

Part2. 웹뷰 띄우기 학습

네이버 앱 클론 코딩 프로젝트

프로젝트1 - 네이버 앱 클론 코딩

학습 목표 및 구성

학습 목표

01

기본적인 ReactNative 사용법을 학습한다

02

ReactNative를 이용해서 스크린 네비게이션을 구현할 수 있다

03

웹뷰로 웹 사이트를 로드하고 컨트롤 할 수 있다 (새로고침, 웹뷰 네비게이션 등)

04

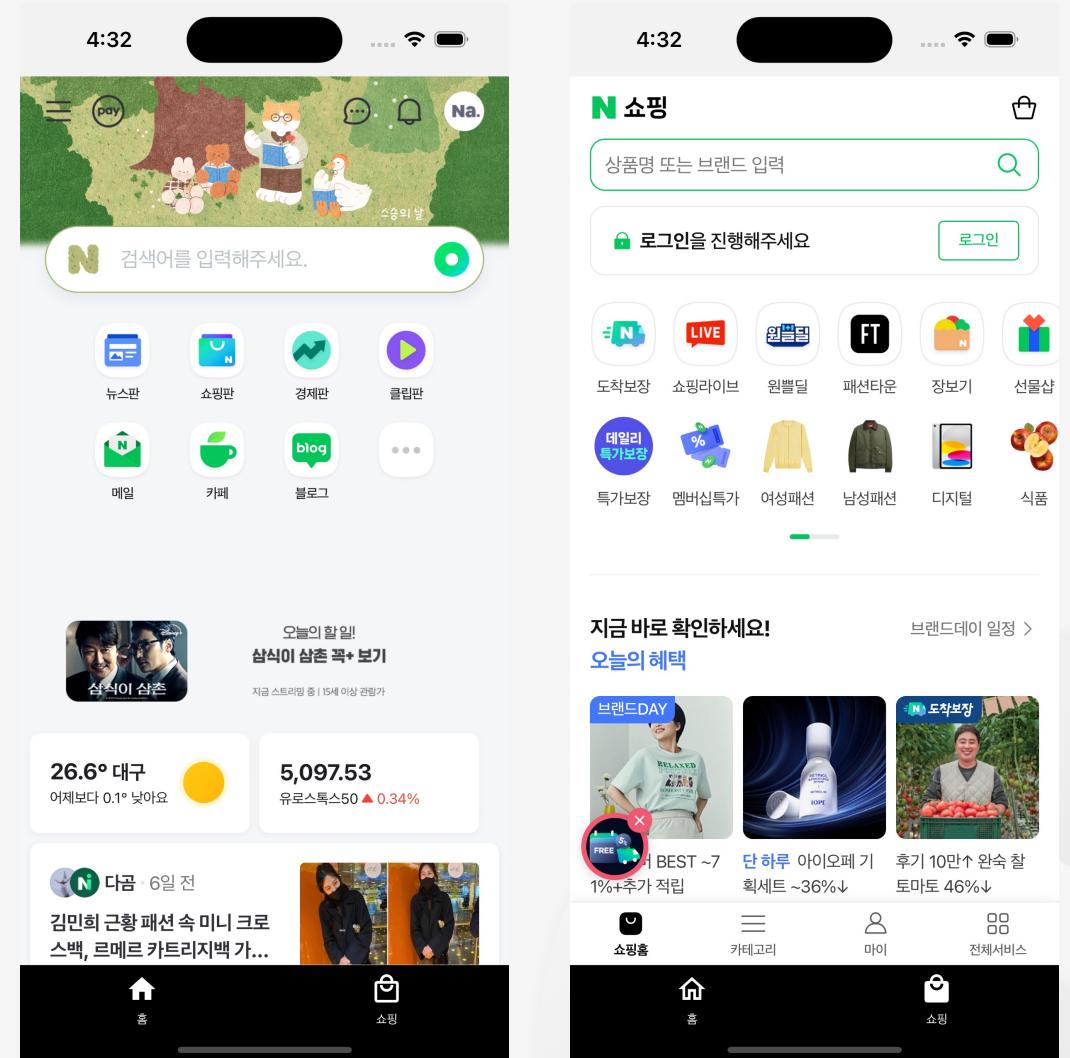
하이브리드 앱 구현을 위해 웹뷰를 최적화 할 수 있다

05

앱 아이콘 및 이름을 변경할 수 있다

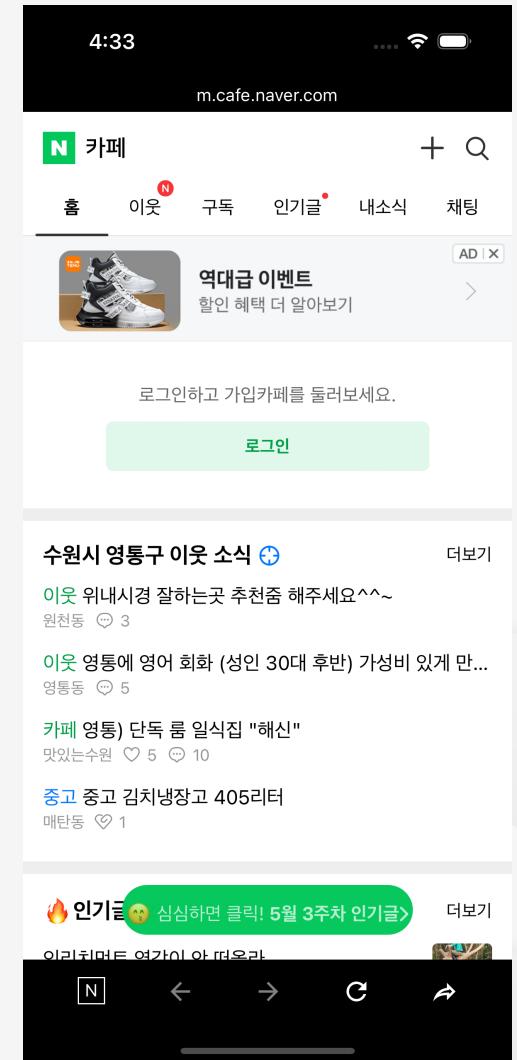
홈 스크린 / 쇼핑 스크린

- 텁 네비게이션 구현
- 아이콘 라이브러리 사용하기
- 웹뷰에 웹 사이트 로드하기
- Pull To Refresh 구현 하기
- 웹 사이트 리퀘스트 핸들링



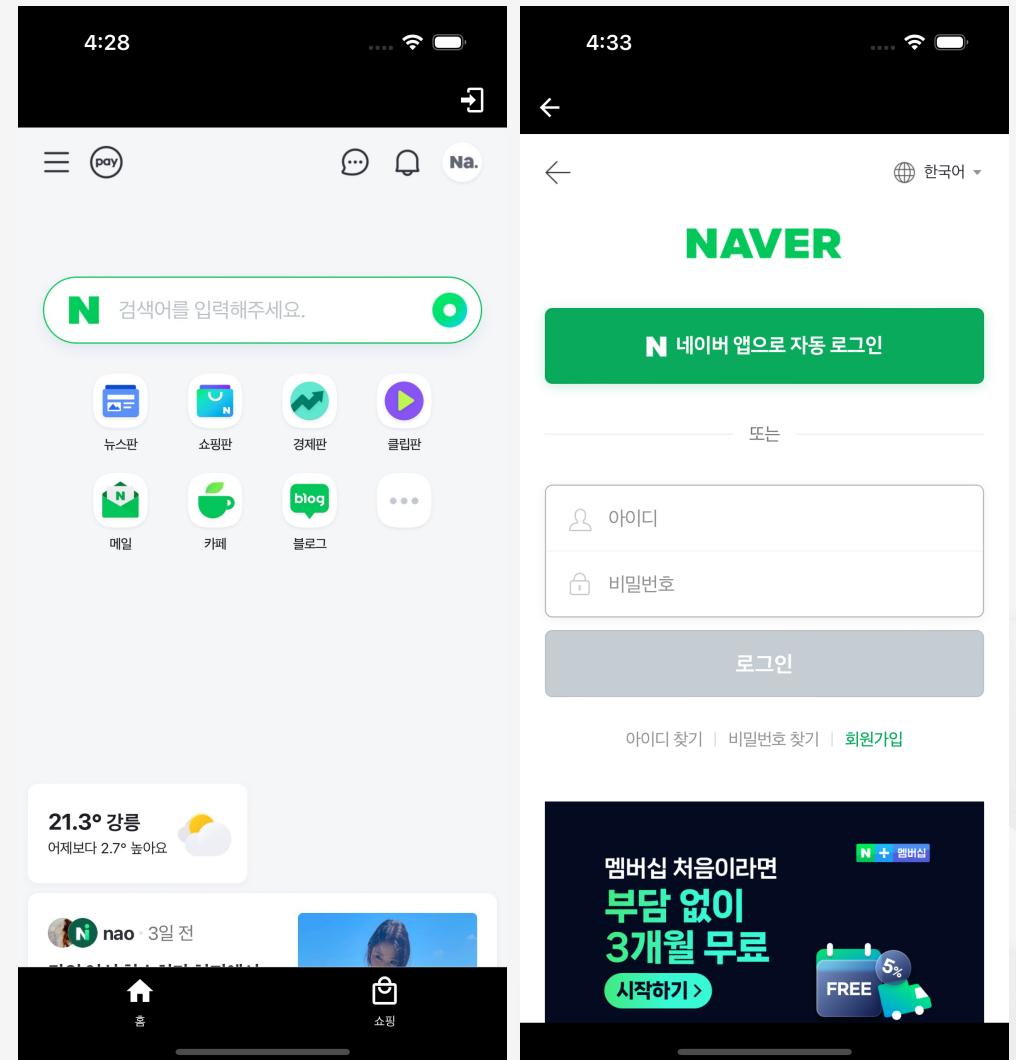
브라우저 스크린

- 웹 사이트 현재 주소 보여주기
- 웹 사이트 로딩 바 구현
- 브라우저 네비게이션 기능 구현 (앞으로, 뒤로, 새로고침)
- iOS, Android 네이티브 공유 기능 구현



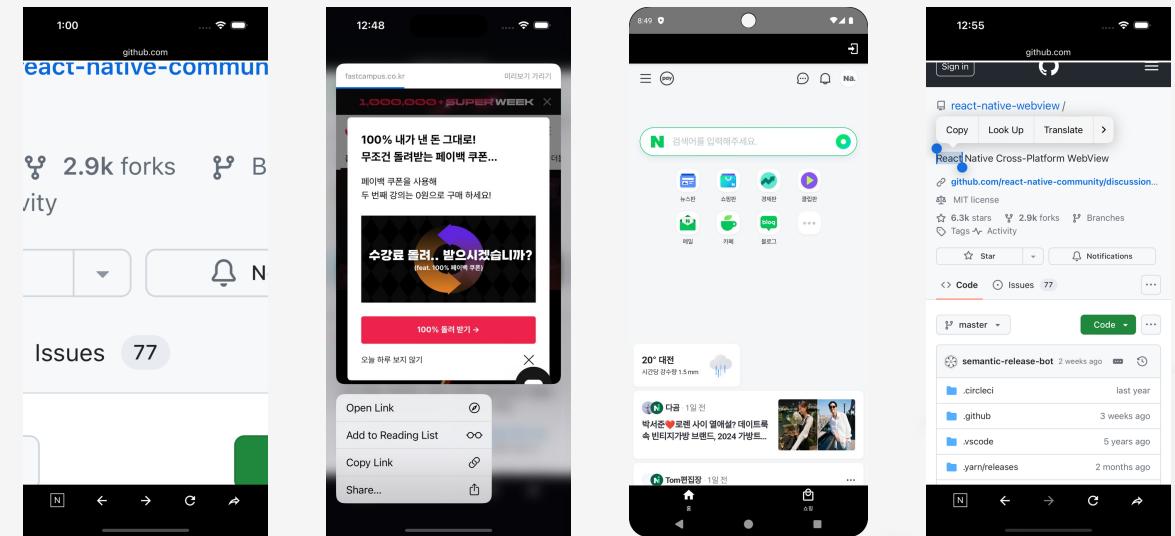
로그인 스크린

- 로그인 후 웹뷰 새로 고침 기능 구현
- 쿠키 읽기 및 쓰기



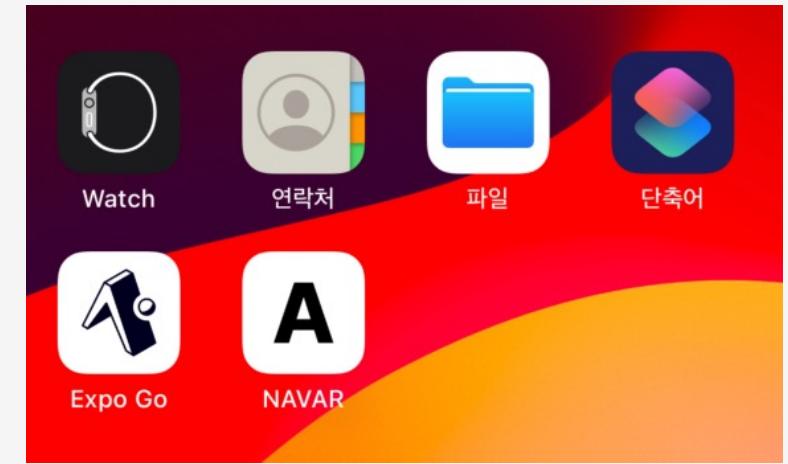
웹앱 최적화

- 핀치 줌/아웃 비활성화 하기
- 링크 롱 프레스 프리뷰 비활성화 하기
- 안드로이드 백버튼과 웹뷰 연결 하기
- 텍스트 롱 프레스 액션 비활성화 하기



앱 설정

- 앱 이름 변경하기
- 앱 아이콘 변경하기



프로젝트1 - 네이버 앱 클론 코딩 (React Native)

프로젝트 셋업

프로젝트 초기화

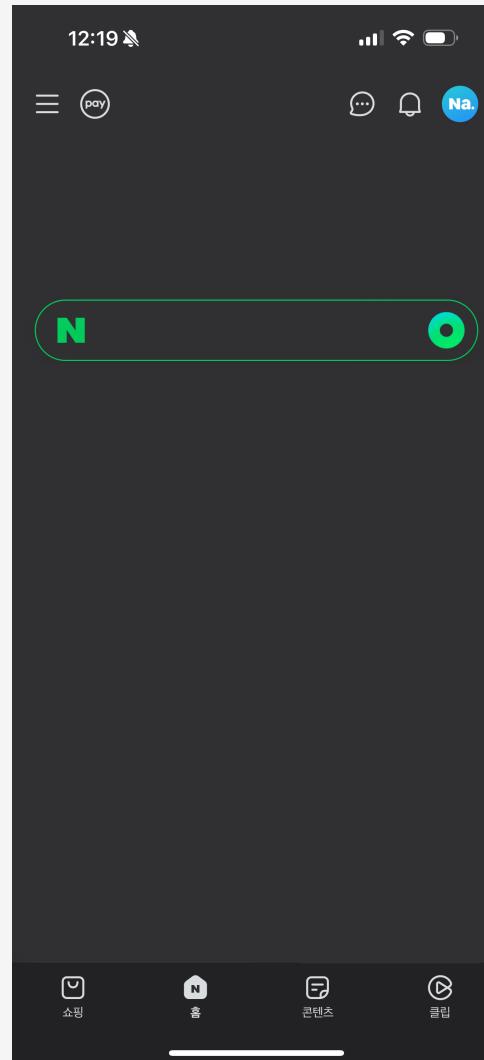
- `npx react-native@0.73.6 init NaverAppClone --version 0.73.6`

프로젝트1 - 네이버 앱 클론 코딩 (React Native)

앱 네비게이션 구현하기

네이버 앱 살펴보기

- 텁 네비게이션
- 스크린 네비게이션



학습 목표

01

React Navigation을 이용하여 탭 네비게이션을 구현할 수 있다

02

React Navigation을 이용하여 스택 네비게이션을 구현할 수 있다

React Navigation

- React Native 앱에서 내비게이션을 위한 라이브러리
- 다양한 네비게이션 패턴 지원 (Stack, Tab, Drawer)

- <https://reactnavigation.org/docs/getting-started>

프로젝트1 - 네이버 앱 클론 코딩 (React Native)

네이버 홈 화면 구현하기

학습 목표

01

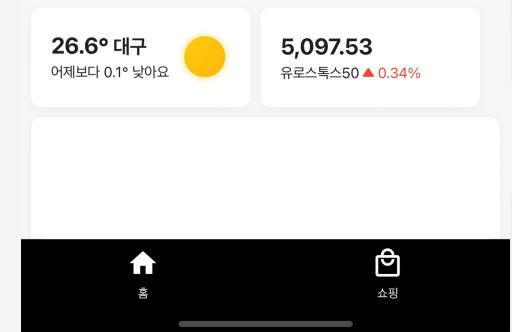
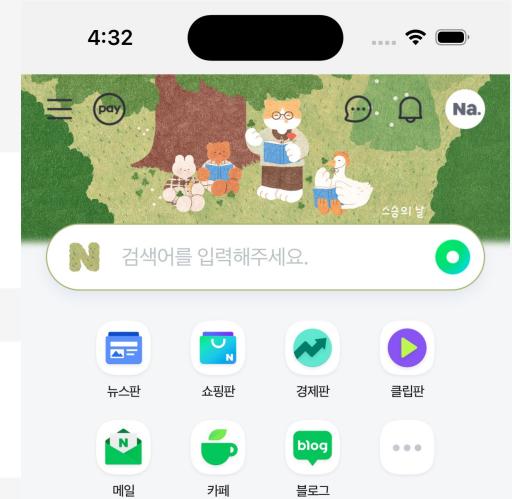
벡터 아이콘을 이용해서 커스텀 탭을 구현할 수 있다

02

웹뷰에 웹 사이트를 로드 할 수 있다

03

웹 사이트 리퀘스트를 처리하여 새로운 스크린으로 이동할 수 있다



React Native Vector Icons

- React Native 애플리케이션에서 벡터 아이콘을 쉽게 사용할 수 있도록 해주는 라이브러리
- 다양한 아이콘 세트 제공 (FontAwesome, MaterialIcons, Ionicons 등)
- 아이콘의 크기, 색상, 스타일 등을 쉽게 변경할 수 있음
- 벡터 그래픽을 사용하여 아이콘이 선명하고, 다양한 해상도에서 잘 보임
- <https://github.com/oblador/react-native-vector-icons>

프로젝트1 - 네이버 앱 클론 코딩 (React Native)

브라우저 기능 구현하기

학습 목표

01

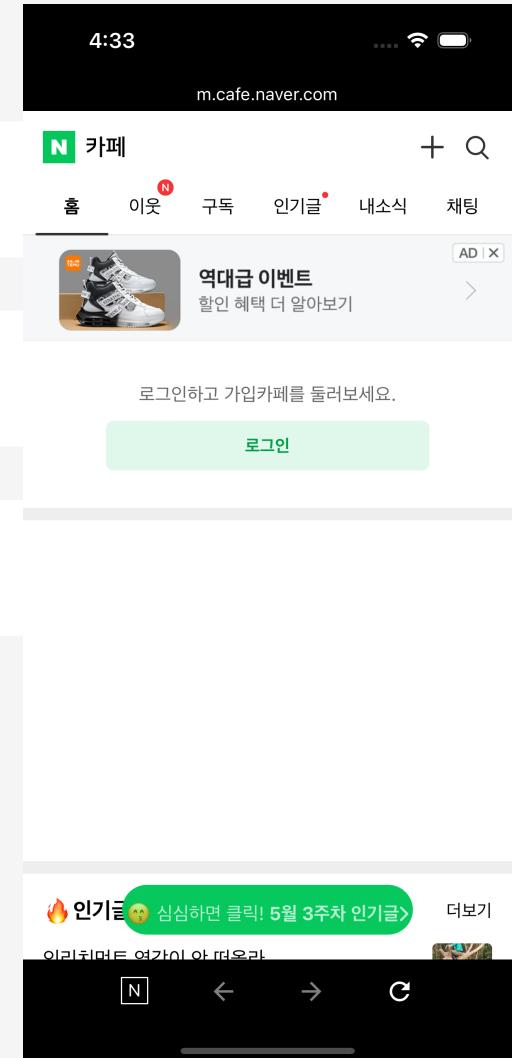
현재 접속 중인 웹 사이트 정보를 나타낼 수 있다

02

웹 사이트 로딩바를 구현 할 수 있다

03

뒤로가기, 앞으로가기, 새로고침 기능을 구현 할 수 있다



프로젝트1 - 네이버 앱 클론 코딩 (React Native)

쇼핑 화면 구현하기

학습 목표

01

웹뷰에 웹 사이트를 로드 할 수 있다

02

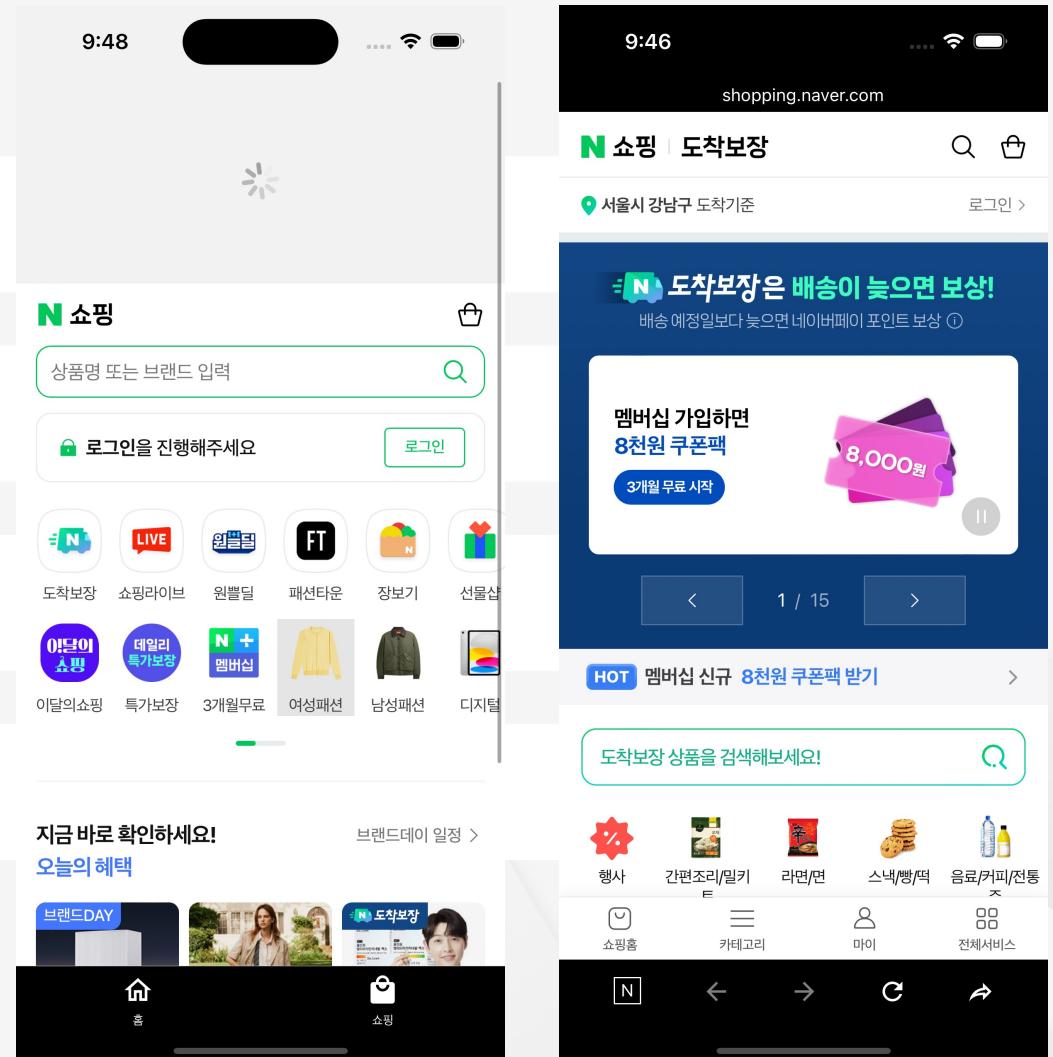
웹 사이트 리퀘스트를 처리하여 링크를 새로운 웹뷰로 띄울 수 있다

03

웹뷰 Pull To Refresh를 구현할 수 있다

04

플랫폼 네이티브 공유 기능을 구현 할 수 있다



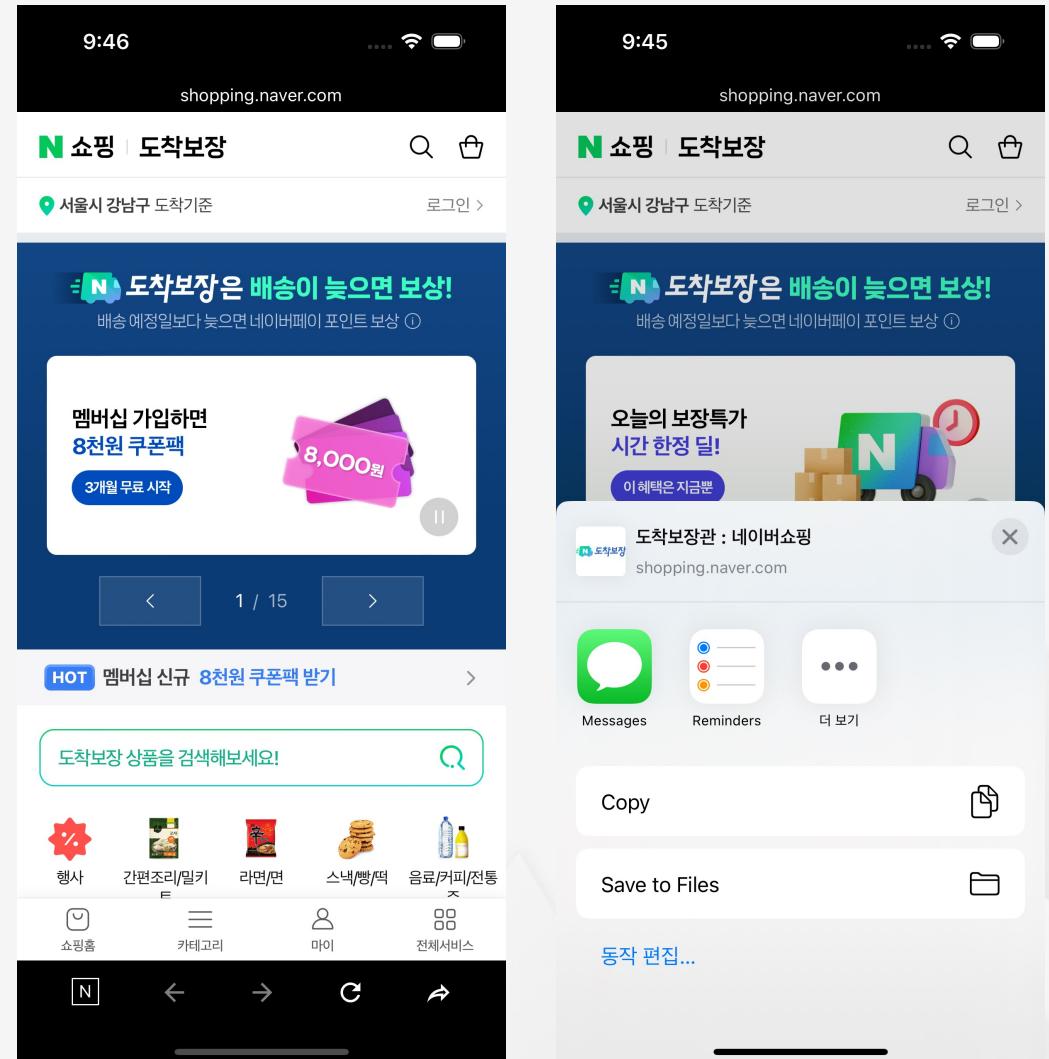
Pull To Refresh

- React Native의 ScrollView와 RefreshControl 컴포넌트 이용
- ScrollView: 스크롤이 가능한 영역 제공
- RefreshControl: ScrollView의 새로고침 상태 표시
- <https://reactnative.dev/docs/scrollview>
- <https://reactnative.dev/docs/refreshcontrol>



공유 기능 구현

- Web Share API가 있지만, 지원하지 않는 브라우저가 있음
 - 하이브리드 앱에 장점! 네이티브 기능을 사용할 수 있는 것
 - React Native Share API를 통해 네이티브 공유 기능을 호출 가능
-
- <https://developer.mozilla.org/en-US/docs/Web/API/Navigator/share>
 - <https://reactnative.dev/docs/share>



프로젝트1 - 네이버 앱 클론 코딩 (React Native)

로그인 가능 구현하기

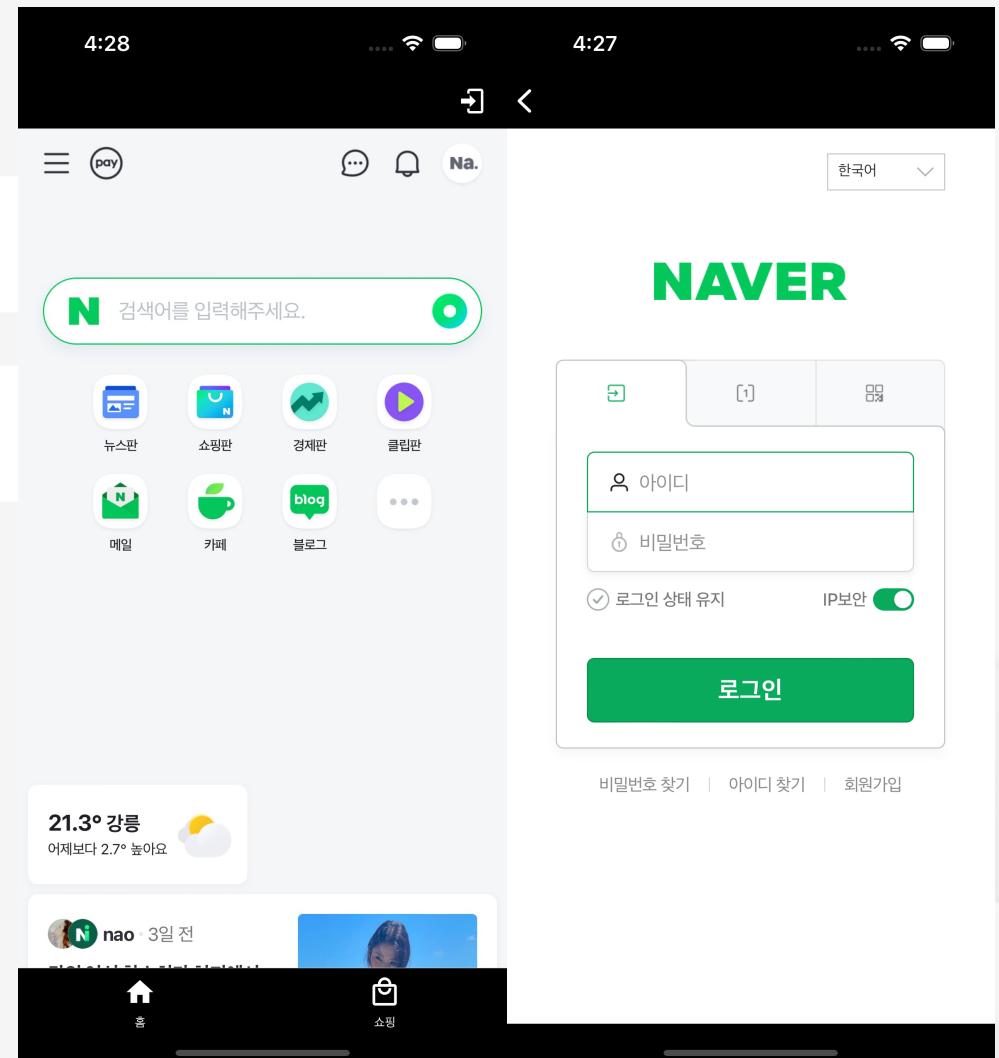
학습 목표

01

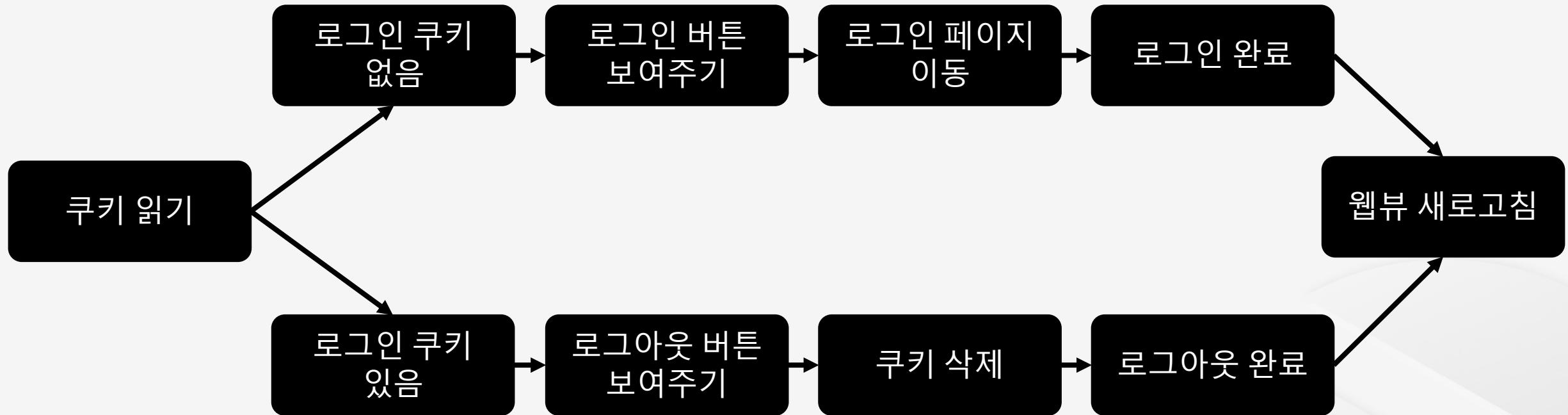
웹뷰에서 쿠키를 다룰 수 있다

02

React Native의 전역 상태 관리 방법을 학습한다



로그인 프로세스



@react-native-cookies/cookies

- React Native 애플리케이션에서 쿠키를 관리하는 라이브러리
- 쿠키 설정, 가져오기, 삭제 지원
- Expo에서는 사용할 수 없음 (eject 하여 사용해야 함)

- <https://github.com/react-native-cookies/cookies>

React Native Context

- Context는 컴포넌트 트리 전체에 걸쳐 상태를 공유할 수 있는 방법을 제공
- Provider와 Consumer
 - Context.Provider로 데이터를 공급
 - Context.Consumer 또는 useContext 훅으로 데이터를 소비

프로젝트1 - 네이버 앱 클론 코딩 (React Native)

웹뷰 최적화 하기

학습 목표

01

핀치 줌/아웃 비활성화 할 수 있다

02

링크 롱 프레스 시 프리뷰 비활성화 할 수 있다

03

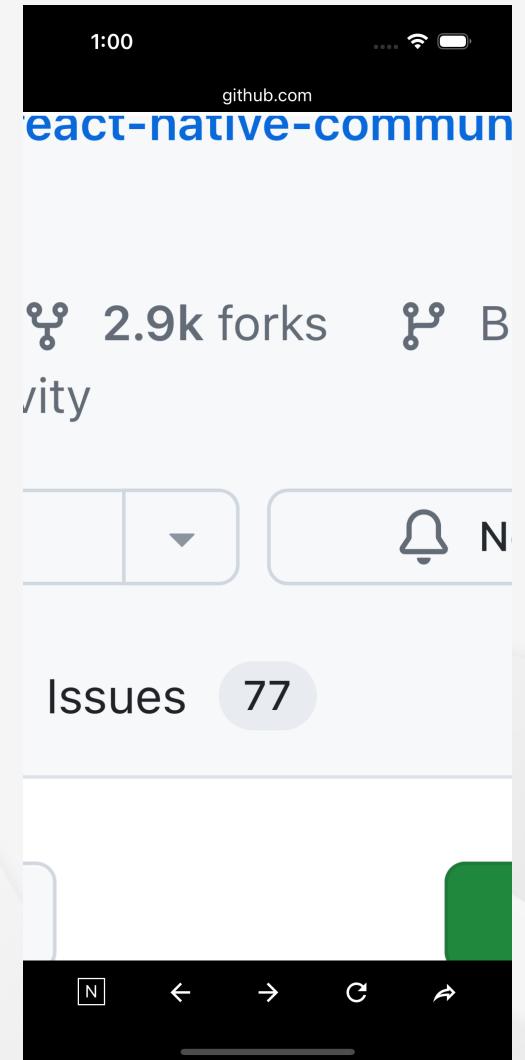
안드로이드 백버튼과 웹뷰 연결할 수 있다

04

텍스트 롱 프레스 액션 비활성화 할 수 있다

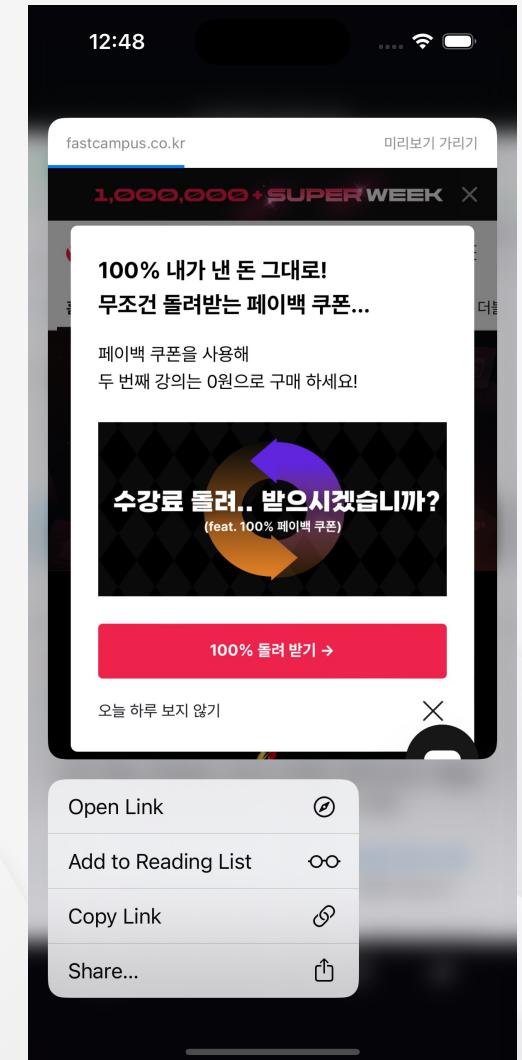
핀치 줌/아웃 비활성화

- 시뮬레이터에서 핀치 줌/아웃 하는 방법
 - iOS: Option 키 누른 상태로
 - Android: Cntl 키 누른 상태로
- 웹 페이지 자체에서 설정해야 함
- 웹 페이지를 수정할 수 없다면?
 - `injectedJavaScript`를 이용해서 웹 페이지에 자바스크립트 주입
 - `onMessage`를 꼭 함께 써줘야함



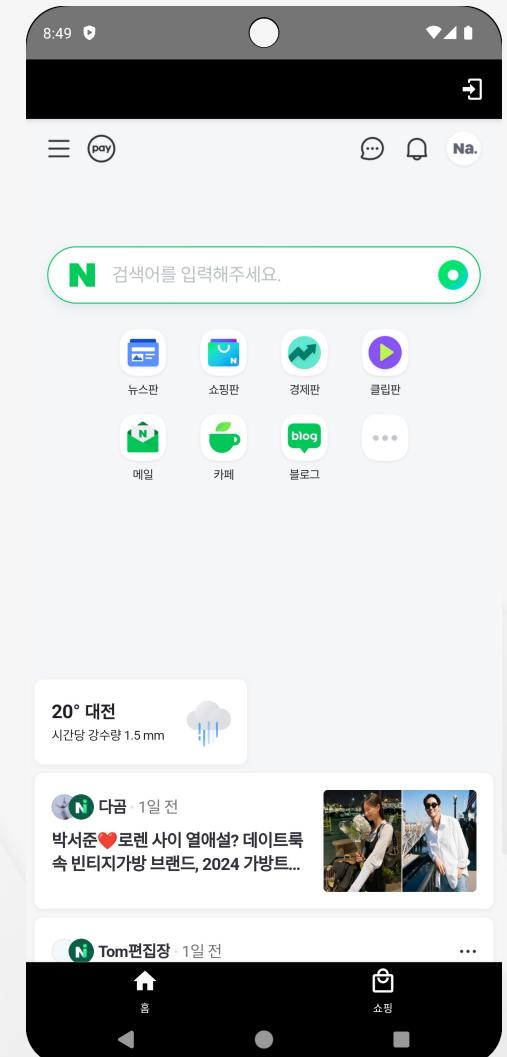
iOS 링크 프리뷰 비활성화

- `allowsLinkPreview`으로 설정 가능
- 시뮬레이터에서는 기본 값이 비활성화로 설정되어 있음



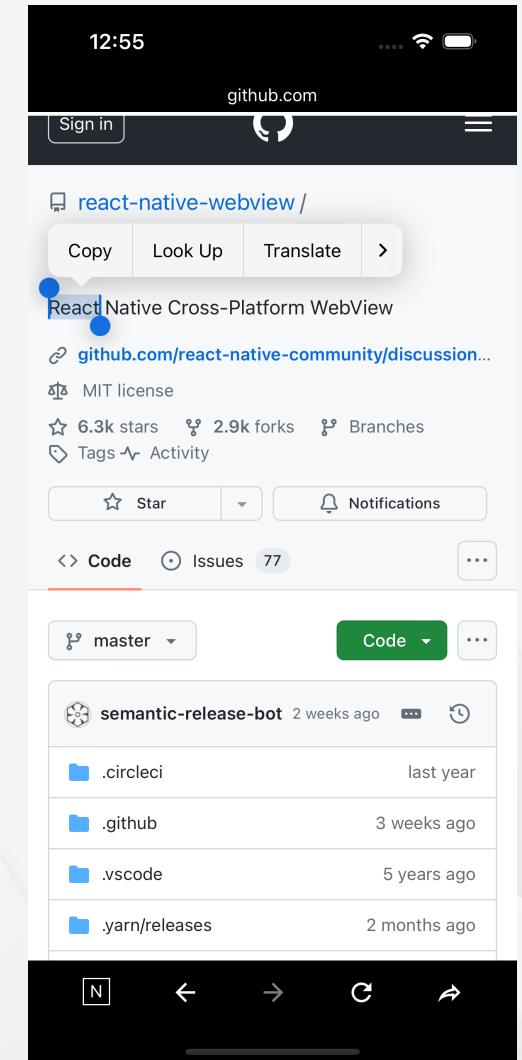
안드로이드 백버튼과 웹뷰 연결

- 안드로이드 기본 백버튼 동작은 앱 스크린 네비게이션
- 앱 스크린 네비게이션이 아니라, 웹뷰 페이지 네비게이션이 되도록 설정 필요
- [@react-native-community/hooks](https://github.com/react-native-community/hooks)을 이용해서 백버튼 동작을 커스텀화 할 수 있음
- <https://github.com/react-native-community/hooks>



텍스트 롱 프레스 비활성화

- 웹 페이지 자체에서 설정해야 함
- 웹 페이지를 수정할 수 없다면?
 - `injectedJavaScript`를 이용해서 웹 페이지에 자바스크립트 주입
 - `onMessage`를 꼭 함께 써줘야함



프로젝트1 - 네이버 앱 클론 코딩 (React Native)

앱 아이콘 설정하기

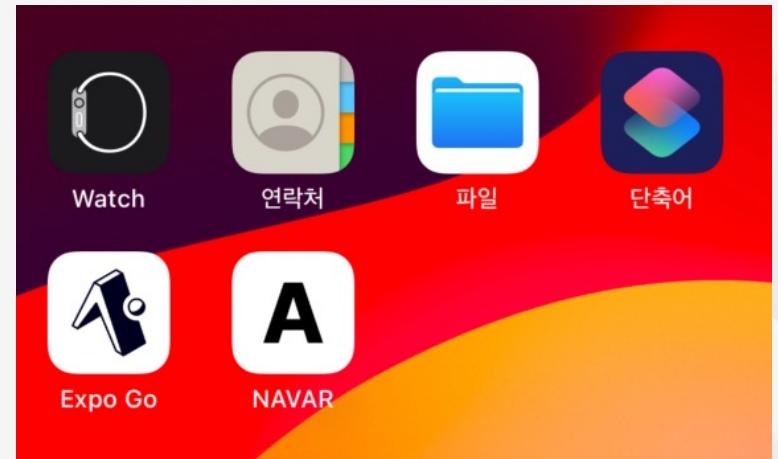
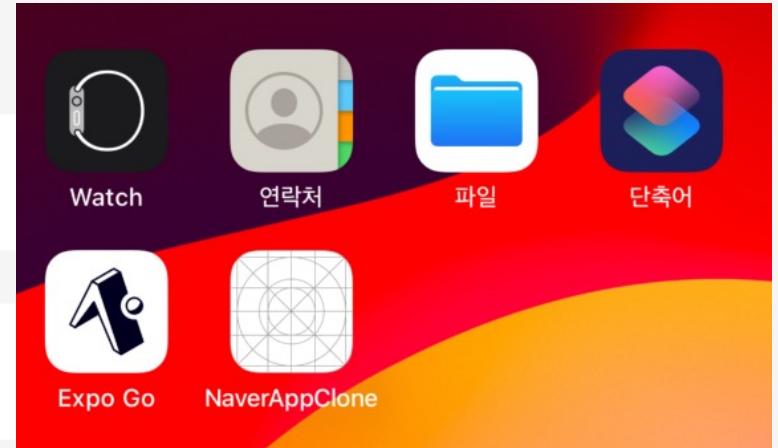
학습 목표

01

앱 이름을 변경 할 수 있다

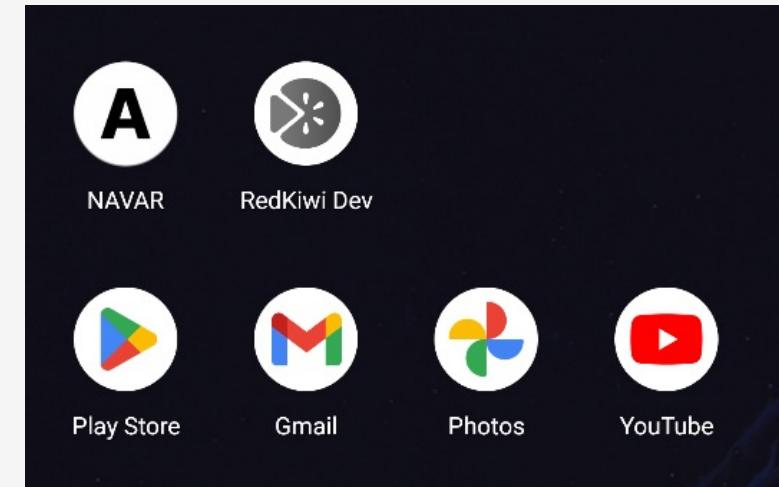
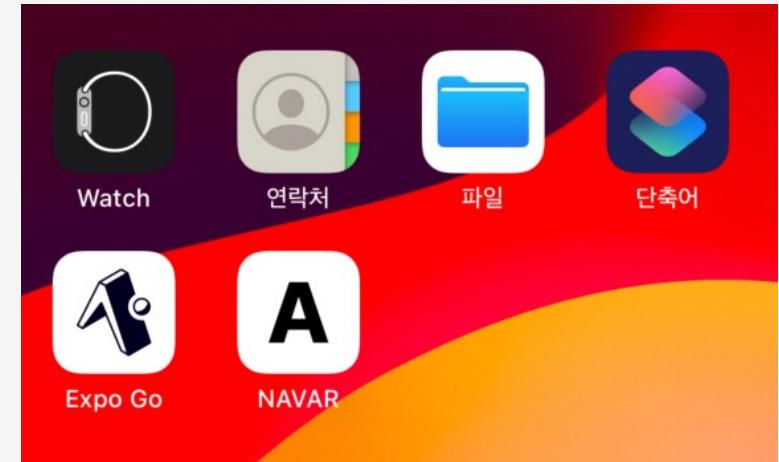
02

앱 아이콘을 변경 할 수 있다



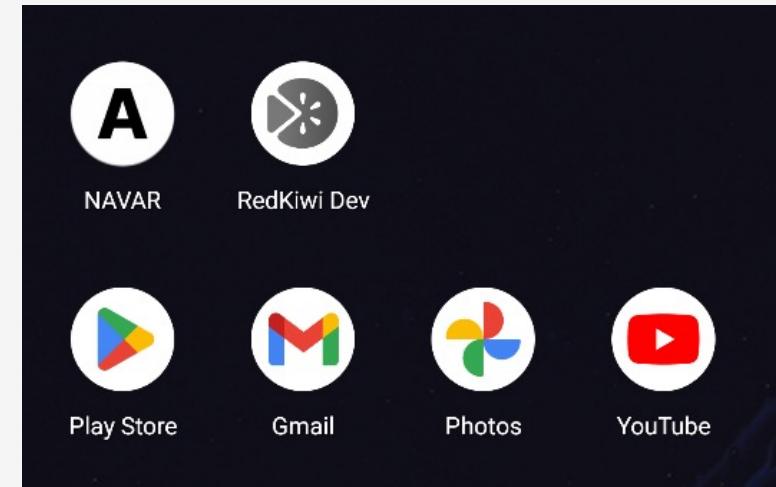
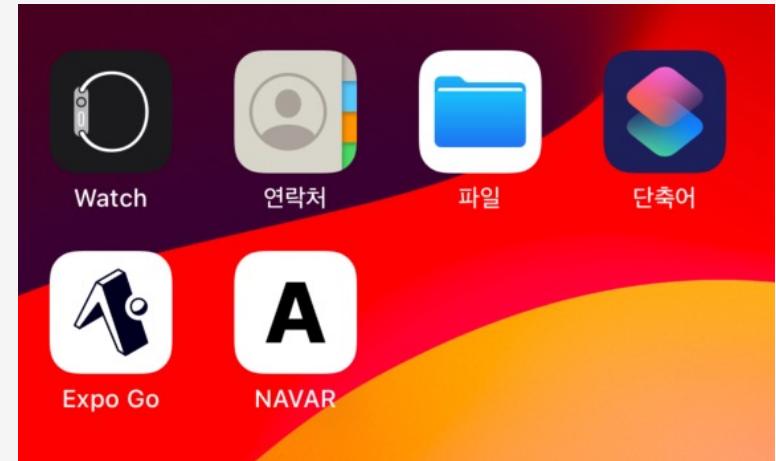
앱 이름 바꾸기

- iOS
 - Info.plist에서 Bundle display name 수정
- Android
 - strings.xml에서 app_name 수정



앱 아이콘 바꾸기

- 여러 해상도의 아이콘 파일 필요
- 단일 아이콘 이미지를 여러 해상도로 변환해주는 웹 서비스
- <https://easyappicon.com/>



프로젝트1 - 네이버 앱 클론 코딩 (Expo)

프로젝트 셋업

프로젝트 초기화

- `yarn create expo --template expo-template-blank-typescript@50`

ESLint 설정

- JavaScript와 TypeScript 코드의 문제를 분석 및 품질을 유지하기 위한 도구
- 주요 기능
 - 문법 검사: 코드에서 문법 오류 감지
 - 코드 스타일 검사: 일관된 코드 스타일을 유지
 - 최적화 제안: 비효율적인 코드 패턴을 감지하고 개선 사항을 제안
 - 자동 수정: 일부 문제를 자동으로 수정
- 장점
 - 일관성 유지: 팀의 코드 스타일을 일관되게 유지할 수 있습니다.
 - 디버깅 시간 절약: 코드 작성 시 실시간으로 오류를 발견하고 수정할 수 있습니다.
 - 확장성: 다양한 플러그인과 설정을 통해 필요에 맞게 확장 가능합니다.

Prettier 설정

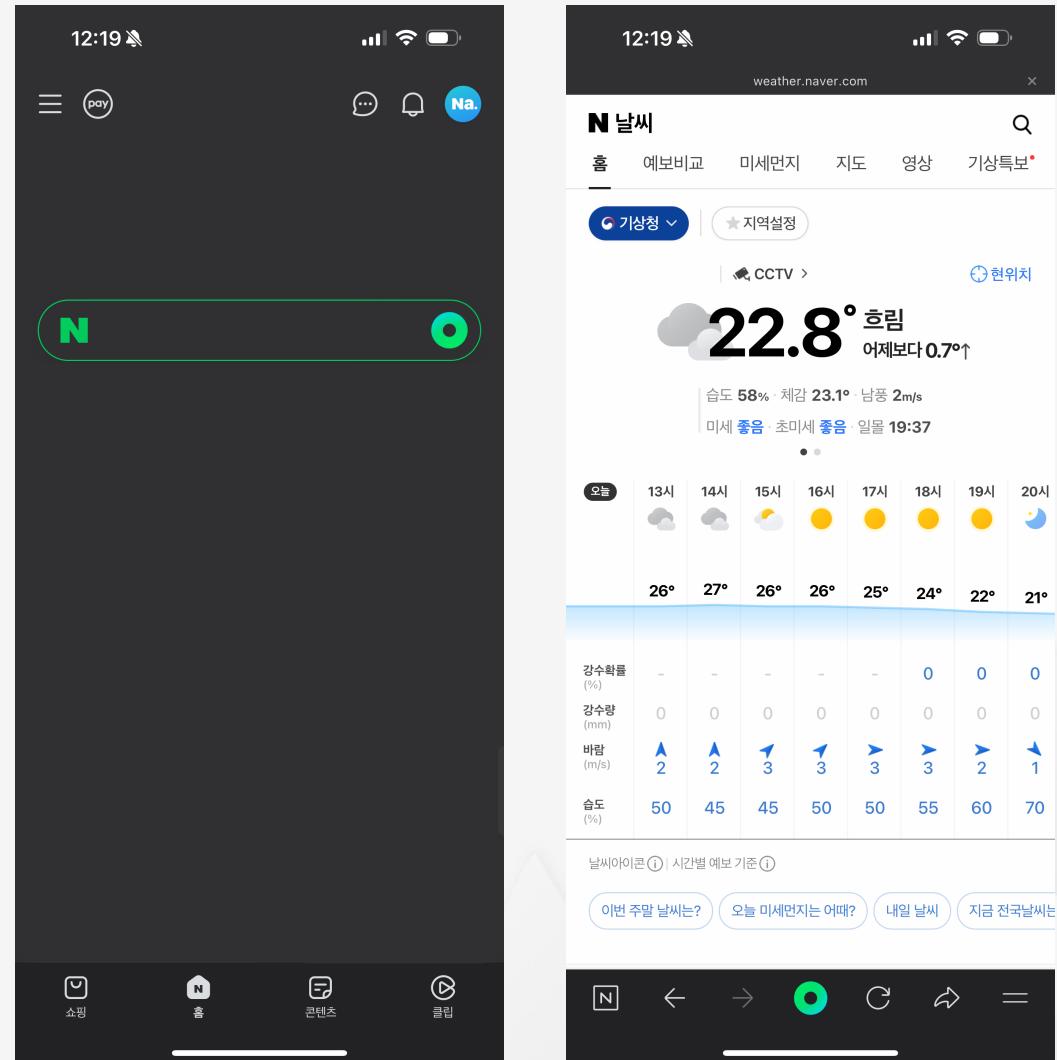
- Prettier는 코드의 일관된 스타일을 유지하기 위한 코드 포매터
- 주요 기능
 - **자동 포매팅**: 지정된 스타일 규칙에 따라 코드를 자동으로 포매팅
 - **일관성 유지**: 팀 내 모든 개발자가 동일한 코드 스타일을 유지하도록 보장
 - **간편한 설정**: 미리 정의된 규칙을 따르며, 별도의 설정 없이도 바로 사용
- 장점
 - **코드 일관성**: 코드 리뷰 시 스타일 관련 논쟁을 없애고, 코드의 가독성을 높임
 - **시간 절약**: 수동으로 코드 스타일을 맞추는 시간을 절약
 - **유연한 통합**: 다양한 에디터와 IDE, 버전 관리 시스템과 쉽게 통합

프로젝트1 - 네이버 앱 클론 코딩 (Expo)

앱 네비게이션 구현하기

네이버 앱 살펴보기

- 텁 네비게이션
- 스크린 네비게이션



학습 목표

01

Expo Router를 이용하여 탭 네비게이션을 구현할 수 있다

02

Expo Router를 이용하여 스택 네비게이션을 구현할 수 있다

Expo Router

- Expo 프로젝트에서 파일 기반 라우팅을 제공하는 라이브러리
 - 다양한 네비게이션 패턴 지원 (Stack, Tab, Drawer)
 - React Navigation 기반
-
- <https://docs.expo.dev/router/introduction/>

프로젝트1 - 네이버 앱 클론 코딩 (Expo)

네이버 홈 화면 구현하기

학습 목표

01

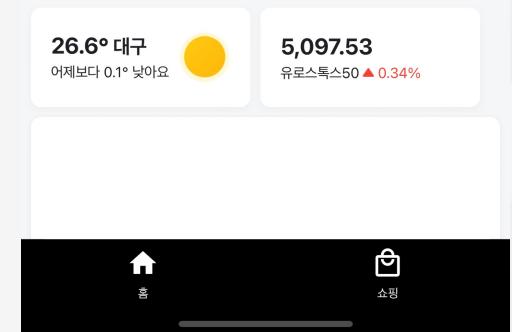
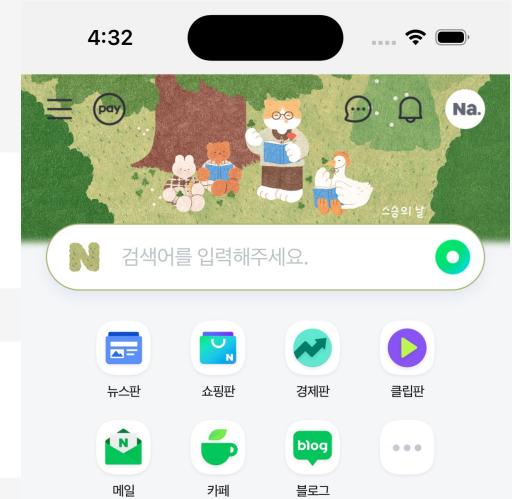
벡터 아이콘을 이용해서 커스텀 탭을 구현할 수 있다

02

웹뷰에 웹 사이트를 로드 할 수 있다

03

웹 사이트 리퀘스트를 처리하여 새로운 스크린으로 이동할 수 있다



Expo Vector Icons

- Expo 애플리케이션에서 벡터 아이콘을 쉽게 사용할 수 있도록 해주는 라이브러리
- 기본으로 설치되어 있어서 바로 사용 가능
- react native vector icons 기반
- 다양한 아이콘 세트 제공 (FontAwesome, MaterialIcons, Ionicons 등)
- 아이콘의 크기, 색상, 스타일 등을 쉽게 변경할 수 있음
- 벡터 그래픽을 사용하여 아이콘이 선명하고, 다양한 해상도에서 잘 보임
- <https://docs.expo.dev/guides/icons/#expovector-icons>

프로젝트1 - 네이버 앱 클론 코딩 (Expo)

브라우저 기능 구현하기

학습 목표

01

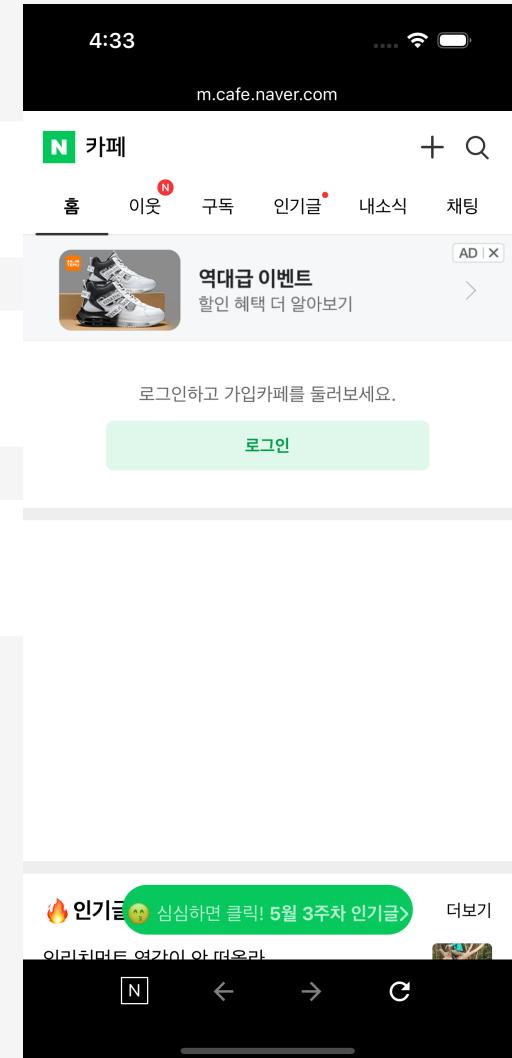
현재 접속 중인 웹 사이트 정보를 나타낼 수 있다

02

웹 사이트 로딩바를 구현 할 수 있다

03

뒤로가기, 앞으로가기, 새로고침 기능을 구현 할 수 있다



프로젝트1 - 네이버 앱 클론 코딩 (Expo)

쇼핑 화면 구현하기

학습 목표

01

웹뷰에 웹 사이트를 로드 할 수 있다

02

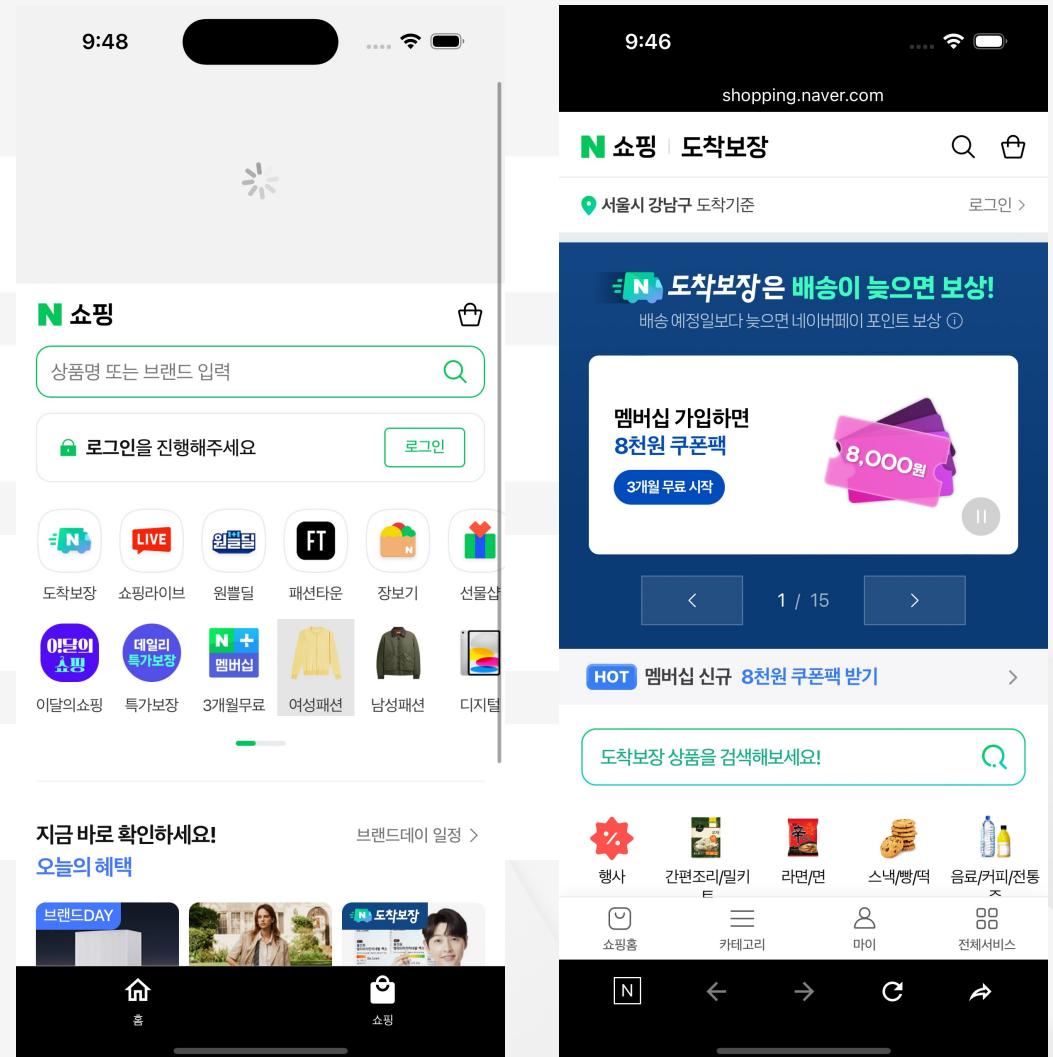
웹 사이트 리퀘스트를 처리하여 링크를 새로운 웹뷰로 띄울 수 있다

03

웹뷰 Pull To Refresh를 구현할 수 있다

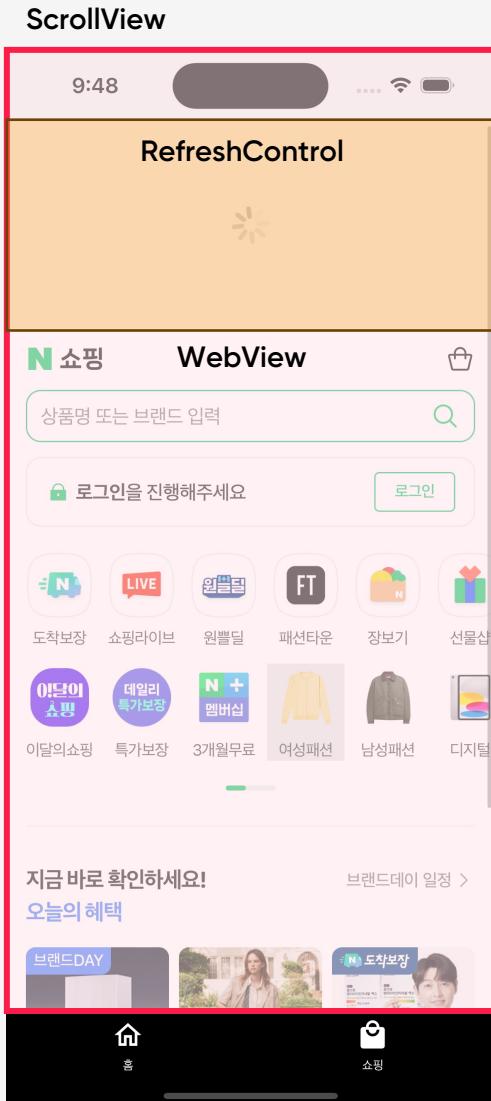
04

플랫폼 네이티브 공유 기능을 구현 할 수 있다



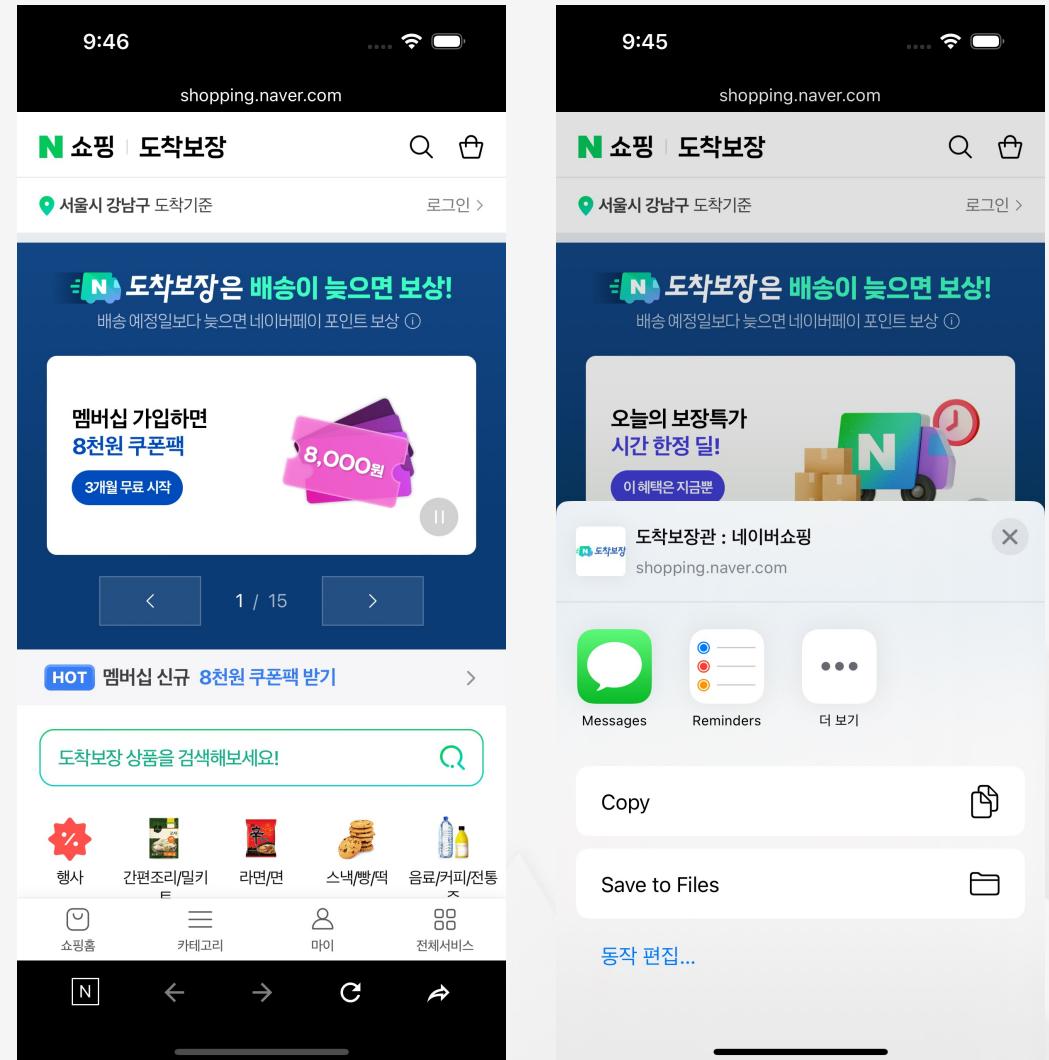
Pull To Refresh

- React Native의 ScrollView와 RefreshControl 컴포넌트 이용
- ScrollView: 스크롤이 가능한 영역 제공
- RefreshControl: ScrollView의 새로고침 상태 표시
- <https://reactnative.dev/docs/scrollview>
- <https://reactnative.dev/docs/refreshcontrol>



공유 기능 구현

- Web Share API가 있지만, 지원하지 않는 브라우저가 있음
 - 하이브리드 앱에 장점! 네이티브 기능을 사용할 수 있는 것
 - React Native Share API를 통해 네이티브 공유 기능을 호출 가능
-
- <https://developer.mozilla.org/en-US/docs/Web/API/Navigator/share>
 - <https://reactnative.dev/docs/share>



프로젝트1 - 네이버 앱 클론 코딩 (Expo)

로그인 가능 구현하기

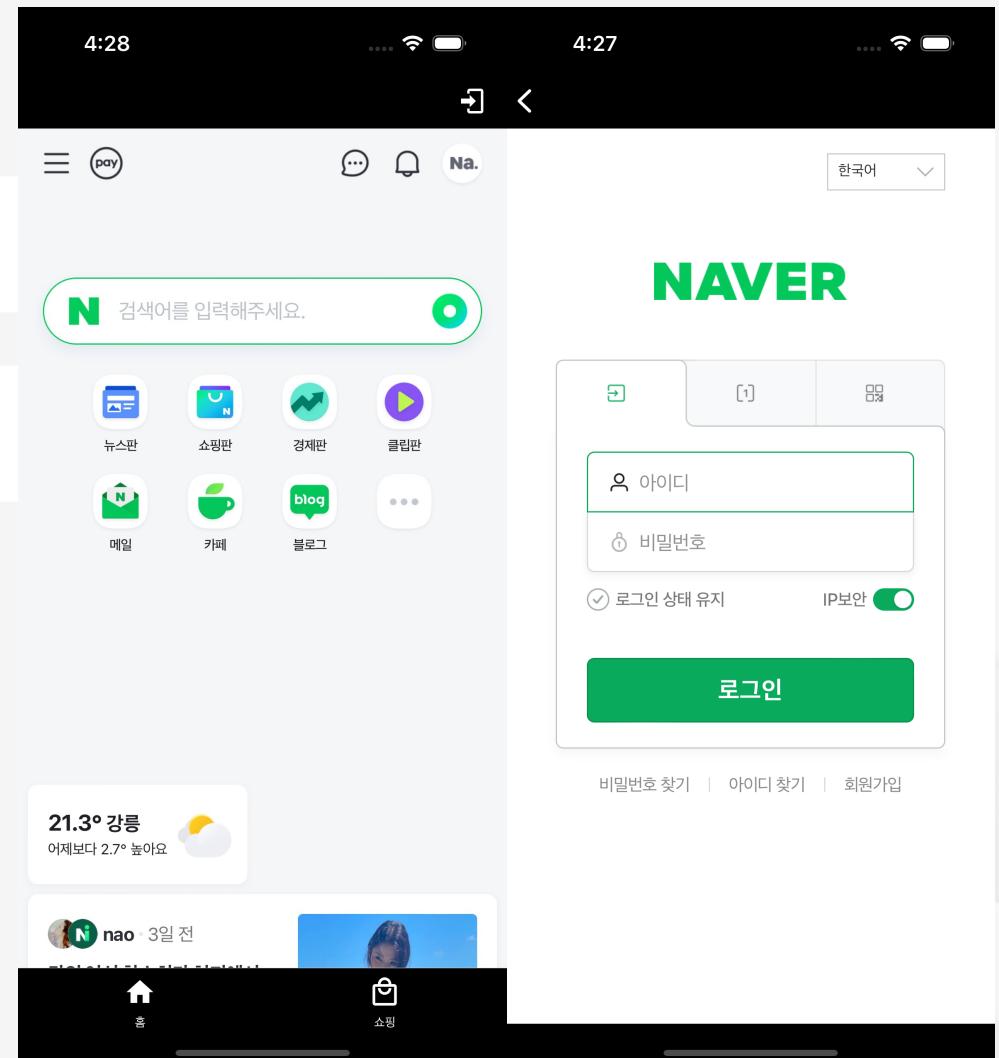
학습 목표

01

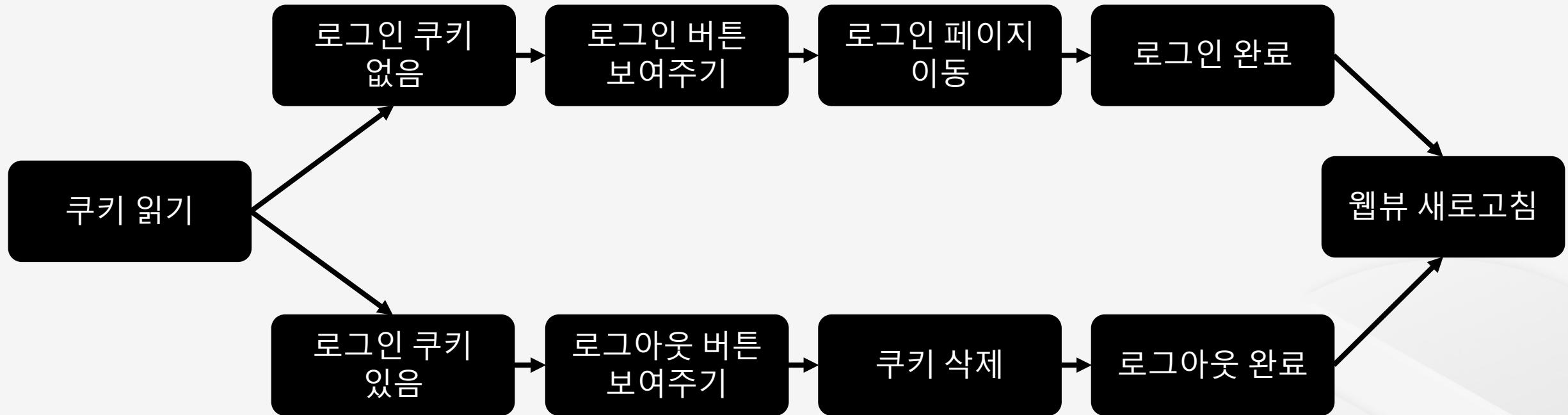
웹뷰에서 쿠키를 다룰 수 있다

02

React Native의 전역 상태 관리 방법을 학습한다



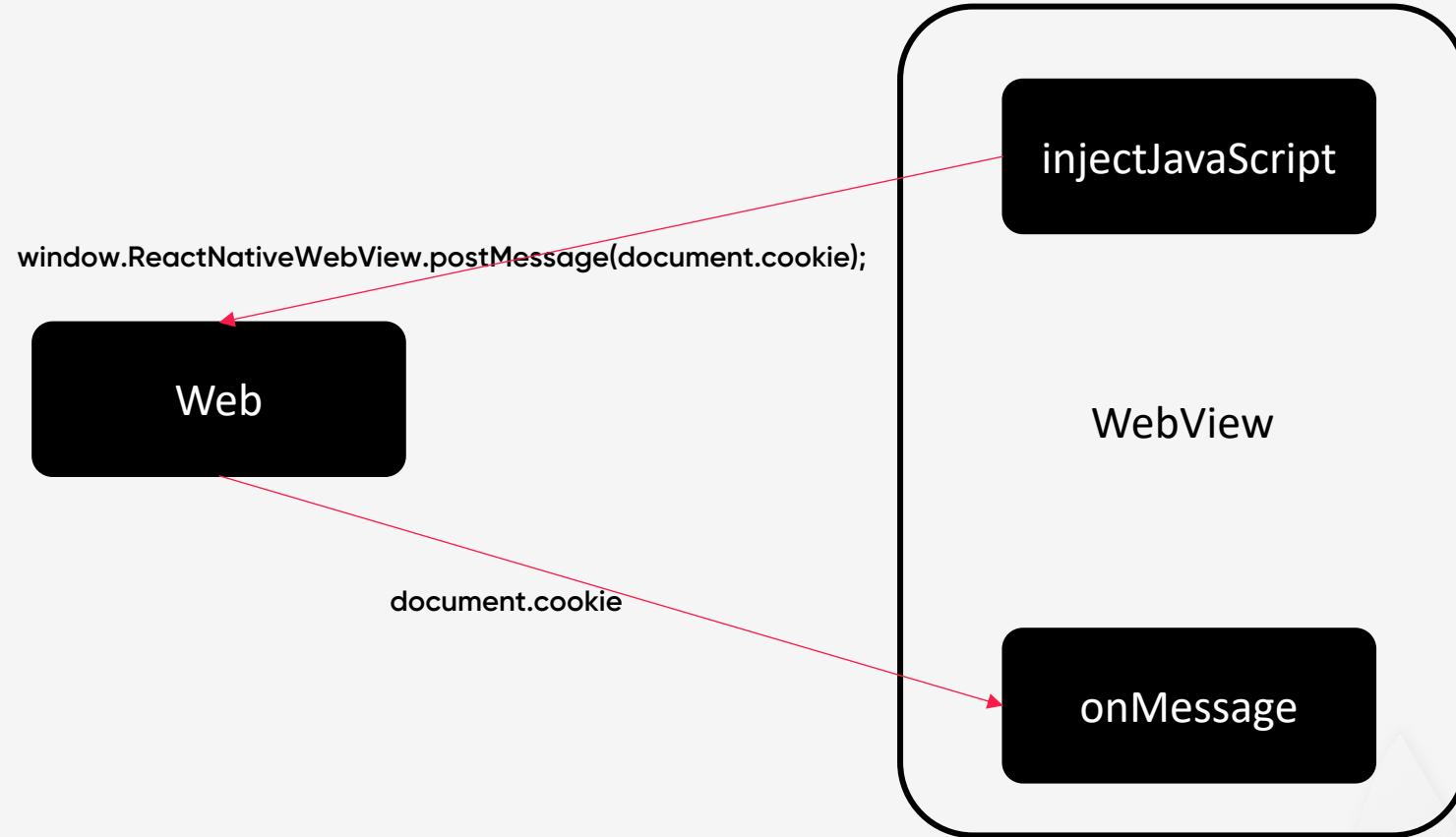
로그인 프로세스



Document: cookie property

- React Native CLI에서는 @react-native-cookies/cookies 이용
- Expo에서는 사용할 수 없음 (eject 하여 사용해야 함)
- 쿠키 읽기
 - cookies = document.cookie
- 쿠키 쓰기
 - document.cookie = newCookie
- <https://developer.mozilla.org/en-US/docs/Web/API/Document/cookie>

쿠키 읽기 로직



쿠키 쓰기 로직

```
document.cookie = 'NID_SES=; expires=Thu, 01 Jan 1970  
00:00:00 UTC; path=/; domain=.naver.com';  
window.ReactNativeWebView.postMessage(document.cookie);
```

Web

injectJavaScript

WebView

onMessage

document.cookie

React Native Context

- Context는 컴포넌트 트리 전체에 걸쳐 상태를 공유할 수 있는 방법을 제공
- Provider와 Consumer
 - Context.Provider로 데이터를 공급
 - Context.Consumer 또는 useContext 훅으로 데이터를 소비

프로젝트1 - 네이버 앱 클론 코딩 (Expo)

웹뷰 최적화 하기

학습 목표

01

핀치 줌/아웃 비활성화 할 수 있다

02

링크 롱 프레스 시 프리뷰 비활성화 할 수 있다

03

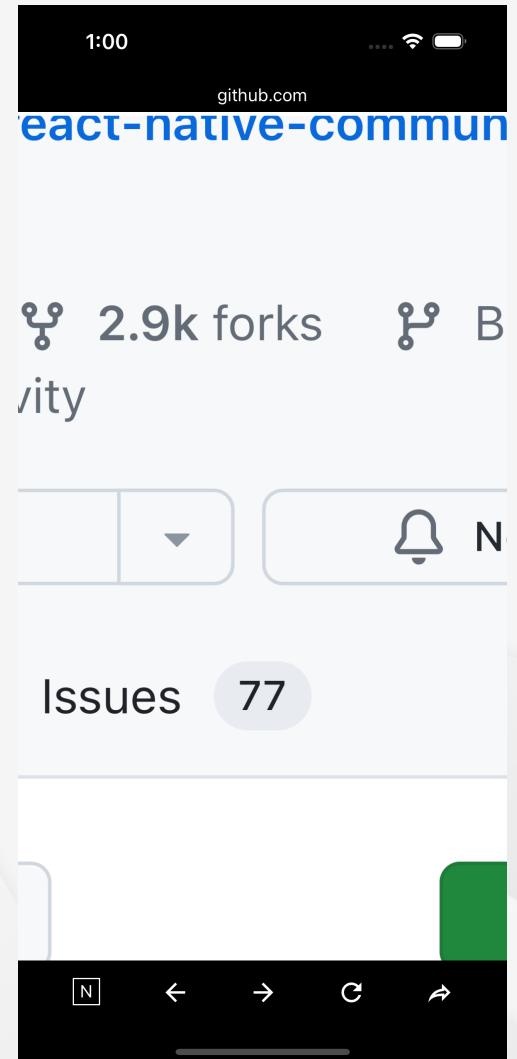
안드로이드 백버튼과 웹뷰 연결할 수 있다

04

텍스트 롱 프레스 액션 비활성화 할 수 있다

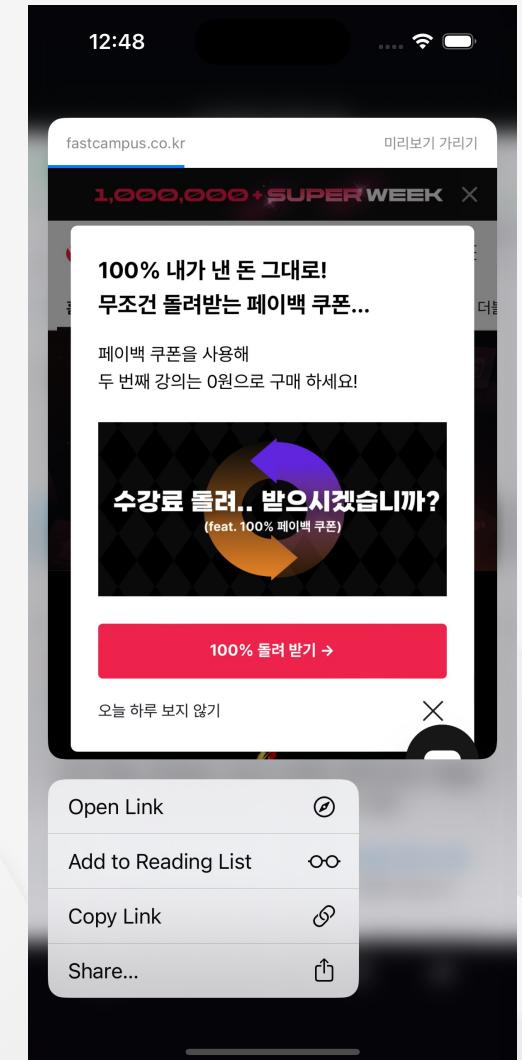
핀치 줌/아웃 비활성화

- 시뮬레이터에서 핀치 줌/아웃 하는 방법
 - iOS: Option 키 누른 상태로
 - Android: Cntl 키 누른 상태로
- 웹 페이지 자체에서 설정해야 함
- 웹 페이지를 수정할 수 없다면?
 - `injectedJavaScript`를 이용해서 웹 페이지에 자바스크립트 주입
 - `onMessage`를 꼭 함께 써줘야함



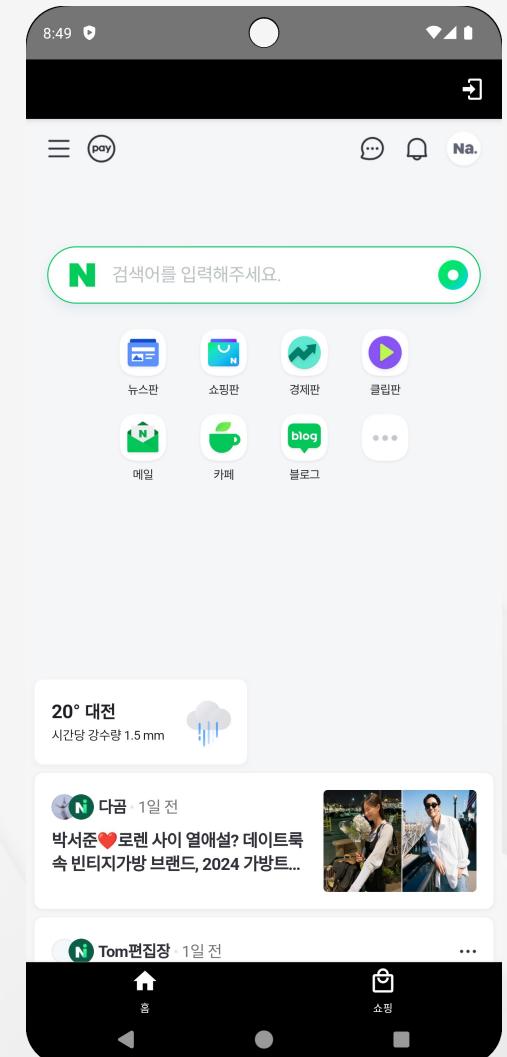
iOS 링크 프리뷰 비활성화

- `allowsLinkPreview`으로 설정 가능
- 시뮬레이터에서는 기본 값이 비활성화로 설정되어 있음



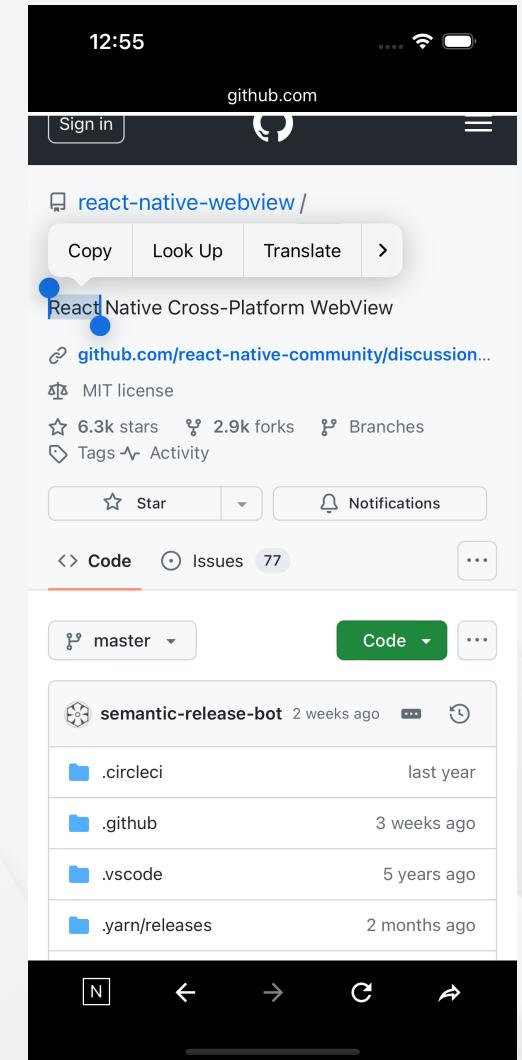
안드로이드 백버튼과 웹뷰 연결

- 안드로이드 기본 백버튼 동작은 앱 스크린 네비게이션
- 앱 스크린 네비게이션이 아니라, 웹뷰 페이지 네비게이션이 되도록 설정 필요
- [@react-native-community/hooks](https://github.com/react-native-community/hooks)을 이용해서 백버튼 동작을 커스텀화 할 수 있음
- <https://github.com/react-native-community/hooks>



텍스트 롱 프레스 비활성화

- 웹 페이지 자체에서 설정해야 함
- 웹 페이지를 수정할 수 없다면?
 - `injectedJavaScript`를 이용해서 웹 페이지에 자바스크립트 주입
 - `onMessage`를 꼭 함께 써줘야함



프로젝트1 - 네이버 앱 클론 코딩 (Expo)

앱 아이콘 설정하기

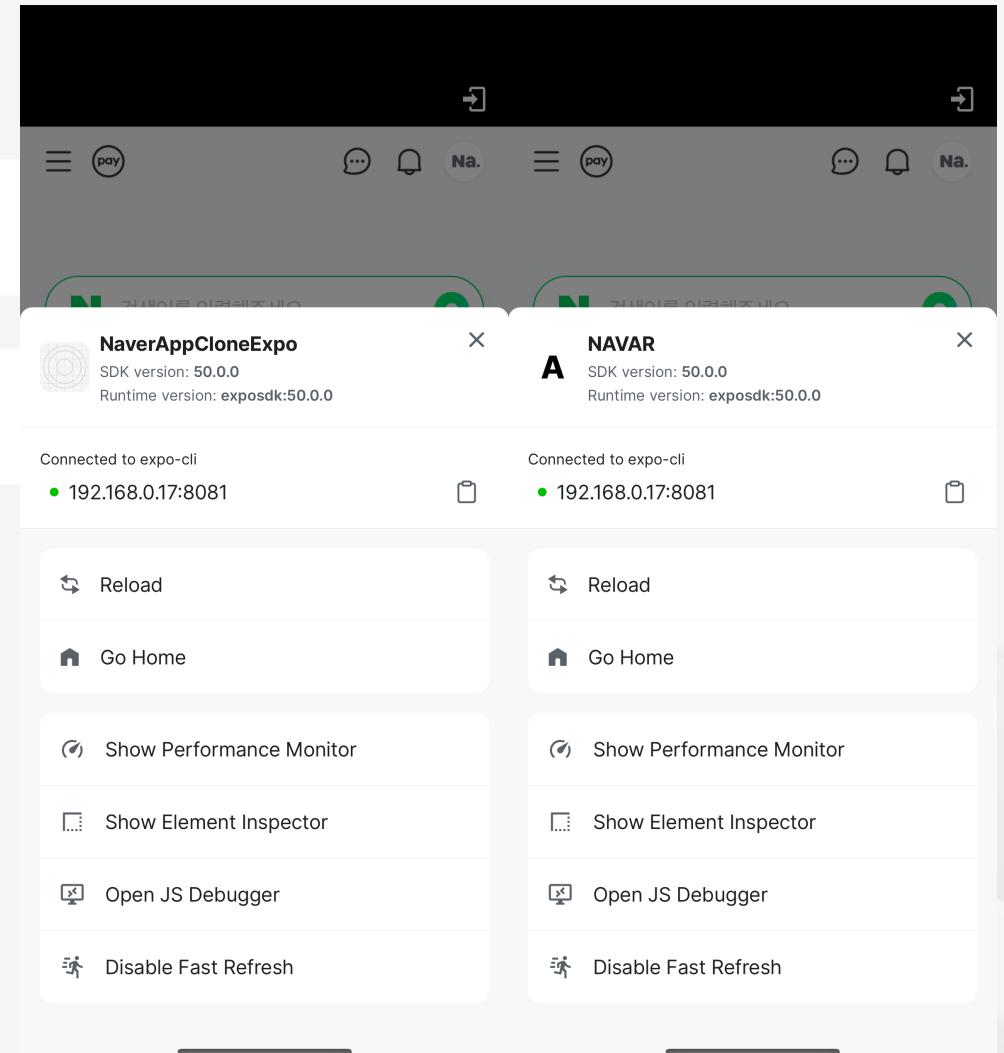
학습 목표

01

앱 이름을 변경 할 수 있다

02

앱 아이콘을 변경 할 수 있다

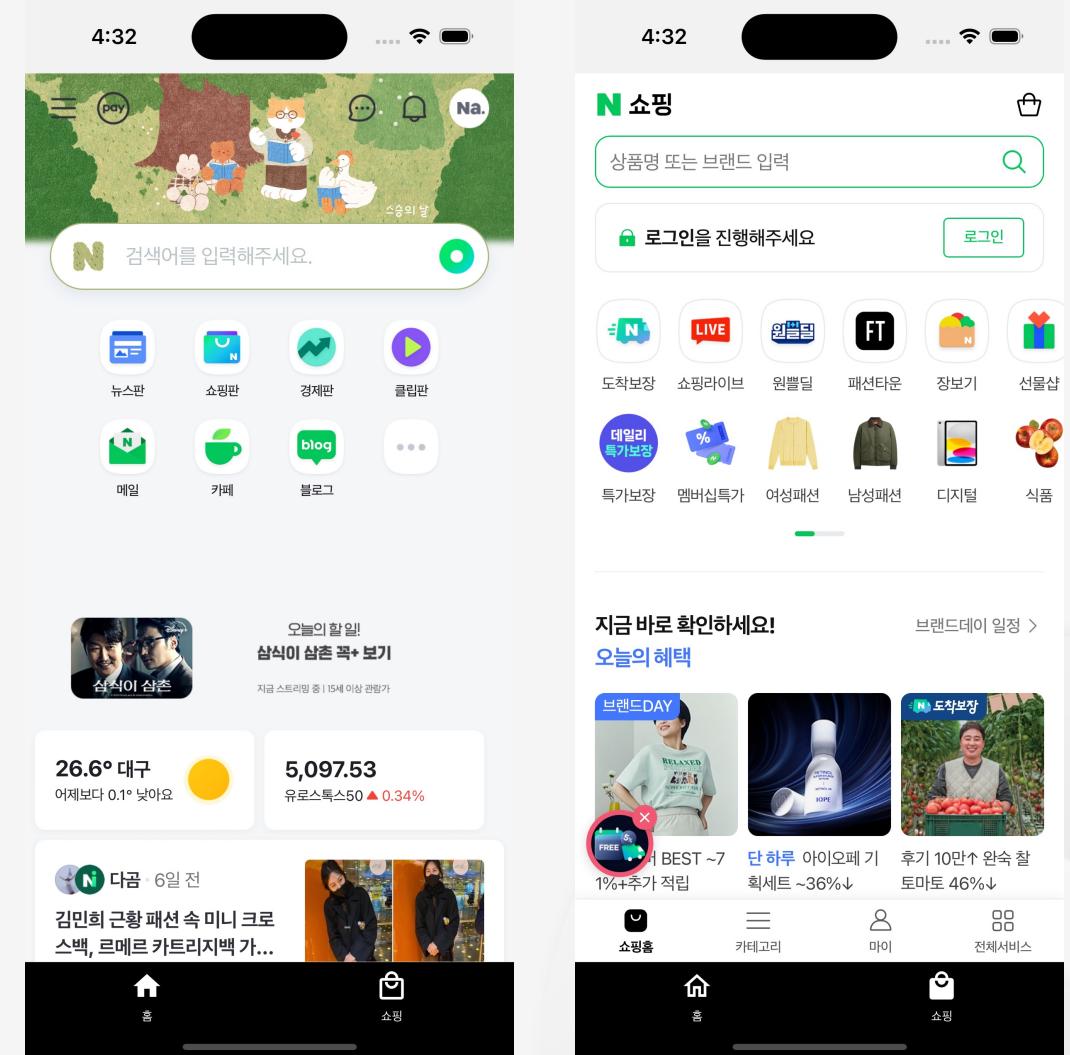


프로젝트1 - 네이버 앱 클론 코딩

학습 리뷰 및 예상 면접 질문

홈 스크린 / 쇼핑 스크린

- 템 네비게이션 구현
 - react-navigation
 - expo-router
- 아이콘 라이브러리 사용하기
 - react-native-vector-icons
 - @expo/vector-icons
- 웹뷰에 웹 사이트 로드하기
- Pull To Refresh 구현 하기
 - ScrollView
 - RefreshControl
- 웹 사이트 리퀘스트 핸들링
 - onShouldStartLoadWithRequest



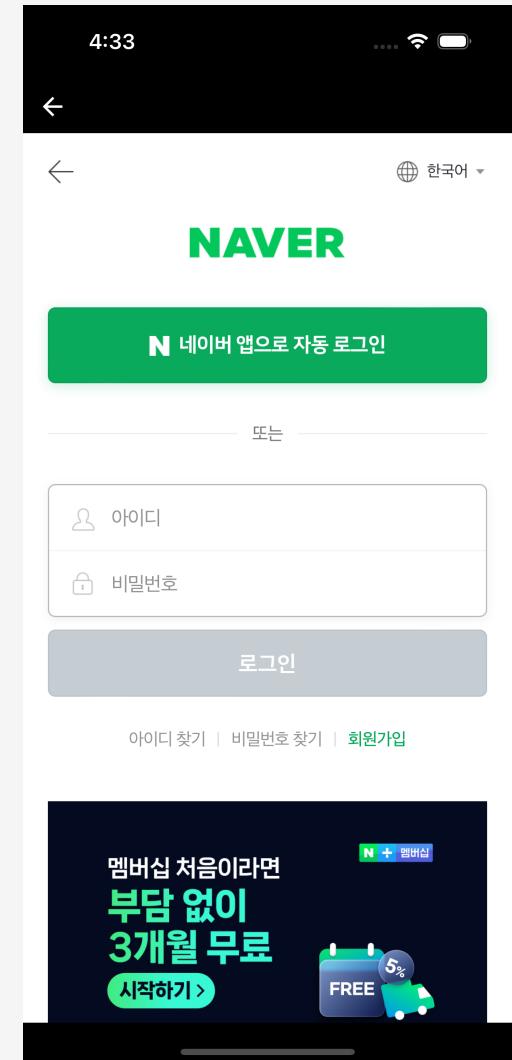
브라우저 스크린

- 웹 사이트 현재 주소 보여주기
 - onNavigationStateChange
 - 현재주소, 뒤로가기 가능, 앞으로가기 가능 상태 접근
- 웹 사이트 로딩 바 구현
 - onLoadProgress, onLoadEnd
- 브라우저 네비게이션 기능 구현 (앞으로, 뒤로, 새로고침)
 - Webview 인스턴스 접근
 - 웹뷰 새로고침 화면 수정: renderLoading
- iOS, Android 네이티브 공유 기능 구현
 - Share API



로그인 스크린

- 로그인 후 웹뷰 새로 고침 기능 구현
 - Context를 이용한 전역적인 상태 관리
- 쿠키 읽기 및 쓰기
 - @react-native-cookies/cookies
 - document.cookie



웹앱 최적화

- 핀치 줌/아웃 비활성화 하기
- 링크 롱 프레스 프리뷰 비활성화 하기
- 안드로이드 백버튼과 웹뷰 연결 하기
- 텍스트 롱 프레스 액션 비활성화 하기

앱 설정

- 앱 이름 변경하기
 - Info.plist, strings.xml
 - app.json 수정
- 앱 아이콘 변경하기
 - AppIcon 이미지 셋 변경, ic_launcher.png 변경
 - assets/icon.png 파일 수정

질문1. React Native를 선택한 이유는 무엇인가요?

- 여러 플랫폼에서 네이티브 애플리케이션을 개발 할 수 있는 효율성과 일관성
- React와 같은 단일 코드베이스를 사용해서 개발 시간과 유지보수 비용 절감 가능
- JavaScript를 사용하여 기존 웹 개발 경험을 활용 할 수 있음
- 높은 효율로 개발 리소스를 이용할 수 있음

질문2. 웹앱으로 만든 이유와 장점은 무엇인가요?

- 기존 웹 애플리케이션을 빠르게 모바일 앱으로 변환할 수 있음
- 기존 웹 개발자가 적은 학습으로 앱 개발을 진행할 수 있음
- WebView를 사용함으로 유지보수가 용이
- 모바일 네이티브 기능을 이용할 수 있음

질문3. Expo와 React Native CLI로 만든 프로젝트의 주요 차이점은 무엇인가요?

- **Expo**
 - 장점: 간편한 설정과 빠른 시작, 내장된 모듈 제공, 빌드 및 배포의 편리함
 - 단점: 네이티브 모듈 제한, 최적화의 한계
- **React Native CLI**
 - 장점: 네이티브 코드 접근 가능, 유연성 및 최적화 가능
 - 단점: 초기 설정 복잡성, 개발 속도 저하 가능성

질문4. Expo에서 네이티브 기능을 사용하는 라이브러리를 사용할 수 없을 때, 어떻게 해결하셨나요?

- Expo 프로젝트를 eject해서 사용할 수 있지만, 그렇게 되면 Expo의 장점을 잃게 됨
- 쿠키를 읽어오는 경우, 네이티브 API 대신 WebView를 통해 document 객체에 접근하여 쿠키를 읽고 사용할 수 있음

질문5. 웹앱을 최적화하기 위해 어떤 작업을 진행하셨나요?

- **핀치 줌/아웃 비활성화:** 모바일 앱에서는 일반적으로 핀치 줌/아웃 기능을 비활성화하는 것이 더 사용자 친화적. 이를 위해 meta 태그를 조작하여 핀치 줌/아웃을 비활성화
- **iOS 프리뷰 기능 비활성화:** iOS에서 링크를 길게 눌렀을 때 나타나는 프리뷰 기능을 비활성화
- **텍스트 선택 비활성화:** 텍스트를 꾹 눌렀을 때 선택되는 기능을 막기. 안드로이드에서는 user-select 스타일을, iOS에서는 -webkit-user-select를 조절하여 구현
- **백버튼 네비게이션:** 백버튼을 누르면 이전 스크린이 아니라 이전 웹 페이지로 이동하도록 WebView의 네비게이션과 연결