fast campus Copyright FASTCAMPUS Corp.All Rights Reserved

초격차 패키지 Online.

안녕하세요
모바일 개발자
남반석 입니다.

PART3 | 활용

오리엔테이션

1. 강사 소개

2. 모바일 앱 개발 개론

3. Part3를 통해서 여러분들이 배우게 될 것들

Fast campus Copyright FASTCAMPUS Corp.All Rights Reserved

1 강사 소개

Naver

2013년 9월 ~

Cafe(인턴)

Post

Blog

Disco

AudioClip

Selective (쇼핑라이브)

Lawfully

2019년 6월 ~
현재

USCIS Case Tracker

Native => Flutter 전환 => 현재

185 국가
100만 다운로드
140만 케이스 등록
28만 Monthly Active User

플러터로 만든 소개 웹페이지

https://bansook.xyz/

Fast campus Copyright FASTCAMPUS Corp.All Rights Reserved

2. 모바일 앱 개발 개론

1 모바일 앱 개발 과정 톺아보기

2 좋은 모바일 개발자와 잘하는 모바일 개발자

3 실전 사례 적용

Fast campus Copyright FASTCAMPUS Corp.All Rights Reserved

1. 모바일 앱 개발 과정 톺아보기

전지적 대표님 입장

로그인 안해도 들을 수 있는
뮤직 플레이어 앱을 만들어줘

뚝딱뚝딱 🔨

기획/디자인/개발

Fast campus Copyright FASTCAMPUS Corp.All Rights Reserved

1. 모바일 앱 개발 과정 톺아보기

Fast campus Copyright FASTCAMPUS Corp.All Rights Reserved

1. 모바일 앱 개발 과정 톺아보기

Fast campus Copyright FASTCAMPUS Corp.All Rights Reserved

갑자기 인기가 많아져서

유저가 폭증

Fast campus Copyright FASTCAMPUS Corp.All Rights Reserved

1. 모바일 앱 개발 과정 톺아보기

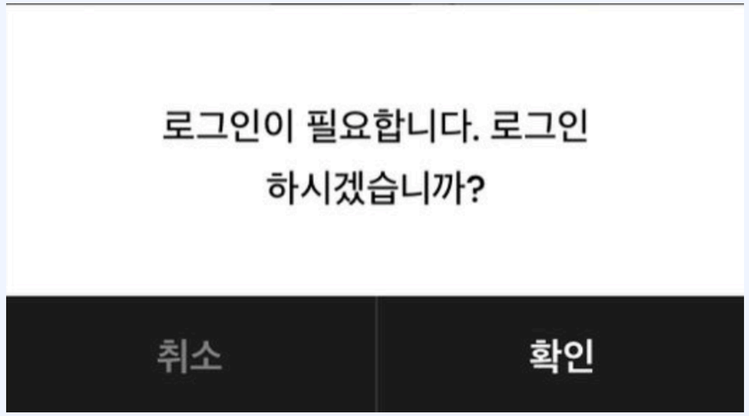
전지적 대표님 혹은 기획자 (Business Developer) Says

유저가 많아졌으니,
회원가입 기능을 추가하고,
좋아요를 로컬이 아닌
서버에 저장시켜주세요

Fast campus Copyright FASTCAMPUS Corp.All Rights Reserved

개발팀과 기획자의 결정

좋아요를 누를때, "로그인을 하시겠습니까?" 다이얼로그를 띄웁시다.

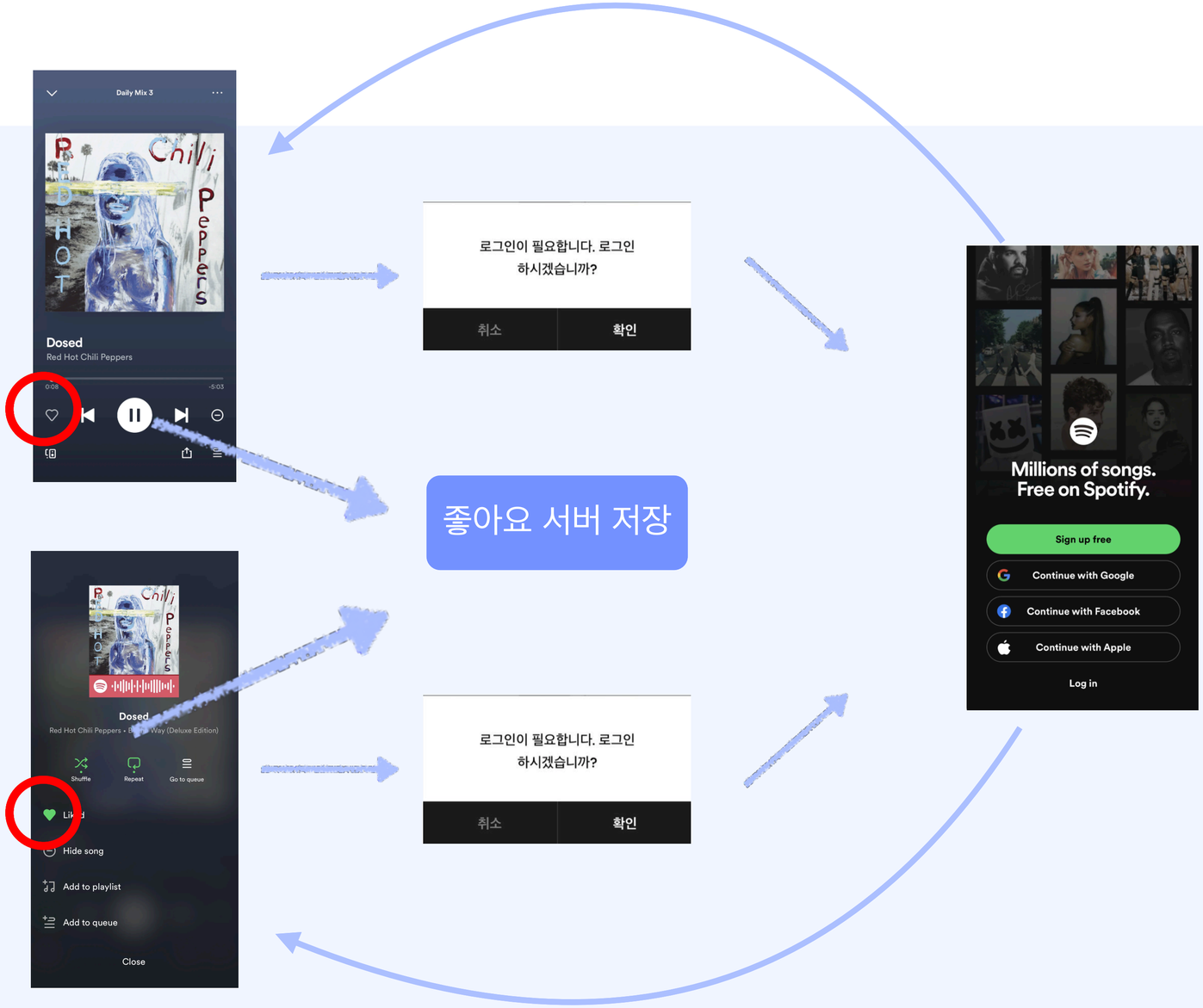


Fast campus Copyright FASTCAMPUS Corp.All Rights Reserved

1. 모바일 앱 개발 과정 톺아보기

A 플레이어 화면

B 곡 더보기 화면



Fast campus Copyright FASTCAMPUS Corp.All Rights Reserved

1. 모바일 앱 개발 과정 톺아보기

Fast campus Copyright FASTCAMPUS Corp.All Rights Reserved

1 좋은 모바일 개발자

2 잘하는 모바일 개발자

Fast campus Copyright FASTCAMPUS Corp.All Rights Reserved

1 좋은 모바일 개발자

1) 안해본 새로운 작업이 맡겨졌을때 해보겠다고 혹은 알아보겠다고 말한다. (ex: 이래서 이래서 안되요 => ❌)

2) 아이디어를 낸 동료와 그 아이디어를 존중한다. (그렇게 꼭 해야되나요? => ❌, 그렇게 하면 코드가 복잡해져요 => ❌)

3) 아이디어가 별로라고 생각한다면,

   더 나은 대안과 생각을 기분 상하지 않게 충분히 잘 설명해줘야한다.

4) 유저의 입장에서 좋은 방향을 계속 고민하고 구현한다.

5) 자신이 알게된 좋은 것들을 다른 개발자와 공유한다.

Fast campus Copyright FASTCAMPUS Corp.All Rights Reserved

2 잘하는 모바일 개발자

1) 다양한 경험과 스펙트럼을 알고 있는 개발자
   - 남들이 몇 주동안 할 일을, 좋은 툴을 찾아서 하루만에 끝낼 수도 있다.
2) 코드 결과물의 가독성이 좋은 개발자 (지속적으로 고민하고 개선해서 더 이상 고칠 부분을 찾기 힘들때까지)
3) 설계를 잘하는 개발자 (스펙이 변경될때도 변경점에 잘 맞추고, 또 앞으로 변경될 상황을 미리 파악해서 설계할 줄 아는 개발자.)
4) 문제 없는 안정적인 코드를 잘 구현하는 개발자.
   - 출시 전에 실수나 버그를 잘 찾는 능력자
   - 출시 후에 문제를 찾았다면, 고치는 것 뿐 아니라
     - 테스트 코드로 재발 방지
     - QA 테스트 규칙들을 추가하여 재발 방지
5) 본인의 작업을 구현하고 남는 시간을 확보해서 지속적인 공부를 하는 개발자.

Fast campus Copyright FASTCAMPUS Corp.All Rights Reserved

실전 사례 적용

1 좋은 모바일 개발자

4) 유저의 입장에서 좋은 방향을 계속 고민하고 구현한다.

2 잘하는 모바일 개발자

3) 설계를 잘하는 개발자

Fast campus Copyright FASTCAMPUS Corp.All Rights Reserved

실전 사례 적용

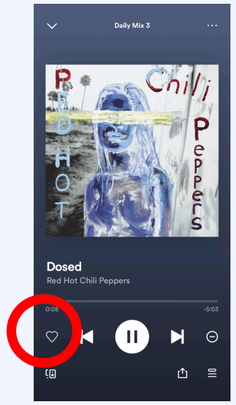
개선 포인트1 - 각 버튼에서 로그인 다이얼로그를 띄우는 코드 합치기

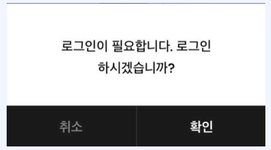
설계 개선 - 중복 로직을 제거

Fast campus Copyright FASTCAMPUS Corp.All Rights Reserved

개선 포인트1 - 각 버튼에서 로그인 팝업을 띄우는 코드 합치기

A 플레이어 화면



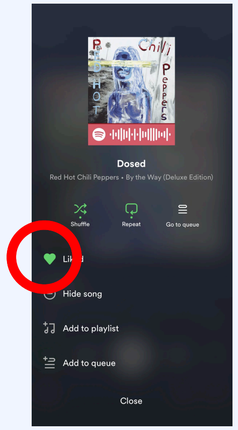


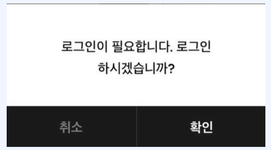

```
LikeButton1(
  onTap: () async {
    if (LoginManager.isNotLoggedIn) {
      final dialogResult = await LoginDialog().show();
      if (dialogResult == null || dialogResult.isFailure) {
        return;
      }
      Nav.push(LoginScreen());
      return;
    }
    await LikeApi.likeSong(songId);
  }),  // LikeButton1
```

1. 로그인을 안한 경우
2. 로그인 유도 다이얼로그
3. 취소한 경우 함수 종료 (return)
4. 로그인 화면 띄우고 종료
5.로그인 안한 경우 서버 API 호출

(예제코드)

B 곡 더보기 화면

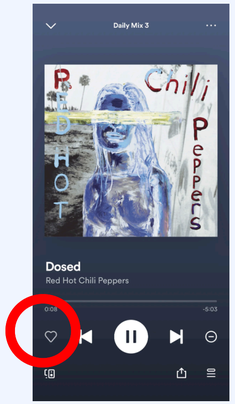




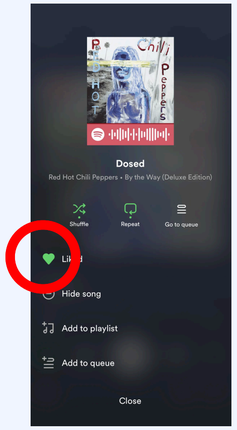

```
LikeButton2(
  onTap: () async {
    if (LoginManager.isNotLoggedIn) {
      final dialogResult = await LoginDialog().show();
      if (dialogResult == null || dialogResult.isFailure) {
        return;
      }
      Nav.push(LoginScreen());
      return;
    }
    await LikeApi.likeSong(songId);
  }),  // LikeButton2
```

Fast campus Copyright FASTCAMPUS Corp.All Rights Reserved

개선 포인트1 - 각 버튼에서 로그인 팝업을 띄우는 코드 합치기




```
child: LikeButton1(
  onTap: () async {
    likeLogic.likeSong(songId, context);
  },
),  // LikeButton1
```


```
child: LikeButton2(
  onTap: () async {
    likeLogic.likeSong(songId, context);
  },
),  // LikeButton2
```

```
class LikeLogic {
  void likeSong(int songId, BuildContext context) async {
    if (LoginManager.isNotLoggedIn) {
      final dialogResult = await LoginDialog().show();
      if (dialogResult == null || dialogResult.isFailure) {
        return;
      }
      Nav.push(LoginScreen());
      return;
    }
    await LikeApi.likeSong(songId);
  }
```

Fast campus Copyright FASTCAMPUS Corp.All Rights Reserved

개선 포인트2 - 좋아요 한번만 눌러도 동작하도록 변경

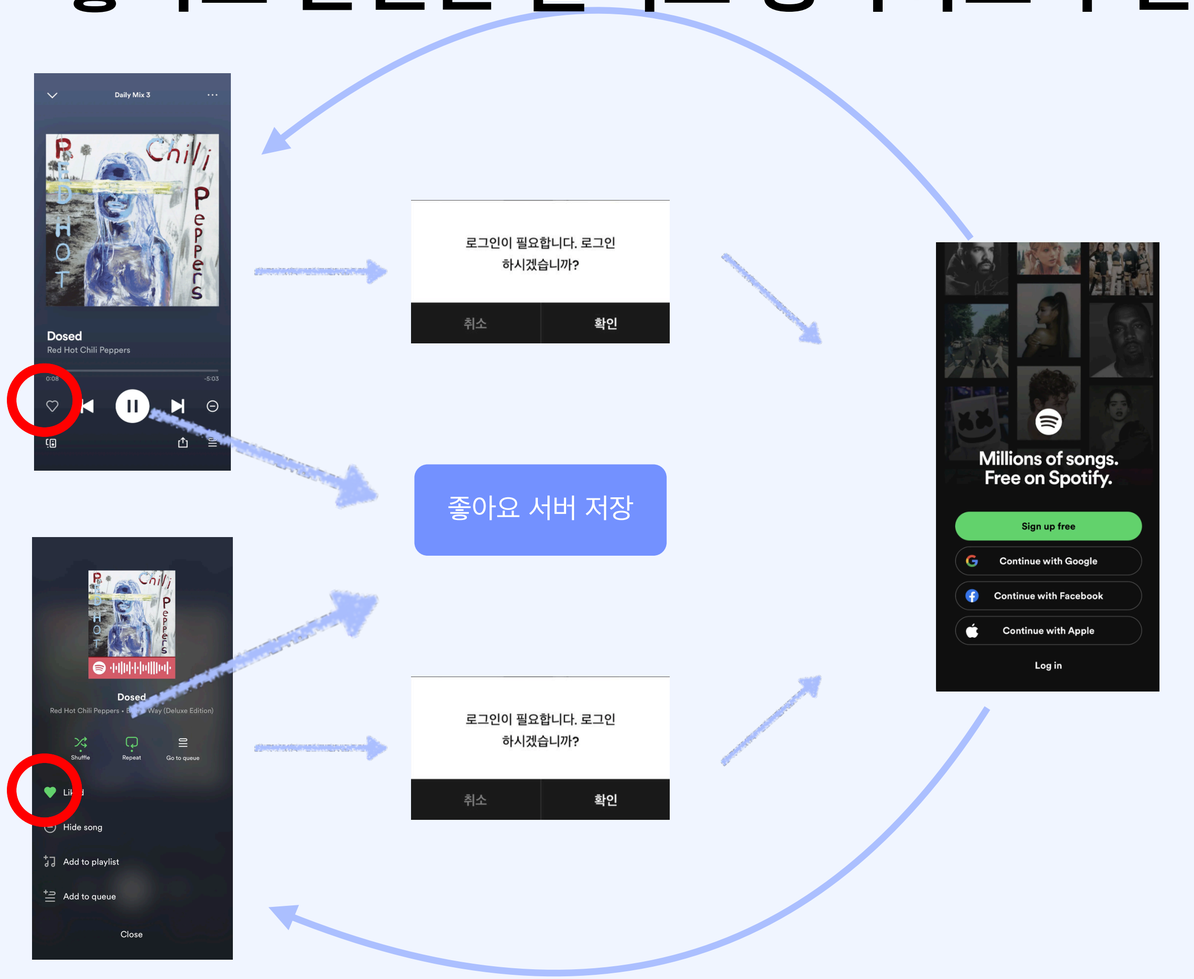
- 유저의 입장에서 좋은 방향을 계속 고민하고 구현한다.
- 설계 개선

Fast campus Copyright FASTCAMPUS Corp.All Rights Reserved

개선 포인트2 - 좋아요 한번만 눌러도 동작하도록 변경

A 플레이어 화면

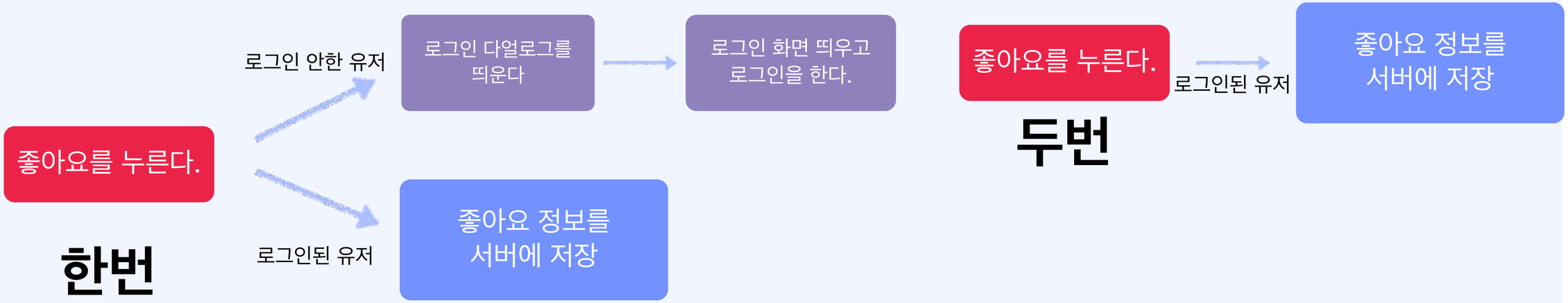
B 곡 더보기 화면



Fast campus Copyright FASTCAMPUS Corp.All Rights Reserved

개선 포인트2 - 좋아요 한번만 눌러도 동작하도록 변경

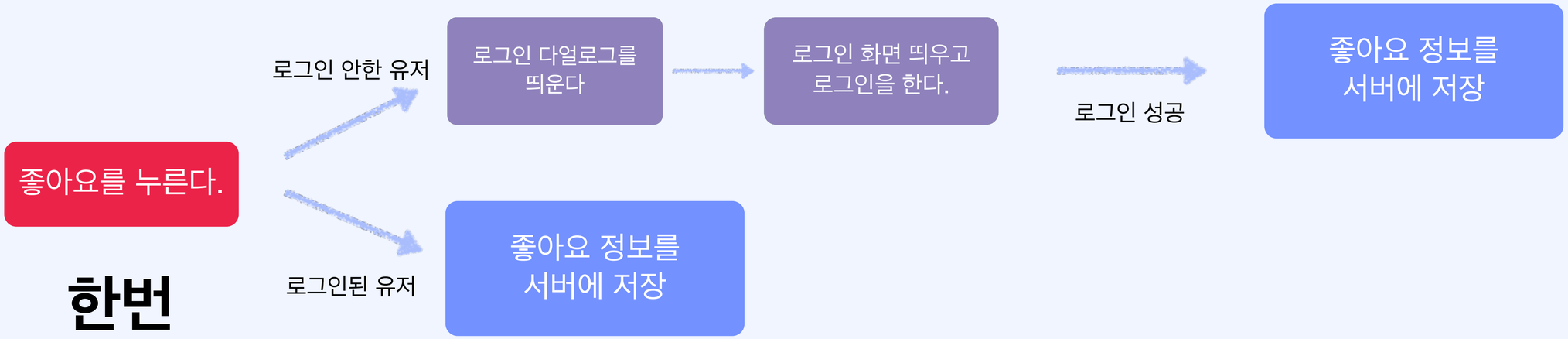
As-Is



Fast campus Copyright FASTCAMPUS Corp.All Rights Reserved

개선 포인트2 - 좋아요 한번만 눌러도 동작하도록 변경

To-Be



Fast campus Copyright FASTCAMPUS Corp.All Rights Reserved

개선 포인트2 - 좋아요 한번만 눌러도 동작하도록 변경

```
void likeSong(int songId, BuildContext context) async {
  if (LoginManager.isNotLoggedIn) {
    final dialogResult = await LoginDialog().show();
    if (dialogResult == null || dialogResult.isFailure) {
      return;
    }
    final SimpleResult? loginResult = await Nav.push(LoginScreen());
    if (loginResult == null || loginResult.isFailure) {
      return;
    }
  }
  await LikeApi.likeSong(songId);
}
```

1. 로그인을 안한 경우에

2. 로그인 유도 다이얼로그

3. 취소한 경우 함수 종료 (return)

4. 로그인 화면 띄움

5. 로그인 화면에서 로그인 안하거나 실패한 경우
함수 종료 (return)

6. 로그인 화면에서 로그인 성공한 경우
서버에 API 호출

Fast campus Copyright FASTCAMPUS Corp.All Rights Reserved

개선 포인트2 - 좋아요 한번만 눌러도 동작하도록 변경

```
class LikeLogic {
  void likeSong(int songId, BuildContext context) async {
    if (await LoginManager.checkIsNotLoggedInAndShowDialog()) {
      return;
    }
    await LikeApi.likeSong(songId);
  }
```

```
class LoginManager {
  static Future<bool> checkIsNotLoggedInAndShowDialog() async {
    if(isLoggedIn) {
      return false;
    }
    final dialogResult = await LoginDialog().show();
    if (dialogResult == null || dialogResult.isFailure) {
      return true;
    }
    final SimpleResult? loginResult = await Nav.push(LoginScreen());
    if (loginResult == null || loginResult.isFailure) {
      return true;
    }
    return false;
  }
```

Fast campus Copyright FASTCAMPUS Corp.All Rights Reserved

Flutter에서는 모든 변화가 함수로 이루어져있고, 통제 가능합니다.

좋아요를 누른다.

- UI 화면 그리기
- 유저 액션 처리

로그인 다얼로그를 띄운다

로그인 화면 띄우고 로그인을 한다.

- 화면 이동
- 화면 결과 처리

좋아요 정보를 서버에 저장

- 비지니스 로직 흐름 처리

Fast campus Copyright FASTCAMPUS Corp.All Rights Reserved

앱 개발이란

- 앱이 실행되고 나서 일어나는 모든 일들을
  다양한 함수로 구성하고, 순서에 맞게 잘 수행되게 하는 작업

- 유저들에게 편의와 가치를 제공하는 일을 하는 것

Fast campus Copyright FASTCAMPUS Corp.All Rights Reserved

3. 여러분이 배우게 될 것들

Fast campus Copyright FASTCAMPUS Corp.All Rights Reserved

3. 여러분이 배우게 될 것들

3.
Part3를 통해서 여러 분들이 배우게 될 것들

Flutter 앱을 만들때 꼭 필요한 기능들과 구현 방법

- Package Manager
- 로컬 데이터 저장
- FCM Push
- Animation
- 네트워크 통신
- Deeplink
- Advanced Dart
- 갤러리 & 카메라 사진
- 다국어 처리
- 상태 관리
  - GetX
  - Bloc
  - RiverPod
  - Flutter Hook
- Navigator

Fast campus Copyright FASTCAMPUS Corp.All Rights Reserved

자 그럼 시작해볼까요?