

놀러!

김소연

- Intro : 개발자 소개



 github.com/aa9390

이름 김소연

학번 2014111517

전공 컴퓨터

E-Mail ese2003@naver.com

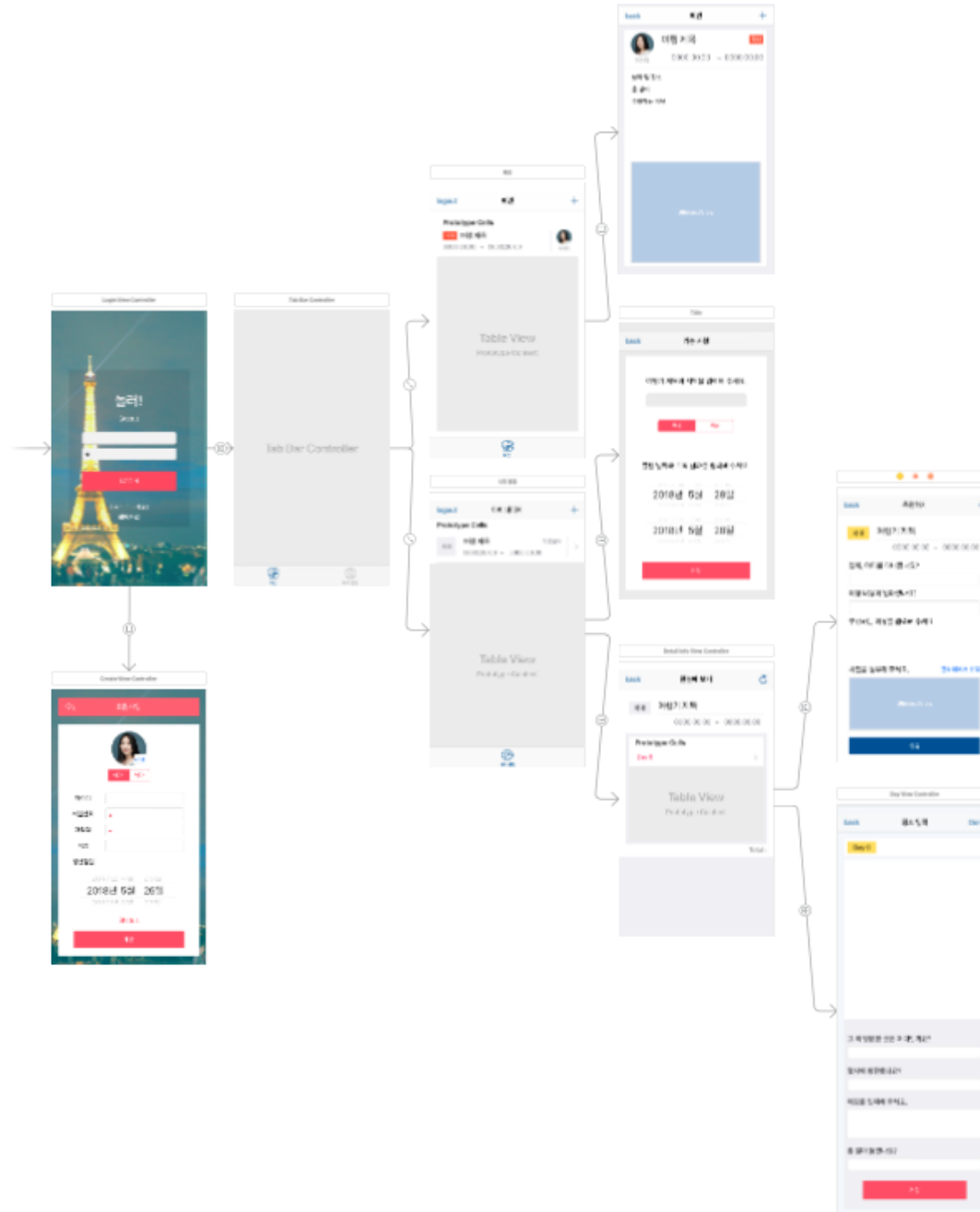
Github ID aa9390

- Intro : 어플리케이션 소개

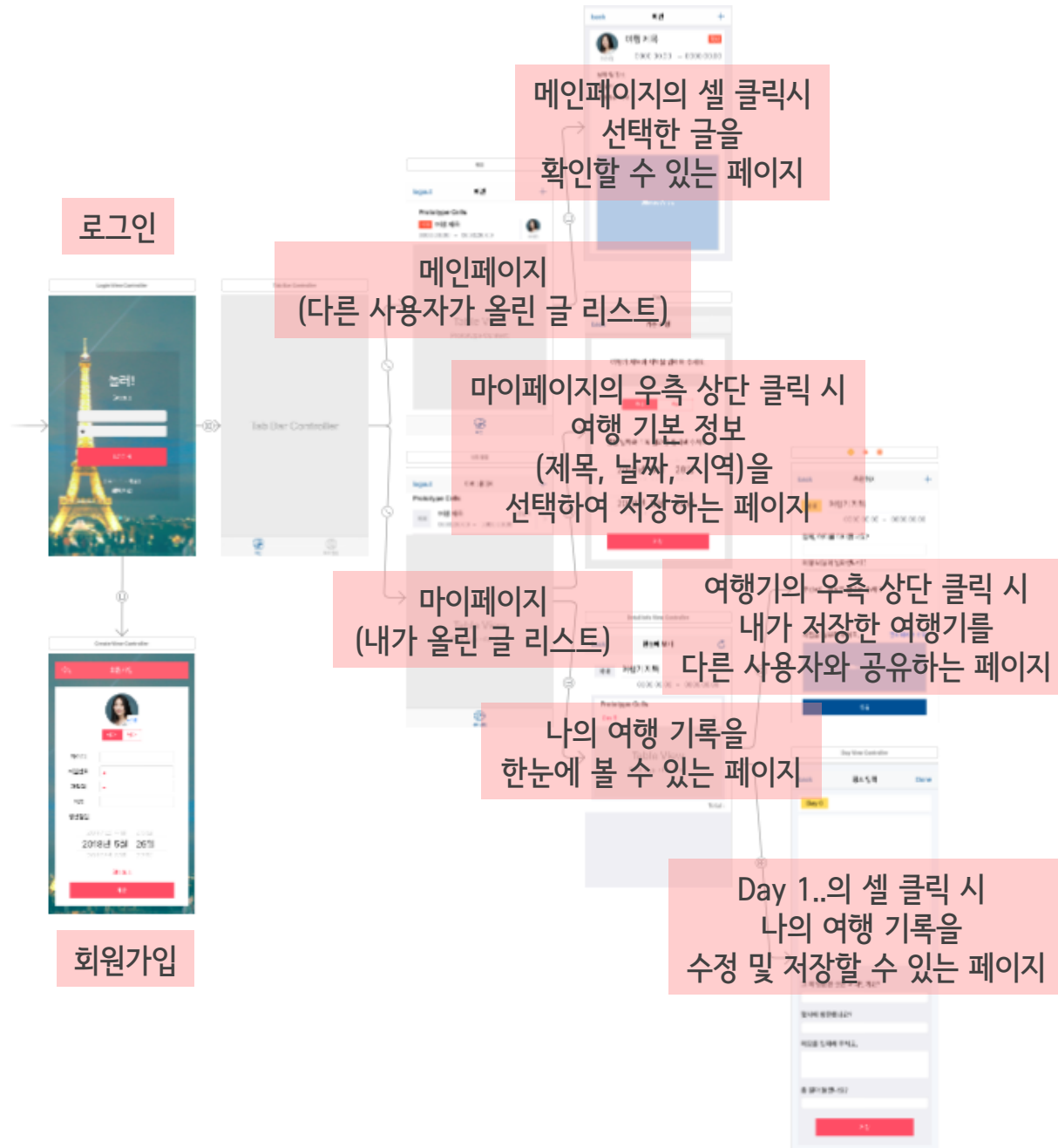
놀이! 란? 여행을 기록하고, 좋았던 여행은 다른 사용자에게 추천할 수 있는
직관적이면서 간편한 여행 스케줄러

앱 개발 동기 휴학 후 한달 간 유럽여행을 간 적이 있는데, 10개가 넘는 나라를 방문하다 보니
한눈에 여행기를 보기 힘들고, 지출 내역 또한 정리하기 어려웠다.
기존 여행 기록 어플은 많이 있으나, 여행 지출 내역을 정리해서 보여주는 어플인
‘세이브트립’과 손쉽게 블로그 글처럼 여행기를 기록할 수 있는 ‘볼로’는 아이폰을 지원하지 않는다.
또한 기타 어플인 ‘핫츠고플랜’, ‘여행노트’는 타 사용자와의 여행기 공유와
혼자만의 여행 기록 둘 중 하나에만 초점을 두고 있기 때문에
모든 기능을 갖춘, 그러나 직관적이고 간단한 어플을 만들고 싶었다.

- Storyboard

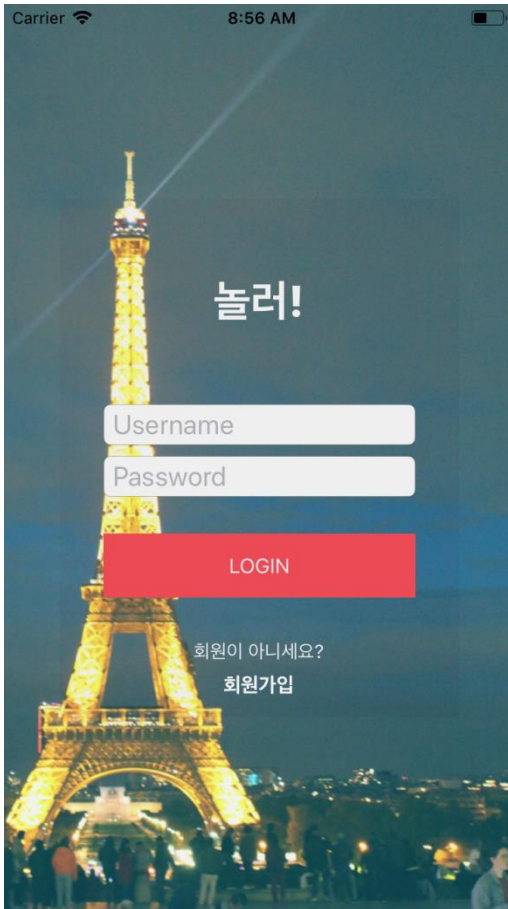


- Storyboard



• 로그인

- LoginViewController.swift
- 테이블의 정보와 내가 입력한 값을 비교하여 로그인을 시도합니다.



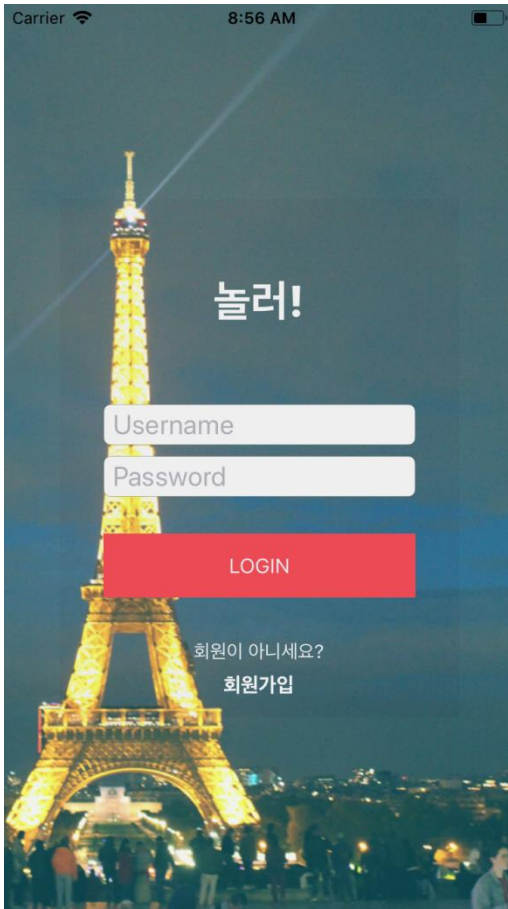
〈입력값 검증〉

```
// 로그인 버튼 눌렀을 시
@IBAction func loginPressed() {
    // 입력값 검증
    if loginUserid.text == "" {
        labelLoginStatus.text = "ID를 입력하세요"
        return
    }
    if loginUserpw.text == "" {
        labelLoginStatus.text = "PW를 입력하세요"
        return
    }
    self.labelLoginStatus.text = " "
```

- Loginuserid, loginUserpw의 텍스트필드 값이 모두 채워지지 않으면 Status 레이블에 에러 메시지를 출력하며, retrurn함.

로그인

- LoginViewController.swift
- 테이블의 정보와 내가 입력한 값을 비교하여 로그인을 시도합니다.



〈서버와의 통신〉

```
//      let urlString: String = "http://localhost:8888/nolleo/login/loginUser.php"
let urlString: String = "http://condi.swu.ac.kr/student/T03nolleo/loginUser.php"
guard let requestURL = URL(string: urlString) else {
    return
}

self.labelLoginStatus.text = " "

var request = URLRequest(url: requestURL)
request.httpMethod = "POST"

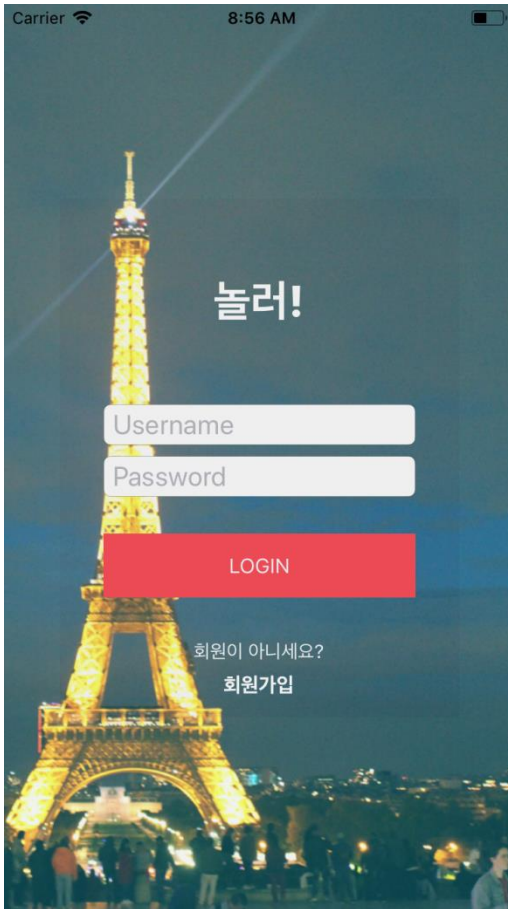
let restString: String = "id=" + loginUserId.text! + "&password=" + loginUserPw.text!
request.httpBody = restString.data(using: .utf8)

let session = URLSession.shared
let task = session.dataTask(with: request) { (responseData, response, responseError) in
    guard responseError == nil else { print("Error: calling POST")
        return }
    guard let receivedData = responseData else { print("Error: not receiving Data")
        return }
```

- 제출 버튼 클릭 시 condi.swu.ac.kr을 서버로 하여 loginUser.php의 동작을 실행한다.

• 로그인

- LoginViewController.swift
- 테이블의 정보와 내가 입력한 값을 비교하여 로그인을 시도합니다.



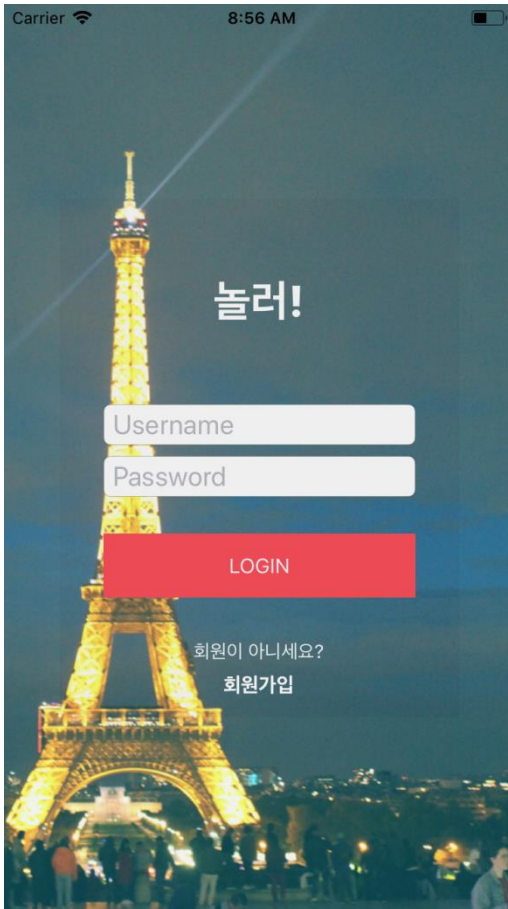
<loginUser.php>

```
<?php
$conn = mysqli_connect('localhost', 'student', 'student1234', 'T03nolleo');
$userid = ($_POST['id']);
$sql = "select user_id, user_pw user_name from user where user_id = '$userid' limit 1";
$result = mysqli_query($conn, $sql);

if (mysqli_num_rows($result) > 0) {
    while ($row = mysqli_fetch_row($result)) {
        $dbUserid = $row[0];
        $dbPassword = $row[1];
        $dbName = $row[2];
        $password = ($_POST['password']);
        $password = md5($password);
        if($userid == $dbUserid && $password == $dbPassword) {
            echo json_encode(array("success"=>"YES", "name"=>$dbName)); } else {
            echo json_encode(array("success"=>"NO", "error"=>"패스워드를 확인해 주세요")); }
        }
    } else {
        echo json_encode(array("success"=>"NO", "error"=>"존재하지 않는 ID입니다")); }
    mysqli_close($conn);
?>
```


로그인

- LoginViewController.swift
- 테이블의 정보와 내가 입력한 값을 비교하여 로그인을 시도합니다.



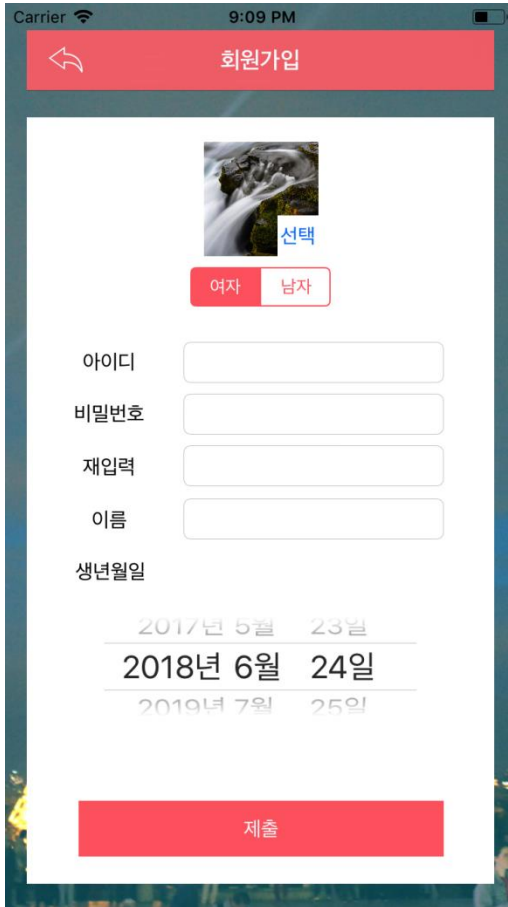
〈서버와의 통신 - 로그인 성공〉

```
// 로그인에 성공했을 경우
// Main 화면으로 이동
if success == "YES" {
    if let name = jsonData["name"] as! String! {
        DispatchQueue.main.async {
            self.labelLoginStatus.text = name + "님 안녕하세요?"
            self.performSegue(withIdentifier: "toLoginSuccess", sender: self)
        }
    }
} else {
    if let errMsg = jsonData["error"] as! String! {
        DispatchQueue.main.async {
            self.labelLoginStatus.text = errMsg
        }
    }
} catch {
    print("Error: \(error)")
}
task.resume()
```

- 로그인을 성공했을 경우 메인 화면으로 이동한다.

• 회원가입

- CreateViewController.swift
- MAMP를 이용해 서버 내 user 테이블에 사용자가 입력한 값을 삽입합니다.



〈입력값 검증〉

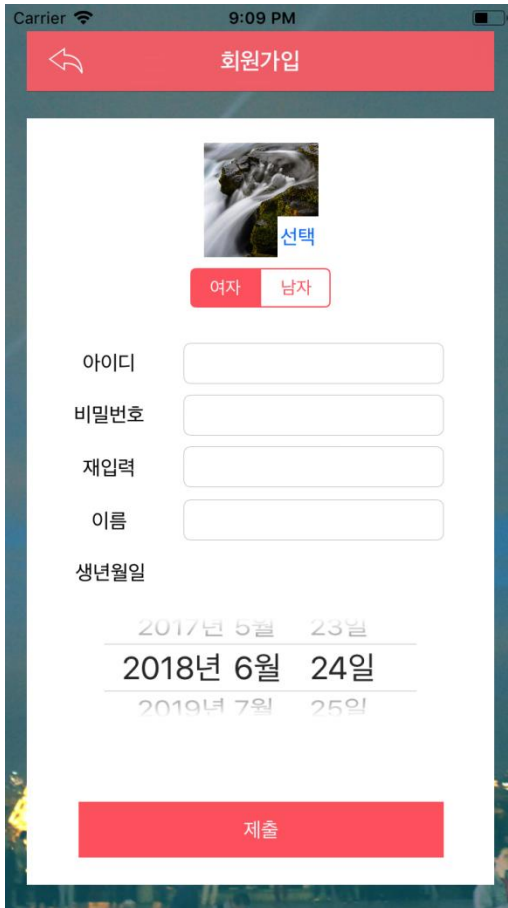
```
@IBAction func buttonSavePressed() {  
    if textCreateId.text == "" {  
        labelCreateStatus.text = "아이디를 입력하세요."  
        return;  
    }  
  
    if textCreatePw.text == "" {  
        labelCreateStatus.text = "비밀번호를 입력하세요."  
        return;  
    }  
  
    if textCreatePwRe.text != textCreatePw.text {  
        labelCreateStatus.text = "비밀번호를 확인해 주세요"  
        return;  
    }  
  
    if textCreateName.text == "" {  
        labelCreateStatus.text = "이름을 입력하세요."  
        return;  
    }  
}
```

- textCreateId, textCreatePw, textCreatePwRe, textCreateName의 값이 모두 채워지지 않으면 status 레이블에 에러 메시지를 출력하며, retrun한다.

• 회원가입

- CreateViewController.swift
- MAMP를 이용해 서버 내 user 테이블에 사용자가 입력한 값을 삽입합니다.

〈프로필 이미지 선택〉



```
// 선택 버튼 클릭
@IBAction func btnProfilePic(_ sender: UIButton) {
    // 갤러리에서 이미지 선택
    let myPicker = UIImagePickerController()
    myPicker.delegate = self;
    myPicker.sourceType = .photoLibrary
    self.present(myPicker, animated: true, completion: nil)
}

func imagePickerController (_ picker: UIImagePickerController, didFinishPickingMediaWithInfo info: [String : Any]) {
    if let image = info[UIImagePickerControllerOriginalImage] as? UIImage {
        self.imageCreateProfile.image = image
    }
    self.dismiss(animated: true, completion: nil)
}

func imagePickerControllerDidCancel (_ picker: UIImagePickerController) {
    self.dismiss(animated: true, completion: nil)
}
```

- 프로필 이미지 하단 선택 버튼을 클릭 시 갤러리에서 사진을 선택하는 화면으로 전환되며, 사진 선택 시 프로필 이미지 부분이 선택한 사진으로 바뀜

• 회원가입

- CreateViewController.swift
- MAMP를 이용해 서버 내 user 테이블에 사용자가 입력한 값을 삽입합니다.

〈서버와의 통신〉

```
// insertUser.php의 Uri String 선언
let urlString: String = "http://localhost:8888/nolleo/login/insertUser.php"
let urlString: String = "http://condi.swu.ac.kr/student/T03nolleo/insertUser.php"

// guard let? null이면 else부분 실행.
// null이 아니면 참인 부분 실행.
// 여기서는 참인 부분이 없으므로 아무것도 실행하지 않음.
guard let requestURL = URL(string: urlString) else {
    return
}

request = URLRequest(url: requestURL)

// POST 메소드를 사용하여 사용자 정보 전송
request.httpMethod = "POST"

let restString: String
    = "&gender=" + gender!
    + "&id=" + textCreateId.text!
    + "&password=" + textCreatePw.text!
    + "&name=" + textCreateName.text!
    + "&birth=" + birth
    + "&profile_img=" + imageFileName

restString = restString + "&birth=" + birth

request.httpBody = restString.data(using: .utf8)

self.executeRequest(request: request)

// 회원가입 성공 시 이전 화면으로 이동
self.dismiss(animated: true, completion: nil)
```

Carrier 9:09 PM

← 회원가입

선택

여자 남자

아이디

비밀번호

재입력

이름

생년월일

2017년 5월 23일

2018년 6월 24일

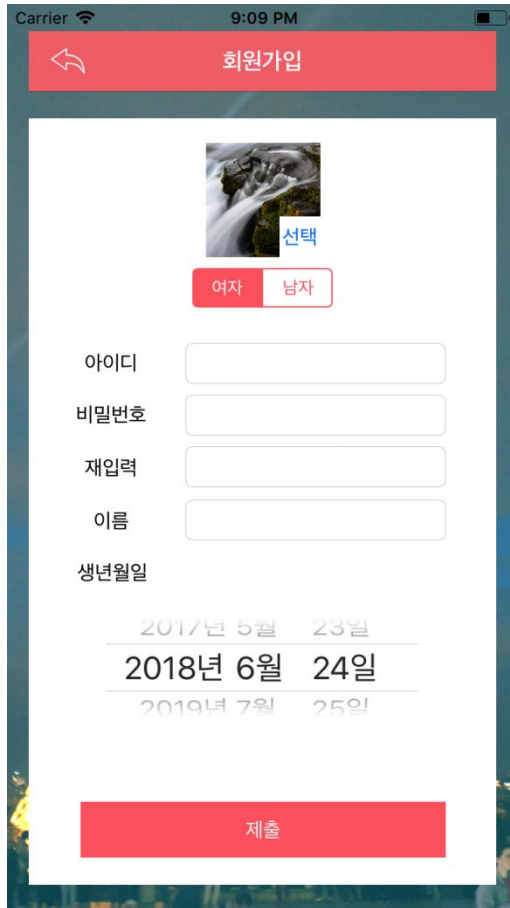
2019년 7월 25일

제출

- 제출 버튼 클릭 시 condi.swu.ac.kr을 서버로 하여 loginUser.php의 동작을 실행한다.
- 입력한 아이디, 비밀번호, 이름, 생년월일을 서버 내 user 테이블에 insert한다.

• 회원가입

- CreateViewController.swift
- MAMP를 이용해 서버 내 user 테이블에 사용자가 입력한 값을 삽입합니다.



A mobile application registration screen. At the top is a red header with a back arrow and the text '회원가입'. Below the header is a profile picture placeholder with a landscape image and a blue '선택' (Select) button. Underneath are two buttons for gender: '여자' (Female) and '남자' (Male). The form includes input fields for '아이디' (ID), '비밀번호' (Password), '재입력' (Re-enter), and '이름' (Name). Below these is a date picker for '생년월일' (Date of Birth) showing '2017년 5월 23일', '2018년 6월 24일', and '2019년 7월 25일'. At the bottom is a large red button labeled '제출' (Submit).

〈insertUser.php〉

```
<?php
// local host, ID, Password, Database
$conn = mysqli_connect('localhost', 'student', 'student1234', 'T03nolleo');

header('Content-Type: charset=utf-8');

$userid = ($_POST['id']);
$password = ($_POST['password']);
// 패스워드 암호화
$password = md5($password);
$name = ($_POST['name']);
$gender = ($_POST['gender']);
$birth = ($_POST['birth']);
$profile_img = ($_POST['profile_img']);
$sql= "insert into user (user_id, user_pw, user_name, user_gender, user_birth, profile_img)
values ('$userid', '$password', '$name', '$gender', '$birth', '$profile_img')";

if (mysqli_query($conn, $sql)) {
    echo '가입되었습니다. ';
}
else
{
    echo mysqli_error($conn);
}
mysqli_close($conn);
?>
```

● 마이페이지

- MypageTableViewCellController.swift
- 사용자가 입력한 값을 Custom Cell 형식으로 보여줍니다.

Carrier 8:27 PM

back 기본 사항

여행기 제목과 지역을 입력해 주세요.

떠나요 유럽으로

국내 국외

출발 날짜와 도착 날짜를 입력해 주세요.

2018년 6월 24일

2018년 7월 10일

저장

Carrier 8:26 PM

logout 마이페이지 +

국외 떠나요 유럽으로 06.24 >

2018.06.24 ~ 2018.07.10

메인 나의 일정

Carrier 8:26 PM

logout 마이페이지 +

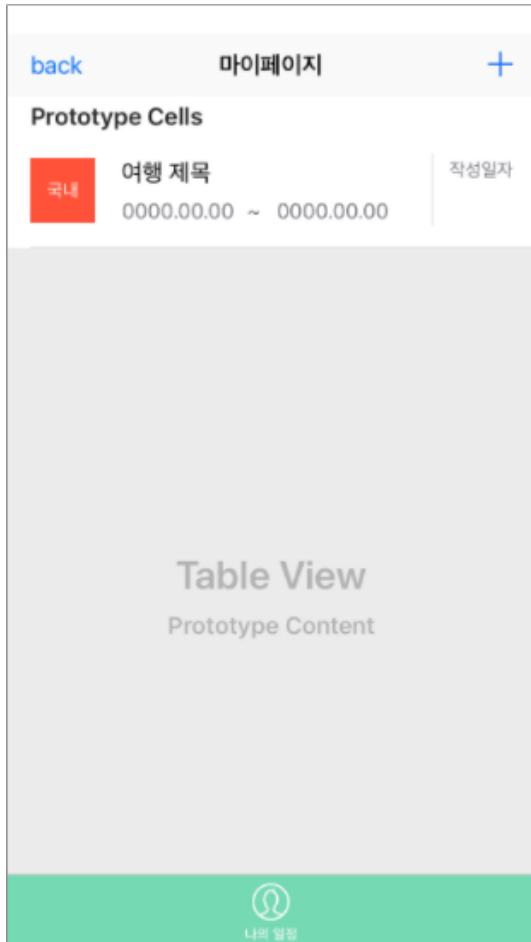
떠나요 유럽으로 06.24 > Delete

2018.06.24 ~ 2018.07.10

메인 나의 일정

• 마이페이지

- MypageTableViewController.swift
- 사용자가 입력한 값을 Custom Cell 형식으로 보여줍니다.



〈BasicInfoTableViewCell.swift〉

```
import UIKit

class BasicInfoTableViewCell: UITableViewCell {

    @IBOutlet var labelArea: UILabel!
    @IBOutlet var labelTitle: UILabel!
    @IBOutlet var labelStartDate: UILabel!
    @IBOutlet var labelEndDate: UILabel!
    @IBOutlet var labelSaveDate: UILabel!

    override func awakeFromNib() {
        super.awakeFromNib()
        // Initialization code
    }

    override func setSelected(_ selected: Bool, animated: Bool) {
        super.setSelected(selected, animated: animated)

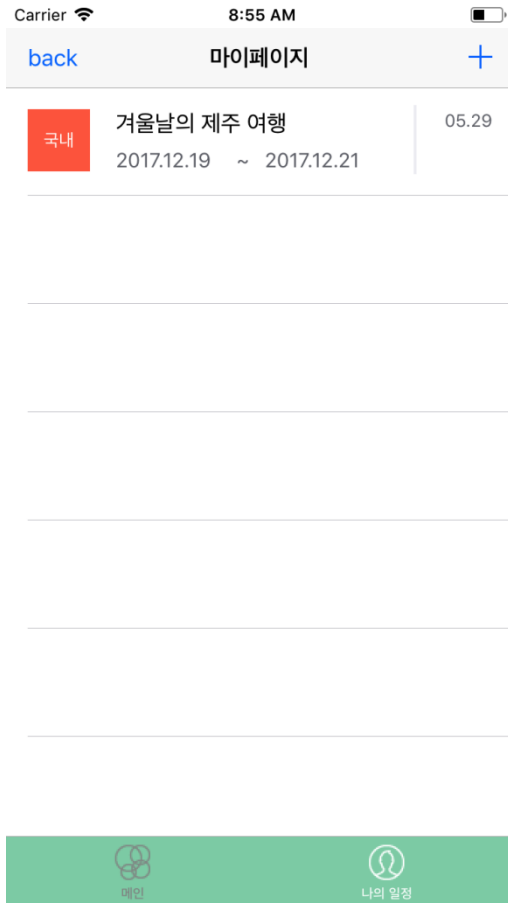
        // Configure the view for the selected state
    }

}
```

- 기본 셀이 아닌 TableCell을 상속받은 swift파일을 만들어, 셀을 커스터마이징 함.

● 마이페이지

- MypageTableViewController.swift
- 사용자가 입력한 값을 Custom Cell 형식으로 보여줍니다.



〈커스텀 셀 사용 명시〉

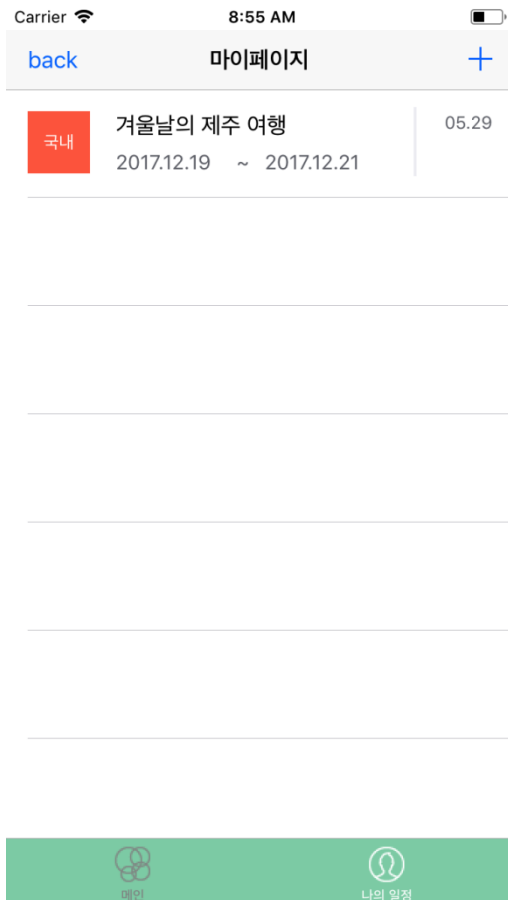
```
override func tableView(_ tableView: UITableView, cellForRowAt indexPath: IndexPath)
-> UITableViewCell {
    // 커스텀 셀 사용함을 명시

    let cell = tableView.dequeueReusableCell(withIdentifier: "Mypage Basic Info
    Cell", for: indexPath) as! BasicInfoTableViewCell
```

- 기본 셀이 아닌 TableCell을 상속받은 swift파일을 만들어, 셀을 커스터마이징 함.

● 마이페이지

- MypageTableViewController.swift
- 사용자가 입력한 값을 Custom Cell 형식으로 보여줍니다.



〈Core Data에서 데이터 가져오기〉

```
var BasicInfo: [NSManagedObject] = []

// View가 보여질 때 자료를 DB에서 가져옴
override func viewDidLoad(_ animated: Bool) {
    super.viewDidLoad(animated)

    let context = self.getContext()
    let fetchRequest = NSFetchRequest<NSManagedObject>(entityName: "BasicInfo")

    do {
        BasicInfo = try context.fetch(fetchRequest)
    }
    catch let error as NSError {
        print("Could not fetch. \(error), \(error.userInfo)")
    }
    self.tableView.reloadData()
}
```

- View가 보여질 때 BasicInfo 엔티티에서 데이터를 가져온다.

• 마이페이지

- MypageTableViewController.swift
- 사용자가 입력한 값을 Custom Cell 형식으로 보여줍니다.

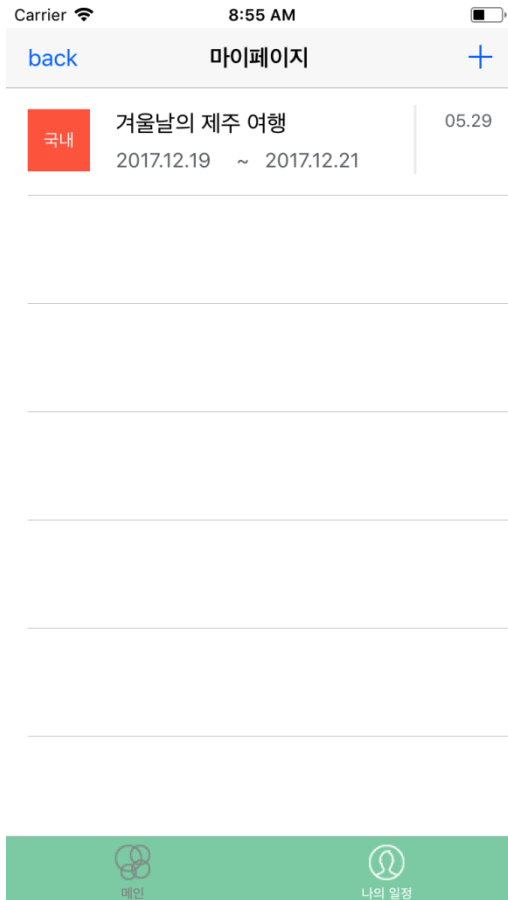
〈셀 삭제〉

```
override func tableView(_ tableView: UITableView, commit editingStyle: UITableViewCellEditingStyle, forRowAt indexPath: IndexPath) {
    if editingStyle == .delete {
        // Core Data 내의 해당 자료 삭제
        let context = getContext()
        context.delete(BasicInfo[indexPath.row])
        do {
            try context.save()
            print("deleted!")
        } catch let error as NSError {
            print("Could not delete \(error), \(error.userInfo)") }
        // 배열에서 해당 자료 삭제
        BasicInfo.remove(at: indexPath.row)
        // 테이블뷰 Cell 삭제
        tableView.deleteRows(at: [indexPath], with: .fade)
    }
}
```

- 셀을 스와이프할 시 삭제가 가능하도록 구현하였다.

● 마이페이지

- MypageTableViewController.swift
- 사용자가 입력한 값을 Custom Cell 형식으로 보여줍니다.



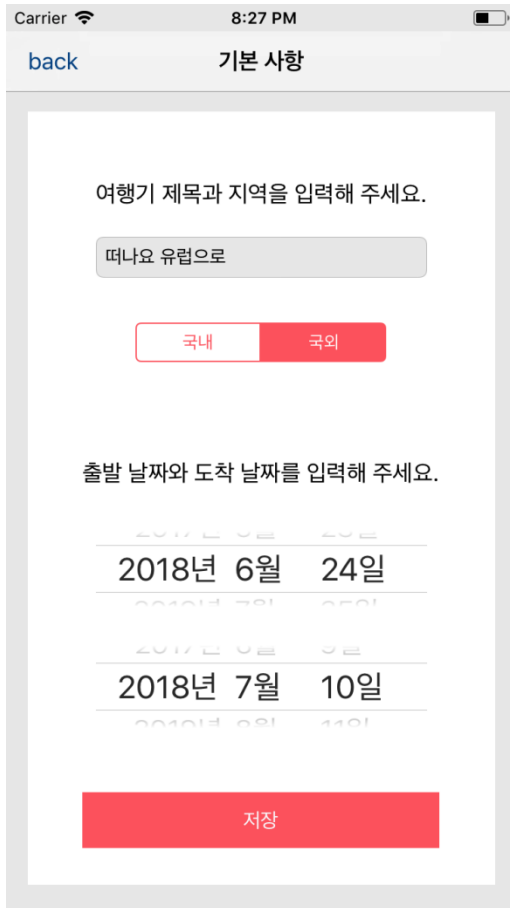
〈셀 클릭 시 클릭한 정보를 배열로 전송〉

```
override func prepare (for segue: UIStoryboardSegue, sender: Any?) {  
    if segue.identifier == "toDetailView" {  
        if let destination = segue.destination as? DetailInfoViewController {  
            if let indexPath = self.tableView.indexPathForSelectedRow {  
                destination.basic = BasicInfo[indexPath.row]  
            }  
        }  
    }  
}
```

- 셀 클릭 시 세부 사항 페이지로 넘어갈 때, basic이라는 배열로 기본 사항을 넘긴다.

• 기본 정보 작성 페이지

- BasicInfoSaveViewController.swift
- 셀을 생성하기 위해 필요한 기본 정보를 작성하여 Core data에 저장합니다.



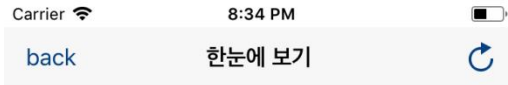
〈기본사항 작성 후 저장 버튼 클릭〉

```
func getContext () -> NSManagedObjectContext {  
    let appDelegate = UIApplication.shared.delegate as! AppDelegate  
    return appDelegate.persistentContainer.viewContext }  
  
// 저장 버튼을 눌렀을 경우  
@IBAction func savePressed() {  
    let context = getContext()  
    let entity = NSEntityDescription.entity(forEntityName: "BasicInfo", in: context)  
    // basicInfo record를 새로 생성함  
    let object = NSManagedObject(entity: entity!, insertInto: context)  
    object.setValue(textTripTitle.text, forKey: "title")  
    object.setValue(segTripArea.titleForSegment(at: segTripArea.selectedSegmentIndex), forKey: "area")  
    object.setValue(pickerStartDate.date, forKey: "startdate")  
    object.setValue(pickerEndDate.date, forKey: "enddate")  
    object.setValue(Date(), forKey: "savedate")  
  
    do {  
        try context.save()  
        print("저장되었습니다.")  
    } catch let error as NSError {  
        print("저장하지 못했습니다. \(error), \(error.userInfo)") }  
  
    self.dismiss(animated: true, completion: nil)  
}
```

- 저장 버튼 클릭 시 텍스트필드, 세그먼트, 데이트피커에 설정된 값을 Core Data에 저장한다.

• 세부 사항 확인 페이지

- DetailInfoViewController.swift
- 생성된 기본 사항을 Core Data에서 불러옵니다.

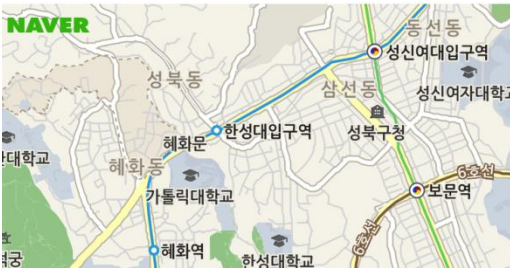


국내 서울 투어!

2018.06.24 ~ 2018.06.26

Day 1	명동, 남산타워	50000	>
Day 2	신당동, 떡볶이거리	12000	>

Total : 62000



<기본 사항을 Core Data에서 불러옴>

```
override func viewDidLoad() {
    super.viewDidLoad()

    if let basicToDetail = basic {
        titleLabel.text = basicToDetail.value(forKey: "title") as? String
        textArea.text = basicToDetail.value(forKey: "area") as? String
        let savedStartDate : Date? = basicToDetail.value(forKey: "startdate") as? Date
        let savedEndDate : Date? = basicToDetail.value(forKey: "enddate") as? Date
        let formatter: DateFormatter = DateFormatter()
        formatter.dateFormat = "yyyy.MM.dd"

        if let unwrapStartDate = savedStartDate {
            let displayStartDate = formatter.string(from: unwrapStartDate as Date)
            startDate.text = displayStartDate
        }

        if let unwrapEndDate = savedEndDate {
            let displayEndDate = formatter.string(from: unwrapEndDate as Date)
            endDate.text = displayEndDate
        }

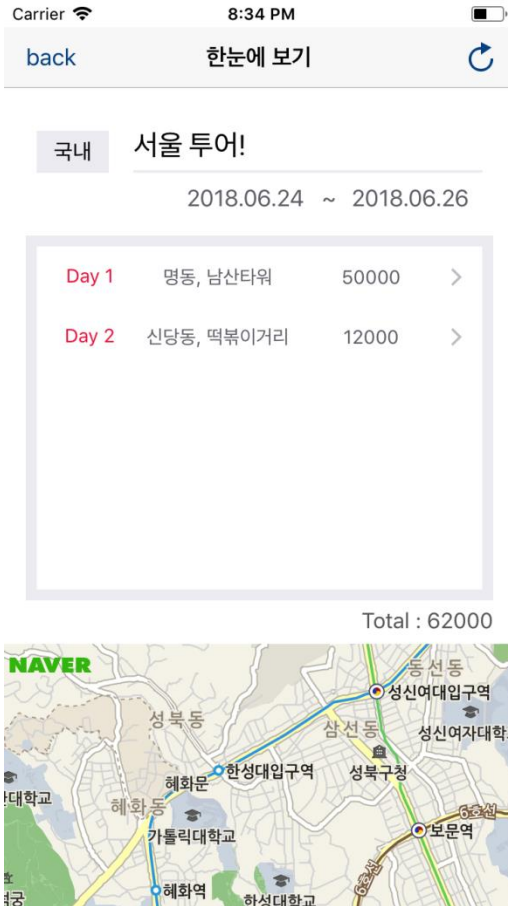
        dayInterval = savedEndDate!.timeIntervalSince(savedStartDate!)
        daysInterval = Int(dayInterval! / 86400)

        titleLabel.text = "\\(daysInterval!)"
    }
    // Do any additional setup after loading the view.
}
```

- viewDidLoad()에서 이전 화면의 테이블뷰에서 받은 indexPath를 넘겨받으면 Core Data에서 해당 NSObject의 데이터를 불러옴. 또한 timeIntervalSince()으로 출발 날짜와 도착 날짜와의 차이를 계산하여 그만큼 테이블 뷰의 DAY1, DAY2, DAY3과 같은 COUNT를 나타내도록 함.

• 세부 사항 확인 페이지

- DetailInfoViewController.swift
- 생성된 기본 사항을 Core Data에서 불러옵니다.



〈테이블 셀 불러오기 & 테이블 셀의 '비용'란의 총 합을 더해 total에 출력〉

```
func tableView(_ tableView: UITableView, cellForRowAt indexPath: IndexPath) -> UITableViewCell {
    if(detailInfo.count >= 1) {
        for i in 0...(detailInfo.count - 1) {
            if (textTitle.text == detailInfo[i].value(forKey: "title")as? String
                && "\(indexPath.row + 1)"
                == detailInfo[i].value(forKey: "daycount")as? String)
            {
                dayInfo = detailInfo[i]

                dayDisplay = (dayInfo.value(forKey: "place")as? String)!
                costDisplay = (dayInfo.value(forKey: "cost")as? String)!

                count = count + 1
                print("count \(count)")

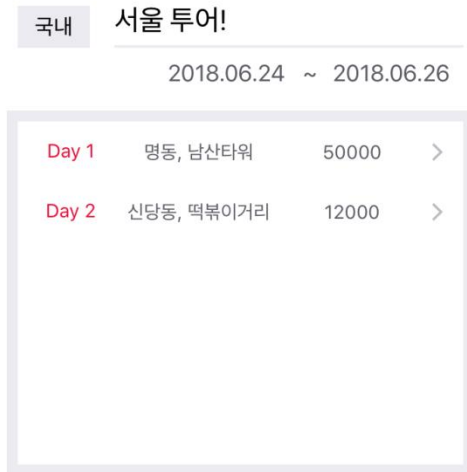
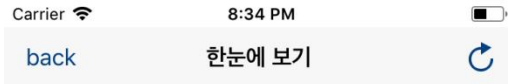
                cell.labelDay?.text = dayDisplay
                cell.labelCost?.text = costDisplay

                total = total + Int(costDisplay)!
            }
            else {
                cell.labelDay?.text = dayDisplay
                cell.labelCost?.text = costDisplay
            }
        }
    }
}
```

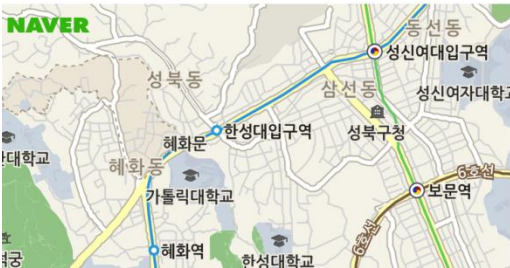
- 선택된 여행기의 제목과 day count가 일치하면 Core data에 저장된 내용을 읽어와 셀에 출력함.
- Total을 int형 초기값 0으로 설정 후, 하나의 셀을 불러올 때마다 total에 costdisplay의 값을 누적시킴.

• 세부 사항 확인 페이지

- DetailInfoViewController.swift
- 생성된 기본 사항을 Core Data에서 불러옵니다.



Total : 62000



〈DetailInfoTableViewCell.swift〉

```
class DetailInfoTableViewCell: UITableViewCell {

    @IBOutlet var labelDayCount: UILabel!
    @IBOutlet var labelDay: UILabel!
    @IBOutlet var labelCost: UILabel!

    override func awakeFromNib() {
        super.awakeFromNib()
        // Initialization code
    }

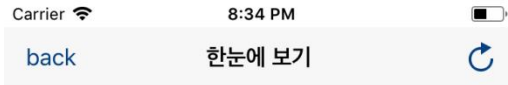
    override func setSelected(_ selected: Bool, animated: Bool) {
        super.setSelected(selected, animated: animated)

        // Configure the view for the selected state
    }
}
```

- 기본 셀이 아닌 TableCell을 상속받은 swift파일을 만들어, 셀을 커스터마이징 함.

• 세부 사항 확인 페이지

- DetailInfoViewController.swift
- 생성된 기본 사항을 Core Data에서 불러옵니다.



국내 서울 투어!

2018.06.24 ~ 2018.06.26

Day 1	명동, 남산타워	50000	>
Day 2	신당동, 떡볶이거리	12000	>

Total : 62000



〈커스텀 셀 사용〉

```
func tableView(_ tableView: UITableView, cellForRowAt indexPath: IndexPath) -> UITableViewCell {  
    // 커스텀 셀 사용함을 명시  
    let cell = tableView.dequeueReusableCell(withIdentifier: "Mypage Detail Info Cell", for: indexPath) as! DetailInfoTableCell  
  
    var dayCountDisplay: String = ""  
    var dayDisplay: String = ""  
    var costDisplay: String = ""  
  
    let formatter: DateFormatter = DateFormatter()  
    formatter.dateFormat = "MM.dd"  
  
    dayCountDisplay = "Day \(count)"  
    dayDisplay = "00.00"  
    costDisplay = "0 KRW"  
  
    cell.labelDayCount?.text = dayCountDisplay  
    cell.labelDay?.text = dayDisplay  
    cell.labelCost?.text = costDisplay  
    if(count <= daysInterval!) {count += 1}  
  
    return cell  
}
```


• 세부 사항 확인 페이지

- DayViewController.swift
- 각 날짜별 세부 정보를 확인 및 저장할 수 있습니다.



〈네이버 지도 객체 설정〉

```
override func viewDidLoad() {  
    super.viewDidLoad()  
  
    // viewWithNMap 안에 지도 설정  
    mapView = NMapView(frame: self.viewWithNMap.frame)  
    if let mapView = mapView {  
        // set the delegate for map view  
        mapView.delegate = self  
        mapView.reverseGeocoderDelegate = self  
  
        // set the application api key for Open MapViewer Library  
        mapView.setClientId("VzFN7JYE7dUDYEh6EgZa")  
        mapView.autoresizingMask = [.flexibleWidth, .flexibleHeight]  
        self.viewWithNMap.addSubview(mapView)  
    }  
  
    // Add Controls.  
    changeStateButton = createButton()  
  
    if let button = changeStateButton {  
        self.viewWithNMap.addSubview(button)  
    }  
}
```

- 네이버 지도 api를 프로젝트에 추가 후, NMapView 객체를 생성하여 해당 페이지에서 지도 관련 작업을 수행할 수 있도록 설정.

• 세부 사항 확인 페이지

- DayViewController.swift
- 각 날짜별 세부 정보를 확인 및 저장할 수 있습니다.



〈네이버 지도의 위치 클릭 시 해당 위치 출력〉

```
// MARK: - NMapReverseGeocoderDelegate Methods
open func location(_ location: NGeoPoint, didFind placemark: NMapPlacemark!) {

    // 클릭한 위치의 주소
    let address = placemark.address
    self.title = address

    labelResult.text = "\(address!)"
    textPlace.text = textPlace.text! + ", \("\(placemark.dongName!)"

}
```

- Api 함수인 `location(_ location: NGeoPoint, didFind placemark: NMapPlacemark!)`을 사용하여 지도 클릭 시 클릭한 위치의 주소를 `address`로 받아 `labelResult`(‘서울특별시 중구 광희동’이 쓰여있는 label)에 클릭한 주소를 출력하고, 동까지의 이름(`placemark.dongName`)을 `textPlace`에 지속적으로 추가함. (Ex. 서울특별시 중구 광희동의 위치 클릭 시 ‘, 광희동’ 이 `textPlace`에 추가됨)

• 세부 사항 확인 페이지

- DayViewController.swift
- 각 날짜별 세부 정보를 확인 및 저장할 수 있습니다.

Carrier 8:37 PM

덕수중학교 충무시장 오장동 서울동대문

수협은행 목정공원 CJ제일저

비축진지구 NAVER Corp. 퇴계로5가 교차로

서울특별시 중구 을지로동

그 외 방문한 곳은 어디인가?

명동, 남산타워

몇시에 방문했나요?

4시

메모를 입력해 주세요.

경치가 너무 예뻐다!

총 얼마 들었나요?

50000

저장

〈viewDidLoad로 Core Data에서 데이터 불러와 진입 화면 세팅〉

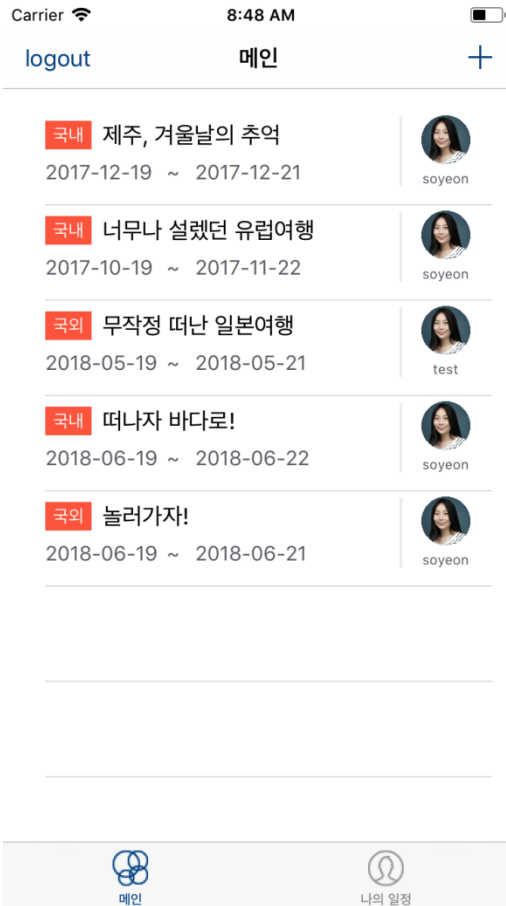
```
let context = self.getContext()
let fetchRequestDetail = NSFetchRequest<NSManagedObject>(entityName: "Detail")
do {
    detailInfo = try context.fetch(fetchRequestDetail)
} catch let error as NSError {
    print("Could not fetch. \(error), \(error.userInfo)")
}

var count: Int = 0
if(detailInfo.count >= 1) {
    for i in 0...(detailInfo.count - 1) {
        if (titleText == detailInfo[i].value(forKey: "title")as? String && daycount == detailInfo[i].value(forKey: "daycount")as? String) {
            dayInfo = detailInfo[i]
            count = count + 1
            print("count \(count)")
            textPlace.text = dayInfo.value(forKey: "place")as? String
            textTime.text = dayInfo.value(forKey: "time")as? String
            textCost.text = dayInfo.value(forKey: "cost")as? String
            textMemo.text = dayInfo.value(forKey: "memo")as? String
        }
    }
}
```

- viewDidLoad()내에서 선택된 여행기의 제목과 day count가 일치하면 Core data에 저장된 내용을 읽어와 관련된 textField를 채움.

• 메인페이지

- MainpageViewController.swift
- 공유가 완료된 여행기를 Server에서 불러옵니다.



〈커스텀 셀 사용〉

```
import UIKit

class MainPageTableViewCell: UITableViewCell {

    @IBOutlet var labelTripArea: UILabel!
    @IBOutlet var labelTripTitle: UILabel!
    @IBOutlet var labelStartDate: UILabel!
    @IBOutlet var labelEndDate: UILabel!
    @IBOutlet var labelId: UILabel!
    @IBOutlet var imgProfile: UIImageView!

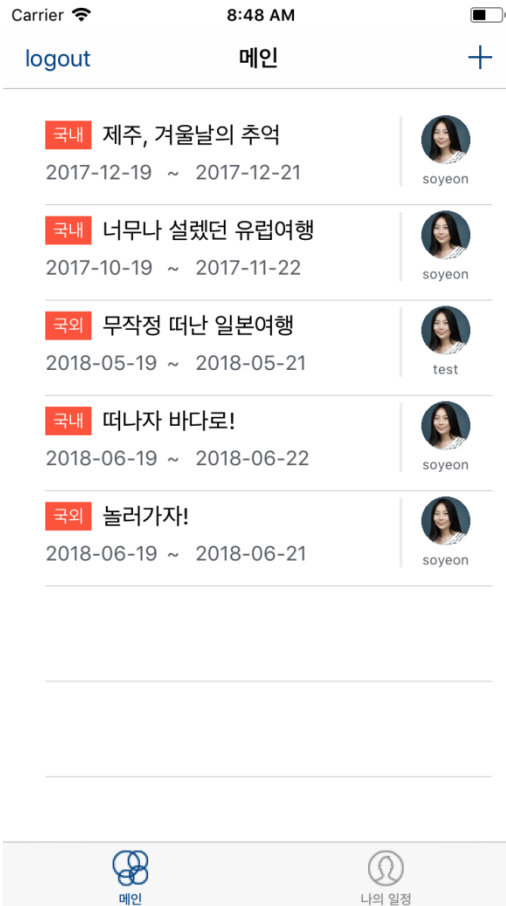
    override func awakeFromNib() {
        super.awakeFromNib()
        // Initialization code
    }

    override func setSelected(_ selected: Bool, animated: Bool) {
        super.setSelected(selected, animated: animated)

        // Configure the view for the selected state
    }
}
```

• 메인페이지

- MainPageViewController.swift
- 공유가 완료된 여행기를 Server에서 불러옵니다.



〈서버에서 데이터 가져옴〉

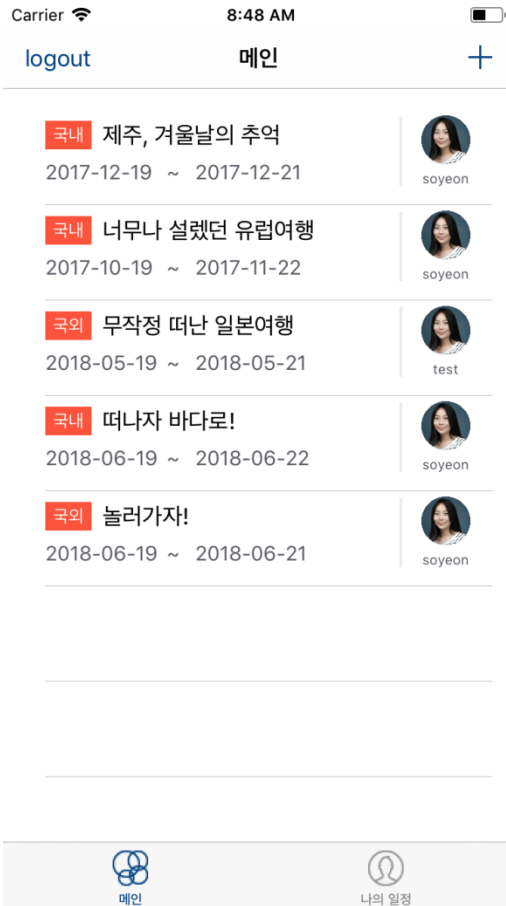
```
var fetchedArray: [BasicInfoData] = Array()
var fetchedArray_user: [UserData] = Array()

override func viewDidLoad() {
    super.viewDidLoad()
    테이블 뷰 높이 지정
    tableView.rowHeight = 70
}

override func viewWillAppear(_ animated: Bool) {
    super.viewWillAppear(animated)
    fetchedArray = [] // 배열을 초기화하고 서버에서 자료를 다시 가져옴
    // 서버에서 데이터 가져옴
    self.downloadDataFromServer()
    self.downloadDataFromServer_user()
}
```

• 메인페이지

- MainpageViewController.swift
- 공유가 완료된 여행기를 Server에서 불러옵니다.



〈서버에서 데이터 가져옴〉

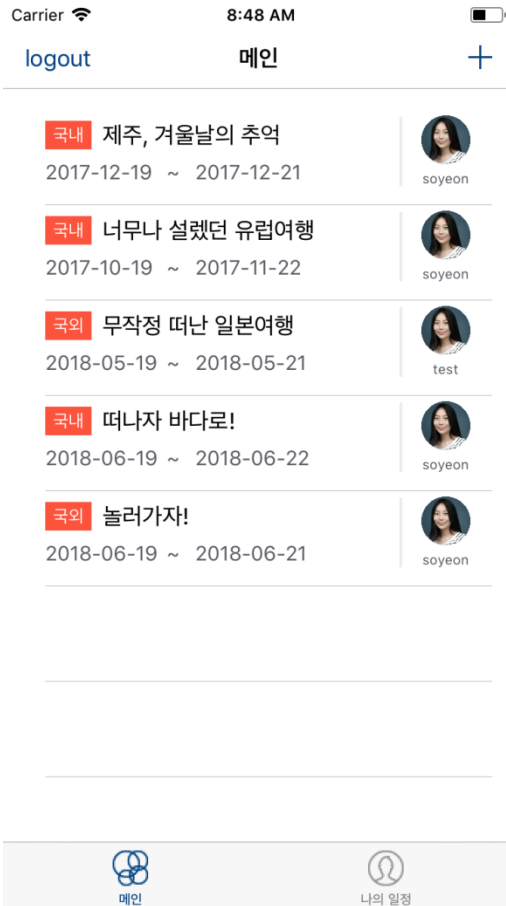
```
// 서버에서 데이터 로드
func downloadDataFromServer() -> Void {
    let urlString: String = "http://localhost:8888/favorite/favoriteTable.php"
    let urlString: String = "http://condi.swu.ac.kr/student/T03nolleo/selectBasicInfo.php"

    guard let requestURL = URL(string: urlString) else { return }
    let request = URLRequest(url: requestURL)
    let session = URLSession.shared
    let task = session.dataTask(with: request) { (responseData, response, responseError) in
        guard responseError == nil else { print("Error: calling POST"); return; }
        guard let receivedData = responseData else {
            print("Error: not receiving Data"); return; }

        let response = response as! HTTPURLResponse
        if !(200...299 == response.statusCode) { print("HTTP response Error!"); return }
        do {
            if let jsonData = try JSONSerialization.jsonObject (with: receivedData,options:.allowFragr
                for i in 0...jsonData.count-1 {
                    let newData: BasicInfoData = BasicInfoData()
                    var jsonElement = jsonData[i]
                    newData.index = jsonElement["index"] as! String
                    newData.title = jsonElement["title"] as! String
                    newData.user_id = jsonElement["user_id"] as! String
                    newData.area = jsonElement["area"] as! String
                    newData.start_date = jsonElement["start_date"] as! String
                    newData.end_date = jsonElement["end_date"] as! String
                    newData.recommend_reason = jsonElement["recommend_reason"] as! String
                    newData.recommend_cost = jsonElement["recommend_cost"]as! String
                    newData.recommend_when_where = jsonElement["recommend_when_where"]as! String
                    newData.recommend_img = jsonElement["recommend_img"]as! String
                    self.fetchedArray.append(newData)
                }
            DispatchQueue.main.async { self.tableView.reloadData() } }
        } catch { print("Error:") } }
    task.resume()
}
```

• 메인페이지

- MainpageViewController.swift
- 공유가 완료된 여행기를 Server에서 불러옵니다.



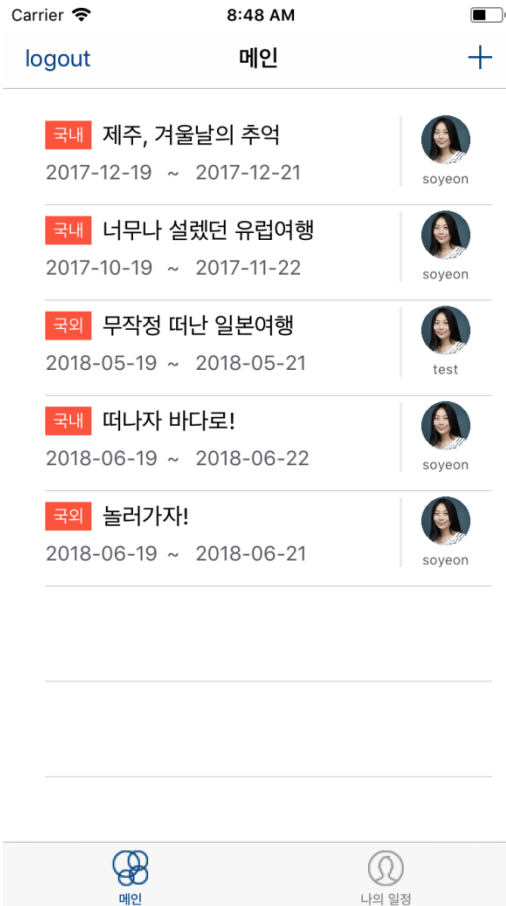
〈가져온 데이터를 table view의 각 cell에 출력〉

```
func tableView(_ tableView: UITableView, cellForRowAt indexPath: IndexPath) -> UITableViewCell {  
  
    let cell = tableView.dequeueReusableCell(withIdentifier: "Mainpage Basic Info Cell", for: indexPath) ;  
  
    let item = fetchedArray[indexPath.row]  
    let item_user = fetchedArray_user[indexPath.row]  
  
    cell.labelId?.text = item.user_id  
    cell.labelTripTitle?.text = item.title  
    cell.labelTripArea?.text = item.area  
    cell.labelStartDate?.text = item.start_date  
    cell.labelEndDate?.text = item.end_date  
  
    var imageName = item_user.profile_img// 숫자.jpg 로 저장된 파일 이름  
    if (imageName != "") {  
        let urlString = "http://condi.swu.ac.kr/student/T03nolleo"  
        imageName = urlString + imageName  
        let url = URL(string: imageName)!  
        if let imageData = try? Data(contentsOf: url) {  
            cell.imgProfile.image = UIImage(data: imageData)  
            // 웹에서 파일 이미지를 접근함  
        }  
    }  
  
    return cell  
}
```

- 가져온 데이터를 각 cell에 출력함.

• 메인페이지

- MainpageViewController.swift
- 공유가 완료된 여행기를 Server에서 불러옵니다.



〈가져온 데이터를 table view의 각 cell에 출력〉

```
override func prepare(for segue: UIStoryboardSegue, sender: Any?) { // Get the new view con
    if segue.identifier == "toSharedInfoView" {
        if let destination = segue.destination as? SharedInfoViewController {
            if let selectedIndex = self.tableView.indexPathsForSelectedRows?.first?.row {
                let data = fetchedArray[selectedIndex]
                destination.selectedData = data
            }
        }
    }
}
```

- Cell 클릭 시 해당 셀의 index에 해당하는 데이터를 다음 화면에 넘겨줌.

• 공유된 내용 확인 페이지

- SharedInfoViewController.swift
- 내가 공유한 여행기와 남들이 공유한 페이지를 확인할 수 있습니다.



〈이전 페이지에서 data 넘겨받아출력〉

```
override func viewDidLoad() {
    super.viewDidLoad()

    guard let sharedData = selectedData else { return }

    labelTitle.text = sharedData.title
    labelArea.text = sharedData.area
    labelUserId.text = sharedData.user_id
    labelStartDate.text = sharedData.start_date
    labelRecommend.text = sharedData.recommend_reason
    labelEndDate.text = sharedData.end_date
    labelWhenWhere.text = sharedData.recommend_when_where
    labelCost.text = sharedData.recommend_cost

    var imageName = selectedData?.recommend_img // 숫자.jpg 로 저장된 파일 이름
    if (imageName != "") {
        let urlString = "http://condi.swu.ac.kr/student/T03nolleo/"
        imageName = urlString + imageName!
        let url = URL(string: imageName!)!
        if let imageData = try? Data(contentsOf: url) {
            imageView.image = UIImage(data: imageData)
            // 웹에서 파일 이미지를 접근함
        }
    }
}
```

- 공유된 내용을 확인할 수 있음.

• 나의 여행을 추천할 수 있는 페이지

- ShareViewController.swift
- 내가 다녀온 여행에 대해 남들과 공유할 수 있습니다.

Carrier 8:48 PM

back 추천하기 +

국외 너무나 설렜던 유럽여행

2017.10.19 ~ 2017.11.22

언제, 어디를 다녀왔나요?

여행 비용이 얼마였나요?

추천하는 이유를 입력해 주세요.

사진을 첨부해 주세요. [갤러리에서 선택](#)

공유

<커스텀 셀 사용>

```
let urlString: String = "http://condi.swu.ac.kr/student/T03nolleo/insertBasicInfo.php"

guard let requestURL = URL(string: urlString) else { return }
request = URLRequest(url: requestURL)
request.httpMethod = "POST"
let appDelegate = UIApplication.shared.delegate as! AppDelegate
guard let userID = appDelegate.ID else { return }

// DB에 들어갈 변수 설정
let title = textTitle.text!
let area = textArea.text!
let startdate = startDate.text!
let enddate = endDate.text!
let recommendText = recommendReason.text!
let recommendwhenwhere = recommendWhenWhere.text!
let recommendcost = recommendCost.text!

var restString: String = "title=" + title + "&user_id=" + userID
restString = restString + "&area=" + area
restString = restString + "&start_date=" + startdate + "&end_date=" + enddate
restString = restString + "&recommend_reason=" + recommendText
restString = restString + "&recommend_when_where=" + recommendwhenwhere
restString = restString + "&recommend_cost=" + recommendcost
restString = restString + "&recommend_img=" + imageFileName
```

• Database 구성 : MAMP, Core Data

〈MAMP - shared_info〉

#	이름	종류	데이터정렬방식	보기	Null	기본값	추가	실행
<input type="checkbox"/>	1 title	varchar(40)	utf8_general_ci	아니오	없음			변경 삭제 기본 고유값 인덱스 Spatial Fulltext 중복되지 않은 값
<input type="checkbox"/>	2 user_id	varchar(40)	utf8_general_ci	아니오	없음			변경 삭제 기본 고유값 인덱스 Spatial Fulltext 중복되지 않은 값
<input type="checkbox"/>	3 area	varchar(40)	utf8_general_ci	아니오	없음			변경 삭제 기본 고유값 인덱스 Spatial Fulltext 중복되지 않은 값
<input type="checkbox"/>	4 start_date	date		아니오	없음			변경 삭제 기본 고유값 인덱스 Spatial Fulltext 중복되지 않은 값
<input type="checkbox"/>	5 end_date	date		아니오	없음			변경 삭제 기본 고유값 인덱스 Spatial Fulltext 중복되지 않은 값
<input type="checkbox"/>	6 index	int(11)		아니오	없음	AUTO_INCREMENT		변경 삭제 기본 고유값 인덱스 Spatial Fulltext 중복되지 않은 값
<input type="checkbox"/>	7 recommend_reason	varchar(600)	utf8_general_ci	아니오	없음			변경 삭제 기본 고유값 인덱스 Spatial Fulltext 중복되지 않은 값
<input type="checkbox"/>	8 recommend_when_where	varchar(40)	latin1_swedish_ci	아니오	없음			변경 삭제 기본 고유값 인덱스 Spatial Fulltext 중복되지 않은 값
<input type="checkbox"/>	9 recommend_cost	varchar(10)	latin1_swedish_ci	아니오	없음			변경 삭제 기본 고유값 인덱스 Spatial Fulltext 중복되지 않은 값
<input type="checkbox"/>	10 recommend_img	text	utf8_general_ci	예	NULL			변경 삭제 기본 고유값 인덱스 Spatial Fulltext 중복되지 않은 값

- Title : 공유한 여행기의 제목
- User_id : 여행기를 공유한 유저의 id
- Area : 여행기의 지역 구분 (국내/국외)
 - Start_date : 여행의 시작 날짜
 - End_date : 여행의 종료 날짜
- Index : 자동으로 증가하는 여행기의 번호
- Recommend_reason : 여행 추천 이유
- Recommend_when_where : 언제, 어디를 갔는지 설명
 - Recommend_cost : 여행에 든 비용
- Recommend_img : 여행기에 첨부한 이미지

- Database 구성 : MAMP, Core Data

〈MAMP - user〉

#	이름	종류	데이터정렬방식	보기	Null	기본값	추가	실행
<input type="checkbox"/> 1	user_id	varchar(10)	utf8_general_ci	아니오	없음		변경 삭제 기본 고유값 인덱스 Spatial Fulltext 중복되지 않은 값	
<input type="checkbox"/> 2	user_pw	varchar(40)	utf8_general_ci	아니오	없음		변경 삭제 기본 고유값 인덱스 Spatial Fulltext 중복되지 않은 값	
<input type="checkbox"/> 3	user_name	varchar(10)	utf8_general_ci	아니오	없음		변경 삭제 기본 고유값 인덱스 Spatial Fulltext 중복되지 않은 값	
<input type="checkbox"/> 4	user_gender	varchar(1)	utf8_general_ci	아니오	없음		변경 삭제 기본 고유값 인덱스 Spatial Fulltext 중복되지 않은 값	
<input type="checkbox"/> 5	user_birth	date		아니오	없음		변경 삭제 기본 고유값 인덱스 Spatial Fulltext 중복되지 않은 값	
<input type="checkbox"/> 6	profile_img	text	utf8_general_ci	예	NULL		변경 삭제 기본 고유값 인덱스 Spatial Fulltext 중복되지 않은 값	

- User_id : 사용자의 아이디
- User_pw : 사용자의 비밀번호
- User_name : 사용자의 이름
- User_gender : 사용자의 성별
- User_birth : 사용자의 생일
- Profile_img : 사용자의 프로필 사진

- Database 구성 : MAMP, Core Data

〈Core Data - BasicInfo : 여행기의 기본 정보〉

ENTITIES	▼ Attributes		
E BasicInfo	Attribute ▼	Type	
FETCH REQUESTS	S title	String	↕
CONFIGURATIONS	D startdate	Date	↕
C Default	D savedate	Date	↕
	D enddate	Date	↕
	S area	String	↕
	+ -		

- Title : 여행기의 제목
- Startdate : 여행의 시작 날짜
- Savedate : 여행기를 저장한 날짜
- Enddate : 여행의 종료 날짜
- Area : 여행기의 지역 구분 (국내/국외)

- Database 구성 : MAMP, Core Data

〈 Core Data - DetailInfo : 여행기의 각 날짜별 세부 정보〉

ENTITIES	
E BasicInfo	
E Detail	
FETCH REQUESTS	
CONFIGURATIONS	
C Default	

▼ Attributes	
Attribute ^	Type
S cost	String
S daycount	String
S memo	String
S place	String
S time	String
S title	String
+ -	

- Cost : 여행에 든 비용
- Daycount : 여행 며칠째인지 알려주는 수 (ex. 첫째 날일 경우 1, 둘째 날일 경우 2...)
 - Memo : 여행한 날마다 기록할 수 있는 메모
 - Place : 방문한 장소
 - Time : 방문한 시각
 - Title : 여행기의 제목