



감정을 음악으로 기록하자

# Musiary

2023145030 박다현  
2023145058 이나윤

감정을 음악으로 기록하자

# Musiary

바쁜 일상 속 감정이 묻히기 쉬운 현대인들이,  
짧은 일기 몇 줄로 자신의 감정을 돌아보고 위로받을 수 있도록 하기위해 제작된 어플



## 01 프로젝트 개요

Musiary 앱의 목적과 전체적인 소개

## 02 기획 배경

기존 앱들과의 차별점 및 기획 의도

## 03 서비스 주요 기능

사용자가 실제로 경험하게 될 핵심 기능 설명

## 04 앱 구조 및 핵심 코드 소개

화면 구성과 함께 주요 화면의 구현 코드 예시

## 05 기술 스택 및 사용 도구

개발 언어, 프레임워크,  
외부 라이브러리, 데이터베이스 등

## 06 기술적 구현 포인트

주요 기술 구현 방법

## 07 문제 해결 및 개선 과정

개발 중 겪은 문제와 그 해결 방법, 리팩토링 사례

## 08 결과 및 시연

실제 앱 실행 화면 시연 및 기능 소개 영상

## 09 향후 계획

추가 개발 예정 기능 및 서비스 확장 방향

ios 프로그래밍

## 01. Project Overview

# 프로젝트 개요

Musiary에 대한 소개



로고

Music + Diary라는 뜻을 가지고있는 Musiary  
감정을 음악으로 기록하는 일기장을 나타내는 로고



주요 타겟 사용자

정신적 힐링이 필요한 10대 ~ 30대



앱 개발 동기

바쁜 일상 속 감정이 묻히기 쉬운 현대인들이,  
짧은 일기 몇 줄로 자신의 감정을 돌아보고  
위로받을 수 있도록 하기위해 제작된 어플



핵심 목표

- 일기 기반 감정 키워드 분석으로 감정을 파악하고 음악 추천
- 단순 플레이리스트가 아닌, '감정 맞춤형 위로 메세지 + 음악' 제공
- 일기 저장 및 감정 흐름 시각화

ios 프로그래밍

## 02. Planning background

# 기획 배경

차별점

일기 기반 감정 키워드 분석으로 감정을 파악하고 음악 추천

spottify



감정 기반 추천 제공 일부 있지만  
감정분석은 없고, 일기 기반 아님

Apple Music



사용자 취향 기반 추천은 있지만  
감정, 기분 반영이 안 되어 있음

MOODA



감정 일기, 친구와 함께 쓰기, 메모 및 사진  
등 기록 남기기 등 기능이 있지만  
음악 추천 기능 없음

## 05. Technical stack and usage tools

# 기술 스택 및 사용 도구

## 개발 언어 및 플랫폼



Xcode(ios 기반 개발)

Swift

## UI 구현 도구

SwiftUI 일부 + UIKit 기반 ViewController 구성

AutoLayout 및 직접 프레임 설정 병행

## 데이터 저장

Firebase Firestore (일기 저장, 감정 태그 등 사용자 데이터 저장)

UserDefaults (임시 데이터 및 설정 정보 저장)

## 추후 계획된 연동 및 고도화 기능

Apple Music API (감정 기반 음악 추천 자동화)

푸시 알림 시스템 연동 (알림 시간 설정 기능 강화)

### 03. Service Key Features

## 서비스 주요 기능

2025.06.09 까지 구현된 기능들입니다.  
이후 기능이 추가될 예정입니다.

로그인 및 회원관리

일기 작성 및  
자동 임시 저장

기분 태그 선택 팝업

오늘의 추천  
문구 출력

감성 UI 및  
줄노트 텍스트뷰

설정 기능  
(비밀번호 변경,  
알림 시간 설정, 앱 잠금)



ios 프로그래밍

## 04. Introduce app structure and core code

# 앱 구조 및 핵심 코드 소개



## 회원가입 코드

```

5 import UIKit
6 import FirebaseAuth //Firebase 인증 사용
7
8
9 class SignUpViewController: UIViewController {
10
11    //필수항
12    @IBOutlet var emailTextField: UITextField!
13    @IBOutlet var passwordTextField: UITextField!
14    @IBOutlet var passwordCheckTextField: UITextField!
15
16    //오류 메시지
17    @IBOutlet var emailErrorLabel: UILabel!
18    @IBOutlet var passwordErrorLabel: UILabel!
19    @IBOutlet var passwordCheckErrorLabel: UILabel!
20
21    override func viewDidLoad() {
22        super.viewDidLoad()
23
24        //초기 상태: 여러 문구 숨김
25        emailErrorLabel.isHidden = true
26        passwordErrorLabel.isHidden = true
27        passwordCheckErrorLabel.isHidden = true
28
29
30        //placeholder 설정
31        emailTextField.placeholder = "이메일을 입력하세요."
32        passwordTextField.placeholder = "비밀번호를 입력하세요."
33        passwordCheckTextField.placeholder = "비밀번호를 다시 입력하세요."
34
35
36        //회원가입 버튼
37        @IBAction func signUpTapped(_ sender: UIButton) {
38            //여러 문구 초기화
39            emailErrorLabel.isHidden = true
40            passwordErrorLabel.isHidden = true
41            passwordCheckErrorLabel.isHidden = true
42
43            //이메일 유효성 검사
44            guard let email = emailTextField.text, !email.isEmpty else {
45                emailErrorLabel.text = "이메일을 입력해주세요."
46                emailErrorLabel.isHidden = false
47                return
48            }
49
50            let emailRegex = "[A-Z0-9a-z._%+-]+@[A-Za-z0-9.-]+\.[A-Za-z]{2,}"
51            let isEmailValid = NSPredicate(format: "SELF MATCHES %@", emailRegex).evaluate(with: email)
52
53            if !isEmailValid {
54                emailErrorLabel.text = "올바르지 않은 이메일 형식입니다."
55                emailErrorLabel.isHidden = false
56                return
57            }
58
59            // 비밀번호 유효성 검사
60            guard let password = passwordTextField.text, !password.isEmpty else {
61                passwordErrorLabel.text = "비밀번호가 입력되지 않았습니다."
62                passwordErrorLabel.isHidden = false
63                return
64            }
65
66            guard let passwordCheck = passwordCheckTextField.text, passwordCheck == password else {
67                passwordCheckErrorLabel.text = "비밀번호가 일치하지 않습니다."
68                passwordCheckErrorLabel.isHidden = false
69                return
70            }
71
72            //Firebase의 계정 생성 요청
73            Auth.auth().createUser(withEmail: email, password: password) { authResult, error in
74                if let error = error {
75                    //에러 처리 시
76                    let alert = UIAlertController(title: "회원가입 실패", message: error.localizedDescription, preferredStyle: .alert)
77                    alert.addAction(UIAlertAction(title: "확인", style: .default))
78                    self.present(alert, animated: true)
79                    return
80                }
81
82                //성공 시 + 로그인 화면으로 전환
83                let alert = UIAlertController(title: "회원가입 완료", message: "로그인 화면으로 이동합니다.", preferredStyle: .alert)
84                alert.addAction(UIAlertAction(title: "확인", style: .default)) { _ in
85                    if let loginVC = self.storyboard?.instantiateViewController(withIdentifier: "LoginView") as? LoginViewController {
86                        (UIApplication.shared.connectedScenes.first?.delegate as? SceneDelegate)?.changeRootViewController(loginVC, animated: true)
87                    }
88                }
89                self.present(alert, animated: true)
90            }
91        }
92    }
}

```

## 04. Introduce app structure and core code

# 앱 구조 및 핵심 코드 소개



## 로그인 코드

```

5 import UIKit
6 import FirebaseAuth
7
8 class LoginViewController: UIViewController {
9
10    @IBOutlet var emailTextField: UITextField!
11    @IBOutlet var passwordTextField: UITextField!
12
13    override func viewDidLoad() {
14        super.viewDidLoad()
15
16        //placeholder 설정
17        emailTextField.placeholder = "이메일을 입력하세요."
18        passwordTextField.placeholder = "비밀번호를 입력하세요."
19    }
20
21    //로그인 버튼
22    @IBAction func goLogin(_ sender: Any) {
23        //입력값 검사
24        guard let email = emailTextField.text, !email.isEmpty,
25              let password = passwordTextField.text, !password.isEmpty else {
26            let alert = UIAlertController(title: "오류", message: "이메일과 비밀번호를 모두 입력해주세요.", preferredStyle: .alert)
27            alert.addAction(UIAlertAction(title: "확인", style: .default))
28            present(alert, animated: true)
29            return
30        }
31
32        //Firebase 로그인 시도
33        Auth.auth().signIn(withEmail: email, password: password) { authResult, error in
34            if let error = error {
35                // 실패 시 경고
36                let alert = UIAlertController(title: "로그인 실패", message: error.localizedDescription, preferredStyle: .alert)
37                alert.addAction(UIAlertAction(title: "확인", style: .default))
38                self.present(alert, animated: true)
39                return
40            }
41
42            //로그인 성공 시: 로그인 상태 저장
43            UserDefaults.standard.set(true, forKey: "isLoggedIn")
44
45            //루트 뷰 전환
46            if let sceneDelegate = UIApplication.shared.connectedScenes.first?.delegate as? SceneDelegate {
47                let storyboard = UIStoryboard(name: "Main", bundle: nil)
48                if let mainVC = storyboard.instantiateViewController(withIdentifier: "MainViewController") as? MainViewController {
49                    sceneDelegate.changeRootViewController(mainVC, animated: true)
50                }
51            }
52        }
53    }
54
55    // 회원가입 이동
56    @IBAction func goToSignUp(_ sender: UIButton) {
57        guard let signUpVC = storyboard?.instantiateViewController(withIdentifier: "SignUpView") as? SignUpViewController else { return }
58        present(signUpVC, animated: true)
59    }
60}
61

```

## 04. Introduce app structure and core code

## 앱 구조 및 핵심 코드 소개



## 메인 화면 핵심 코드

```

36 // MARK: - 메뉴바 구성 함수
37 func configureTabBar() {
38     writeDiaryImageView.image = UIImage(named: "write")
39     writeDiaryImageView.isUserInteractionEnabled = true
40     writeDiaryImageView.addGestureRecognizer(UITapGestureRecognizer(target: self, action: #selector(showDiaryPopup)))
41
42     homeImageView.image = UIImage(named: "home")
43     homeImageView.isUserInteractionEnabled = true
44     homeImageView.addGestureRecognizer(UITapGestureRecognizer(target: self, action: #selector(goToHome)))
45
46     calendarImageView.image = UIImage(named: "calendar")
47     calendarImageView.isUserInteractionEnabled = true
48     calendarImageView.addGestureRecognizer(UITapGestureRecognizer(target: self, action: #selector(goToCalendar)))
49
50     settingsImageView.image = UIImage(named: "settings")
51     settingsImageView.isUserInteractionEnabled = true
52     settingsImageView.addGestureRecognizer(UITapGestureRecognizer(target: self, action: #selector(goToSettings)))
53 }
54
55 @objc func showDiaryPopup() {
56     guard let vc = storyboard?.instantiateViewController(withIdentifier: "DiaryViewController") as? DiaryViewController else { return }
57     vc.modalPresentationStyle = .pageSheet
58     present(vc, animated: true)
59 }
60
61 @objc func goToHome() {
62     guard let vc = storyboard?.instantiateViewController(withIdentifier: "MainViewController") as? MainViewController else { return }
63     vc.modalPresentationStyle = .fullScreen
64     present(vc, animated: false)
65 }
66
67 @objc func goToCalendar() {
68     guard let vc = storyboard?.instantiateViewController(withIdentifier: "CalendarViewController") as? CalendarViewController else { return }
69     vc.modalPresentationStyle = .fullScreen
70     present(vc, animated: false)
71 }
72
73 @objc func goToSettings() {
74     guard let vc = storyboard?.instantiateViewController(withIdentifier: "SettingsViewController") as? SettingsViewController else { return }
75     vc.modalPresentationStyle = .fullScreen
76     present(vc, animated: false)
77 }

```

## 04. Introduce app structure and core code

## 앱 구조 및 핵심 코드 소개



## 메인 화면 핵심 코드

```

79 // MARK: - 기본 선택
80 @objc func showFeelingSelector() {
81     guard let vc = storyboard?.instantiateViewController(withIdentifier: "FeelingSelectView") as? FeelingSelectViewController else { return }
82     vc.modalPresentationStyle = .pageSheet
83     vc.delegate = self
84     present(vc, animated: true)
85 }
86
87 func didSelectFeeling(_ feeling: String) {
88     feelingLabel.text = feeling
89     showQuote(from: [feeling])
90 }
91
92 // MARK: - 추천 명언
93
94 let quoteDatabase: [String: [String]] = [
95     "설렘": ["두근거림은 살아있다는 증거야.", "작은 설렘이 하루를 바꿔줄 수 있어."],
96     "기쁨": ["기쁨은 나누면 배가 돼!", "웃는 네가 제일 예뻐!"],
97     "행복": ["행복은 지금 이 순간에도 널 기다려.", "너는 이미 많은 걸 가지고 있어."],
98     "그리움": ["그리움은 사랑의 또 다른 이름이야.", "그때의 네가 지금도 소중해."],
99     "편안": ["오늘은 그냥 숨 쉬는 것도 잘한 거야.", "편안함도 너의 능력이야."],
100    "우울": ["괜찮아, 울어도 돼.", "오늘 하루 버틴 너는 충분히 대단해."],
101    "비참": ["지금 느끼는 감정도 네 일부야.", "비참함 속에서도 빛나는 너가 있어."],
102    "지침": ["쉬어도 괜찮아. 네 몸이 하는 말이야.", "지쳤을 땐 스스로를 안아줘."],
103    "화남": ["화난 너도 괜찮아.", "감정은 느낄 수 있는 거니까 소중해."]
104 ]
105
106 func showQuote(from selectedTags: [String]) {
107     var quotes: [String] = []
108     for tag in selectedTags {
109         if let messages = quoteDatabase[tag] {
110             quotes += messages
111         }
112     }
113     quotesLabel.text = quotes.randomElement() ?? "오늘 하루도 수고했어요 ❤"
114 }

```

## 04. Introduce app structure and core code

## 앱 구조 및 핵심 코드 소개



## 다이어리 화면 핵심 코드

```

① @IBAction func saveButtonTapped(_ sender: UIButton) {
53     guard let uid = Auth.auth().currentUser?.uid else { return }
54     let text = diaryTextView.text ?? ""
55
56     //플레이스홀더 상태면 저장 안 함
57     if isPlaceholderVisible { return }
58
59     let dateFormatter = DateFormatter()
60     dateFormatter.dateFormat = "yyyy-MM-dd"
61     let today = dateFormatter.string(from: Date())
62
63     let db = Firestore.firestore()
64     db.collection("diaries").document(uid).collection("entries").document(today).setData([
65         "text": text,
66         "timestamp": Timestamp(date: Date())
67     ]) { error in
68         if let error = error {
69             print("저장 실패: \(error.localizedDescription)")
70         } else {
71             print("저장 성공!")
72             UserDefaults.standard.removeObject(forKey: "diaryDraft")
73             UserDefaults.standard.set(true, forKey: "diarySaved")
74             self.dismiss(animated: true)
75         }
76     }
77 }
78
79 //UITextViewDelegate 따로 확장
80 extension DiaryViewController: UITextViewDelegate {
81
82     func textViewDidBeginEditing(_ textView: UITextView) {
83         if isPlaceholderVisible {
84             textView.text = ""
85             textView.textColor = .black
86             isPlaceholderVisible = false
87         }
88     }
89
90     func textViewDidEndEditing(_ textView: UITextView) {
91         if textView.text.trimmingCharacters(in: .whitespacesAndNewlines).isEmpty {
92             textView.text = placeholderText
93             textView.textColor = .lightGray
94             isPlaceholderVisible = true
95         }
96     }
97
98     func textViewDidChange(_ textView: UITextView) {
99         saveDraft(textView.text)
100    }
101 }
102 }
```

## 04. Introduce app structure and core code

## 앱 구조 및 핵심 코드 소개



## 다이어리 화면 핵심 코드

```

① @IBAction func saveButtonTapped(_ sender: UIButton) {
53     guard let uid = Auth.auth().currentUser?.uid else { return }
54     let text = diaryTextView.text ?? ""
55
56     //플레이스홀더 상태면 저장 안 함
57     if isPlaceholderVisible { return }
58
59     let dateFormatter = DateFormatter()
60     dateFormatter.dateFormat = "yyyy-MM-dd"
61     let today = dateFormatter.string(from: Date())
62
63     let db = Firestore.firestore()
64     db.collection("diaries").document(uid).collection("entries").document(today).setData([
65         "text": text,
66         "timestamp": Timestamp(date: Date())
67     ]) { error in
68         if let error = error {
69             print("저장 실패: \(error.localizedDescription)")
70         } else {
71             print("저장 성공!")
72             UserDefaults.standard.removeObject(forKey: "diaryDraft")
73             UserDefaults.standard.set(true, forKey: "diarySaved")
74             self.dismiss(animated: true)
75         }
76     }
77 }
78
79 //UITextViewDelegate 따로 확장
80 extension DiaryViewController: UITextViewDelegate {
81
82     func textViewDidBeginEditing(_ textView: UITextView) {
83         if isPlaceholderVisible {
84             textView.text = ""
85             textView.textColor = .black
86             isPlaceholderVisible = false
87         }
88     }
89
90     func textViewDidEndEditing(_ textView: UITextView) {
91         if textView.text.trimmingCharacters(in: .whitespacesAndNewlines).isEmpty {
92             textView.text = placeholderText
93             textView.textColor = .lightGray
94             isPlaceholderVisible = true
95         }
96     }
97
98     func textViewDidChange(_ textView: UITextView) {
99         saveDraft(textView.text)
100    }
101 }
102 }
```

## 04. Introduce app structure and core code

## 앱 구조 및 핵심 코드 소개



## 달력 화면 핵심 코드

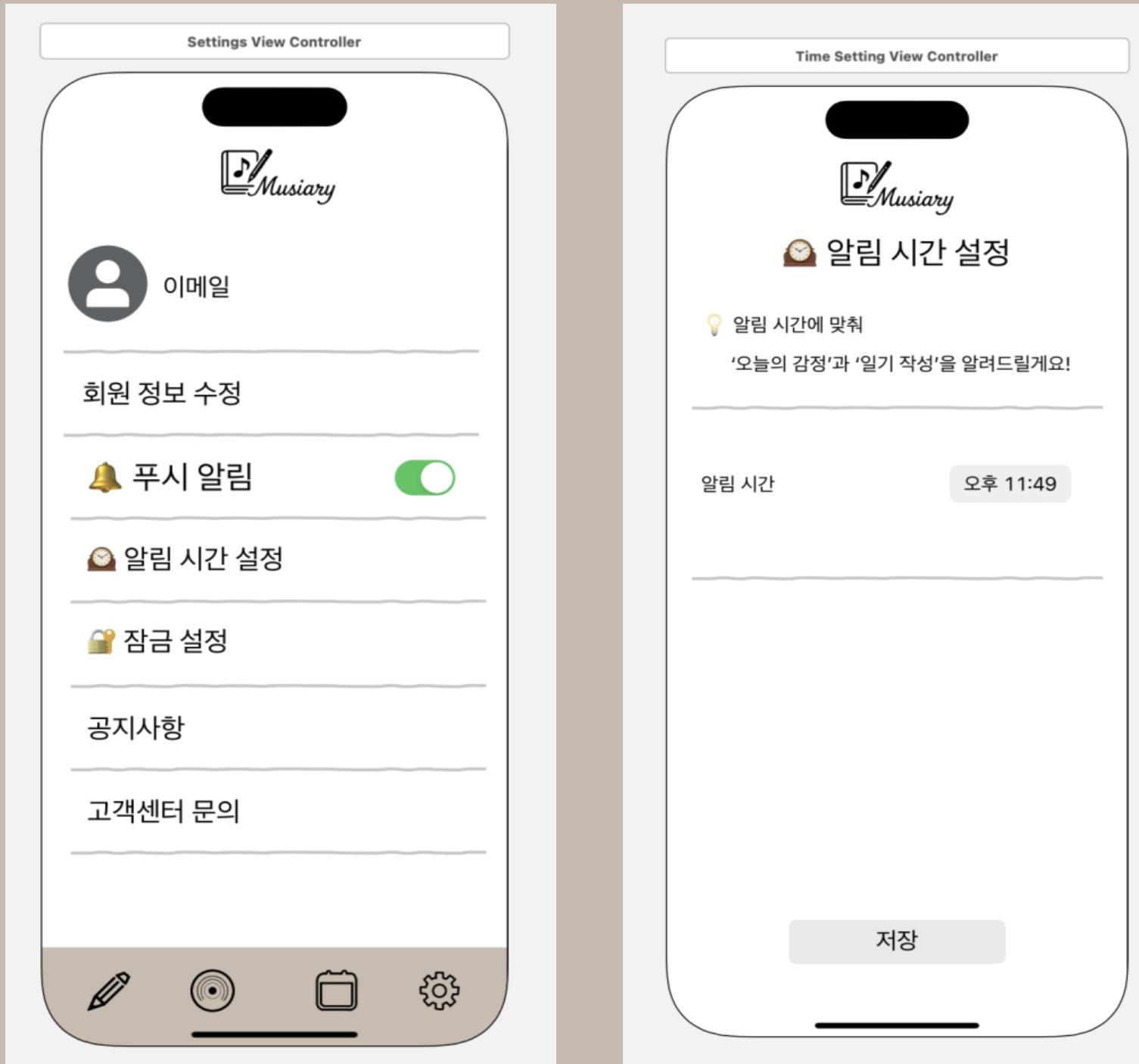
```

31 // MARK: - 달력 관련
32 func updateMonth() {
33     let formatter = DateFormatter()
34     formatter.dateFormat = "M월"
35     monthLabel.text = formatter.string(from: currentDate)
36
37     daysInMonth = generateDays(for: currentDate)
38     collectionView.reloadData()
39 }
40
41 func generateDays(for date: Date) -> [String] {
42     var result: [String] = []
43
44     let calendar = Calendar.current
45     let range = calendar.range(of: .day, in: .month, for: date)!
46     let firstDay = calendar.date(from: calendar.dateComponents([.year, .month], from: date))!
47     let weekday = calendar.component(.weekday, from: firstDay)
48
49     // 공백 추가 (1일 전까지)
50     result += Array(repeating: "", count: weekday - 1)
51     result += range.map { String($0) }
52
53     return result
54 }
55
56 @IBAction func prevMonth(_ sender: UIButton) {
57     currentDate = Calendar.current.date(byAdding: .month, value: -1, to: currentDate)!
58     updateMonth()
59 }
60
61 @IBAction func nextMonth(_ sender: UIButton) {
62     currentDate = Calendar.current.date(byAdding: .month, value: 1, to: currentDate)!
63     updateMonth()
64 }
65
66 // MARK: - CollectionView DataSource
67 func collectionView(_ collectionView: UICollectionView, numberOfItemsInSection section: Int) -> Int {
68     return daysInMonth.count
69 }
70
71 func collectionView(_ collectionView: UICollectionView, cellForItemAt indexPath: IndexPath) -> UICollectionViewCell {
72     let cell = collectionView.dequeueReusableCell(withIdentifier: "dayCell", for: indexPath)
73     if let label = cell.viewWithTag(1) as? UILabel {
74         label.text = daysInMonth[indexPath.item]
75     }
76     return cell
77 }

```

## 04. Introduce app structure and core code

## 앱 구조 및 핵심 코드 소개



## 설정 화면 핵심 코드

```

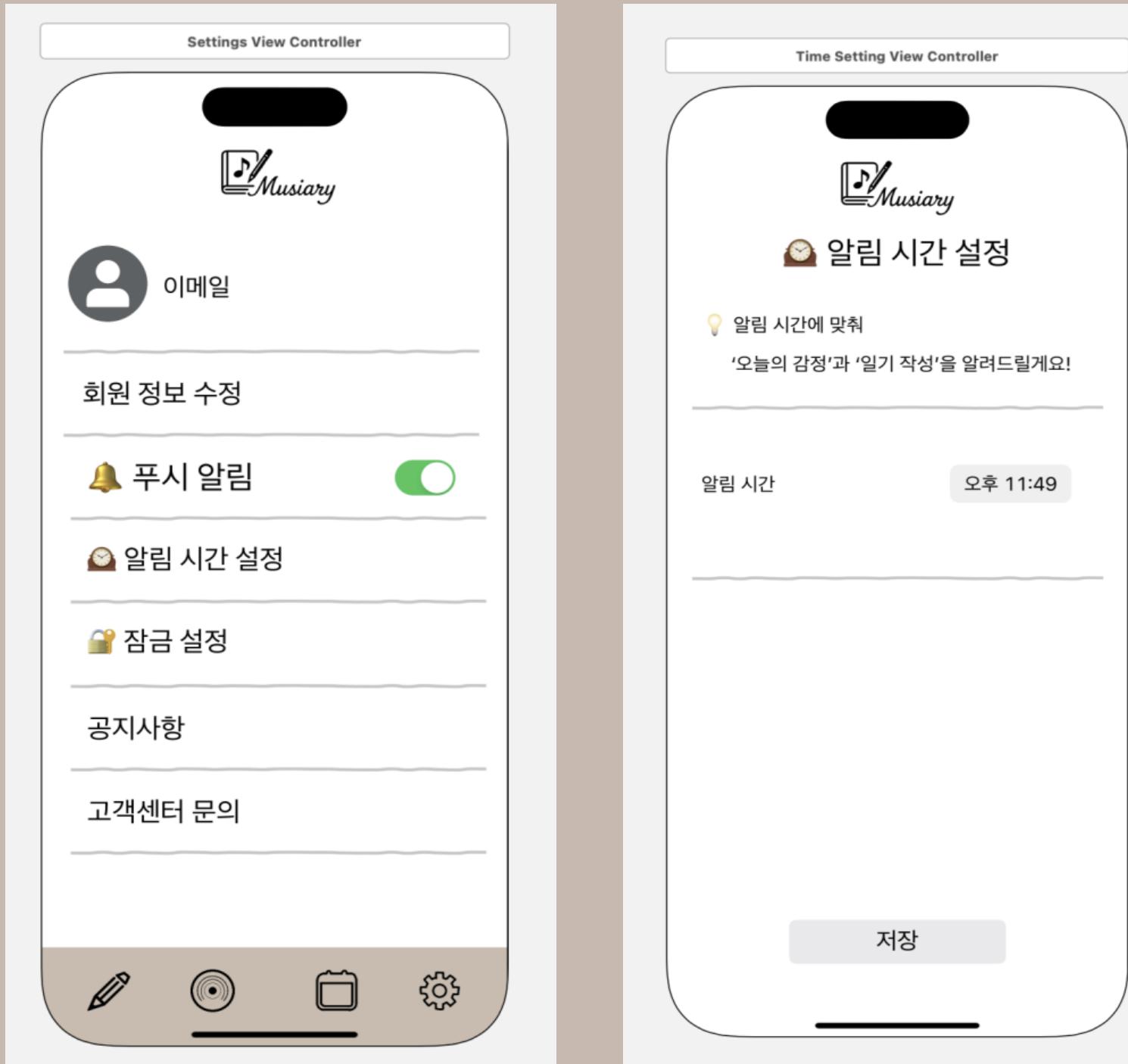
33 // MARK: - 푸시 알림 스위치 변경
34 @IBAction func pushSwitchChanged(_ sender: UISwitch) {
35     UserDefaults.standard.set(sender.isOn, forKey: "pushEnabled")
36
37     if sender.isOn {
38         UNUserNotificationCenter.current().requestAuthorization(options: [.alert, .sound, .badge]) { granted, _ in
39             print(granted ? "🔔 알림 허용됨" : "🔕 알림 거부됨")
40         }
41     } else {
42         print("알림 비활성화됨")
43     }
44 }

45 // MARK: - 팝업 메뉴 항목
46 @IBAction func openPasswordChange(_ sender: UIButton) {
47     presentPopup("PasswordAuthViewController")
48 }
49
50 @IBAction func openTimeSetting(_ sender: UIButton) {
51     presentPopup("TimeSettingViewController")
52 }
53
54 @IBAction func openAppLockSetting(_ sender: UIButton) {
55     presentPopup("AppLockViewController")
56 }
57
58 func presentPopup(_ identifier: String) {
59     guard let vc = storyboard?.instantiateViewController(withIdentifier: identifier) else { return }
60     vc.modalPresentationStyle = .pageSheet
61     present(vc, animated: true)
62 }
63

```

## 04. Introduce app structure and core code

## 앱 구조 및 핵심 코드 소개



## 알림 시간 설정 화면 핵심 코드

```

17 // 이전에 저장된 알림 시간 불러오기
18 if let savedDate = UserDefaults.standard.object(forKey: "alertTime") as? Date {
19     timePicker.date = savedDate
20 }
21
22 @IBAction func saveButtonTapped(_ sender: UIButton) {
23     let isPushEnabled = UserDefaults.standard.bool(forKey: "pushEnabled")
24     let selectedTime = timePicker.date
25
26     // 푸시 알림이 꺼져 있을 경우 경고
27     if !isPushEnabled {
28         showAlert(message: "⚠️ 푸시 알림이 꺼져있어 알림이 올리지 않아요!")
29         return
30     }
31
32     // 알림 시간 저장
33     UserDefaults.standard.set(selectedTime, forKey: "alertTime")
34
35     // 알림 예약
36     scheduleNotification(at: selectedTime)
37     dismiss(animated: true)
38 }
39
40
41 func scheduleNotification(at date: Date) {
42     let content = UNMutableNotificationContent()
43     content.title = "📝 오늘의 감정과 일기 작성하셨나요?"
44     content.body = "오늘 하루를 기록해보세요!"
45     content.sound = .default
46
47     var dateComponents = Calendar.current.dateComponents([.hour, .minute], from: date)
48     dateComponents.second = 0
49
50     let trigger = UNCalendarNotificationTrigger(dateMatching: dateComponents, repeats: true)
51     let request = UNNotificationRequest(identifier: "dailyReminder", content: content, trigger: trigger)
52
53     UNUserNotificationCenter.current().add(request) { error in
54         if let error = error {
55             print("🔴 알림 등록 실패: \(error.localizedDescription)")
56         } else {
57             print("✅ 알림 등록 완료!")
58         }
59     }
60 }

```

## 04. Introduce app structure and core code

## 앱 구조 및 핵심 코드 소개



## 현재 비밀번호 확인 화면 핵심 코드

```

1 // PasswordAuthViewController.swift
2 // Musiary
3 // Created by 박다현
4
5
6 import UIKit
7 import FirebaseAuth
8
9 class PasswordAuthViewController: UIViewController {
10
11     @IBOutlet weak var passwordTextField: UITextField!
12     @IBOutlet weak var errorLabel: UILabel!
13     @IBOutlet weak var confirmPasswordErrorLabel: UILabel!
14
15     override func viewDidLoad() {
16         super.viewDidLoad()
17         errorLabel.text = ""
18     }
19
20     @IBAction func confirmButtonTapped(_ sender: UIButton) {
21         guard let password = passwordTextField.text,
22               let email = Auth.auth().currentUser?.email else { return }
23
24         let credential = EmailAuthProvider.credential(withEmail: email, password: password)
25
26         Auth.auth().currentUser?.reauthenticate(with: credential) { result, error in
27             if let error = error {
28                 self.errorLabel.text = "X 비밀번호가 틀렸습니다."
29                 print("Reauthentication failed: \(error.localizedDescription)")
30             } else {
31                 // 비밀번호가 맞으면 비밀번호 변경 화면으로 전환
32                 guard let vc = self.storyboard?.instantiateViewController(withIdentifier: "PasswordChangeViewController") else { return }
33                 vc.modalPresentationStyle = .pageSheet
34                 self.present(vc, animated: true)
35             }
36         }
37     }
38 }
```

## 04. Introduce app structure and core code

## 앱 구조 및 핵심 코드 소개



## 회원 정보 수정 화면 핵심 코드

```

23 // MARK: - 저장 버튼 클릭
24 @IBAction func saveButtonTapped(_ sender: UIButton) {
25     let newPassword = newPasswordTextField.text ?? ""
26     let confirmPassword = confirmPasswordTextField.text ?? ""
27
28     var hasError = false
29
30     // 비밀번호 입력 확인
31     if newPassword.isEmpty {
32         newPasswordErrorLabel.text = "비밀번호가 입력되지 않았습니다."
33         newPasswordErrorLabel.isHidden = false
34         hasError = true
35     } else {
36         newPasswordErrorLabel.isHidden = true
37     }
38
39     // 비밀번호 일치 확인
40     if newPassword != confirmPassword {
41         confirmPasswordErrorLabel.text = "입력한 비밀번호가 일치하지 않습니다."
42         confirmPasswordErrorLabel.isHidden = false
43         hasError = true
44     } else {
45         confirmPasswordErrorLabel.isHidden = true
46     }
47
48     if hasError { return }
49
50     // 비밀번호 길이 검사
51     if newPassword.count < 6 {
52         showAlert(message: "비밀번호는 최소 6자 이상이어야 합니다.")
53         return
54     }
55
56     // 파이어베이스 비밀번호 변경
57     Auth.auth().currentUser?.updatePassword(to: newPassword) { error in
58         if let error = error {
59             print("❌ 변경 실패: \(error.localizedDescription)")
60             self.showAlert(message: "비밀번호 변경에 실패했습니다.")
61         } else {
62             self.showAlert(message: "비밀번호가 변경되었습니다.") {
63                 self.dismiss(animated: true)
64             }
65         }
66     }
67 }

```

## 04. Introduce app structure and core code

## 앱 구조 및 핵심 코드 소개



## 잠금 설정 화면 핵심 코드

```

5 import UIKit
6
7 class AppLockViewController: UIViewController {
8
9     @IBOutlet weak var passwordTextField: UITextField!
10    @IBOutlet weak var lockSwitch: UISwitch!
11
12    override func viewDidLoad() {
13        super.viewDidLoad()
14
15        // 저장된 상태 불러오기
16        lockSwitch.isOn = UserDefaults.standard.bool(forKey: "appLockEnabled")
17        passwordTextField.text = UserDefaults.standard.string(forKey: "appLockPassword")
18        passwordTextField.keyboardType = .numberPad
19    }
20
21    @IBAction func saveButtonTapped(_ sender: UIButton) {
22        let isLockEnabled = lockSwitch.isOn
23        let password = passwordTextField.text ?? ""
24
25        if isLockEnabled {
26            // 잠금 설정 O + 비번 미입력
27            if password.isEmpty {
28                showAlert("비밀번호가 입력되지 않았습니다.")
29                return
30            }
31            if password.count != 4 || Int(password) == nil {
32                showAlert("숫자 4자리로 입력해주세요.")
33                return
34            }
35
36            UserDefaults.standard.set(true, forKey: "appLockEnabled")
37            UserDefaults.standard.set(password, forKey: "appLockPassword")
38        } else {
39            UserDefaults.standard.set(false, forKey: "appLockEnabled")
40        }
41
42        dismiss(animated: true)
43    }
44
45    func showAlert(_ message: String) {
46        let alert = UIAlertController(title: "오류", message: message, preferredStyle: .alert)
47        alert.addAction(UIAlertAction(title: "확인", style: .default))
48        present(alert, animated: true)
49    }
50 }

```

## 04. Introduce app structure and core code

## 앱 구조 및 핵심 코드 소개



## 잠금 화면 핵심 코드

```

23 // **** 업데이트
24 func updatePasswordDots() {
25     let labels = [passwordLabel1, passwordLabel2, passwordLabel3, passwordLabel4]
26     for i in 0..<4 {
27         labels[i]!.text = i < input.count ? "●" : ""
28     }
29 }
30
31 // 숫자 버튼 (0~9)
32 @IBAction func numberButtonTapped(_ sender: UIButton) {
33     guard let digit = sender.titleLabel?.text, input.count < 4 else { return }
34     input += digit
35     updatePasswordDots()
36     errorLabel.isHidden = true // 오류 숨김
37
38     if input.count == 4 {
39         checkPassword()
40     }
41 }
42
43 // 삭제 버튼: 마지막 문자 삭제
44 @IBAction func deleteTapped(_ sender: UIButton) {
45     guard !input.isEmpty else { return }
46     input.removeLast()
47     updatePasswordDots()
48     errorLabel.isHidden = true
49 }
50
51 // 취소 버튼: 전체 초기화
52 @IBAction func cancelTapped(_ sender: UIButton) {
53     input = ""
54     updatePasswordDots()
55     errorLabel.isHidden = true
56 }
57
58 // 비밀번호 검증
59 func checkPassword() {
60     let savedPassword = UserDefaults.standard.string(forKey: "appLockPassword") ?? ""
61
62     if input == savedPassword {
63         dismiss(animated: true) // ✅ 성공
64     } else {
65         input = ""
66         updatePasswordDots()
67         errorLabel.text = "✗ 비밀번호가 틀렸습니다"
68         errorLabel.isHidden = false
69     }
70 }
71

```

06. Technical implementation point

## 기술적 구현 포인트

### 로그인 및 자동 로그인 기능

Firebase Authentication 기반 이메일 로그인

앱 재실행 시 자동 로그인 유지

### 메인 페이지 구성

미니 달력 UI

오늘의 감정 키워드, 명언 표시

### 일기 작성 기능

일기 입력 창 팝업 (pageSheet) 형태로 구현

일기 저장 및 Firebase Firestore 연동

입력 중인 일기 임시 저장 기능 (UserDefaults)



## 06. Technical implementation point

# 기술적 구현 포인트



## 기분 선택 기능

9가지 감정 태그 중 선택

선택된 감정이 '오늘의 기분' 박스에 반영

## 앱 설정 기능

푸시 알림 ON/OFF 스위치

알림 시간 설정 팝업

앱 잠금 설정 및 비밀번호 변경 기능

로그인된 이메일 표시 및 로그아웃

## 하단 메뉴바 기능

홈, 달력, 설정, 일기작성 버튼 구성

현재 위치한 화면의 아이콘 상태 반영

전환 방식: 전체 화면 전환 or 팝업 선택

## 07. Problem solving and improvement process

## 문제 해결 및 개선 과정

**스토리보드 연결**

"Storyboard doesn't contain a view controller with identifier" 오류 발생

→ 각 ViewController에 Storyboard ID를 명확하게 지정하여 해결

**UICollectionView 오류**

"must register a nib or a class for the identifier" 오류 발생

→ 스토리보드 내에서 셀에 Identifier (dayCell) 부여하여 해결

**자동 로그인 및 로그아웃 문제**

로그인 유지 안됨 → UserDefaults에 로그인 상태 저장 및 SceneDelegate로 처리

로그아웃 시 루트 뷰 전환 → SceneDelegate의 changeRootViewController 활용

**반응형 UI 문제**

실제 기기에서 비율 깨짐 발생

→ StackView 및 Constraint 조정으로 해결

아직 해결되지 않은 문제점

달력 UI 관련 문제: 일부 날짜 정렬이나 셀 배치가 원하는 디자인과 다르게 표시됨

잠금화면 연결 오류: 앱 실행 시 잠금화면이 정상적으로 호출되지 않는 오류가 발생 중

08. Results and demonstration

# 결과 및 시연

09. Future plan

## 향후 계획



### 인공지능(AI) 기반 감정 분석 기능 도입

사용자가 작성한 일기 내용을 AI가 분석하여 감정 키워드를 추출

감정에 맞는 음악을 자동으로 추천하는 시스템 개발

단순한 감정 태그 선택이 아닌, 텍스트 기반 감정 인식 → 음악 추천으로 기능 고도화 예정

ios 프로그래밍



감정을 음악으로 기록하자

# Musiary

2023145030 박다현  
2023145058 이나윤