', 'Main Dancer', 'https://static.inews24.com/v1/375f6ab250cda0.jpg', 'https://live.staticflickr.com/65535/49524476483\_1795eb9ec0\_o.jpg'),

(2, 'Wendy', '1994-02-21', 'Main Vocalist', 'https://image.xportsnews.com/contents/images/upload/article/2023/0619/mb\_1687130233331295.jpg', 'https://dimg.donga.com/wps/SPORTS/IMAGE/2023/06/09/119692309.3.jpg'),

(3, 'Wonyoung', '2004-08-31', 'Lead Vocalist', 'https://talkimg.imbc.com/TVianUpload/tvian/TViews/image/2022/10/01/gppKEHUiAyIO638002448398716484.jpg', 'https://static.news.zumst.com/images/139/2023/04/11/a6e7c3ec631e4c16a7e8638ab3d81771.jpg'),

(3, 'Sakura', '1998-03-19', 'Lead Rapper', 'https://talkimg.imbc.com/TVianUpload/tvian/TViews/image/2023/05/01/5b689644-4d48-4911-abf1-e21ae88c519e.jpg', 'https://entertainimg.kbsmedia.co.kr/cms/uploads/PERSON\_20220405075028\_3b9104d6de492e3833dd9e4c60b60c7a.jpg'),

(3, 'Yuri', '2003-10-22', 'Main Vocalist', 'https://cdn.topstarnews.net/news/photo/202206/14696380\_801236\_148.jpg', 'https://newsimg.hankookilbo.com/cms/articlerelease/2021/08/30/97910b03-91fc-4dff-bae1-6eaf52a93fda.jpg'),

(4, 'Karina', '2000-04-11', 'Leader, Main Dancer', 'https://cdn.theproof.co.kr/news/photo/202308/592\_830\_2319.jpg', 'https://talkimg.imbc.com/TVianUpload/tvian/TViews/image/2022/09/18/1e586277-48ba-4e8a-9b98-d8cdbe075d86.jpg'),

(4, 'Giselle', '2000-10-30', 'Main Rapper', 'https://img.hankyung.com/photo/202306/BF.33648689.1.jpg', 'https://cdn.mhnse.com/news/photo/202211/156012\_150456\_17.jpg'),

(4, 'Winter', '2001-10-28', 'Lead Vocalist', 'https://kpopping.com/documents/12/1/2267/220624-aespa-Winter-documents-1(1).jpeg?v=e7118', 'https://dispatch.cdnser.be/cms-content/uploads/2023/05/22/7130a80b-fa35-42bd-8195-588221cc8ffc.jpg'),

(5, 'Yeji', '2000-12-20', 'Leader, Main Rapper', 'https://image.kmib.co.kr/online\_image/2019/1225/611811110014067007\_1.jpg', 'https://talkimg.imbc.com/TVianUpload/tvian/TViews/image/2022/01/03/XlaLfTVZ5iIM637768407379571191.jpg'),

(5, 'Lia', '2000-07-21', 'Main Vocalist', 'https://file.mk.co.kr/meet/neds/2023/09/image\_readtop\_2023\_709255\_16950132025630215.jpg', 'https://image.ytn.co.kr/general/jpg/2023/0918/202309181355182175\_d.jpg'),

(5, 'Ryujin', '2001-04-17', 'Main Rapper', 'https://news.nateimg.co.kr/orgImg/sh/2022/02/27/kda20220227230827\_O\_01\_C\_1.jpg', 'https://newsimg.hankookilbo.com/2020/05/23/202005230175330276\_1.jpg'),

(6, 'Jisoo', '1995-01-03', 'Lead Vocalist', 'https://talkimg.imbc.com/TVianUpload/tvian/TViews/image/2022/04/30/3e6ad71b-ace2-489e-a139-38156851d0ae.jpg', 'https://cdn.topstarnews.net/news/photo/202104/869240\_604006\_560.jpg'),

(6, 'Jennie', '1996-01-16', 'Main Rapper', 'https://cphoto.asiae.co.kr/listimglink/1/2023041419525789689\_1681469577.jpeg', 'https://spnimage.edaily.co.kr/images/Photo/files/NP/S/2020/09/PS20092300066.jpg'),

(6, 'Rosé', '1997-02-11', 'Main Vocalist', 'https://dimg.donga.com/wps/NEWS/IMAGE/2021/02/10/105377249.2.jpg', 'https://image.ytn.co.kr/general/jpg/2022/0307/202203071005559220\_d.jpg'),

(7, 'Haerin', '2006-05-15', 'Lead Vocalist', 'https://ojsfile.ohmynews.com/STD\_IMG\_FILE/2022/1008/IE003061577\_STD.jpg', 'https://talkimg.imbc.com/TVianUpload/tvian/TViews/image/2022/10/08/b0bb64c1-989a-4ff6-abeb-fd92e81eda16.jpg'),

(7, 'Minji', '2004-05-07', 'Main Rapper', 'https://talkimg.imbc.com/TVianUpload/tvian/TViews/image/2022/08/19/50d2cee1-f43e-4cf7-910e-9151e36b623d.jpg', 'https://file.mk.co.kr/meet/neds/2023/09/image\_readtop\_2023\_673574\_16938843015612457.jpg'),

(7, 'Hanni', '2004-10-06', 'Main Vocalist', 'https://encrypted-tbn0.gstatic.com/images?q=tbn:ANd9GcRhj-mc\_HOvPzZLiaLpeAot\_qP1xxf4\_K6rxg&usqp=CAU', 'https://img.tvreportcdn.de/cms-content/uploads/2023/11/17/4d7c5abc-d9fe-443b-a974-5f7ac444f75a.jpg');

**-- 기존의 뷰 삭제**

DROP VIEW IF EXISTS GroupMembersInfoWithID;

**-- 새로운 뷰 생성**

CREATE VIEW GroupMembersInfoWithID AS

SELECT m.MemberID, g.GroupName, g.DebutDate, g.Company, m.MemberName, m.BirthDate, m.Position, m.FullBodyImageURL, m.FaceImageURL

FROM GirlGroups g

JOIN Members m ON g.GroupID = m.GroupID;

**-- 새로운 뷰 데이터 조회**

SELECT \* FROM GroupMembersInfoWithID;

**5. 결 론**

**개발 내용 및 실험 결과 요약**

주어진 목표에 따라 Android 앱을 개발하여, 사용자들이 걸그룹 멤버 정보를 스크롤하며 확인할 수 있는 편리한 UI를 제공했습니다. 리스트를 통해 멤버들의 간단한 정보를 제공하고, 상세 정보 화면을 통해 더 자세한 정보를 표시하도록 구현하였습니다. 이에 따라 사용자들은 UI를 통해 쉽게 걸그룹 멤버들의 정보를 찾고 탐색할 수 있었습니다.

**향후 개선 과제**

검색 기능 추가: 멤버를 이름이나 소속 그룹 등으로 검색하여 특정 멤버를 빠르게 찾을 수 있는 기능을 추가할 필요가 있습니다.

UI/UX 개선: 사용자 경험을 향상시키기 위해 UI/UX 측면에서의 개선이 필요합니다. 예를 들어, 더 효과적인 네비게이션, 레이아웃, 색상, 애니메이션 등의 추가적인 요소를 고려할 수 있습니다.

외부 데이터 연동: 외부 데이터 소스와의 연동을 통해 실시간으로 업데이트되는 정보를 제공하는 기능을 추가할 수 있습니다.

**기술적, 사회적, 경제적 파급 효과 및 기대 효과**

기술적 효과: MVVM 아키텍처를 적용하여 코드의 모듈화와 유지 보수성을 높이는데 도움이 되었습니다. 안정적이고 효율적인 코드를 유지하는 데 도움이 됩니다.

사회적 효과: K-pop에 대한 글로벌 관심이 높아지고 있으며, 해당 앱을 통해 사용자들은 더 쉽게 걸그룹 멤버들에 대한 정보를 접할 수 있어 사회적 관심을 끌 수 있습니다.

경제적 효과: K-pop 열풍 속에서 관련된 앱들은 사용자들에게 유료로 판매되거나 광고 수익을 얻을 수 있는 경제적 가치를 가질 수 있습니다. 사용자들의 활발한 이용으로 인해 비즈니스적 가치를 창출할 수 있습니다.

**6. 자체 분석과 평가**

- 수행 수준의 자기평가

이 프로젝트는 사용자들에게 걸그룹 멤버들의 정보를 쉽게 탐색하고 확인할 수 있는 플랫폼을 제공합니다. 사용자들은 멤버의 프로필, 이미지, 데뷔 날짜 등 다양한 정보를 확인함으로써 걸그룹에 대한 풍부한 이해와 통찰을 얻을 수 있습니다. 이러한 앱은 K-pop에 입문하는 사용자부터 열정적인 팬까지 다양한 사용자들에게 유익한 정보를 제공하고자 합니다.

현재 이 앱은 기본적인 기능을 구현하여 사용자들이 멤버 리스트를 확인하고 각 멤버의 상세 정보를 조회할 수 있습니다. 하지만 여전히 몇 가지 부분에서 개선의 여지가 있습니다. 예를 들어, 사용자 경험을 더욱 향상시킬 수 있는 추가적인 기능들을 구현하고, UI/UX를 개선하여 사용자들이 더욱 손쉽게 정보를 찾을 수 있도록 할 계획입니다.

앞으로 이 앱은 K-pop에 대한 지식을 쌓고자 하는 사용자들에게 많은 도움을 줄 것으로 예상됩니다. 따라서 이러한 프로젝트를 통해 K-pop 문화에 관심을 가진 사용자들이 보다 편리하게 정보를 습득하고 K-pop의 다양한 매력을 경험할 수 있도록 돕는 데 기여하고자 합니다.

**어려웠던 점과 해결 과정**

프로젝트를 수행하면서 가장 어려웠던 부분은 기존에 사용하던 클라우드 서비스가 유료로 전환되면서 새로운 데이터베이스 서버를 찾아야 했던 점이었습니다. 초기에는 AWS나 Google Cloud와 같은 대형 클라우드 서비스를 찾아 시도하는데 많은 시간을 투자하게 되었습니다. 3일간의 분투 끝에 세계적으로 유명한 클라우드 서비스를 사용하는 데 성공했지만, 데이터베이스에서 직접 데이터를 불러오는 것이 보안과 안정성 문제가 있다는 것을 알게 되었습니다.

이에 따라, 기존에 사용한 방식인 데이터베이스 정보를 JSON 형식으로 웹 페이지에 노출시키고, 애플리케이션이 해당 웹페이지의 JSON 데이터를 불러오는 방법을 사용해야 하는 점을 깨달았습니다. 그러나 데이터베이스와 서버를 별도로 구성할 경우 예상치 못한 추가 서버 비용 문제가 발생할 수 있다는 것을 알게 되었습니다.

이를 극복하기 위해, 기존에 사용하던 CloudType에서 프리 티어를 활용하여 백엔드 서버를 구성하고자 했습니다. 이를 위해 CloudType에 신용카드를 등록하고, 프리 티어를 사용하여 백엔드를 구축하는 방법을 선택했습니다. 이를 통해 비용 문제를 예상 가능한 수준으로 낮추고 프로젝트를 성공적으로 진행할 수 있었습니다.

**7. 부록**

https://github.com/26solitude/SmartAppHomeWork

**참고문헌**

https://nano5.notion.site/Week09-Database-4f95cf4133cf4375bdb447d0bf248291?pvs=4

https://nano5.notion.site/Week10-Backend-300c043e6349429592e3a1d5a790a88a?pvs=4

https://nano5.notion.site/Week13-01-Song-List-53472b09f76f46cd9d3805ec52e9b6bf?pvs=4

https://nano5.notion.site/Week13-02-AsyncImage-556f54b34dbd43df8105331c518eaa43?pvs=4

https://nano5.notion.site/Week14-02-Song-Detail-Screen-c12d837638a14b3094e3b82be24d01a0?pvs=4