

2018-1 모바일앱프로그래밍

프로젝트 #2 중간보고서

컴퓨터학과 14 김소연

- Intro : 보다 가벼운 국내, 해외 여행 일정 관리 - 놀러! (nolleo!)

이름 김소연

학번 2014111517

전공 컴퓨터

E-Mail ese2003@naver.com

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.

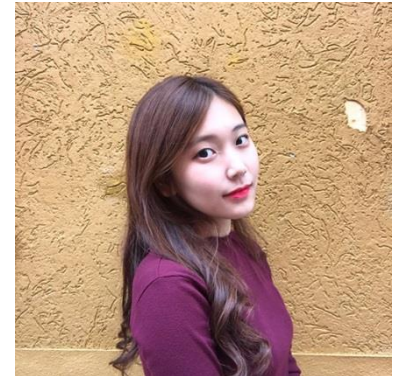
기존 여행 기록 어플은 많이 있으나, 여행 지출 내역을 정리해서 보여주는 어플인

‘세이브트립’과 손쉽게 블로그 글처럼 여행기를 기록할 수 있는 ‘볼로’는 아이폰을 지원하지 않는다.

또한 기타 어플인 ‘핫츠고플랜’, ‘여행노트’는 타 사용자와의 여행기 공유와

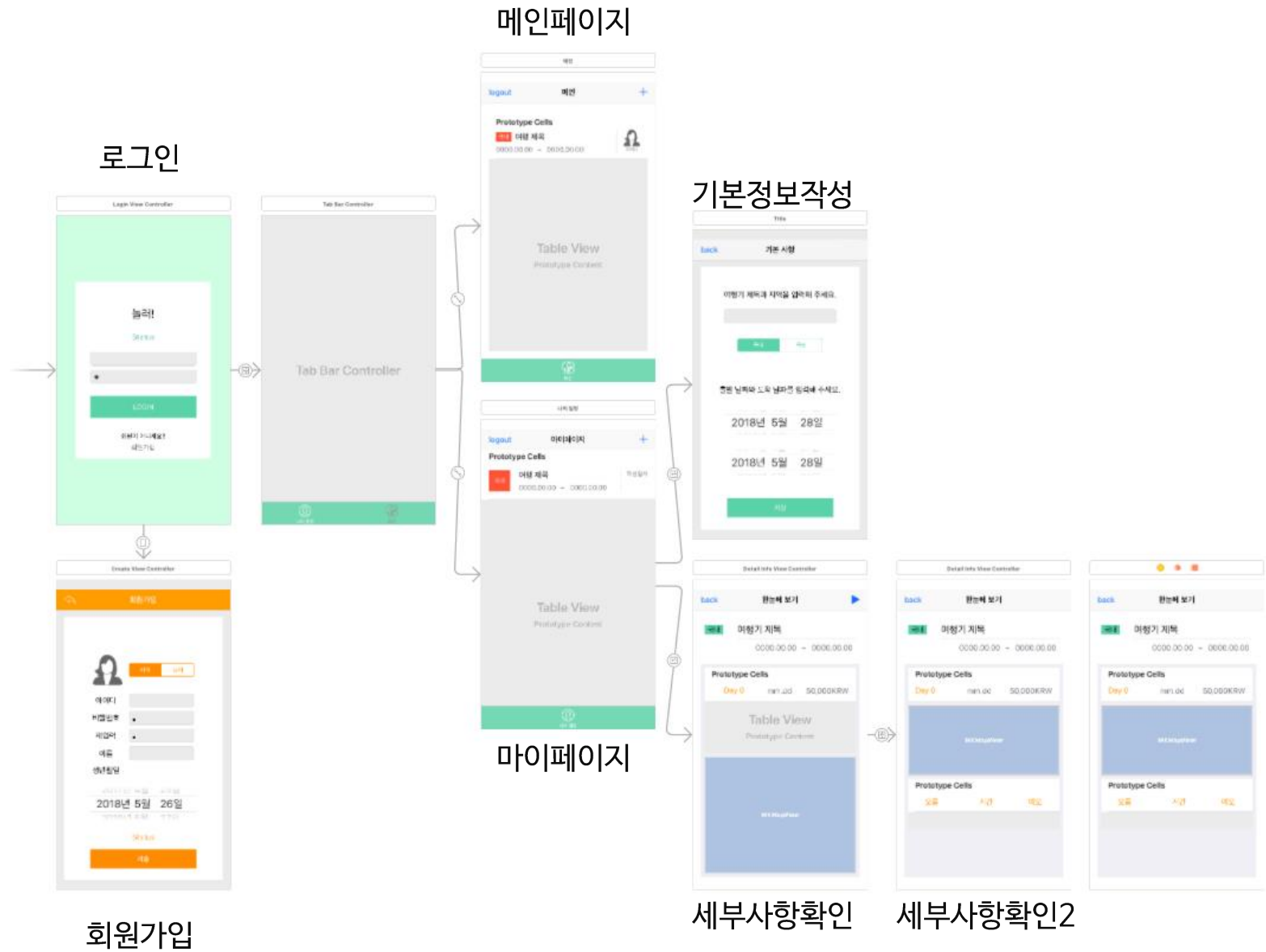
혼자만의 여행 기록 둘 중 하나에만 초점을 두고 있기 때문에

모든 기능을 갖춘, 그러나 직관적이고 간단한 어플을 만들고 싶었다.



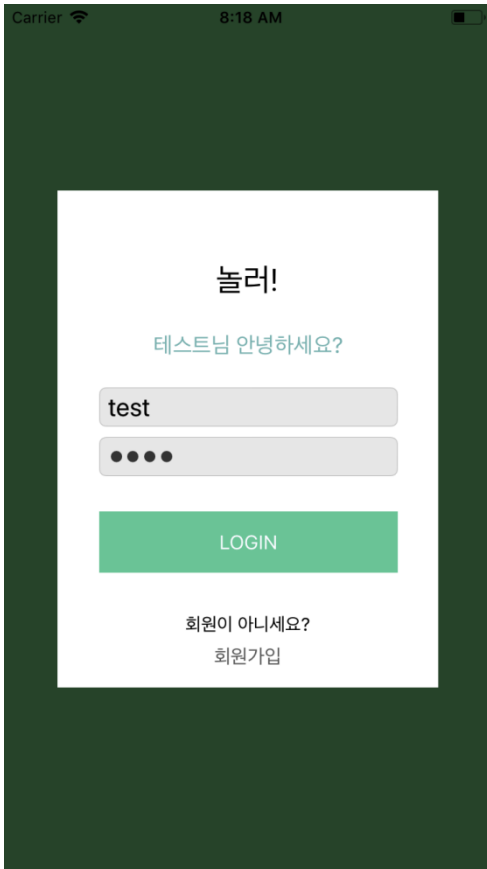
 github.com/aa9390

- Storyboard



• 로그인

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



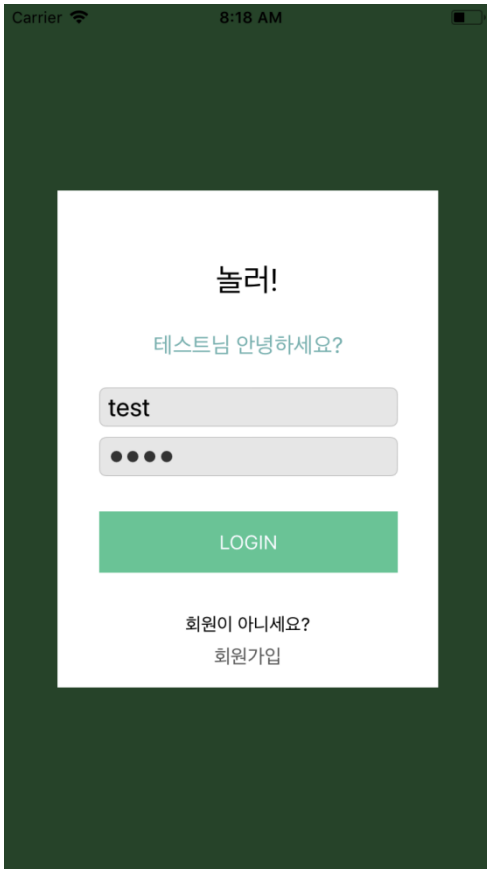
〈입력값 검증〉

```
// 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, return함.

• 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"

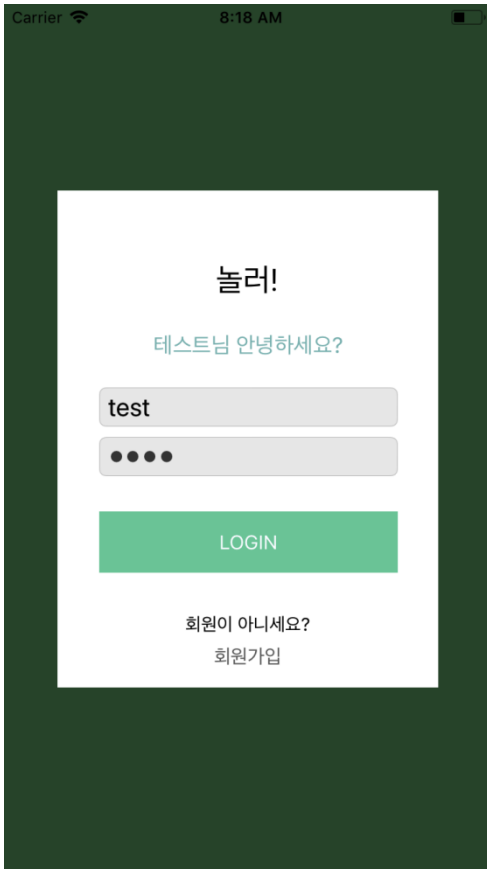
let restString: String = "id=" + loginUserid.text! + "&password=" + loginUserpw.text!
request.httpBody = restString.data(using: .utf8)

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.



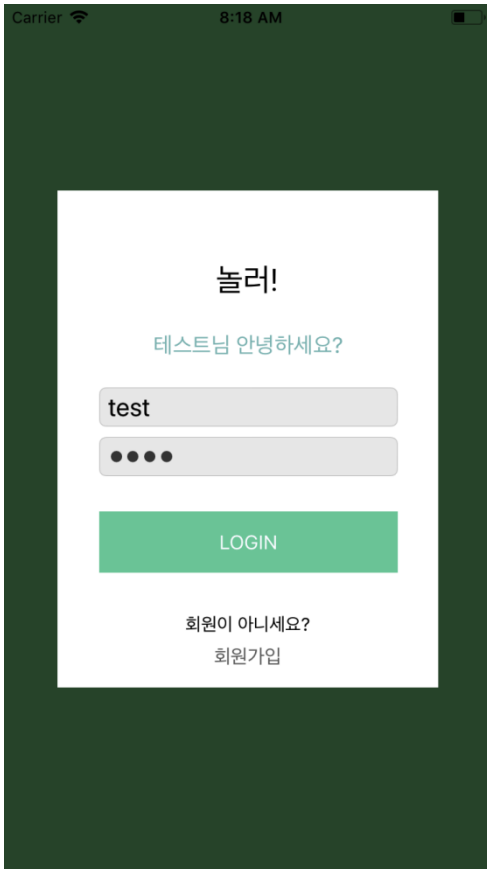
<loginUser.php>

```
<?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.



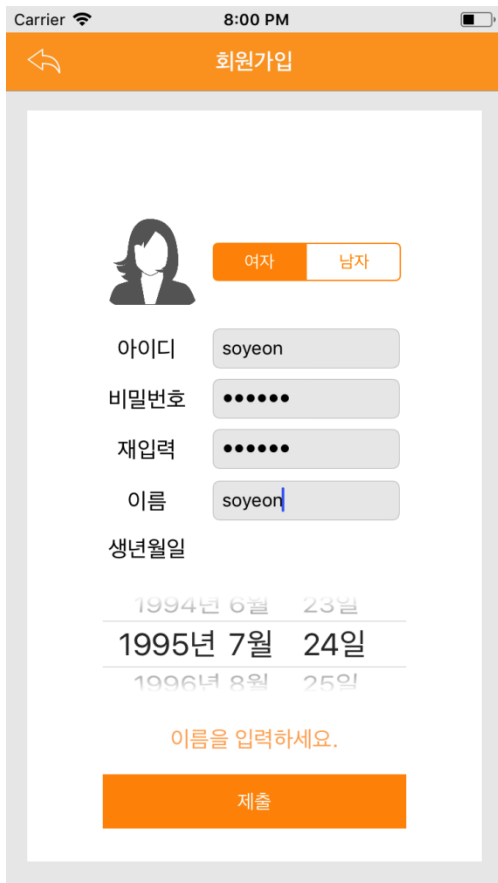
〈서버와의 통신 - 로그인 성공〉

```
// 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.



The image shows a mobile app interface for user registration. At the top, there's a status bar with 'Carrier', signal strength, '8:00 PM', and battery level. Below it is an orange header bar with a back arrow and the text '회원가입'. The main content area is white and contains a registration form. On the left is a female silhouette icon. To its right are two buttons: '여자' (selected) and '남자'. Below these are input fields for '아이디' (soyeon), '비밀번호' (masked with dots), '재입력' (masked with dots), and '이름' (soyeon). At the bottom, there's a date picker showing '1995년 7월 24일' and a red error message '이름을 입력하세요.' (Please enter your name). At the very bottom is an orange '제출' (Submit) button.

〈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.

〈서버와의 통신〉

Carrier 8:00 PM

회원가입

여자 남자

아이디 soyeon

비밀번호

재입력

이름 soyeon

생년월일

1994년 6월 23일

1995년 7월 24일

1996년 8월 25일

이름을 입력하세요.

제출

```
// 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.
// 여기서의 참인 부분이 없으므로 아무것도 실행하지 않음.
guard let requestURL = URL(string: urlString) else {
    return
}

var 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

//
restString = restString + "&birth=" + birth

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

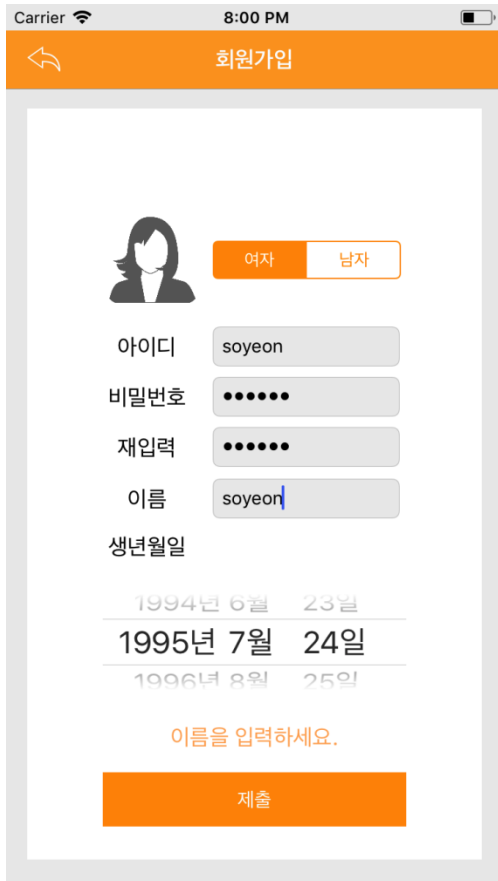
self.executeRequest(request: request)

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

- 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.



The image shows a mobile app interface for user registration. At the top, there's a status bar with 'Carrier', signal strength, and time '8:00 PM'. Below it is an orange header bar with a back arrow and the text '회원가입'. The main content area has a white background with a gray border. It features a profile icon placeholder, gender selection buttons ('여자' and '남자'), and input fields for '아이디' (soyeon), '비밀번호' (masked with dots), '재입력' (masked with dots), '이름' (soyeon), and '생년월일' (1995년 7월 24일). A red text prompt '이름을 입력하세요.' is visible, and an orange '제출' button is at the bottom.

<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']);
$sql= "insert into user (user_id, user_pw, user_name, user_gender, user_birth)
values ('$userid', '$password', '$name', '$gender', '$birth')";

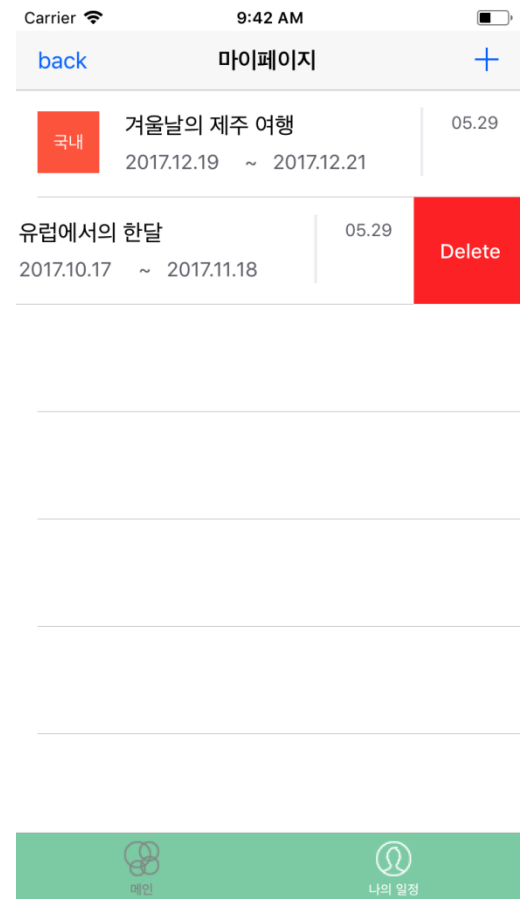
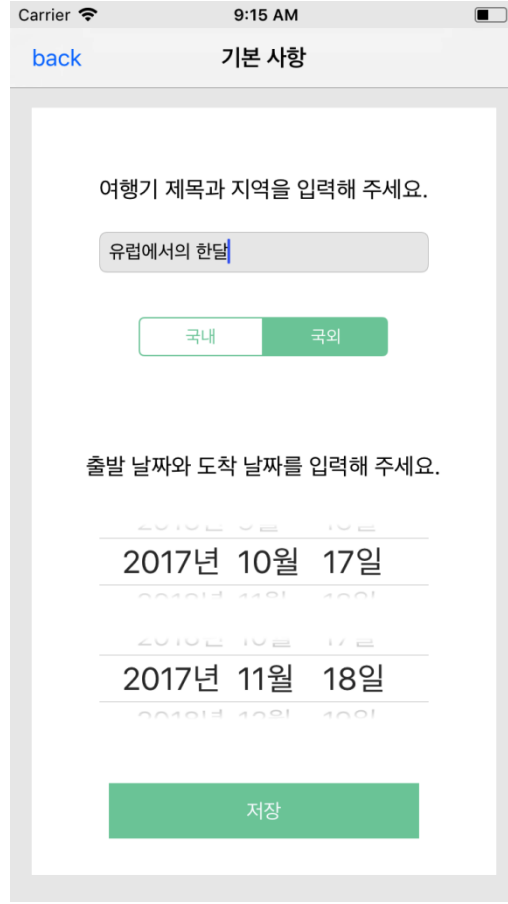
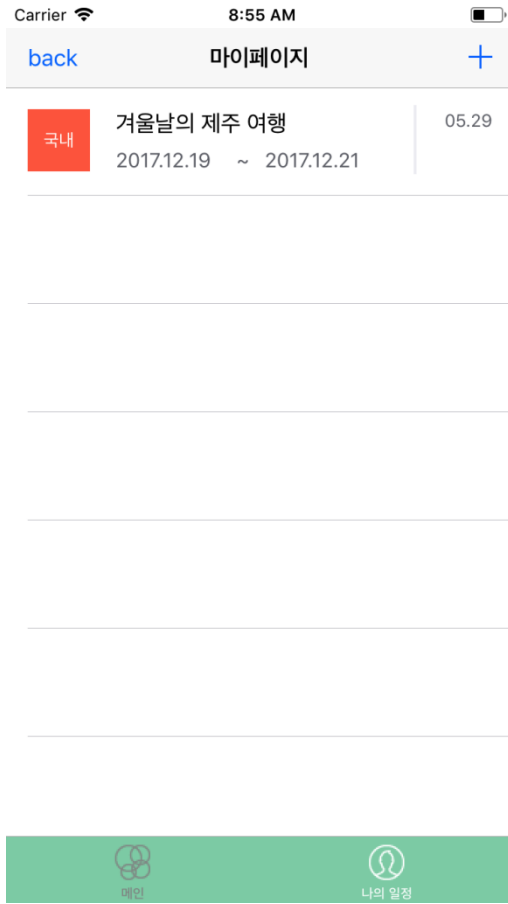
if (mysqli_query($conn, $sql)) {
    echo '가입되었습니다. ';
}
else
{
    echo mysqli_error($conn);
}
mysqli_close($conn);
?>
```

+ 옵션

		user_id	user_pw	user_name	user_gender	user_birth
<input type="checkbox"/>	수정	soyeon	74502e030dba1d1dad42db721faafc7b	ê±€i+CEi—°	i	1995-07-24
<input type="checkbox"/>	수정	test	098f6bcd4621d373cade4e832627b4f6	test	i	2018-06-03

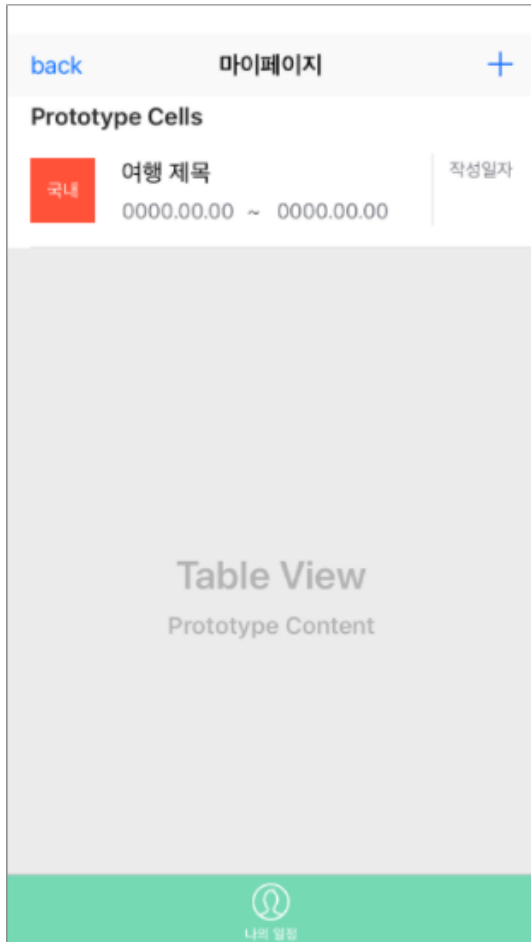
● 마이페이지

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



• 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)

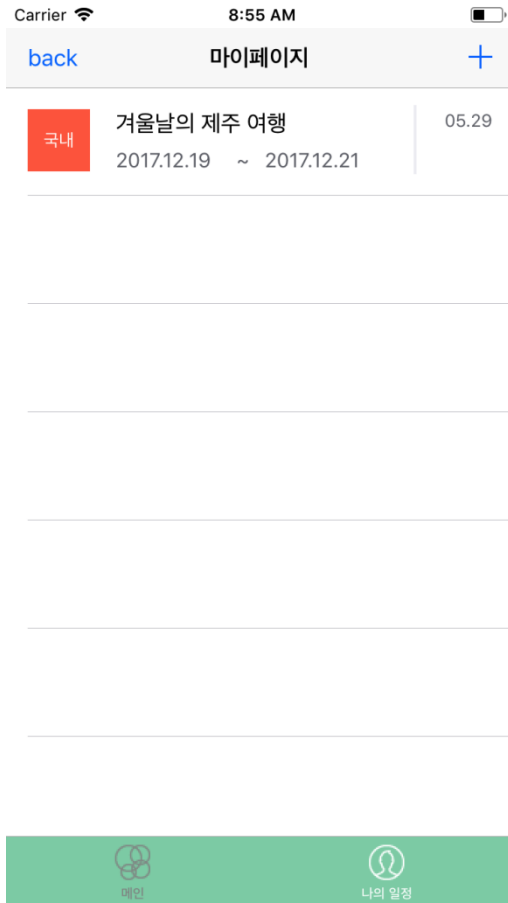
        // Configure the view for the selected state
    }

}
```

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

● 마이페이지

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



〈커스텀 셀 사용 명시〉

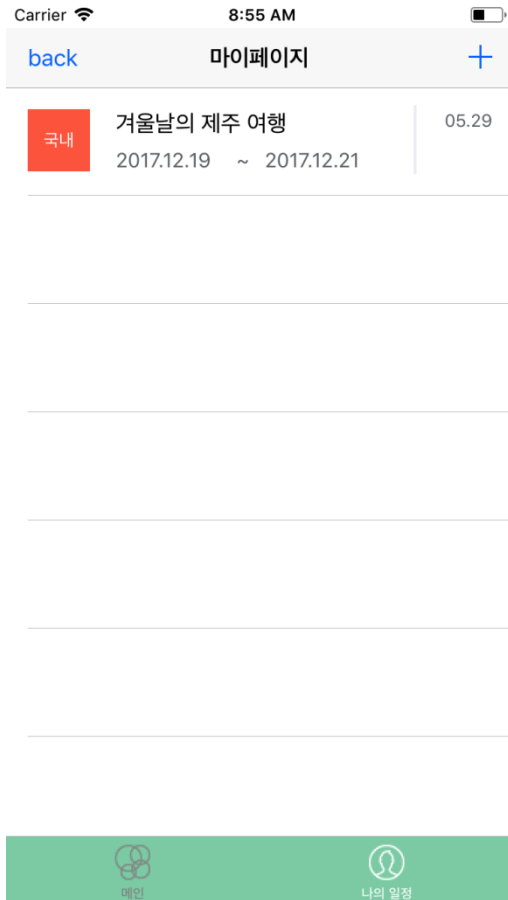
```
override func tableView(_ tableView: UITableView, cellForRowAt indexPath: IndexPath)
-> UITableViewCell {
    // 커스텀 셀 사용함을 명시

    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.

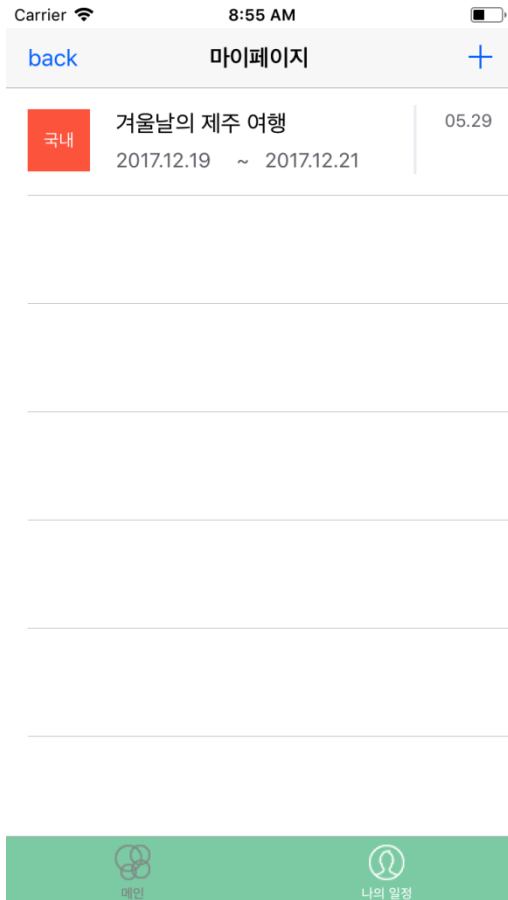
〈셀 삭제〉

```
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.



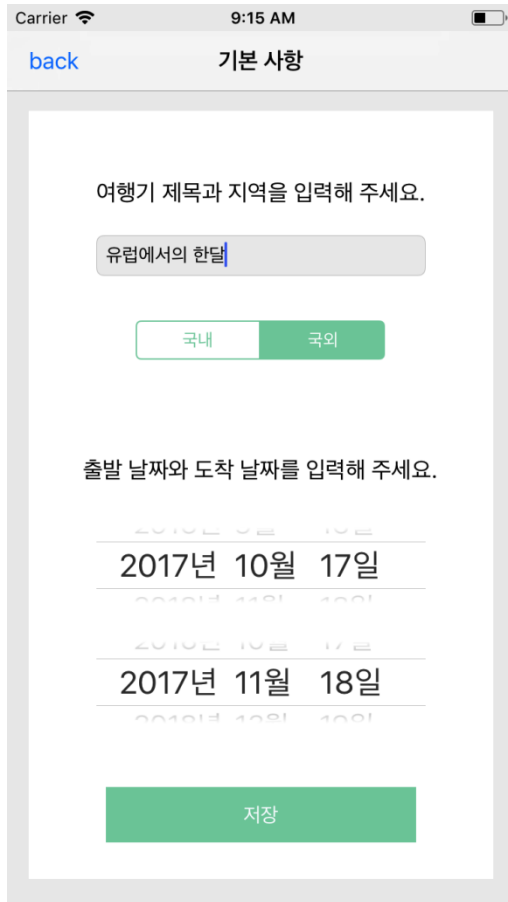
〈셀 클릭 시 클릭한 정보를 배열로 전송〉

```
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")

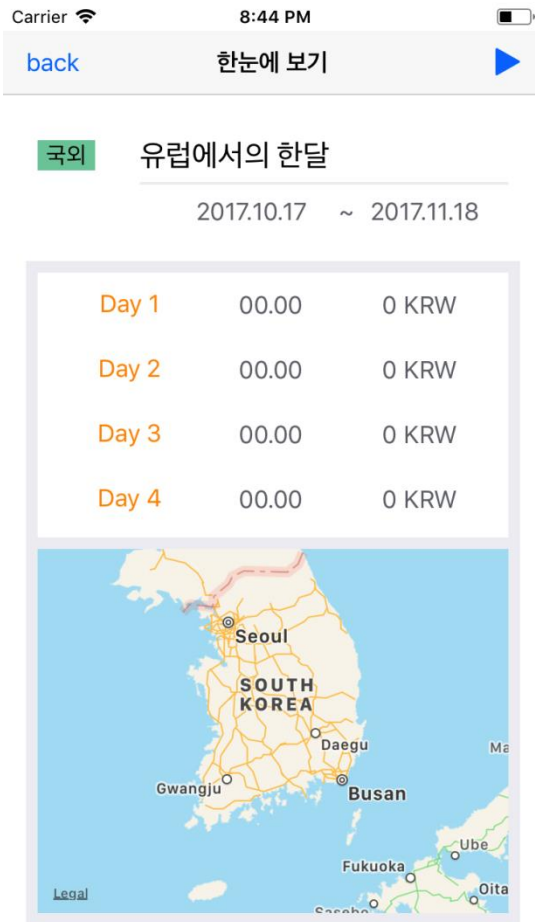
    do {
        try context.save()
        print("저장되었습니다.")
    } catch let error as NSError {
        print("저장하지 못했습니다. \(error), \(error.userInfo)") }

    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.



〈기본 사항을 Core Data에서 불러옴〉

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

    if let basicToDetail = basic {
        textTitle.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
        }

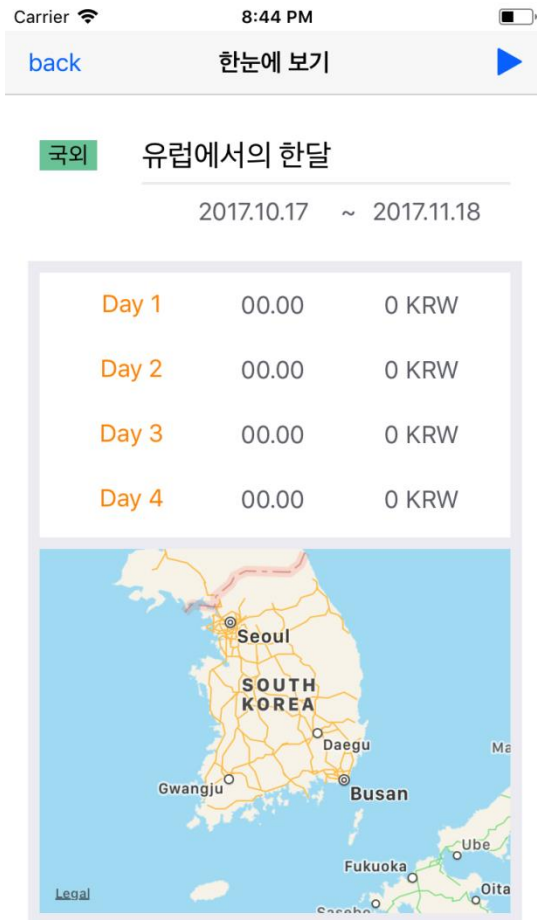
        dayInterval = savedEnddate!.timeIntervalSince(savedStartdate!)
        daysInterval = Int(dayInterval! / 86400)

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

- viewDidLoad()에서 기본 사항을 불러온다.
- daysInterval은 여행 종료일자와 시작 일자를 계산하여 총 며칠 여행했는지를 저장하는 변수이다.

• 세부 사항 확인 페이지

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



<DetailInfoTableViewCell.swift>

```
class DetailInfoTableViewCell: UITableViewCell {

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

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

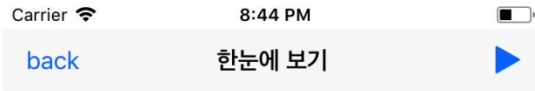
    override func setSelected(_ selected: Bool, animated: Bool) {
        super.setSelected(selected, animated: animated)

        // 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.



국외 유럽에서의 한달

2017.10.17 ~ 2017.11.18

Day 1	00.00	0 KRW
Day 2	00.00	0 KRW
Day 3	00.00	0 KRW
Day 4	00.00	0 KRW

〈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
}
```

• Database 구성 : MAMP, Core Data

- 앞으로 더 추가할 예정입니다.

〈MAMP〉

#	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	

〈Core Data〉

ENTITIES	▼ Attributes
BasicInfo	
FETCH REQUESTS	
CONFIGURATIONS	
Default	
	Attribute ▾
	Type
	title String
	startdate Date
	savedate Date
	enddate Date
	area String
	+ -