



## 김 건 형

(Kim Geon Hyeong)  
iOS Developer

1994.04.07

아주대학교 소프트웨어학과 졸업  
Ajou University  
Computer Science and Engineering

geonhyeong.dev@gmail.com

010-9866-6577

<https://github.com/GeonHyeongKim>

## Skill Stack

이미지/영상, 실시간 채팅



## Keyword



경험 (인턴 2, 대회 3, 팀 프로젝트 3)



노력



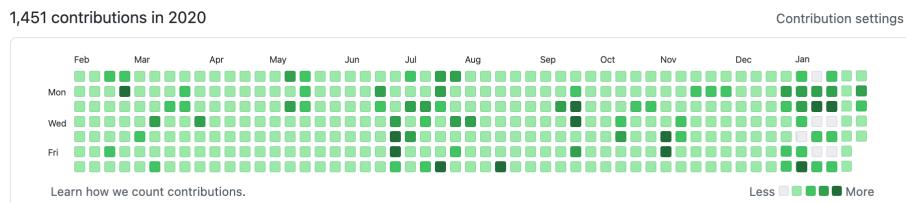
협업



친화력



리더십



## Activities

- 2022 (활동중) 멋쟁이 사자처럼 앱 스쿨 1기
- 2022 (활동중) 알고리즘 스터디 - Swift, Python
- 2022 해커톤 구름톤 3기 - 대상
- 2022 YAPP 20기 활동 가족을 위한 추억 공유 프로젝트 (iOS)
- 2022 Smile Gate STOVE Dev Camp 2기 인턴 Slack + Notion 협업 툴 (iOS)
- 
- 2021 CS 스터디 활동 - 운영체제, 네트워크
- 2021 이노베이션 아카데미 42 Seoul 4기
- 
- 2021 패스트캠퍼스 iOS 토이 프로젝트
- 
- 2020 아주대학교 TLO 연구원
- 2020 프로그래머스 Swift 온라인 교육 이수
- 2020 KISA 핀테크 한국인터넷진흥원 보험 추천 프로젝트
- 2020 1년 1일 1 커밋 - Swift Algorithm 풀이
- 
- 2019 자기주도 프로젝트  
이미지 & 영상 편집 프로젝트 (iOS)
- 2019 주) 앱플러 iOS 인턴 개발  
Awesome Day 프로젝트 & QA

# Abstract

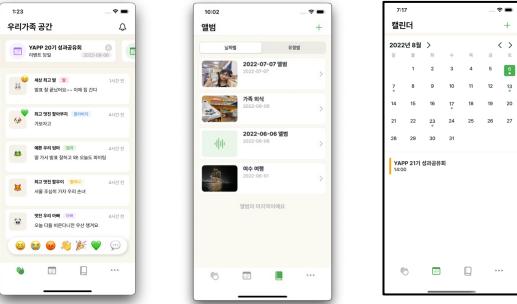


## Sofa, 가족 추억 기록 APP

기간 : 2022.04 ~ 2022.08

- UIKit + SwiftUI
- MVVM
- lottie-ios
- KakaoSDK
- Alamofire
- URLImage

추억 공간 (사진/음성)



실시간 프로필

공유 캘린더



## Snack(Slack + Notion) 협업 툴

기간 : 2021.12 ~ 2022.02

- RxSwift
  - MVVM
  - Alamofire
  - WebView
  - StompClinetLib
- : 채팅 & Presence 통신
- SnapKit
  - Then
  - MessageKit

채팅 / 실시간 Presence 실시간 문서편집



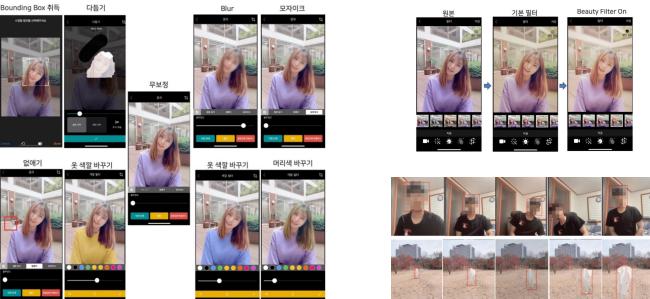
## 이미지 & 영상 편집 APP

기간 : 2019.09 ~ 2019.12

- UIKit
  - GPUImage2
  - OpenCV
  - Apple Vision
- (Tracking Objects)

: 영상에서 전경 Tacking

이미지의 전경과 배경 분리 이중 필터 / 전경 추적



## 도롱도롱

기간 : 2022.11.15 ~ 2022.11.18

- SwiftUI
- MVVM
- lottie-ios
- Alamofire
- URLImage

보이스 등록, 제주설화, 제주소리, 태교코칭



# Project 1



## Sofa, 가족 추억 기록 APP > 소개 & 기술 스택

기간 : 2021.04.02 ~ 2022.09.14



**제작 :** YAPP 20기 활동으로 PM, 디자이너와 함께 프로젝트

진행하고, 가족과 소식을 공유하고 싶은 20대 자녀를 메인 타겟으로 가족끼리 보다 쉽고 편리하게 소통의 니즈 반영

**목적 :** 우리 가족만을 위한 공용 일정, 사진 및 음성 공유, 소식 전달까지도 쉽게 나눌 수 있는 폐쇄형 가족 소통 서비스

**구성 :** PM, 디자이너, iOS 3, BE 2

**역할 :** 기획, iOS 개발, 추억 공유 탭(이미지 & 음성 처리), 네트워크 통신에 대한 Base Code 작성, 권한 처리에 대한 코드제공

**언어 :** SwiftUI + UIKit

**모델 :** MVVM

**Git :** <https://github.com/YAPP-Github/20th-iOS-Team-2-FE>

### 기술 스택

- UIKit + SwiftUI
- MVVM
- lottie-ios
- KakaoSDK
- Alamofire
- SwiftKeychainWrapper
- URLImage : SwiftUI에서 URL 이미지를 받아올 때 사용
- Introspect : UIKit 또는 AppKit을 사용

# Project 1

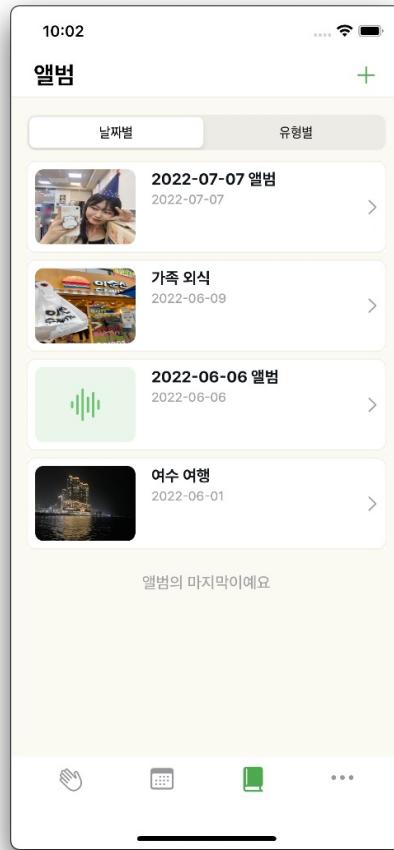
## Sofa, 가족 추억 기록 APP > 기능

### 기능



#### 가족 최신 정보

- 애니메이션
- 말풍선 기능
- D-Day
- Notice
- 손 흔들기(알림)



추억 공간 (사진/음성)

#### 추억 공간(사진 / 음성)

- 앨범 처리 및 모아보기
- 날짜별/유형별 사진 모으기
- 이미지/음성 업로드
- 음성에 대한 애니메이션
- 대표사진, 날짜 수정, 즐겨찾기, 저장
- 게시물에 대한 댓글



공유 캘린더

#### 공유 캘린더

- 구글 캘린더 가져오기
- 개인마다 고유한 색상 제공

# Project 2



## Snack(Slack + Notion) 협업 툴 > 소개 & 기술 스택

기간 : 2021.12.13 ~ 2022.02.25

**제기 :** 기존의 여러 협업 툴들을 따로 사용하면 창 간 전환이 너무 빈번하고 별도의 메신저를 사용해야 해서 생산성이 떨어짐

**목적 :** 워크스페이스와 채널을 기반으로 Slack의 채팅 기능과 Notion의 실시간 문서 편집을 할 수 있는 협업 툴 제공



메신저 형태의 실시간 네트워크 통신 서비스 구현



문서관리, 실시간 공동 편집 기능

**구성 :** iOS 1, Web 1, BE 2

**역할 :** 팀장, iOS 개발, 알림 서버 개발

**언어 :** RxSwift + UIKit

**모델 :** MVVM

**Git :** [https://github.com/njsh4261/SGS\\_Last\\_Punch/tree/dev/src/frontend/ios](https://github.com/njsh4261/SGS_Last_Punch/tree/dev/src/frontend/ios)

### 기술 스택

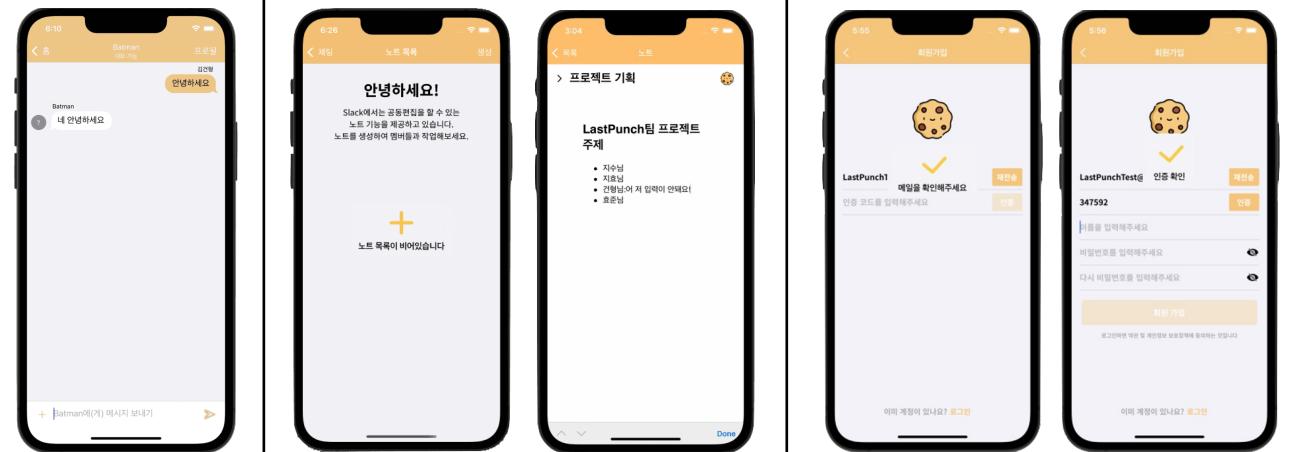
- RxSwift
- RxDataSources
- RxCocoa
- SnapKit
- MVVM
- Then
  
- Alamofire
- WebView
  
- KeyChain : 암호나 API Token과 같이 민감한 사용자 데이터에 대해 암호화 시켜 안전하게 보관
- ProgressHUD : 커스터마이징 되어진 Activity Indicator 제공
- PasscodeKit : Face ID 제공
  
- **MessageKit** : Collection View를 base로 message에 필요한 말풍선, 입력 창 등을 제공하면서 Custom 가능한 라이브러리
- **StompClinetLib** : 실시간 채팅 & Presence 통신을 위한 라이브러리

# Project 2



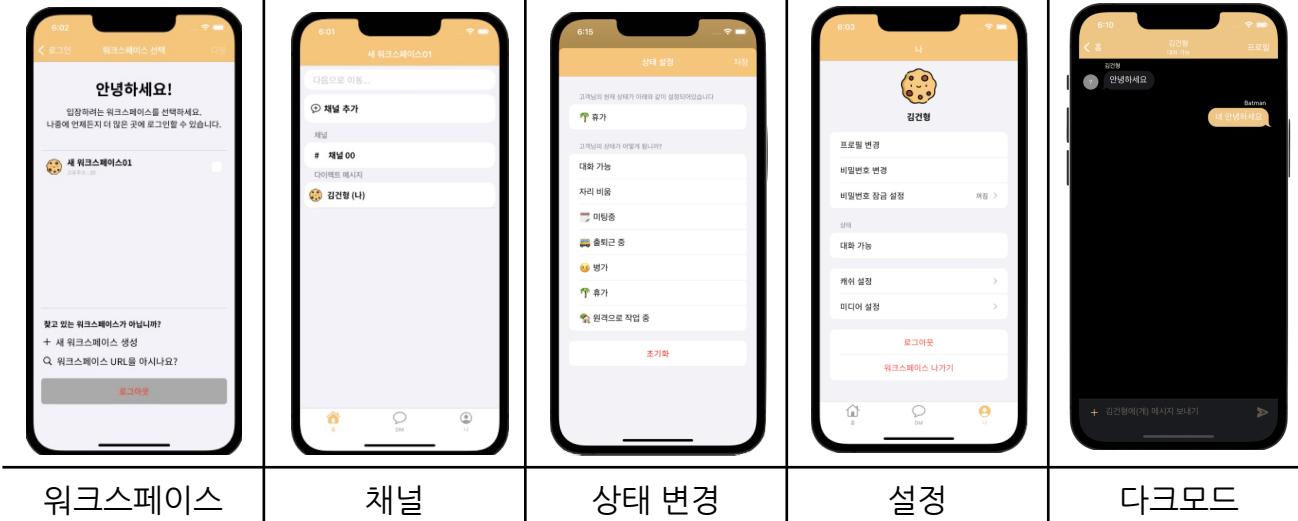
## Snack(Slack + Notion) 협업 툴 > 기능 & 구조

### 기능

WebSocket /  
그룹 채팅, DM

WebView / 실시간 문서 편집

회원가입 로직



워크스페이스

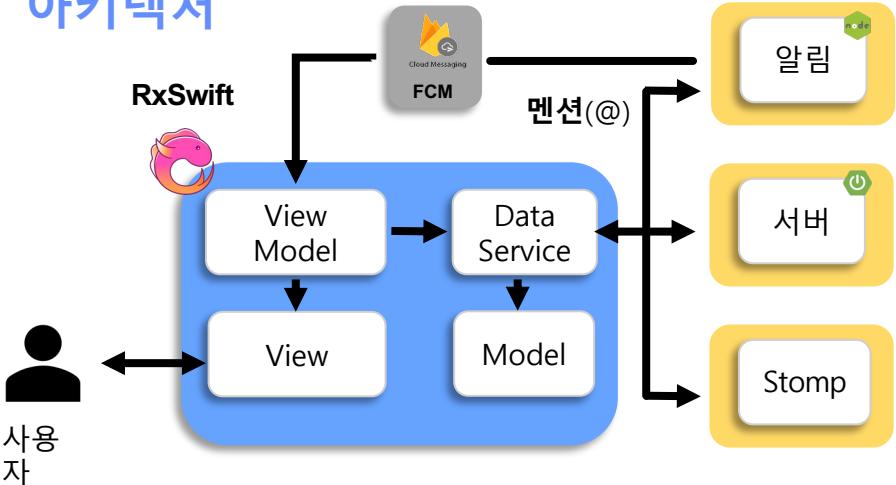
채널

상태 변경

설정

다크모드

### 아키텍처



### View 구조 - 4개의 함수

- init()** 가장 초기에 실행되며, Data를 넘겨 받거나 Token을 호출
- bind()** ViewModel을 바인딩
- attribute()** UI 객체의 속성을 정의, Then 활용
- layout()** UI 객체들의 constraints를 정의, Snapkit 활용

### ViewModel 구조 - Input, Output 객체

ViewModel은 init()으로 input, output 객체를 생성하고

View에서 바인딩하는 방법으로 구현

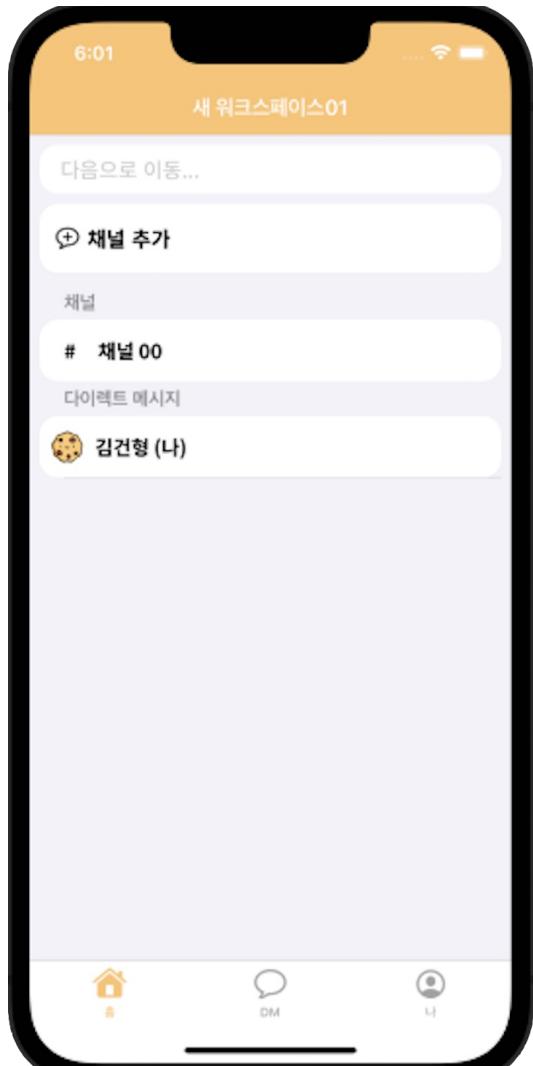
\* **input**은 PublishSubject, **output**은 PublishRelay로 구성

# Project 2



## Snack(Slack + Notion) 협업 툴 > 기술적 고민 & 해결

### 1. DataSource – Section별 Item 정의



1) Codable, Equatable을 상속받은 각 Model을 통한 Section 정의

```
struct HomeSection {
    typealias Model = SectionModel<HomeSection, HomeItem>

    enum HomeSection: Equatable {
        case chennel
        case member
    }

    enum HomeItem: Equatable {
        case channel(Channel)
        case member(Member)
    }
}
```

2) RxDataSource의 TableSectionReloadDataSource를 통한 Section별 Cell Setting 하여 해결

```
private var dataSource: RxTableViewSectionedReloadDataSource<HomeSection.Model>!

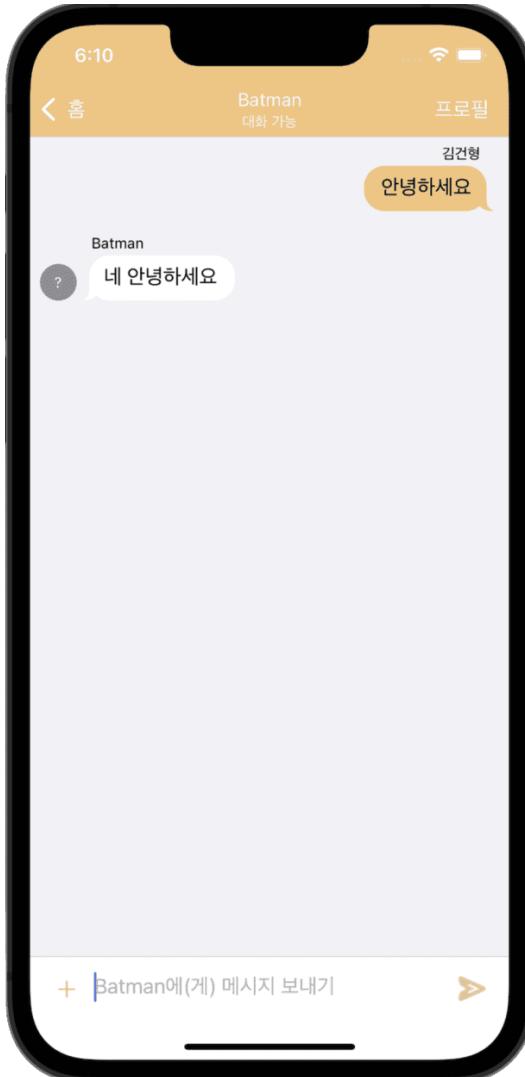
dataSource = RxTableViewSectionedReloadDataSource<HomeSection.Model> {
    dataSource, tableView, indexPath, item in
    switch item {
        case .chennel(let chennel):
            let cell = tableView.dequeueReusableCell(withIdentifier: "ChannelListCell", for: indexPath) as! ChannelListCell
            cell.setChennel(chennel)
            return cell
        case .member(let member):
            let cell = tableView.dequeueReusableCell(withIdentifier: "MemberListCell", for: indexPath) as! MemberListCell
            cell.setMember(member, indexPath.row)
            return cell
    }
}
```

# Project 2



## Snack(Slack + Notion) 협업 툴 > 기술적 고민 & 해결

### 2. WebSocket의 실시간 Data 반영 – 채팅, 입력 중 표시, 사용자 상태



1) View Model에서 Singleton Pattern으로 Data를 감지 후, 각각을 Bind

```
// socket
ChatStompWebsocket.shared.message
    .filter {$0.channelId == self.channel.id.description}
    .bind(to: output.socketMessage)
    .disposed(by: disposeBag)

// typing
ChatStompWebsocket.shared.typing
    .filter {$0.channelId == self.channel.id.description}
    .bind(to: output.socketTyping, output.socketEndTyping)
    .disposed(by: disposeBag)

// presence
PresenceWebsocket.shared.presence
    .bind(to: output.socketPresence)
    .disposed(by: disposeBag)
```

# Project 2



## Snack(Slack + Notion) 협업 툴 > 기술적 고민 & 해결

### 3. (시간적 문제) 실시간 문서 편집을 위한 Web과 iOS의 OSS의 동기화 문제



#### 1) WebView를 통한 해결

```
override func viewDidLoad() {
    super.viewDidLoad()
    loadUrl()
    webKitView!.navigationDelegate = self
    // 스와이프를 통해서 뒤로가기 앞으로가기를 할수 있게 해주는 설정값입니다.
    self.webKitView?.allowsBackForwardNavigationGestures = true
    webKitView?.evaluateJavaScript("localStorage.setItem(\"access_token\", \"\\"(\self.accessToken)\\"")", completionHandler: nil)
    webKitView?.evaluateJavaScript("localStorage.setItem(\"refresh_token\", \"\\"(\self.refreshToken)\\"")", completionHandler: nil)
}

// MARK: - Func
func loadUrl() {
    if let url = URL(string: url!) {
        var urlRequest = URLRequest(url: url)
        headers.forEach {urlRequest.setValue($0.value, forHTTPHeaderField: $0.key)}
        webKitView?.load(urlRequest)
    } else {
        // 예러처리문.. 예를들어서 alert를 띄워주거나..
        print("에러")
    }
}
```

#### 2) Access Token 만료 문제

WebView Delegate의 didStartProvisionation을 통해 localStorage에 access\_token과 refresh\_token 미리 저장

# Project 3



## 🎥 이미지 & 영상 편집 APP > 소개 & 기술 스택

기간 : 2019.09.01 ~ 2019.12.15

**제기 :** 스트리밍 영상 송출 기술을 플랫폼으로 하는 2019 하계 현장실습의 영상처리 기술을 활용한 iOS 개발 경험을 기반으로,  
영상을 편집할 때, 특정 영역의 object recognition 하지 못하여 위치 변화에 따라 수동으로 효과를 넣어주고 있음.  
기존의 처리 방식은 모바일이 아닌 Computer를 이용하거나 전문성을 요구함

**목적 :** 정적 이미지 편집을 확장하여 동영상에 적용할 수 있는 iOS 앱으로, 이미지 및 영상 편집 전문가가 아니더라도 모바일 환경에서 자동으로 동영상 편집에 도움을 주는 것.

**구성 :** iOS 1

**역할 :** 기획, 개발, OSS의 R&D

**언어 :** UIKit

**모델 :** MVC

**Git :** [https://github.com/GeonHyeongKim/cinemagraph\\_editor](https://github.com/GeonHyeongKim/cinemagraph_editor)

## 기술 스택

- UIKit
- TOCropViewController : 사용자가 원하는 전경을 포함한 사각형 추출
- GPUImage2 : 이미지 & 영상 필터, 커스텀 필터를 구현할 수 있으며, 이중 필터 기능까지 고려함
- OpenCV : 이미지 & 영상 편집, GrabCut 알고리즘 : 이용한 이미지의 전경과 배경 분리할 수 있음
- Apple Vision (Tracking Objects) : 동영상에서 오는 일련의 프레임을 통해 개체 또는 직사각형을 감지하고 추적할 수 있음

# Project 3



## 이미지 & 영상 편집 APP > 기능 & 과정

### 기능



구간선택, 명도&amp; 채도 &amp; 크기 조절, 필터



전경 선택, 전경 추가 제거 및 편집



### 과정

#### Step1. Create Story Board of View

MVC 패턴의 View 부분으로 유저 플로우를 파악해 UI 설계 및 구현함

#### Step2. Apply Video Filter

GPUImage2 라이브러리를 활용하여, OpenGL을 이해하고 상속 받아 필터를 직접 구현함. 이중 필터를 적용하기 위해 Base가 되는 기본 필터를 구현하고 인물의 피부를 보정하고 전체 색감을 연출하기 위해 Beauty Filter 구현



색상변경, 영역 지우기, Blur, 모자이크



이중 필터 / 전경 추적 후 영상에 저장



#### Step3. Extract Object in Image

OpenCV 라이브러리를 활용하여, 이미지에서 전경과 배경을 구분한 뒤 전경 영역에 모자이크, Blur, 가리기, 색깔 변경을 가능하게 만듦

#### Step4. Object Tracking in Video

Apple develop에서 제공하는 Vision tracking API를 통해 bounding box를 추득할 수 있도록 함

# Project 3



## 이미지 & 영상 편집 APP > 기술적 고민 & 해결

### 1. 처리된 프레임 segmentation의 정확도가 떨어지는 문제



### AI 기술의 landmark처럼 color를 중심으로 문제를 해결

- 1) 영상편집을 위한 이미지 편집단계에서 openCV의 알고리즘 중 grab cut을 이용하여 객체를 추출
- 2) 추출과 동시에 객체와 배경의 색상을 저장
- 3) 이를 통해 매 프레임의 bounding box에서 색상의 좌표와 영역을 계산
- 4) 또한, 모든 프레임에 대해 좌표를 구하는 것은 시간이 오래 걸렸기에 약 50프레임마다 bounding box 위치와 크기를 기준으로, 큰 변화가 있는 프레임에만 적용

그 결과, segmentation의 정확도가 완벽하지는 않지만 적용하기 전보다 개선 확인

객체가 영상에서 사라지지 않은 영상 및 객체를 포함한 프레임이 깨지지 않는 영상에서 더욱더 효과적으로 개선되었습니다.

# Project 3



이미지 & 영상 편집 APP > 기술적 고민 & 해결

## 2. 프레임 계산 처리속도와 UI 스레드 출력의 속도가 맞지 않아 시각적으로 버벅임이 발생하는 문제 : 영상에 대한 비동기 처리를 하여 처리

```
protocol VisionTrackerProcessorDelegate: class {
    func displayFrame(_ frame: CVPixelBuffer?, withAffineTransform transform: CGAffineTransform, rects: [TrackedPolyRect]?)  

    func displayFrameCounter(_ frame: Int)  

    func didFinifshTracking()  

}
```

- 1) VisionTrackerProcessor Class에서 비디오에 대한 해당 object를 tracking 하는 과정에서 displayFrame()을 계속해서 호출
- 2) 이때, Delegate Protocol을 선언해 놓고 displayFrame라는 함수 생성해 놓음

```
/// 6.Tracking Processor Delegate  
extension GrabCutViewController: VisionTrackerProcessorDelegate {  
    func displayFrame(_ frame: CVPixelBuffer?, withAffineTransform transform: CGAffineTransform,  
                      rects: [TrackedPolyRect]?) {  
        DispatchQueue.global(qos: .default).sync {  
            if let frame = frame {  
                let ciImage = CIImage(cvPixelBuffer: frame).transformed(by: transform)  
                let uiImage = UIImage(cgImage: self.convertCIImageToCGImage(inputImage: ciImage))  
                //ciimage -> cgimage  
                var rect = CGRect()  
                var resultImage = UIImage()  
  
                // OpenCV - CGRect() 변환  
                // MARK: frame에서 openCV를 활용하여 원하는 이미지 검출 코드 생략  
  
                DispatchQueue.main.async {  
                    self.trackingView.image = resultImage  
                }  
  
                DispatchQueue.main.async {  
                    self.trackingView.setNeedsDisplay()  
                }  
            }  
        }  
    }  
}
```

- 1) VisionTrackerProcessorDelegate을 선택한 뒤, **displayFrame()**을 부를 때마다,  
처리된 이미지들을 비동기처리
- 2) **setNeedsDisplay()**를 통해 View를 다시 UI Update를 시스템에 알려  
매끄러운 화면을 표시할 수 있도록 처리

# Project 4



## 도롱도롱 > 소개 & 기술 스택

**기간 :** 2022.11.15 ~ 2022.11.18

**제기 :** 제주도, 클라우드, 교육이라는 3가지 키워드로 해커톤을 만든 2박 3일간의 프로젝트

**목적 :** 도롱도롱은 아이가 곤히 잠자는 소리를 연상시키는 말입니다. 기존 고전적인 교육방식인 퀴즈와 강의에서 벗어나 오로지 '소리'만을 이용해 태아를 교육하는 서비스

**구성 :** PM 1, Designer 1, iOS 1, AOS 1 BE 1

**역할 :** iOS Part 개발, 개발 Leader

**언어 :** SwiftUI

**모델 :** MVVM

**Git :** [https://github.com/dorongdorong2022/DorongDorong\\_iOS](https://github.com/dorongdorong2022/DorongDorong_iOS)

**영상 :** <https://www.youtube.com/watch?v=GW6ONQgK8hs>



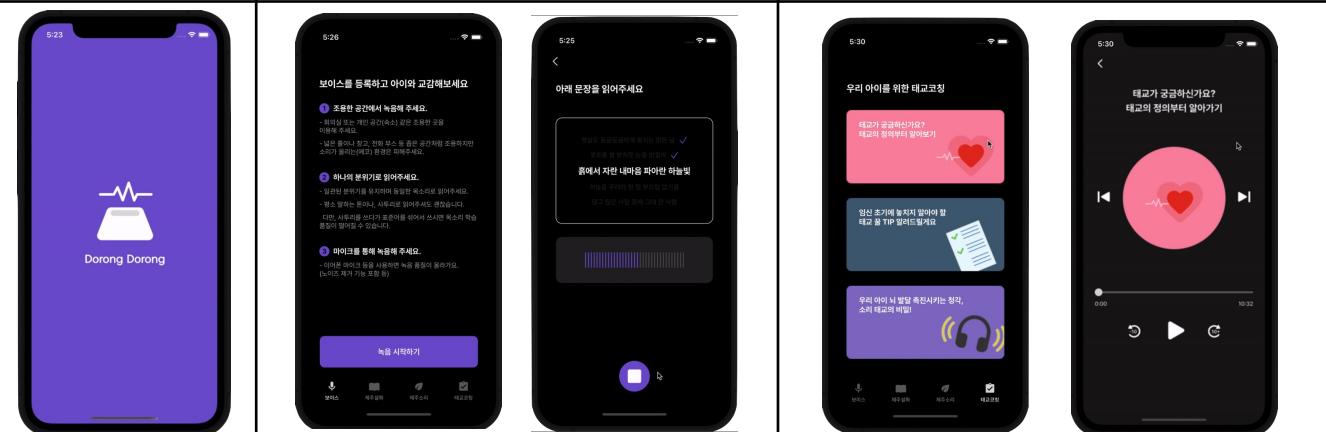
## 기술 스택

- SwiftUI
- MVVM
- lottie-ios
- Alamofire
- SwiftKeychainWrapper
- URLImage

# Project 4



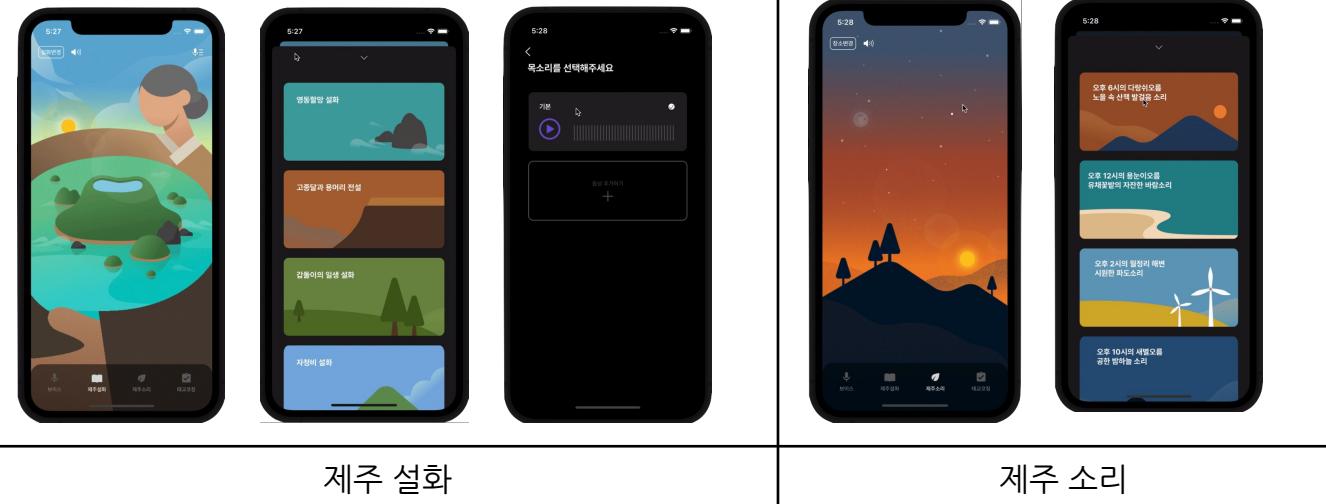
## 도룡도룡 > 소개 & 기술 스택



스플래쉬

TTS 등록

태교코칭



제주 설화

제주 소리

## 기능소개

### 제주소리

름다운 제주 자연의 소리를 아기에게 들려주는 기능  
산모가 제주로 여행을 갔던 당시의 감정, 풍경을 태아에게  
간접적으로 전해주기 위해 '소리'라는 매체를 선택

### 제주설화

제주소리에 나오는 자연 명소들에 깃든 설화들을 아기에게  
들려주는 기능

### 보이스

산모와 아이의 정서적 교감을 위해 도룡도룡이 내세우는 핵심  
가치인 '소리'를 가장 잘 살려준 기능

### 태교코칭

서로가 서로에게 처음인만큼 임산부가 태아에게 어떻게 행동해야  
좋을지 알려주는 코칭 기능

# Project 5



## Awesome Day, 여행 가이드 어플 APP > 소개 & 기술 스택

**기간 :** 2019.07.01 ~ 2019.08.27

**계기 :** Link+ 학업 연계 인턴 교육으로 iOS를 처음 접하게 됨, 회사의 특허를 낸 알고리즘을 사용하여 비디오 스트리밍 커머스로 라이브러리를 사용하여 쉽게 라이브 송출

**목적 :** 여행객이자, 가이드가 되어, 전세계 그 어느 곳이든 잘 알고 있다면 여행을 공유하고 여행상품을 직접 만들어 가이드로서 수입을 얻을 수 있는 앱

**구성 :** PM, 디자이너, iOS 3, BE 2

**역할 :** 인턴, 개발중인 앱의 QA와 실시간 스트리밍 기술 기반의 'AwesomeDay(iOS)' 앱 기능 개발

- 개발 팀장님에게 할당 받은 스프린트를 기간내에 개발
- 기존에 있는 안드로이드 버전의 앱을 iOS 버전으로 개발하는 프로젝트
- 필요한 API는 기존의 JSON형식으로 통신하고, 클라이언트의 요구사항으로 추가되는 새로운 기능들은 새롭게 구현(예약 기능)

**언어 :** UIKit

**모델 :** MVP

### 기술 스택

- UIKit, AutoLayout
- Swift의 다양한 UI 및 Interaction 개발 기술
- 외부 API 통신, JSON 이해
- GPUImage2 : 이미지 & 영상 필터
- FSCalendar API
- 비디오 오디오 관리

# Project 5



## Awesome Day, 여행 가이드 어플 APP > 기능

### 기능



1. 배경에 녹화된 영상 반복 송출
2. 사용자 정보에 따른 상품 정보를 가져와 인기순으로 랜덤 표시
3. 예약 기능에 대한 추가 개발로 FSCalendar API 응용



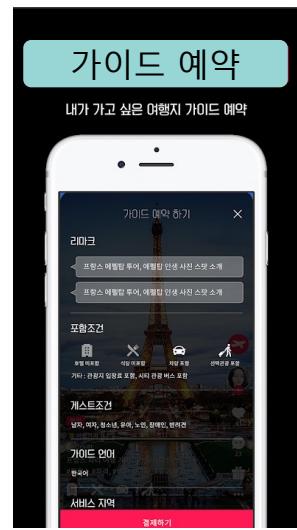
1. 가이드가 여행지를 선택하는 화면으로 여러 조건을 구현
2. 필수가 아닌 조건들에 대한 처리가 필요하여 사용자에게 알려 줄 수 있는 효율적인 방안을 찾음



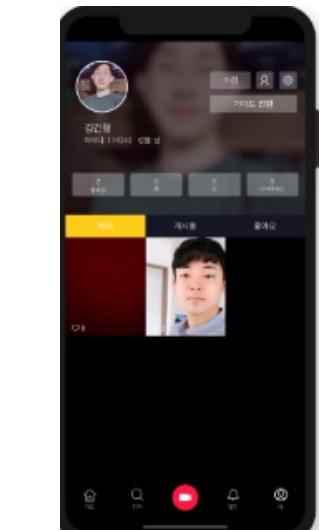
1. 내 프로필 화면으로 collectionView와 tableView를 활용
2. collectionView FlowLayout을 사용하여 더욱더 유동적인 화면 처리로 구현



1. 필터 기능으로 예외 처리를 고려해 구현



1. 예약 날짜에 대한 처리를 위해 FSCalendar API를 사용
2. Client의 요구 사항에 맞추어 Custom하여 구현



3. 사용자가 데이터가 많을 경우, 시간 단축에 대해 멘토와 상의하여 최적화



## 펫 미용 예약과 펫 보험 추천 서비스 &gt; 소개 &amp; 기술 스택

기간 : 2020.06.30 ~ 2020.08.27

계기 : KISA 핀테크 한국인터넷진흥원에서 진행을 통해 은행 API를 기반으로  
핀테크 관련 프로젝트 진행, 카카오 헤어샵처럼 펫 미용도 예약의 통합 서비스의 니즈와  
각 보험사에 따른 다른 조건으로 추천 서비스가 필요함

목적 : 펫 미용 예약 시스템과 펫 보험 추천 서비스 개발

구성 : 풀스택 5 (BE + FE)

역할 : 팀장, collaborative filtering & combination filter 구현

언어 : Python, NodeJS

## 기술 스택

- 핀테크 뱅킹 API
- 간편 결제(지급 결제) & 자산 관리
- QR 코드 결제
- 블록체인
- P2P 금융
- 인슈어테크



## 펫 미용 예약과 펫 보험 추천 서비스 > 기능

### Collaborative filtering

사용자 기반 협업 필터링을 이용한 보험 추천 서비스 알고리즘 로직 구현

사용자가 회원가입때 입력한 품종, 성별, 나이를 전달받아 유사한 사용자들이

가장 많이 가입해 있는 보험을 추천해주는 기능, 유사한 사용자는 피어슨 상관 계수를 적용하여

유사도를 계산하였으며, 보험의 추천도는 피어슨 상관 계수가 0.5 이상인 그룹에서 보험마다

사용자들의 피어슨 상관 계수를 더하여 가장 큰 값을 가지는 보험 2개를 추천

```
exports.basicDataSet = [
  {
    '신준수': {
      '강아지/고양이': 1,
      '품종': '90.0',
      '나이': (getAgeFromBirthday('2019/03/04') / 10),
      '성별': 1
    },
    '박보경': {
      '강아지/고양이': 1,
      '품종': '90.0',
      '나이': (getAgeFromBirthDay('2017/03/04') / 10),
      '성별': 1
    }
  }
];
```

다른 사용자들 정보  
(견종, 나이, 성별,  
중성화, 등록)



```
insurances: [
  {
    "id": 0,
    "fullName": "롯데손해보험 롯데마이펫보험",
    "company": "롯데손해보험",
    "insuranceName": "롯데마이펫보험",
    "img": "images/lotte.png",
    "assurance": [
      {
        "가장"
      },
      {
        "배상"
      },
      {
        "월"
      }
    ],
    "price": [
      "치과비(수술) 150만원",
      "치료비(입원) 10만원",
      "장례비자금비 15만원"
    ],
    "color": "#ff076e",
    "category": "no-medical unregistered ok-funeral"
  }
];
```

보험 정보



```
exports.insuranceDataSet = [
  {
    '신준수': {
      'insurance': '삼성화재 애니펫'
    },
    '김철수': {
      'insurance': '삼성화재 애니펫'
    },
    '박보경': {
      'insurance': '삼성화재 애니펫'
    },
    '김미령': {
      'insurance': 'DB손해보험 아이러브펫보험'
    }
  },
  {
    '가장'
  },
  {
    '배상'
  },
  {
    '월'
  }
];
```

다른 사용자들  
선택 정보

```
'신준식': {
  'insurance': '삼성화재 애니펫'
},
'신준일': {
  'insurance': '삼성화재 애니펫'
},
'신준이': {
  'insurance': '삼성화재 애니펫'
},
'신준상': {
  'insurance': '삼성화재 애니펫'
},
'김소일': {
  'insurance': 'NH 농협손해보험 반려동물장례비보험'
},
```

다른 사용자들  
선택 정보



최우선 / 차우선  
보험 추천

피어슨  
상관 계수  
구하기

추천  
로직

```
var personCorrelation = function (dataset, p1, p2) {
  var existP1p2 = {};
  for (item in dataset[p1]) {
    if (item in dataset[p2]) {
      existP1p2[item] = 1
    }
  }
  var numExistence = len(existP1p2);
  if (numExistence == 0) return 0;
  //store the sum and the square sum of both p1 and p2
  //store the product of both
  var p1_sq_sum = 0,
  p2_sq_sum = 0,
  p1_p2_sq_sum = 0,
  prod_p1p2 = 0;
  //calculate the sum and square sum of each data point
  //and also the product of each point
  for (var item in existP1p2) {
    p1_sq_sum += dataset[p1][item];
    p2_sq_sum += dataset[p2][item];
    p1_p2_sq_sum += Math.pow(dataset[p1][item], 2);
    p2_p2_sq_sum += Math.pow(dataset[p2][item], 2);
    prod_p1p2 += dataset[p1][item] * dataset[p2][item];
  }
  var numerator = prod_p1p2 - [p1_sq_sum * p2_sq_sum / numExistence];
  var st1 = p1_sq_sum * Math.pow(p1_sq_sum, 2) / numExistence;
  var st2 = p2_sq_sum * Math.pow(p2_sq_sum, 2) / numExistence;
  var denominator = Math.sqrt(st1 * st2);
  if (denominator == 0) return 0;
  else {
    var val = numerator / denominator;
    return val;
  }
}
```



### Combination filter

데이터에 Hash값을 미리 입력해 키워드에 따른 data filter 가능

```
"insurances": [
  {
    "id": 0,
    "fullName": "롯데손해보험 롯데마이펫보험",
    "company": "롯데손해보험",
    "insuranceName": "롯데마이펫보험",
    "img": "images/lotte.png",
    "assurance": [
      {
        "가장"
      },
      {
        "배상"
      },
      {
        "월"
      }
    ],
    "price": [
      "치료비(수술) 150만원",
      "치료비(입원) 10만원",
      "장례비자금비 15만원"
    ],
    "color": "#ff076e",
    "category": "no-medical unregistered ok-funeral"
  }
],
```

보험에  
Category  
값 설정





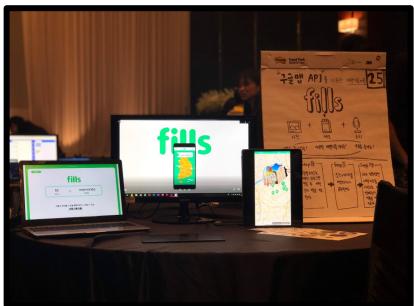
## Fills (추억을 채워가는 여행기록장) > 소개 & 기술



**기간 :** 2019.06.26 ~ 2019.06.28



**제기 :** 2019 오픈소스 해커톤에 참여하여 만든 팀 프로젝트  
 1. 여행의 추억을 다양한 매개체로 저장하고 싶음  
 2. 내가 갔던 여행지 별로 사진을 편하게 다시 보고 싶음  
 3. 다음 여행지를 추천



### 기술 스택

- Google 지도 API

**목적 :** 여행의 추억을 가지고 있는 사진을 지도 모양 및 크기로  
저장

**구성 :** 디자이너 1, AOS 5

**역할 :** 팀장, 디자이너와 대표 소통, AOS,  
Google 지도 API의 지도 좌표값 추출, 지역 filter 기능

**언어 :** Java

**Git :** <https://github.com/DivisonOfficer/iksan>



## Fills (추억을 채워가는 여행기록장) > 기능

### 기능

Launch	원하는 지역을 선택하거나 직접 검색	Side Bar	지역 필터	
사진, 메모 기능	플라로이드 형식 저장	지역 모양으로 사진 저장	지역 선택	사진, 메모, 음성, 녹음, 첨부



## 미유밋 (me you meet) > 소개 & 기술



기간 : 2019.03.01 ~ 2019.06.20

### 제기 :

- 여러 명의 대학생들이 주중에 약속을 잡을 때,  
모두가 만족하는 시간을 찾기 힘듦
- 가능 여부를 확인하기 위해서 개개인이 일일이 확인을 해야  
함



목적 : 인원수가 많은 모임에 대해서 자동으로 손쉽게  
모임원들이 공통적으로 만족하는 시간을 찾아주고, 없으면  
최적의 방법을 찾아서 다른 대안을 제안해주는 앱

구성 : BE 1, FE 3

역할 : 팀장, BE(PHP), AWS, DB 설계, 리눅스 환경 구축

언어 : PHP

Git : [https://github.com/jehoLee/CapD\\_America\\_ver3](https://github.com/jehoLee/CapD_America_ver3)

### 기술 스택

- OpenCV
- Google 캘린더
- KakaoAPI



## 미유밋 (me you meet) > 기능

### 기능

시간표 등록	일정 시간 선택	모임 / 일정 알림	모임원간의 일정
캡처한 시간표 이미지를 업로드 해주세요			

#### Service 1 이미지 업로드를 통한 시간표 분석

개인 및 타인의 시간표 이미지를 upload하여 쉽게 정보 저장

#### Service 2 대학생 모임 관리

모임 구성원들에게 모임 시간 공지 및 개개인의 스케줄에 따른 모임 시간 변경 용이

#### Service 3 모임 일정 시간 자동 추천

모임 구성원들의 시간표 정보를 자동으로 분석하여 빈 시간을 알아내 추천

#### Service 4 개인 일정 관리

등록 된 시간을 구글 캘린더에 기록하여 일정 관리 용이

### KPI

사용자가 로그인 후에 상대방의 모임 일정을 생성하기까지의 걸린 click 횟수 카카오톡에서 모임을 만들기 위해서는 각자의 시간표를 공유하고, 그룹원들끼리 일정을 조율하는 것에 시간을 소요 하지만 미유밋에서는 공강시간 분석 및 모임 생성을 앱에서 자동으로 처리하여 시간 단축



16회 이상



3회