

# Part4. 앱 리소스 사용 학습

## ChatGPT API 활용 클로바 스타일 녹음요약 앱 프로젝트

**프로젝트3 - 내용 요약해주는 AI 녹음기**

# **학습 목표 및 구성**

## 학습 목표

01

웹 서비스와 앱 서비스를 동시에 개발 할 수 있다

02

Nextjs를 이용하여 웹 서비스를 개발할 수 있다

03

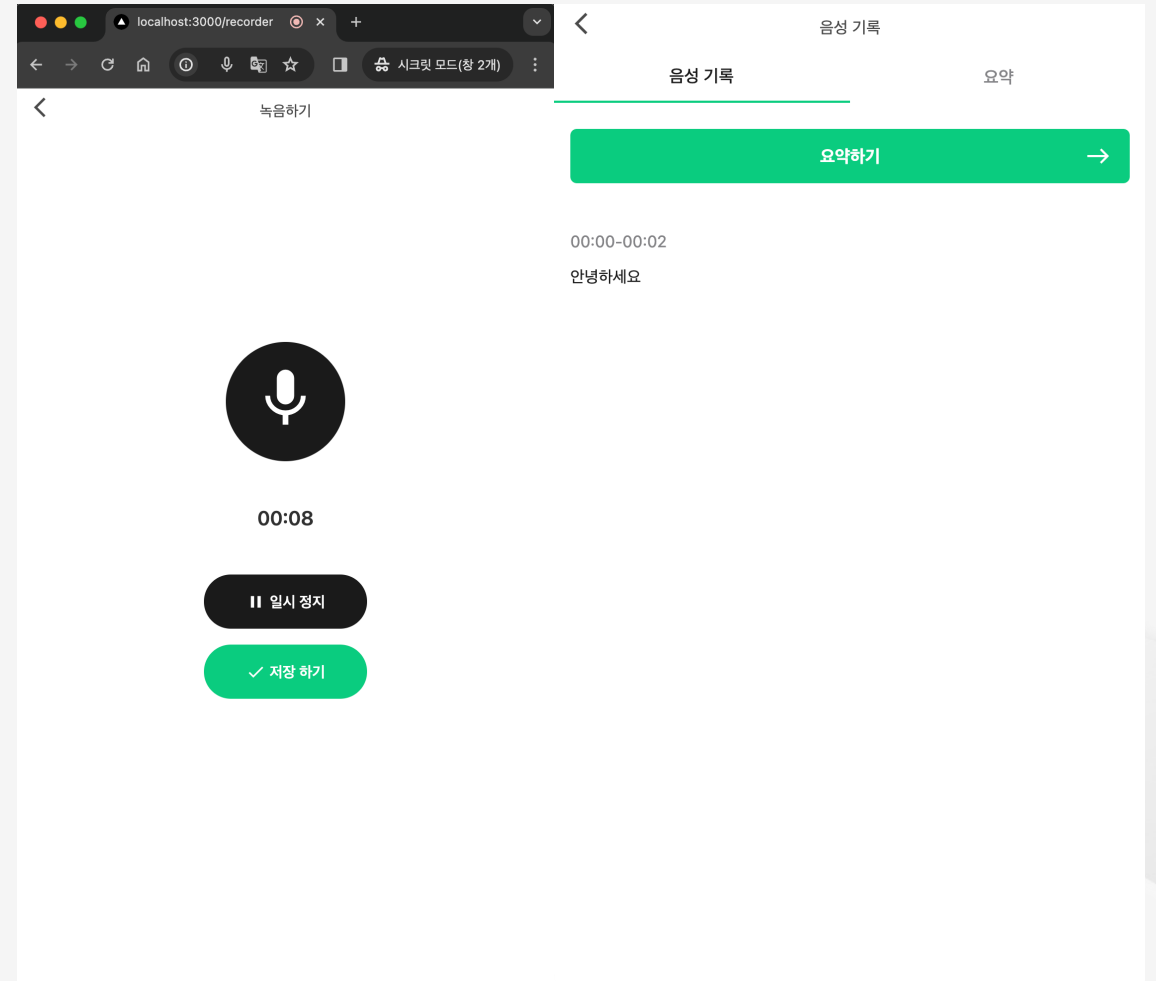
웹 사이트와 앱 리소스(녹음, 카메라, 스토리지)를 함께 사용하는 하이브리드 앱을 구현해본다

04

OpenAI API를 이용하여 AI 기능(Speech To Text, 내용 요약)을 구현한다

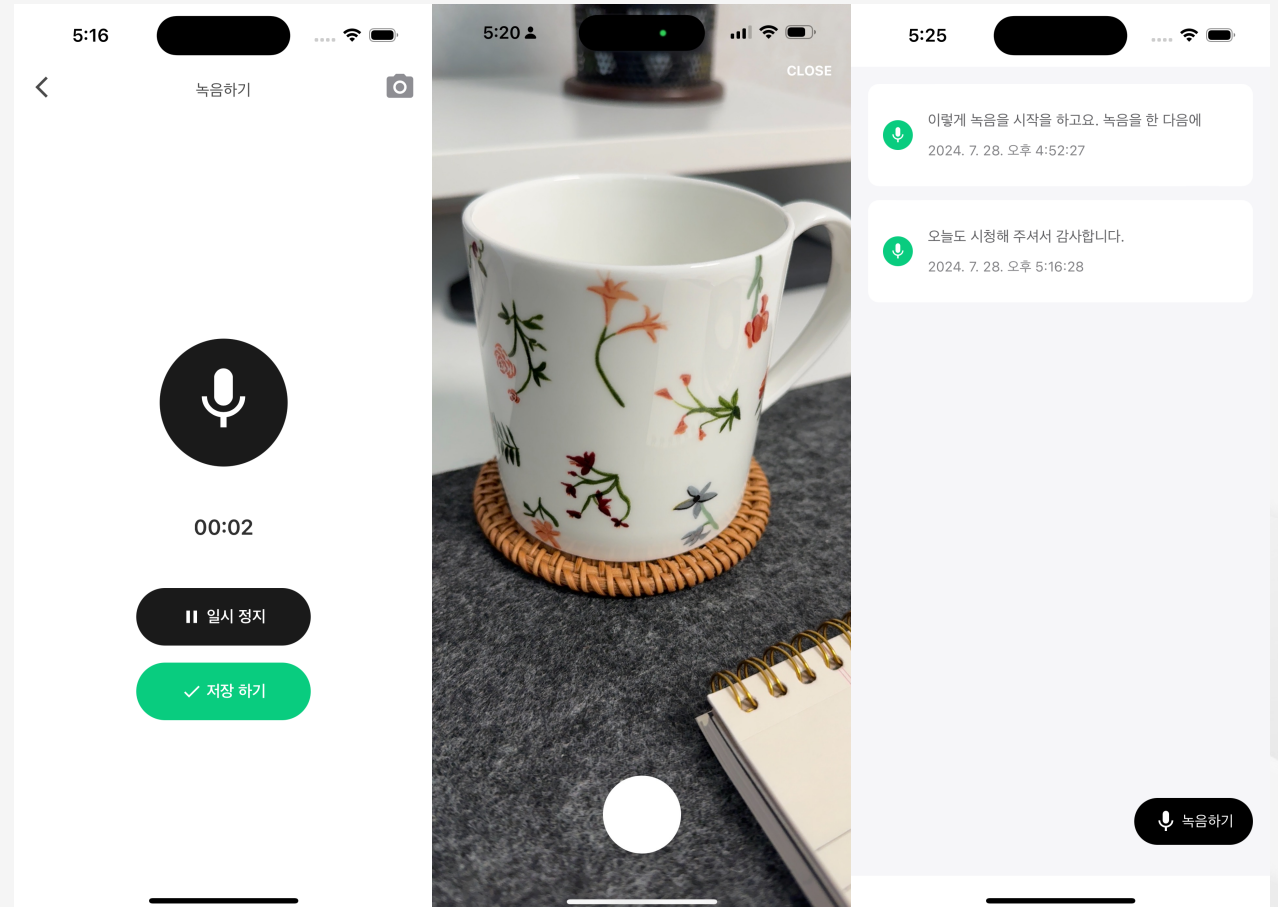
# 내용 요약해주는 AI 녹음기 (웹)

- 오디오 녹음 기능
- 스크립트 추출
- 스크립트 요약



# 내용 요약해주는 AI 녹음기 (앱)

- 오디오 녹음 기능
- 사진 첨부 기능
- 데이터 저장 기능



# Overview

## **Web**

강의 클립 1-7

## **App(RN CLI) + Web**

강의 클립 8-12

## **App (Expo)**

강의 클립 13-17

**프로젝트3 - 내용 요약해주는 AI 녹음기 (Web)**

# **React 웹 프로젝트 셋업**

# 프로젝트 초기화

- 웹 프레임워크: Next.js

- React의 확장판: Next.js는 React를 더 쉽게 사용하고 배포할 수 있게함
- 자동 라우팅: pages 폴더에 파일을 만들면 자동으로 라우팅이 설정.
- 서버 사이드 렌더링 (SSR): 서버에서 페이지를 렌더링해 초기 로딩 속도를 빠르게 하고 SEO를 개선
- 정적 사이트 생성 (SSG): 빌드 시점에 미리 HTML 파일을 생성해 빠른 페이지 로딩을 제공
- API Routes: 백엔드 로직을 간단히 구현할 수 있는 API 엔드포인트를 만들 수 있음
- <https://nextjs.org/>

- CSS 프레임워크: Tailwind

- 유틸리티 클래스 기반: 미리 세팅된 클래스를 이용해서 HTML 요소에 직접 적용할 수 있게 도와줌
- 생산성 향상: 필요한 스타일을 직접 HTML에 작성하므로, 별도의 CSS 파일을 관리할 필요가 없음
- 재사용성: 유틸리티 클래스를 사용해 여러 요소에 동일한 스타일을 적용할 수 있어, 코드 중복을 줄일 수 있음
- <https://tailwindcss.com/>

- 설치 명령어

- `npx create-next-app@14.2.4`



# Tailwind 적용 예시

```
import React from 'react';

const styles = {
  container: {
    display: 'flex',
    justifyContent: 'center',
    alignItems: 'center',
    minHeight: '100vh',
    backgroundColor: '#f8f9fa',
  },
  card: {
    backgroundColor: 'fff',
    padding: '20px',
    borderRadius: '8px',
    boxShadow: '0 4px 6px rgba(0, 0, 0, 0.1)',
    textAlign: 'center',
  },
  cardTitle: {
    fontSize: '1.5rem',
    fontWeight: 'bold',
    color: '#333',
  },
  cardContent: {
    fontSize: '1rem',
    color: '#666',
  },
};

function App() {
  return (
    <div style={styles.container}>
      <div style={styles.card}>
        <h1 style={styles.cardTitle}>Hello, World!</h1>
        <p style={styles.cardContent}>This is a simple card component.</p>
      </div>
    </div>
  );
}

export default App;
```

Tailwind 적용 전

```
import React from 'react';
import 'tailwindcss/tailwind.css';

function App() {
  return (
    <div className="flex items-center justify-center min-h-screen bg-gray-100">
      <div className="bg-white p-6 rounded-lg shadow-md text-center">
        <h1 className="text-2xl font-bold text-gray-800">Hello, World!</h1>
        <p className="text-gray-600">This is a simple card component.</p>
      </div>
    </div>
  );
}

export default App;
```

Tailwind 적용 후

**프로젝트3 - 내용 요약해주는 AI 녹음기 (Web)**

# **녹음 화면 구현**

## 학습 목표

01

React를 이용해서 녹음 기능을 구현할 수 있다

02

Next.js를 이용하여 페이지 라우팅을 구현할 수 있다

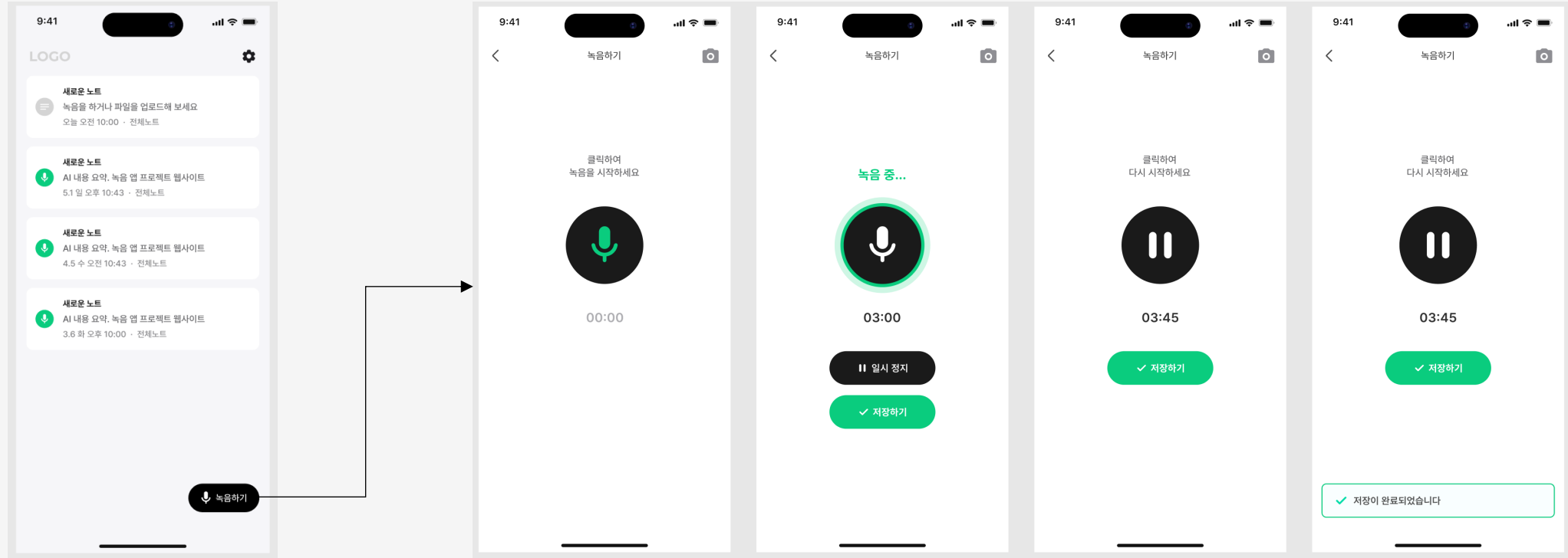
03

기본적인 Tailwind 사용법을 익힌다

04

웹 환경에서 Material Icons 사용할 수 있다

## 구현 할 스크린 소개



/

/recorder

# Blob

- 브라우저와 서버 간에 파일을 주고받기 위해 사용되는 데이터 타입
- Blob의 주요 특징
  - 바이너리 데이터 저장: 텍스트, 이미지, 비디오 등 다양한 바이너리 데이터를 저장
  - 크기 및 형식 지정: 파일의 크기와 MIME 타입(예: image/jpeg, text/plain)을 지정
- Blob의 사용 사례
  - 파일 업로드 및 다운로드: 웹 애플리케이션에서 사용자가 파일을 업로드하거나 다운로드할 때 Blob을 사용
  - 이미지 및 비디오 처리: 이미지를 캡처하고 이를 Blob으로 변환하여 서버로 전송하거나, 브라우저에서 표시 가능

**프로젝트3 - 내용 요약해주는 AI 녹음기 (Web)**

# **녹음 스크립트 추출하기**

## 학습 목표

**01** Speech To Text API를 이용해서 음성을 텍스트로 변환할 수 있다

**02** Next.js를 이용하여 API를 구현할 수 있다

# Speech To Text

- **STT**: 음성을 텍스트로 변환하는 기술
- OpenAI Whisper
  - 유료: \$0.006 / minute
- OpenAI에 가입하고, 크레딧 결제 필요
  - <https://platform.openai.com/>
  - Settings -> Billing -> Add to credit balance




P

Personal / Default project

Dashboard


Docs


API reference



D

SETTINGS

 Your profile

 Organization


General

Members

2

Billing

Limits

 Project

General

Members

Limits

+ Create project

Billing

Overview

Payment methods

Billing history

Preferences

Pay as you go

Credit balance ⓘ

\$9.95

☐ Auto recharge is off

When your credit balance reaches \$0, your API requests will stop working. Enable automatic recharge to automatically keep your credit balance topped up.

Enable auto recharge

3

Add to credit balance

Cancel plan

**프로젝트3 - 내용 요약해주는 AI 녹음기 (Web)**

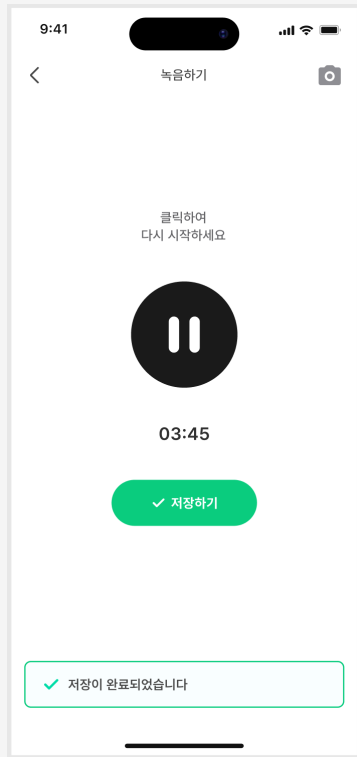
# **스크립트 컴포넌트 구현**

# 학습 목표

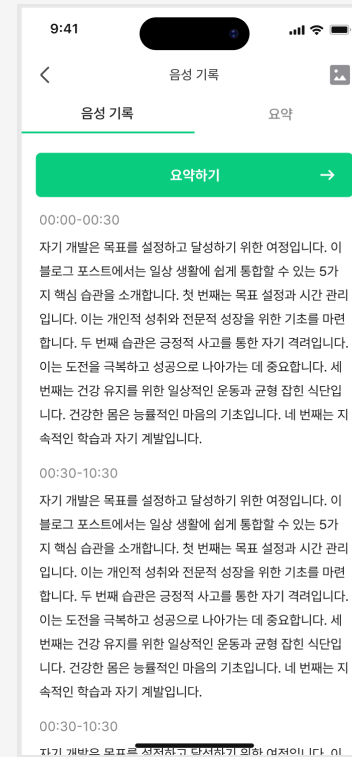
01

다른 페이지 간에 데이터를 공유할 수 있다

## 구현 할 스크린 소개



/recorder



/recording/[id]

# 페이지 간 데이터 공유

- URL 파라미터 (동적 경로)

- URL 경로에 데이터를 포함시켜 다른 페이지로 전달하는 방법
- 간단하고 직관적이지만, 여러 데이터 전달이 어려움
- 예시) /recording/id

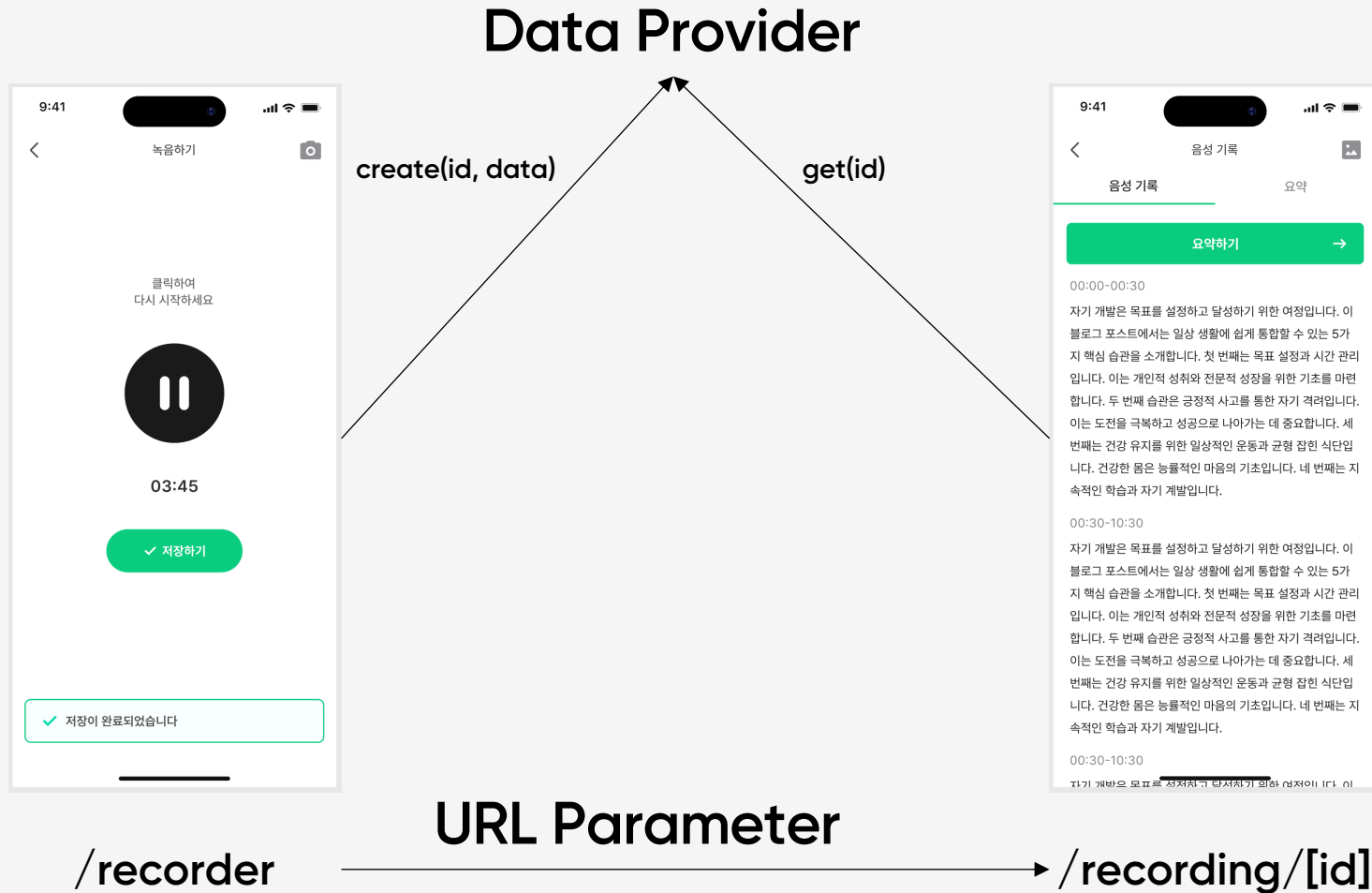
- 쿼리 파라미터

- URL에 데이터를 포함시켜 다른 페이지로 전달하는 방법
- 간단하고 직관적이지만, URL 길이에 제한이 있어 큰 데이터 전달이 어려움
- 예시) /recording/id?text=안녕하세요&startTime=0&endTime=0

- React Context API

- 전역적으로 데이터를 관리하고 공유
- **Provider**: 데이터를 제공하는 컴포넌트로, 자식 컴포넌트들에게 데이터를 전달합니다.
- **Consumer**: 데이터를 소비하는 컴포넌트로, Provider에서 제공하는 데이터를 사용합니다.
- 계층 구조가 깊어도 데이터 직접 전달할 필요 없이, Context를 통해 손쉽게 데이터 공유 가능

# 페이지 간 데이터 공유



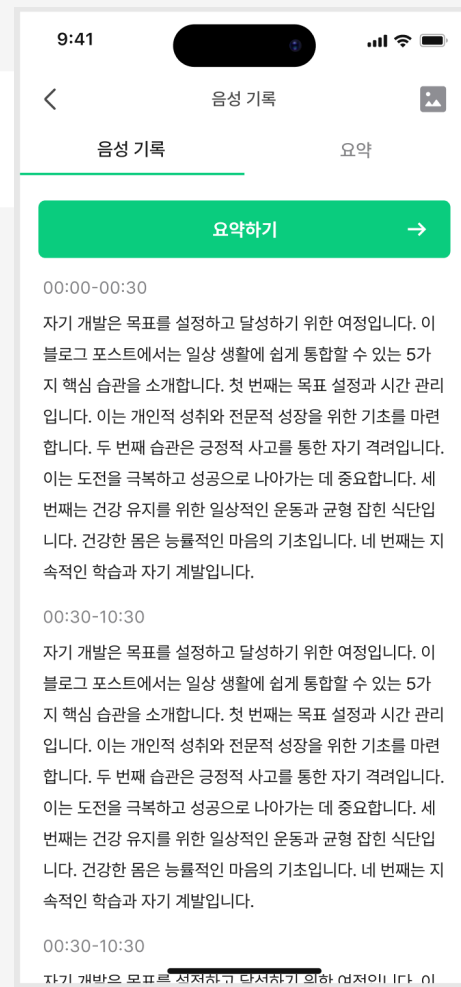
**프로젝트3 - 내용 요약해주는 AI 녹음기 (Web)**

# **스크립트 요약하기**

# 학습 목표

## 01

ChatGPT API를 이용해서 텍스트를 요약하는 API를 만들 수 있다





**프로젝트3 - 내용 요약해주는 AI 녹음기 (Web)**

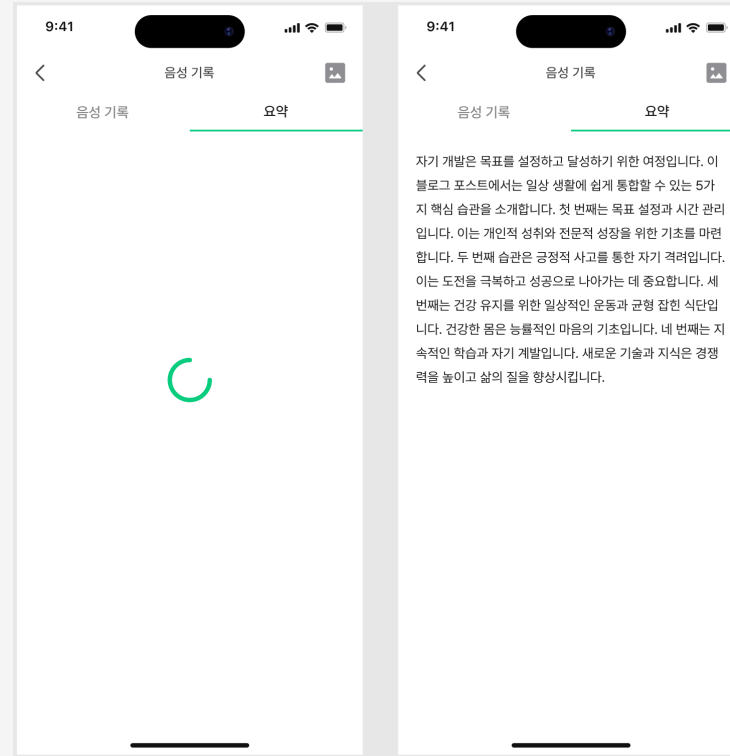
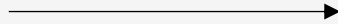
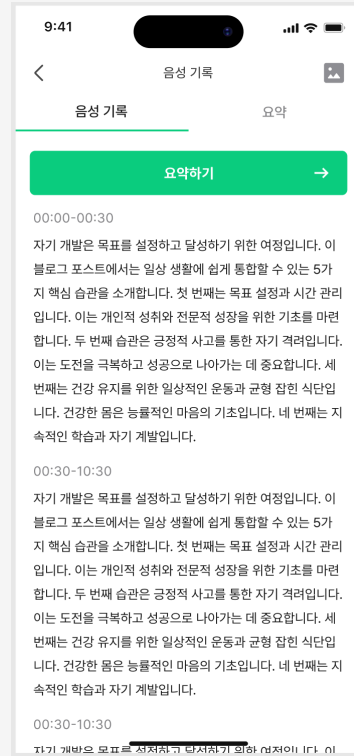
**스크립트 요약 컴포넌트 구현**

# 학습 목표

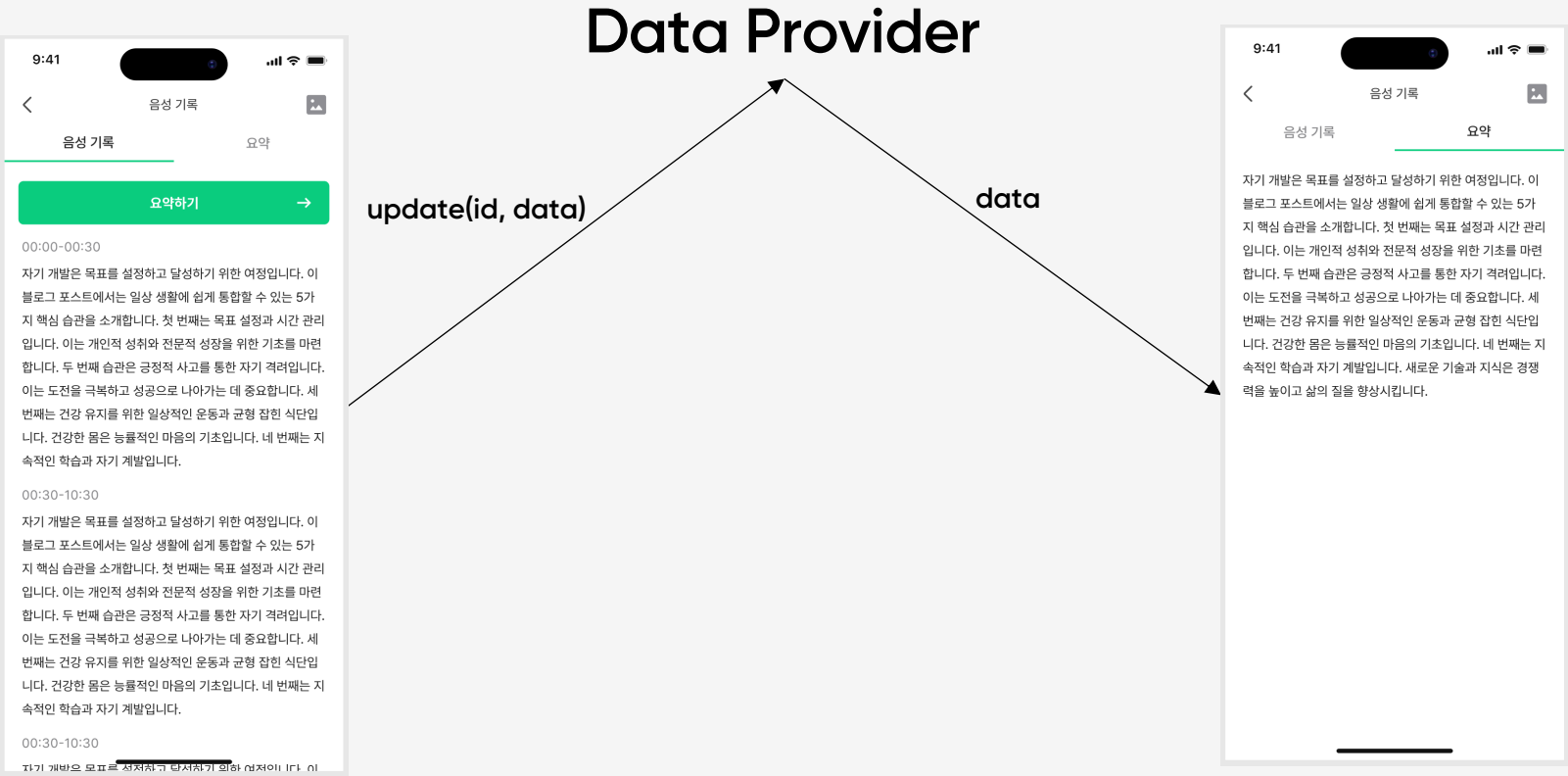
01

Context API를 이용하여 데이터를 업데이트 한다

# 구현 할 스크린 소개



# 데이터 업데이트



**프로젝트3 - 내용 요약해주는 AI 녹음기 (React Native CLI)**

# **앱 프로젝트 셋업**

# Overview

- AI 녹음기를 웹 서비스를 앱으로 제공하는 프로젝트
  - 기존 서비스의 기능인 녹음 및 스크립트 추출 기능 지원
  - 새로운 기능: 녹음 중 카메라 촬영 기능 지원
  - 새로운 기능: 녹음 스크립트 히스토리 저장

## 1. 앱에서 웹 로드

웹뷰를 이용해서 웹 로드

## 2. 녹음 기능

디바이스 관계없이 지원하기 위해  
웹이 아닌 앱의 녹음 API 사용

## 3. 부가 기능: 카메라

앱의 카메라 API를 이용하여  
사진 촬영 기능 구현

## 4. 부가 기능: 히스토리 저장

데이터를 디바이스에 저장하여  
녹음 리스트 제공

## 프로젝트 초기화

- `npx react-native@0.73.6 init AiRecorderApp --version 0.73.6`
- 패키지 설치: `react-native-webview`

**프로젝트3 - 내용 요약해주는 AI 녹음기 (React Native CLI)**

# **웹 프로젝트 웹앱으로 로드하기**



# 학습 목표

01

개발 중인 웹 사이트를 웹뷰로 로드 할 수 있다

## 개발 중인 웹 사이트 로드

- 개발 중인 웹사이트 URL
  - `http://localhost:3000`
- 에뮬레이터에서 개발
  - iOS: `http://localhost:3000`
  - Android: `http://10.0.2.2:3000`
- 디바이스에서 개발
  - localhost 접근 불가능
  - 실제 서버에 호스팅: 테스트 할 때 마다 배포를 해야 하는 불편함
  - 내부 IP 찾기: 예시) `http://192.168.0.102:3000`
  - 터널링 이용: 로컬 서버에 공용 URL을 할당하여 외부에서 접근가능하게 해줌 (ngrok)

# Ngrok

- 로컬에서 실행 중인 웹 서버를 안전하게 공개 인터넷에 노출시켜주는 도구
- 주요 기능
  - 터널링: ngrok은 터널링을 통해 로컬 서버에 공용 URL을 할당
  - 보안: ngrok은 HTTPS를 지원하여 데이터 전송 시 보안을 유지
  - 접속 로그: ngrok은 모든 트래픽을 기록하고, 개발자가 이를 실시간으로 모니터링
  - 검사 및 디버깅: ngrok의 웹 인터페이스를 통해 HTTP 요청 및 응답을 검사하고 디버깅
  - 사용 용이성: ngrok은 간단한 설치와 명령어로 쉽게 사용
- <https://ngrok.com/>

```
ngrok
Help shape K8s Bindings https://ngrok.com/new-features-update?ref=k8s

Session Status      online
Account             daehoon.dev@gmail.com (Plan: Free)
Version             3.13.0
Region              Japan (jp)
Web Interface        http://127.0.0.1:4040
Forwarding           https://0fb7-61-254-209-36.ngrok-free.app -> http://localhost:3000

Connections          ttl      opn      rt1      rt5      p50      p90
0                   0        0.00     0.00     0.00     0.00
```

**프로젝트3 - 내용 요약해주는 AI 녹음기 (React Native CLI)**

# **웹앱에서 녹음 기능 구현**

## 학습 목표

- 01** 웹과 앱에서 동작이 다르게 웹 사이트를 구현할 수 있다
- 02** 앱에서 녹음 기능을 구현할 수 있다
- 03** 웹과 앱이 통신하여 앱에서 녹음한 오디오를 웹으로 전송할 수 있다

## 왜 녹음 기능을 따로 구현해야 할까?

- 앱에서 마이크 권한을 받으면 웹 녹음 API를 바로 사용 가능
- 하지만, 특정 OS 버전 및 플랫폼에 따라서 지원되지 않을 수 있음
  - 웹에서 webm으로 녹음을 하면 사파리에서 지원이 안됨 -> iOS 사용 불가능
- 따라서, 앱 자체의 녹음 API를 이용하는 것이 호환성 측면에서 유리

## 앱 <-> 웹 통신

코드를 실행 할 때!



## 앱 <-> 웹 통신

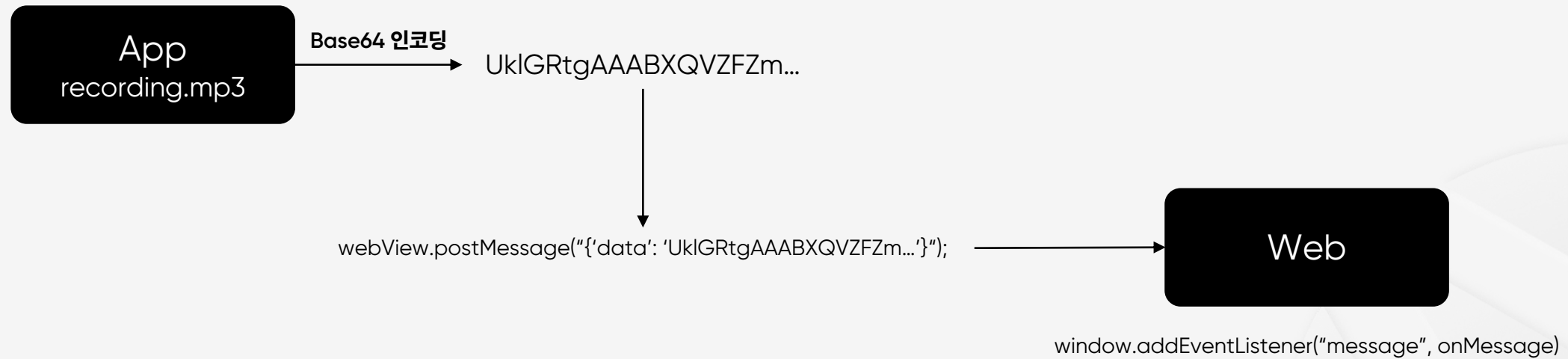




# 앱에서 녹음 기능 구현하기

- 오디오 녹음 및 재생 라이브러리
  - react-native-audio-recorder-player
  - 오디오 녹음 및 재생 지원
  - <https://github.com/hyochan/react-native-audio-recorder-player>
- 퍼미션 요청 라이브러리
  - react-native-permissions
  - 특정 기능에 대한 권한 요청
  - <https://github.com/zoontek/react-native-permissions>
- React Native 파일 시스템 접근 라이브러리
  - react-native-fs
  - 녹음된 오디오 파일을 읽을 수 있음
  - <https://github.com/itinance/react-native-fs>

## 앱에서 오디오를 웹으로 전달하기



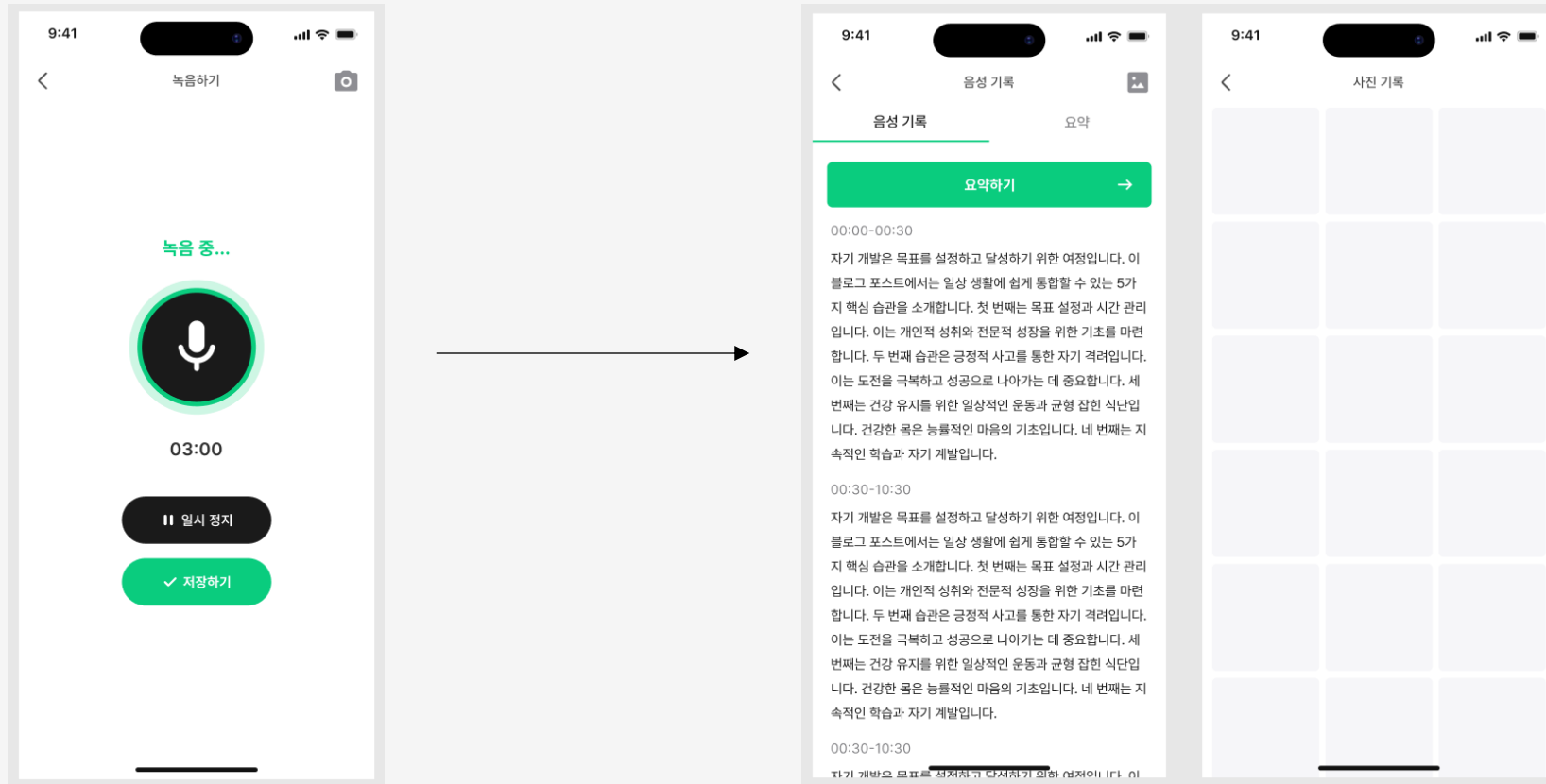
**프로젝트3 – 내용 요약해주는 AI 녹음기 (React Native CLI)**

# **웹앱에서 사진 첨부 기능 구현**

## 학습 목표

- 01** 웹과 앱에서 동작이 다르게 웹 사이트를 구현할 수 있다
- 02** 앱에서 카메라를 이용하여 사진을 촬영하는 기능을 구현할 수 있다
- 03** 웹과 앱이 통신하여 앱에서 촬영한 사진을 웹으로 전송할 수 있다

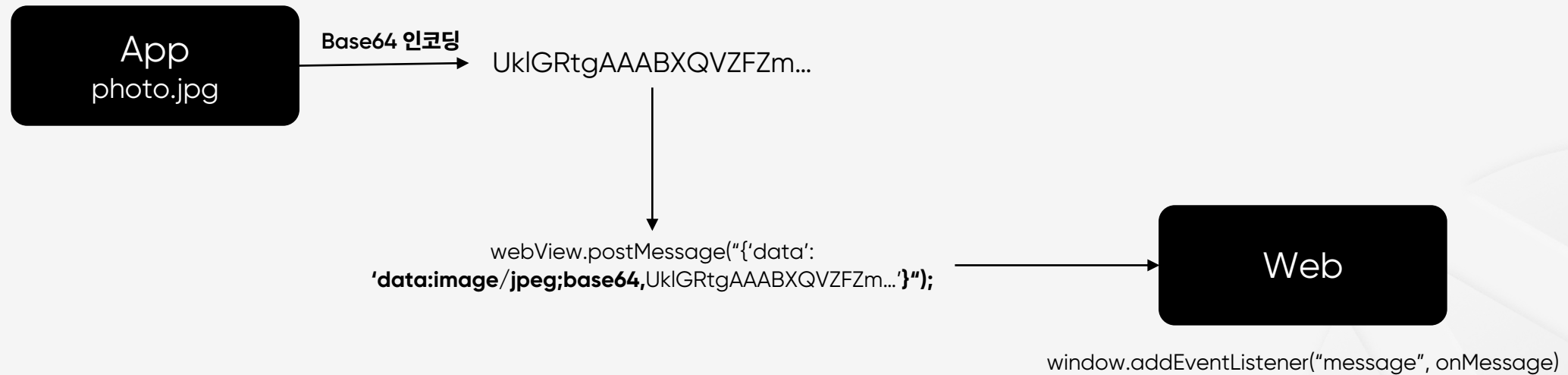
## 구현 할 스크린 소개



# 앱에서 사진 촬영 기능 구현하기

- 사진 촬영 라이브러리
  - react-native-vision-camera
  - 사진 및 비디오 촬영 지원
  - <https://github.com/mrousavy/react-native-vision-camera>

## 앱에서 이미지를 웹으로 전달하기



**프로젝트3 - 내용 요약해주는 AI 녹음기 (React Native CLI)**

# **웹앱에 데이터 저장 기능 구현**



## 학습 목표

01

앱 스토리지에 데이터를 저장할 수 있다

02

웹과 앱이 통신하여 앱에 데이터를 저장하고 이를 가져올 수 있다.

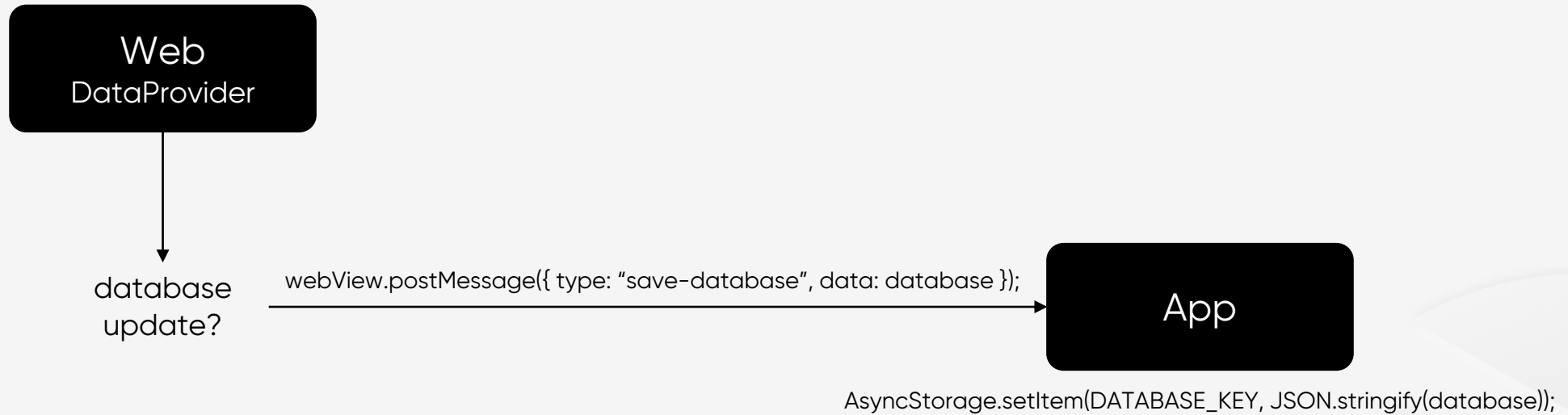
## 구현 할 스크린 소개



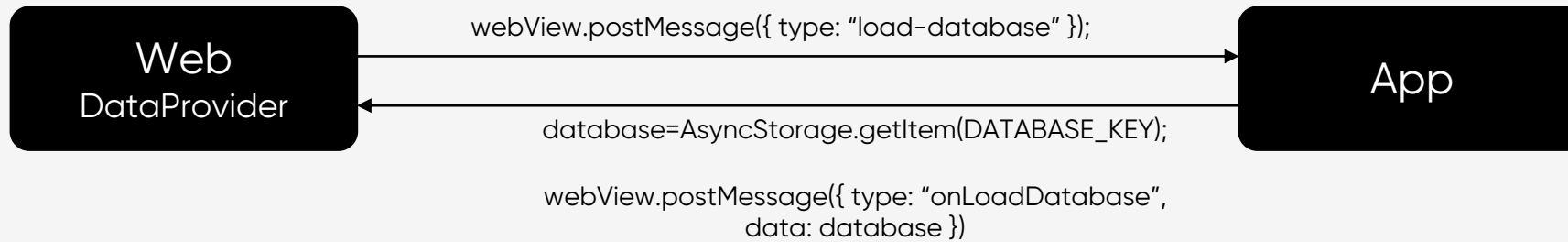
# 앱 스토리지에 데이터 저장

- 앱 스토리지 라이브러리
  - 암호화되지 않은 데이터를 Key-Value 형태로 비동기로 저장
  - 앱이 종료되어도 데이터를 저장하고 싶은 경우 사용
  - <https://github.com/react-native-async-storage/async-storage>

## 앱-웹 데이터 저장 개요



## 앱-웹 데이터 로드 개요



**프로젝트3 - 내용 요약해주는 AI 녹음기 (Expo)**

# **앱 프로젝트 셋업**

# Overview

- AI 녹음기를 웹 서비스를 앱으로 제공하는 프로젝트
  - 기존 서비스의 기능인 녹음 및 스크립트 추출 기능 지원
  - 새로운 기능: 녹음 중 카메라 촬영 기능 지원
  - 새로운 기능: 녹음 스크립트 히스토리 저장
- 웹 파트 구현을 위해 이전 클립 시청 필수

## 1. 앱에서 웹 로드

웹뷰를 이용해서 웹 로드

## 2. 녹음 기능

디바이스 관계없이 지원하기 위해  
웹이 아닌 앱의 녹음 API 사용

## 3. 부가 기능: 카메라

앱의 카메라 API를 이용하여  
사진 촬영 기능 구현

## 4. 부가 기능: 히스토리 저장

데이터를 디바이스에 저장하여  
녹음 리스트 제공

### Web

강의 클립 1-7

### App(RN CLI) + Web

강의 클립 8-12

### App (Expo)

강의 클립 13-17

## 프로젝트 초기화

- `yarn create expo --template expo-template-blank-typescript@51`
- Eslint
  - <https://docs.expo.dev/guides/using-eslint/#setup>
- Prettier
  - <https://docs.expo.dev/guides/using-eslint/#installation>



**프로젝트3 - 내용 요약해주는 AI 녹음기 (Expo)**

# **웹 프로젝트 웹앱으로 로드하기**

# 학습 목표

01

개발 중인 웹 사이트를 웹뷰로 로드 할 수 있다

## 개발 중인 웹 사이트 로드

- 개발 중인 웹사이트 URL
  - `http://localhost:3000`
- 에뮬레이터에서 개발
  - iOS: `http://localhost:3000`
  - Android: `http://10.0.2.2:3000`
- 디바이스에서 개발
  - localhost 접근 불가능
  - 실제 서버에 호스팅: 테스트 할 때 마다 배포를 해야 하는 불편함
  - 내부 IP 찾기: 예시) `http://192.168.0.102:3000`
  - 터널링 이용: 로컬 서버에 공용 URL을 할당하여 외부에서 접근가능하게 해줌 (ngrok)

# Ngrok

- 로컬에서 실행 중인 웹 서버를 안전하게 공개 인터넷에 노출시켜주는 도구
- 주요 기능
  - 터널링: ngrok은 터널링을 통해 로컬 서버에 공용 URL을 할당
  - 보안: ngrok은 HTTPS를 지원하여 데이터 전송 시 보안을 유지
  - 접속 로그: ngrok은 모든 트래픽을 기록하고, 개발자가 이를 실시간으로 모니터링
  - 검사 및 디버깅: ngrok의 웹 인터페이스를 통해 HTTP 요청 및 응답을 검사하고 디버깅
  - 사용 용이성: ngrok은 간단한 설치와 명령어로 쉽게 사용
- <https://ngrok.com/>

```
ngrok
Help shape K8s Bindings https://ngrok.com/new-features-update?ref=k8s

Session Status      online
Account             daehoon.dev@gmail.com (Plan: Free)
Version             3.13.0
Region              Japan (jp)
Web Interface        http://127.0.0.1:4040
Forwarding           https://0fb7-61-254-209-36.ngrok-free.app -> http://localhost:3000

Connections          ttl      opn      rt1      rt5      p50      p90
0                   0        0.00     0.00     0.00     0.00
```

**프로젝트3 - 내용 요약해주는 AI 녹음기 (Expo)**

# **웹앱에서 녹음 기능 구현**

## 학습 목표

01

앱에서 녹음 기능을 구현할 수 있다

02

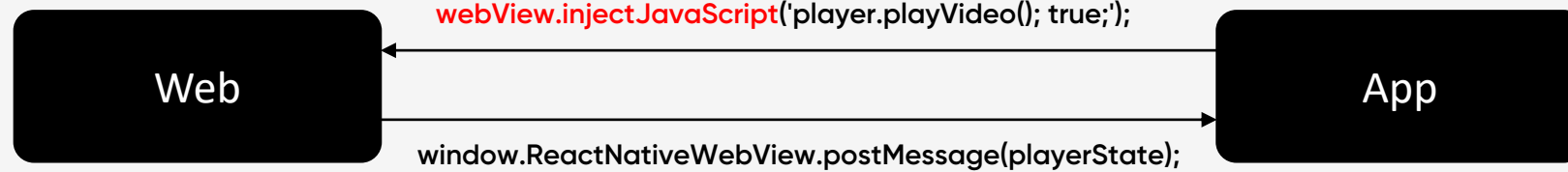
웹과 앱이 통신하여 앱에서 녹음한 오디오를 웹으로 전송할 수 있다

## 왜 녹음 기능을 따로 구현해야 할까?

- 앱에서 마이크 권한을 받으면 웹 녹음 API를 바로 사용 가능
- 하지만, 특정 OS 버전 및 플랫폼에 따라서 지원되지 않을 수 있음
  - 웹에서 webm으로 녹음을 하면 사파리에서 지원이 안됨 -> iOS 사용 불가능
- 따라서, 앱 자체의 녹음 API를 이용하는 것이 호환성 측면에서 유리

## 앱 <-> 웹 통신

코드를 실행 할 때!





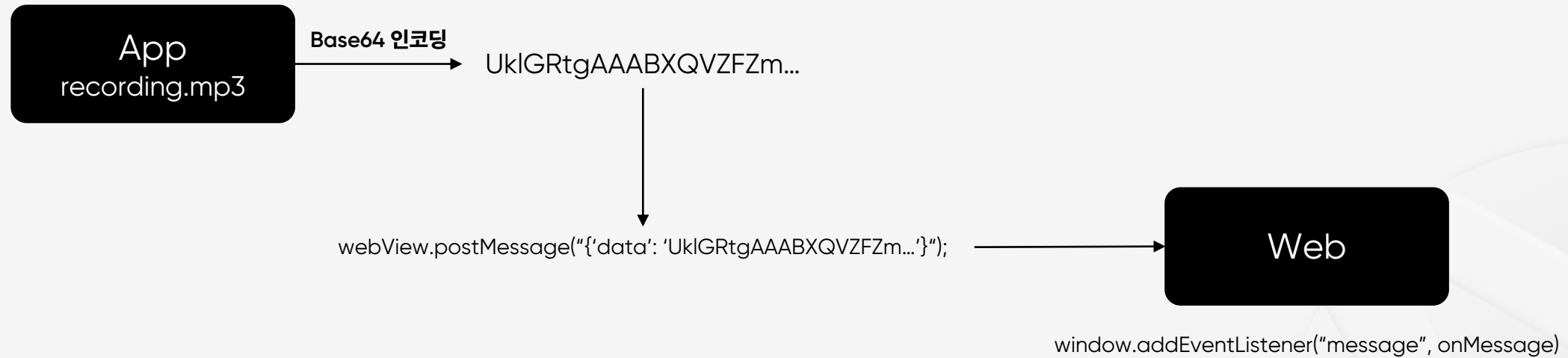
## 앱 <-> 웹 통신



# 앱에서 녹음 기능 구현하기

- 오디오 녹음 및 재생 라이브러리
  - expo-av
  - 오디오 녹음 및 재생 지원
  - 퍼미션 요청 기능 내장
  - <https://docs.expo.dev/versions/latest/sdk/audio/>
- 앱 파일 시스템 접근 라이브러리
  - expo-file-system
  - 녹음된 오디오 파일을 읽을 수 있음
  - <https://docs.expo.dev/versions/latest/sdk/filesystem/>

# 앱에서 오디오를 웹으로 전달하기



**프로젝트3 - 내용 요약해주는 AI 녹음기 (Expo)**

# **웹앱에서 사진 첨부 기능 구현**

## 학습 목표

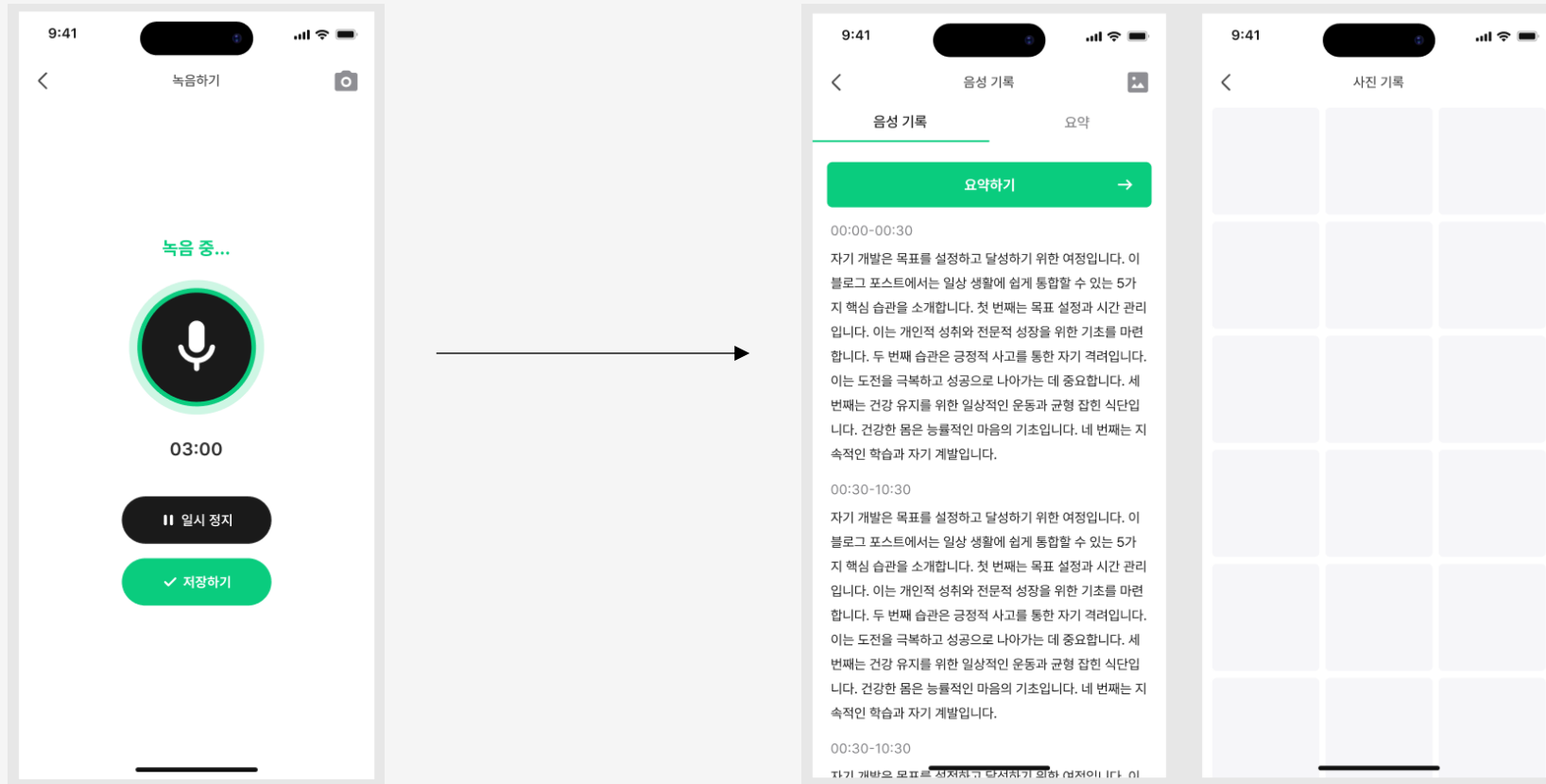
01

앱에서 카메라를 이용하여 사진을 촬영하는 기능을 구현할 수 있다

02

웹과 앱이 통신하여 앱에서 촬영한 사진을 웹으로 전송할 수 있다

## 구현 스크린 소개

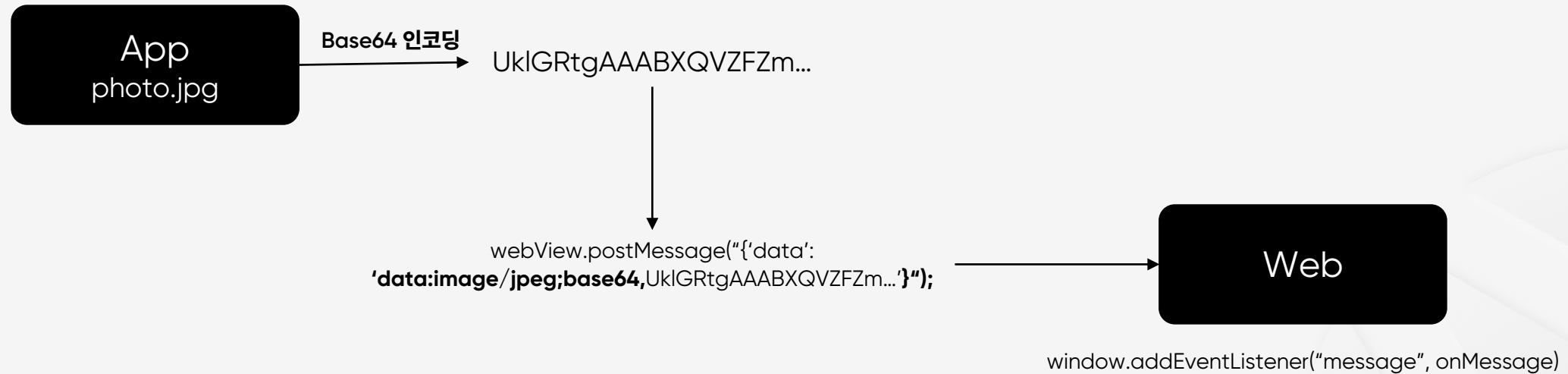


11번 웹앱에서 사진 첨부 기능 구현 (React Native CLI) 클립에서 진행

# 앱에서 사진 촬영 기능 구현하기

- 사진 촬영 라이브러리
  - expo-camera
  - 사진 및 비디오 촬영 지원
  - 퍼미션 요청 기능 내장
  - <https://docs.expo.dev/versions/latest/sdk/camera/>

## 앱에서 이미지를 웹으로 전달하기





**프로젝트3 - 내용 요약해주는 AI 녹음기 (Expo)**

# **웹앱에 데이터 저장 기능 구현**

## 학습 목표

01

앱 스토리지에 데이터를 저장할 수 있다

02

웹과 앱이 통신하여 앱에 데이터를 저장하고 이를 가져올 수 있다.

## 구현 스크린 소개

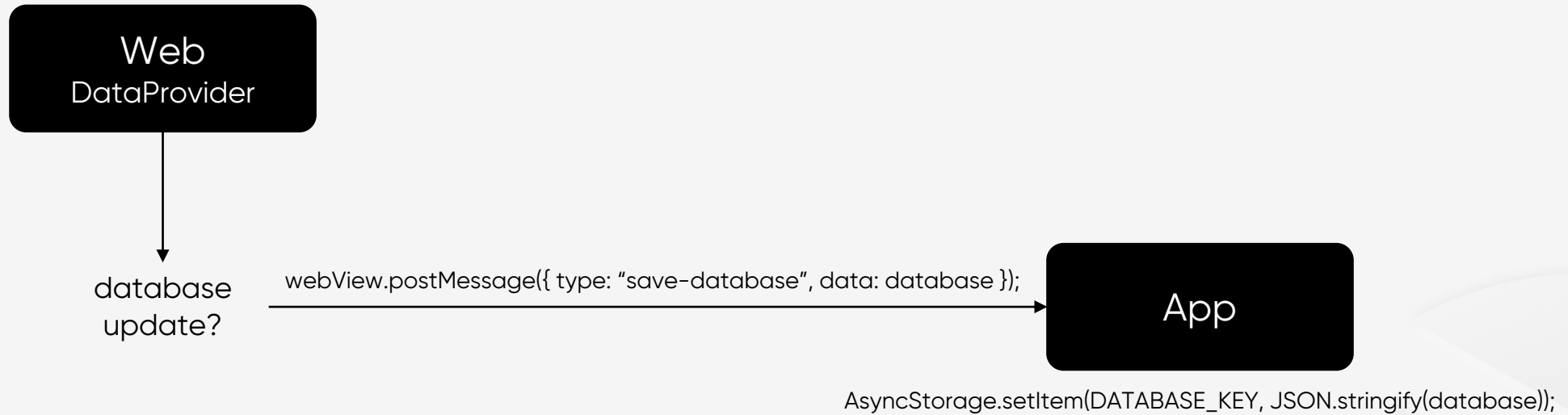


12번 웹앱에 데이터 저장 기능 구현 (React Native CLI) 클립에서 진행

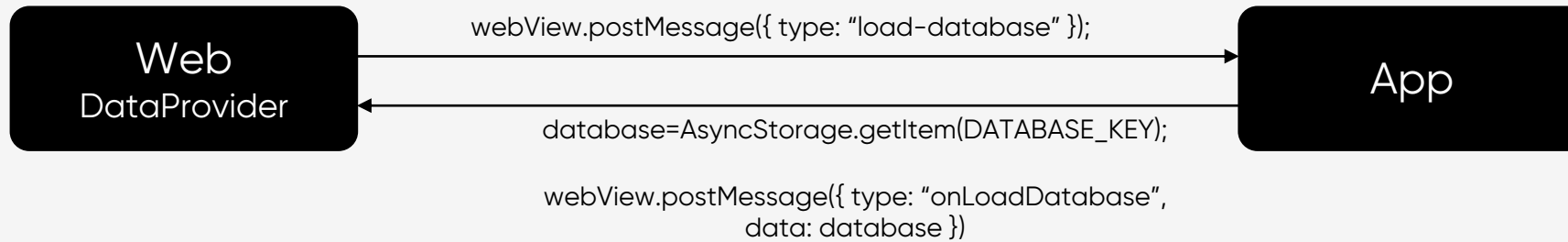
# 앱 스토리지에 데이터 저장

- 앱 스토리지 라이브러리
  - 암호화되지 않은 데이터를 Key-Value 형태로 비동기로 저장
  - 앱이 종료되어도 데이터를 저장하고 싶은 경우 사용
  - <https://docs.expo.dev/versions/latest/sdk/async-storage/>

## 앱-웹 데이터 저장 개요



## 앱-웹 데이터 로드 개요

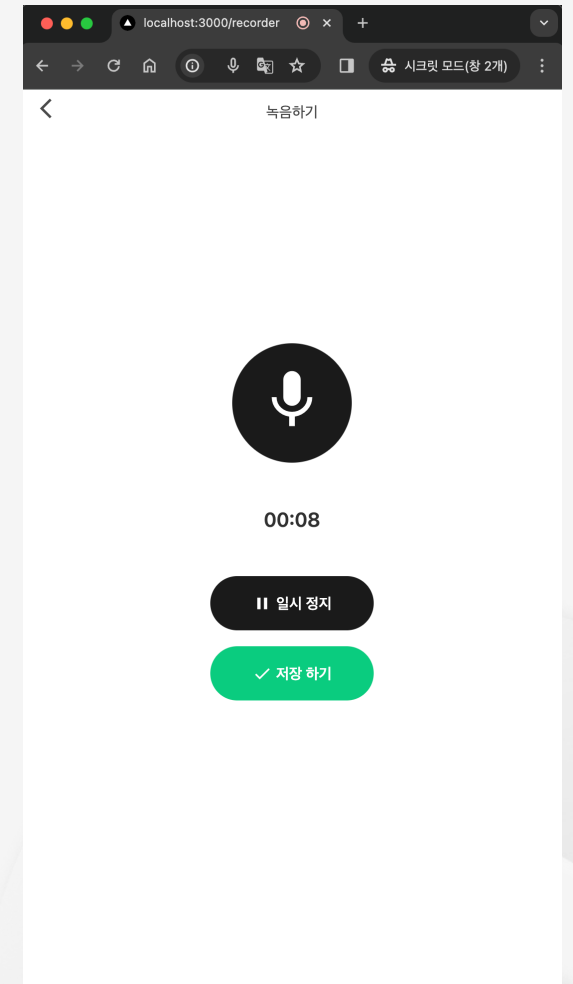


**프로젝트3 - 내용 요약해주는 AI 녹음기**

**학습 리뷰 및 예상 면접 질문**

## 웹 - 녹음 기능 구현

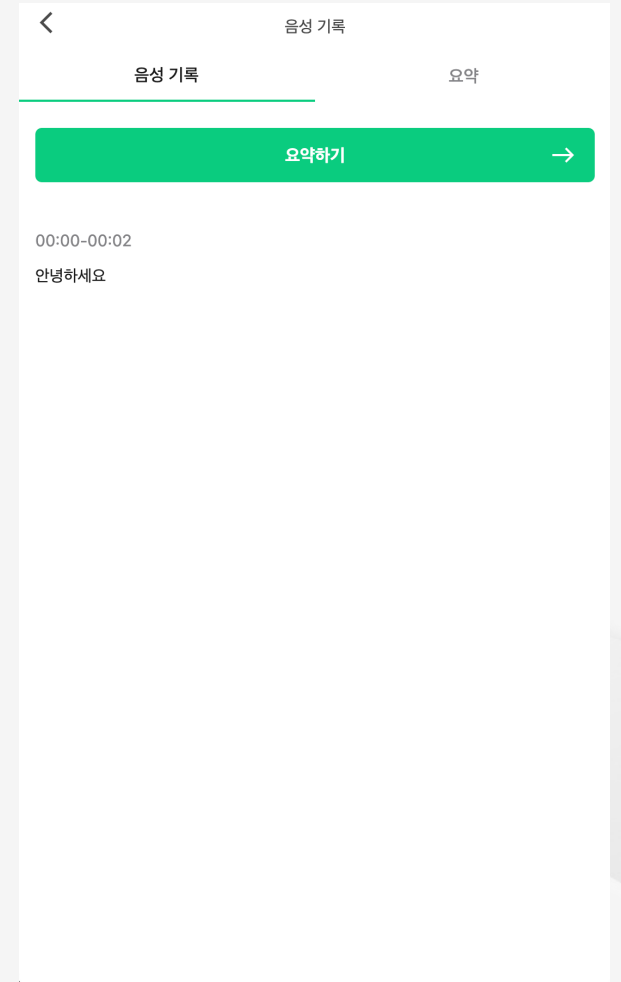
- Nextjs를 이용한 웹 프로젝트 초기화
- Tailwind를 이용한 CSS 작업
- MediaRecorder API를 이용한 웹 녹음 기능 구현





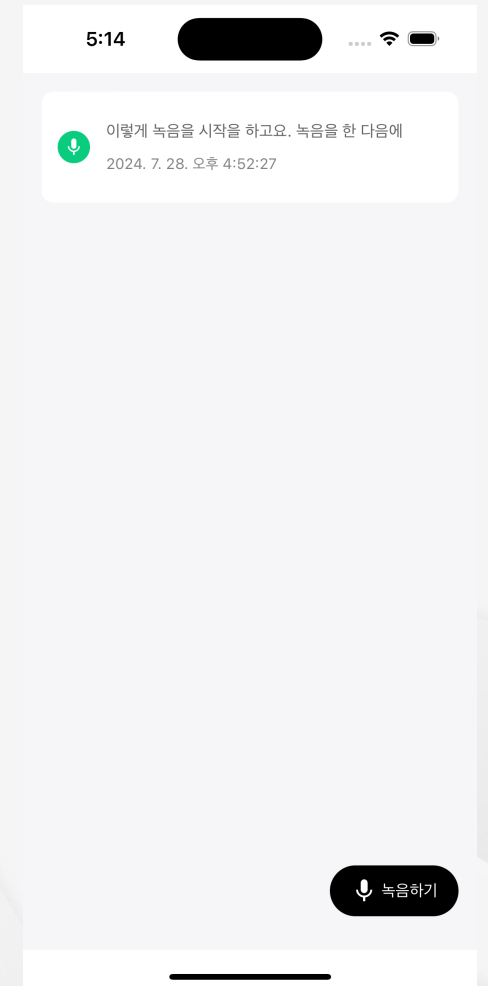
## 웹 - 스크립트 추출 및 요약

- OpenAI의 Whisper API를 이용하여 Speech To Text 구현
- OpenAI의 ChatGPT API를 이용하여 텍스트 요약 기능 구현



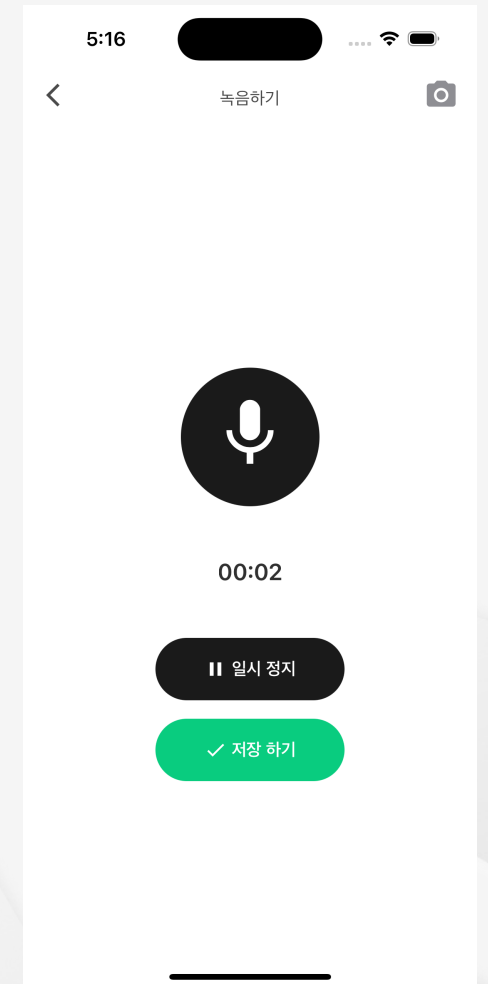
## 앱 - 웹 사이트 로드하기

- 웹뷰를 이용하여 웹 사이트 로드
- 디바이스에서 테스트 하기 위해서 터널링 서비스 이용 (ngrok)



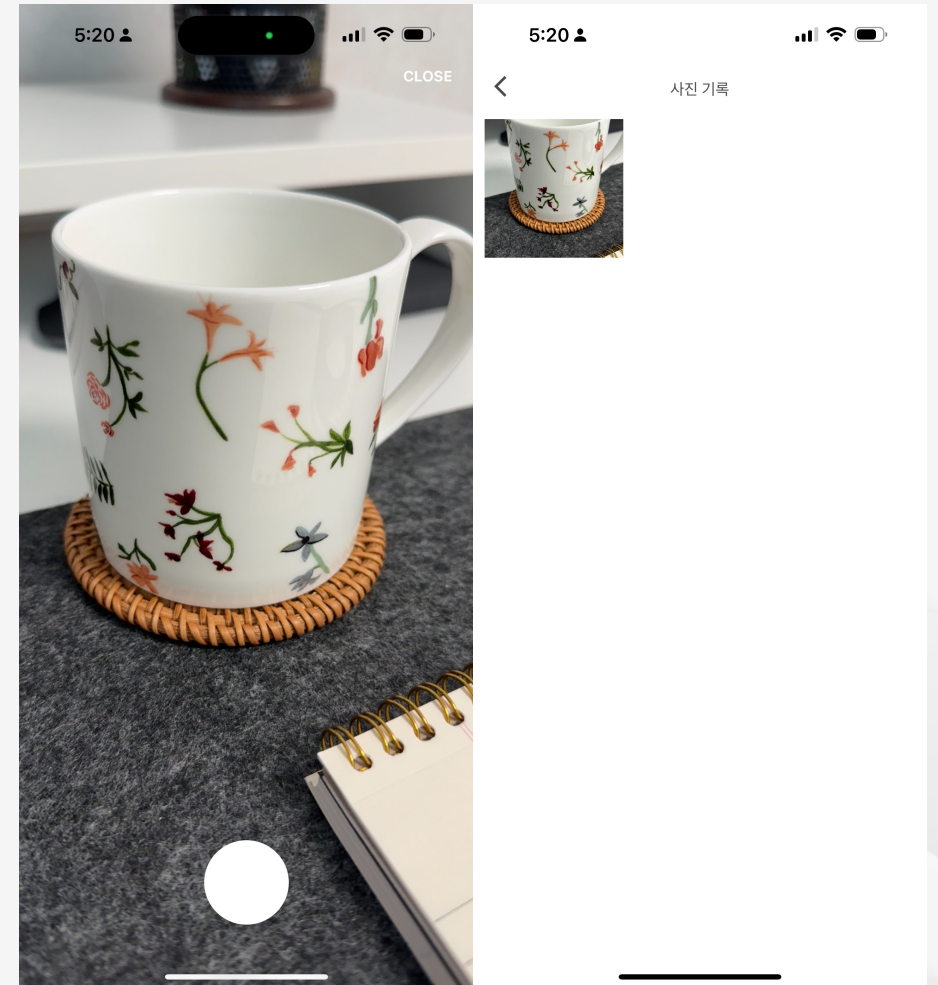
## 앱 - 녹음 기능 구현

- 웹과 앱 동작 분기하여 웹 프로젝트 구현
  - `window.ReactNativeWebView` 존재 여부 확인
- 웹에서 앱으로 녹음 요청
- 앱의 녹음 API를 이용하여 오디오 녹음
  - `react-native-audio-recorder-player` / `expo-av`
  - `react-native-permissions`
  - `react-native-fs` / `expo-file-system`
- 앱의 녹음 파일을 웹으로 전달
  - Base64 인코딩
  - `webview.postMessage`



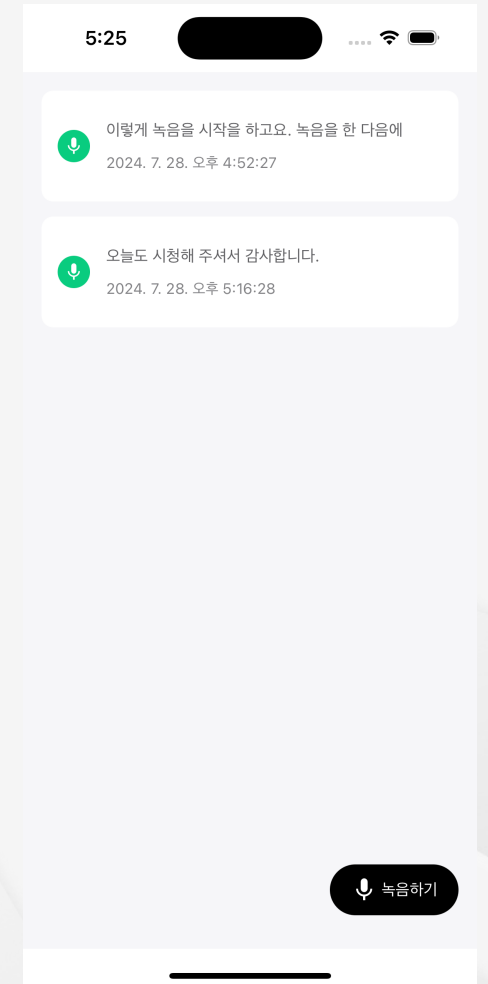
## 앱 - 사진 첨부 기능 구현

- 웹에서 앱으로 사진 촬영 요청
- 앱의 카메라 API를 이용하여 사진 촬영
  - react-native-vision-camera / expo-camera
  - react-native-fs / expo-file-system
- 앱의 사진 파일을 웹으로 전달
  - Base64 인코딩
  - `webView.postMessage`



## 앱 - 데이터 저장 기능 구현

- 웹의 데이터를 앱 스토리지에 영구 저장
  - react-native-async-storage



## 질문1. 개발중인 웹사이트를 이용하여 웹앱을 만들 때 발생 할 수 있는 어려운 점과 해결책?

- 개발 중인 웹사이트는 보통 localhost로 접근합니다. 시뮬레이터에서 개발할 경우 localhost에 바로 접근할 수 있지만, **실제 디바이스에서 테스트할 때는 로컬호스트에 접근할 수 없습니다.** 이 문제를 해결하기 위해 여러 방법이 있습니다. 개발 웹사이트를 실제로 웹에 호스팅할 수 있지만, 이는 개발할 때마다 웹에 다시 배포해야 하므로 불편할 수 있습니다. 대신 ngrok 같은 **터널링 서비스를 사용하면** 로컬호스트에 외부에서 접근 가능한 URL을 제공해 개발 과정을 더 편리하게 할 수 있습니다.

## 질문2. 웹 서비스를 이용하여 앱 서비스를 개발 할 때 고려할 점?

- **웹에서 제공되는 서비스가 앱에서도 제대로 동작하는지를 고려해야 합니다.** 웹 기능이 앱에서 완벽하게 동작하지 않을 수 있습니다. 예를 들어, 녹음 기능이나 사진 촬영 기능은 플랫폼에 따라 웹 서비스의 API가 아닌 **앱 자체의 리소스를 이용해 구현해야 할 때가 있습니다.** 이 경우 **웹 서비스를 개발할 때 브라우저 접근과 웹앱 접근을 구분하여 서비스를 구현해야 합니다.** 보통 웹에서 앱으로 특정 동작을 요청하는 메시지를 보내고, 앱에서는 이를 처리한 후 결과를 웹으로 전송하는 방식으로 하이브리드 앱을 구현합니다.

## 질문3. 오디오나 사진과 같은 데이터를 앱과 웹이 공유해야 할 때 주의해야 할 점은 무엇인가요?

- 웹과 앱이 통신할 때 주로 문자열 형태로 데이터를 주고받기 때문에 **이진 바이너리 파일을 바로 전송할 수 없습니다**. 이를 해결하려면 바이너리 데이터를 텍스트로 변환해야 하는데, **base64 인코딩을 사용하여** 처리할 수 있습니다. base64로 변환된 데이터를 전송한 후, 이를 다시 blob과 같은 이진 바이너리 데이터로 변환하여 사용할 수 있습니다.