COOKIE-APP

**Convergent Capstone Design 1
Final Progress Report**

Group 5 Park JongBeum & Baek SeungHeon, June 20th 2023

Motivation

Why Cookie

Instagram
Problems with existing SNS

- Photo Based

- Ads

- Show Off

- Time Spending

- No Good for mental health

- Main Purpose of SNS blurred

KakaoTalk
Problems with existing SNS

- Chat Based

- Too many useless features

- Used also for Public Purpose

Our Goal

Our Goal
C🍪🍪KIE-APP

- Back to the ESSENCE of SNS

- Real Connection between actual "Friends"

  - Less Online, More Offline

  - What we need for "More Offline"?

    - Sharing Schedules & Location

    - Of course, private informations should be under user's control

Challenges

Framework & Language

Frontend Framework
Challenges in Choosing Framework & Language

- Flutter (Dart)

- React Native (JavaScript / TypeScript)

- PWA (JavaScript / TypeScript)

- iOS Native (Swift / ObjC)

- Android Native (Java / Kotlin)

React Native
Pros & Cons

- Easy for React Developers

- Needs Lots of External Libraries

- Harder (than flutter) to develop Cross-Platform App

  - Platform-Dependent Logics

PWA (Progressive Web App)
Pros & Cons

- Easy to develop since it's just Web Programming

- Less Support for Hardware Control

- Still getting updates

  - Supported actions are still dependent on platform

Why Flutter?
Challenges in Choosing Framework & Language

- Easy to Design UI even for beginners

  - Similar to CSS

  - Pre-Defined design: material, cupertino etc.

- Less Platform-Dependent than React-Native

- Single Codebase for all Platforms

- Reduced Development Time

- Native-like Performance

Backend Framework
Challenges in Choosing Framework & Language

- nodeJS based (JavaScript / TypeScript)

  - Express

  - Koa

  - NestJS

- Flask (Python)

- Django (Python)

- Spring (Java)

Why Express?
Challenges in Choosing Framework & Language

- Large Community

- Lots of Libraries (npm community)

- Express Middlewares

- Easy to use & learn

- Faster Development

Building Test Server

Backend Specifications
Creating Backend Server

- Ubuntu 22.04

- NGINX

- HTTPS (Let's Encrypt)

- mongoDB cloud

  - cookiecluster.vz4toss.mongodb.net

- AWS Route53

  - test.parkjb.com

Backend Reliability

TypeScript Migration
Backend Reliability

- Trial & Error

  - JavaScript → TypeScript Migration

- JavaScript will not warn you about Type Errors

  - Which leads to more critical Errors

JOI
Validate Requested Body / Query Type

- JOI: Validation Library

- Validates if request is requested in proper type before running business logics

- Which leads to stability

Backend Security

HTTP
Backend Security - Layers in HTTP

Application Layer

| HTTP | FTP | NNTP | XMPP |

Transport Layer

TCP/IP
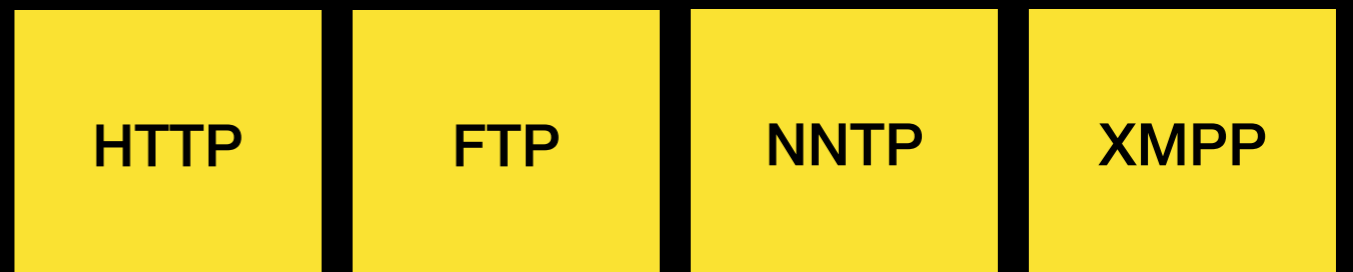
HTTPS
Backend Security - Layers in HTTPS

Application Layer

| HTTP | FTP | NNTP | XMPP |

SSL / TLS → Handshake Layer

| HandShake | HandShake | HandShake |

SSL / TLS → Record Layer

| Record |

Transport Layer

| TCP/IP |

Results
Test from ssllabs.com

JWT
Backend Security - JSON Web Token

- Authorization Token

- Header

  - Authorization: Bearer eyJhbGciOiJIUzI1NiI
sInR5cCI6IkpXVCJ9.eyJ1c2VyaWQiOiJ0ZXN0aWQxIi
widXNlcm5hbWUiOiJ0ZXN0MSIsImlhdCI6MTY4NzE4
Njk0OSwiZXhwIjoxNjg3NzkxNzQ5LCJpc3MiOiJ0ZXN0
LnBhcmtqYi5jb20iLCJzdWIiOiJ1c2VySW5mbyJ9.oia7l
uSztiAgT4O5gHNbfUBHbkxnZza5lojXIqvzN2k

UI Design

Can any device have the same ratio?
Applying Responsive Widget

- Automatically resize

  - Flexible widget

  - Expanded widget

- Indicates the screen size

  - Media Query
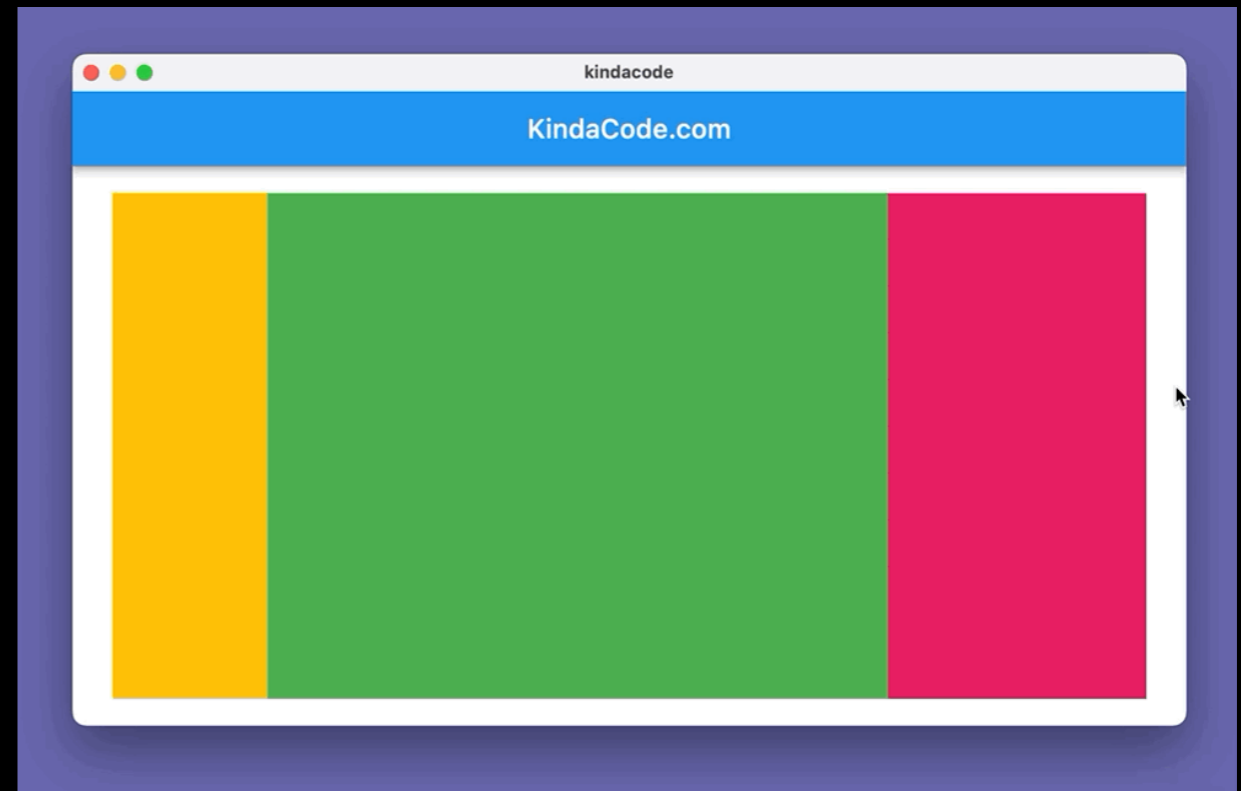
How should we manage the Theme?
Fonts, Colors, Border, Shapes...

- ThemeData()

  - Defines the configuration of the overall visual Theme within the app

  - No need to set each element one by one

  - Default theme

  - Dark mode theme

How should we manage the Theme?
Fonts, Colors, Border, Shapes...

```
1   ThemeData defaultThemeData = ThemeData(
2     brightness: Brightness.light,
3     appBarTheme: const AppBarTheme(
4       backgroundColor: Colors.orangeAccent,
5       iconTheme: IconThemeData(color: Colors.white),
6       centerTitle: true,
7       titleSpacing: 50.0,
8     ),
```

Frontend Structure

Why Design Architecture is needed?
Importance of Design Architecture

- Improved Code Maintenance

- Increased Reusability

- Clarity of Code Structure

- Flexibility and Scalability

Before MVVM

- UI & Business Logics are combined

  - Hard to modify the logics

- Copy & Pasted for same components

  - Low Reusability

- Types were not defined clearly

MVVM
Applying Model-View-ViewModel Structure

Data Binding

Update

| View | ← Data Binding → | ViewModel | → Update → | Model |

View

ViewModel

Model

Event
(Command Pattern)

Notify
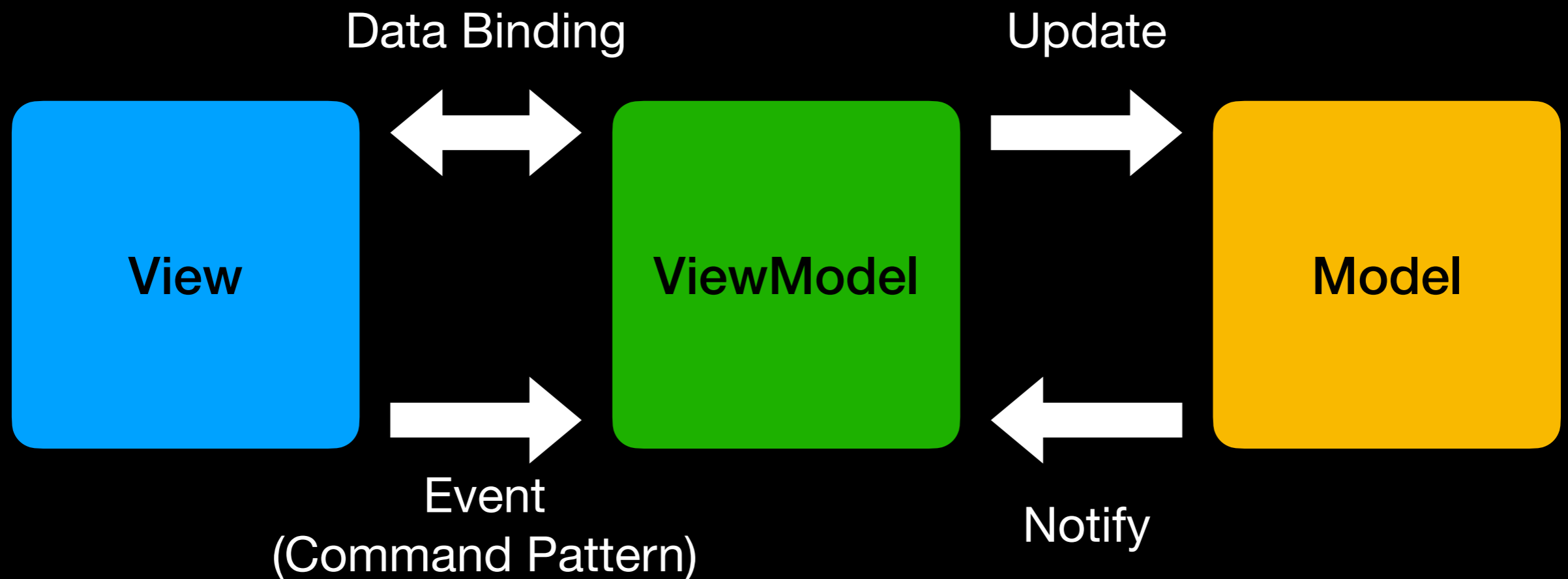
MVVM
Applying Model-View-ViewModel Structure

✓ Faster Screen Loading

✓ Improved Performance

✓ Reusable Components

✓ Testability

✗ Overkilling for simple UI operation

✗ Higher Learning Curve

MVVM
Applying Model-View-ViewModel Structure

```
∨ 📕 model
  ∨ 📂 account
      🔷 account_info.dart
      ⬡ account_info.g.dart
  ∨ 📒 chat
      🔷 message.dart
      ⬡ message.g.dart
      🔷 room.dart
      ⬡ room.g.dart
    🔷 jwt.dart
```

```
∨ 📙 view
  > 🟩 components
  ∨ 📙 pages
    > 📁 chatroom
    > 📁 club
    > 📁 friends
    > 📁 maps
    > 🟦 settings
      🔷 signin.dart
      🔷 signup.dart
    🔷 mainwidget.dart
```

```
∨ 📁 viewmodel
  ∨ 📒 chat
      🔷 message.viewmodel.dart
      🔷 room.viewmodel.dart
    🔷 account.viewmodel.dart
    🔷 auth.viewmodel.dart
    🔷 base.viewmodel.dart
    🔷 map.viewmodel.dart
    🔷 theme.viewmodel.dart
```
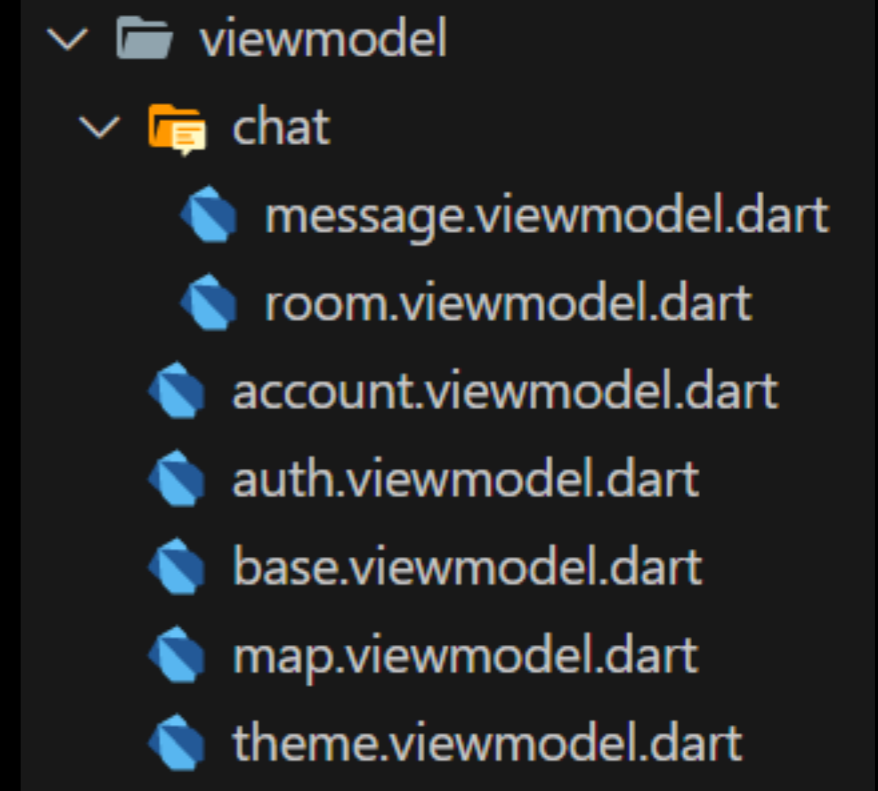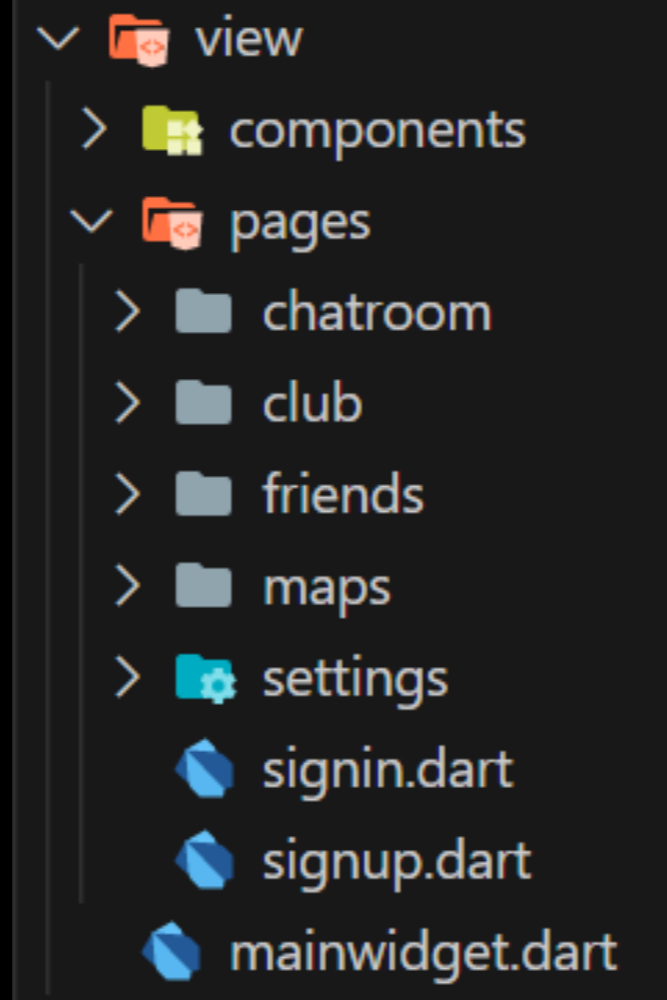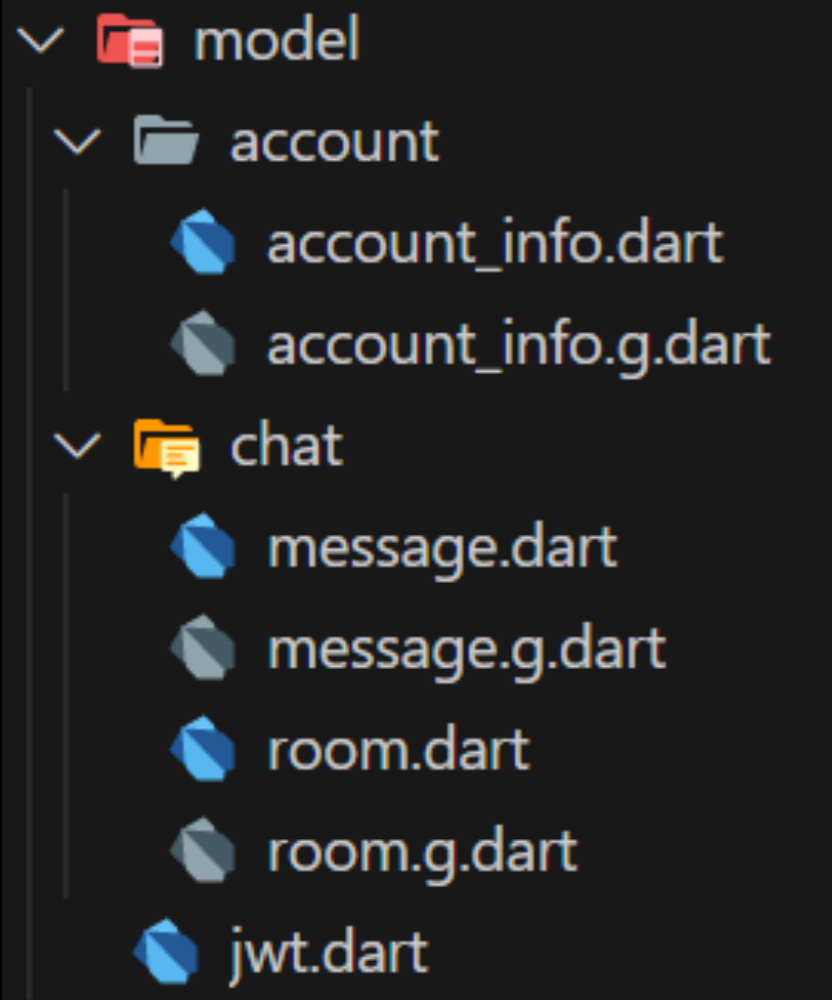
State management

Why we need State management?
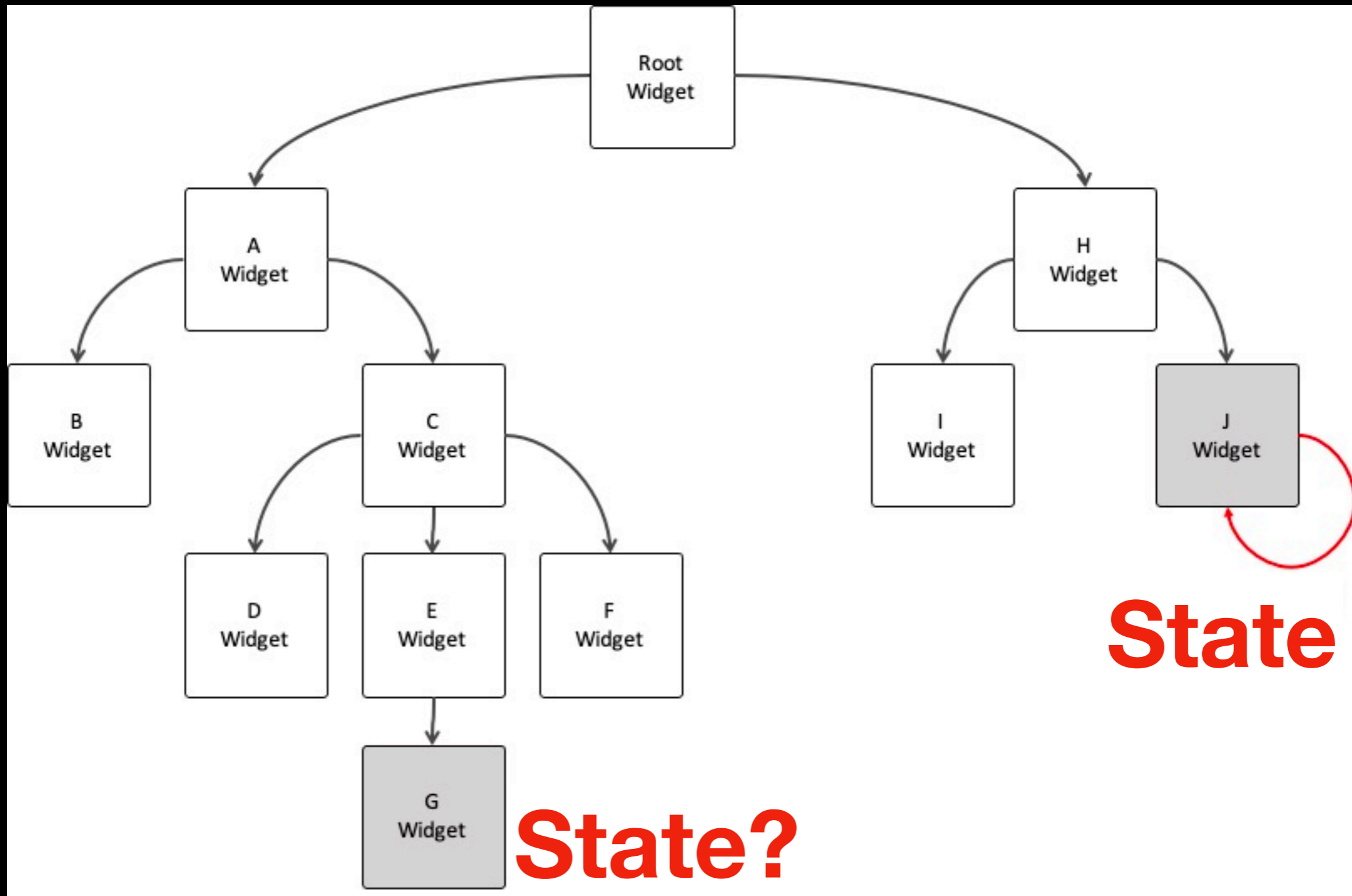
State management
Centralize your data & Improve performance
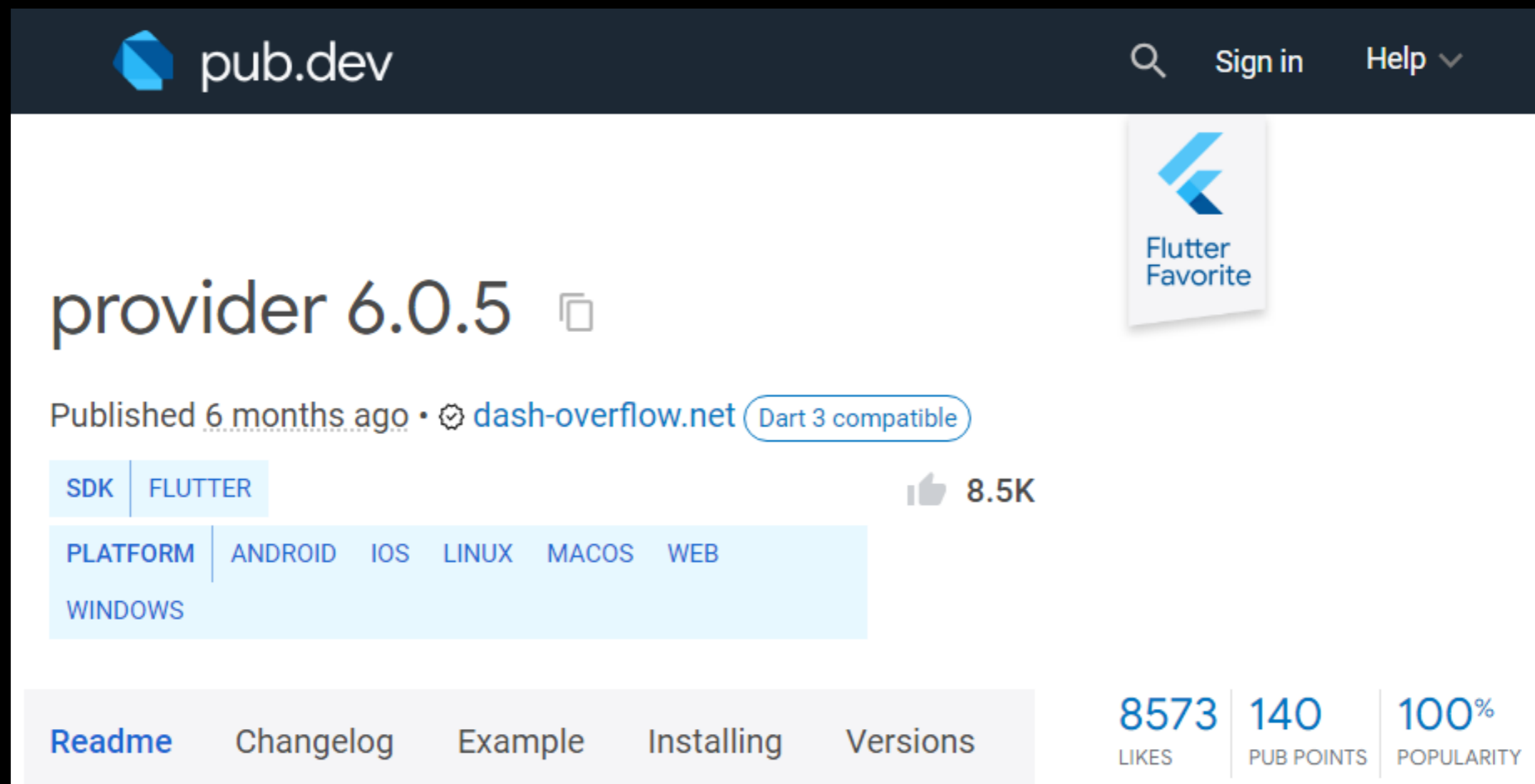
- Some packages

    - BLoC

    - Provider

    - Redux

    - GetX

    - RiverPod
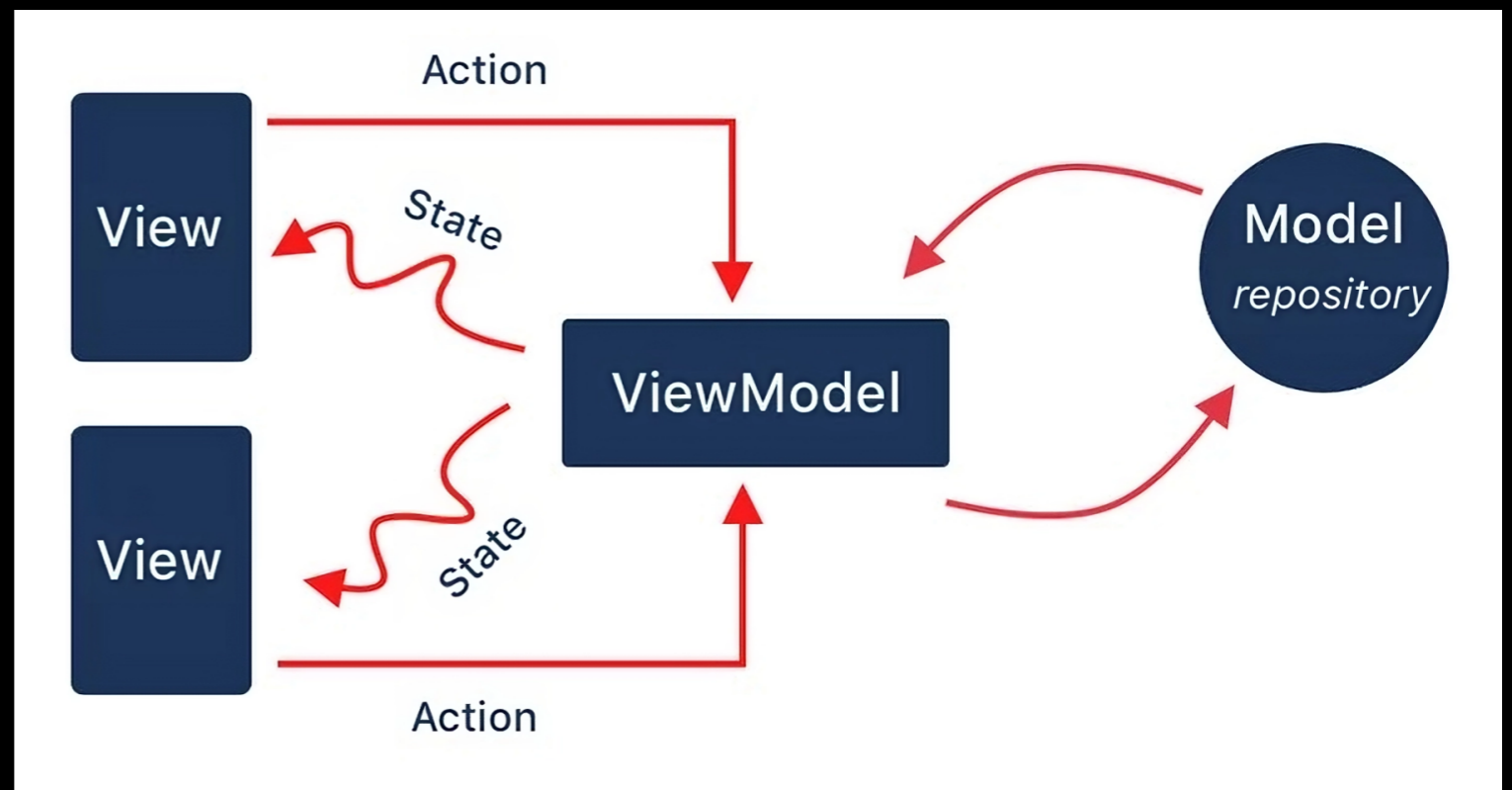
Why Provider?
Challenges in Choosing State management package

- Simple and readable & Easy to use

- Widely used package in the Flutter ecosystem

Outputs

github
cookie-devel (cookie_app & cookie_server)
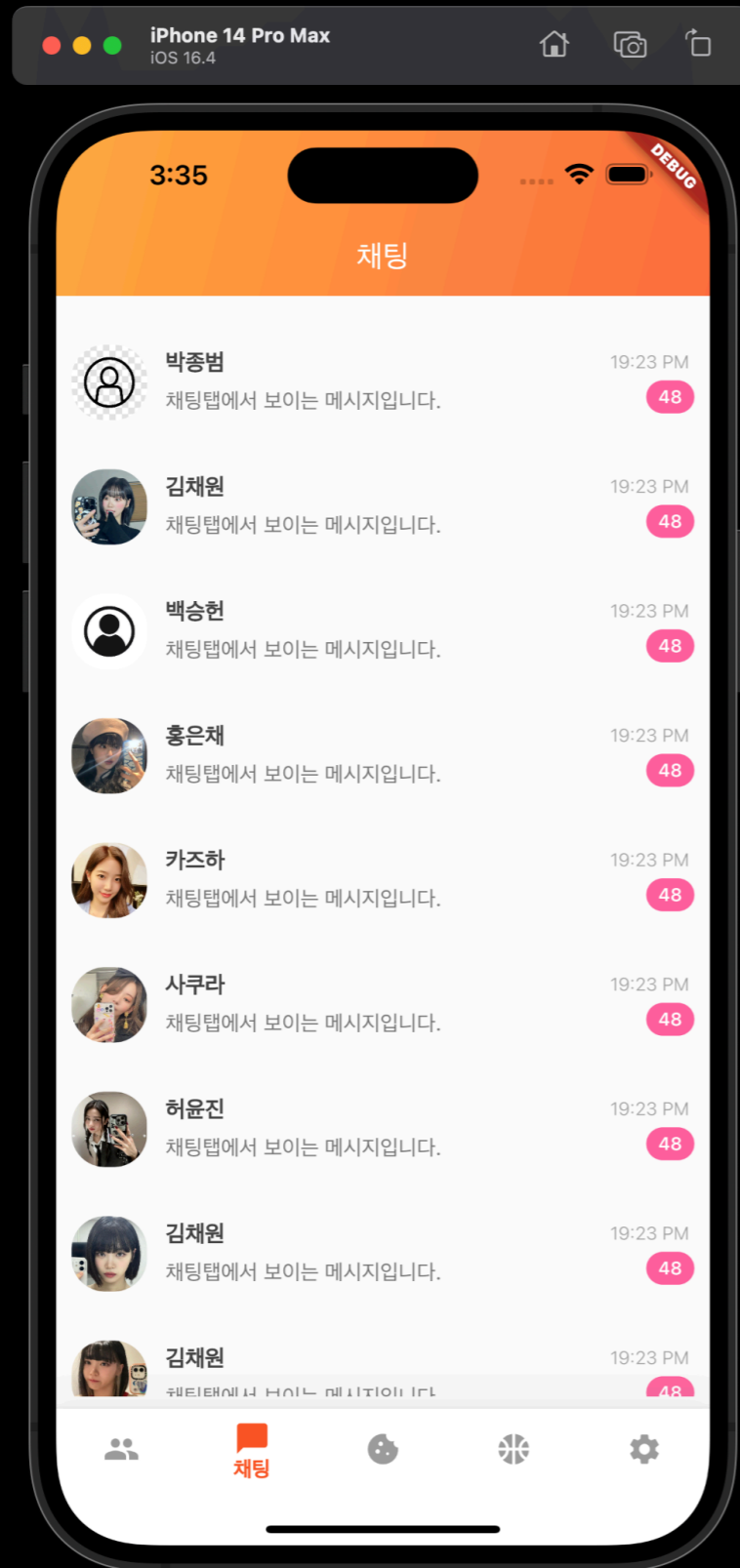

41

Sign In & Sign Up

Friends Tab
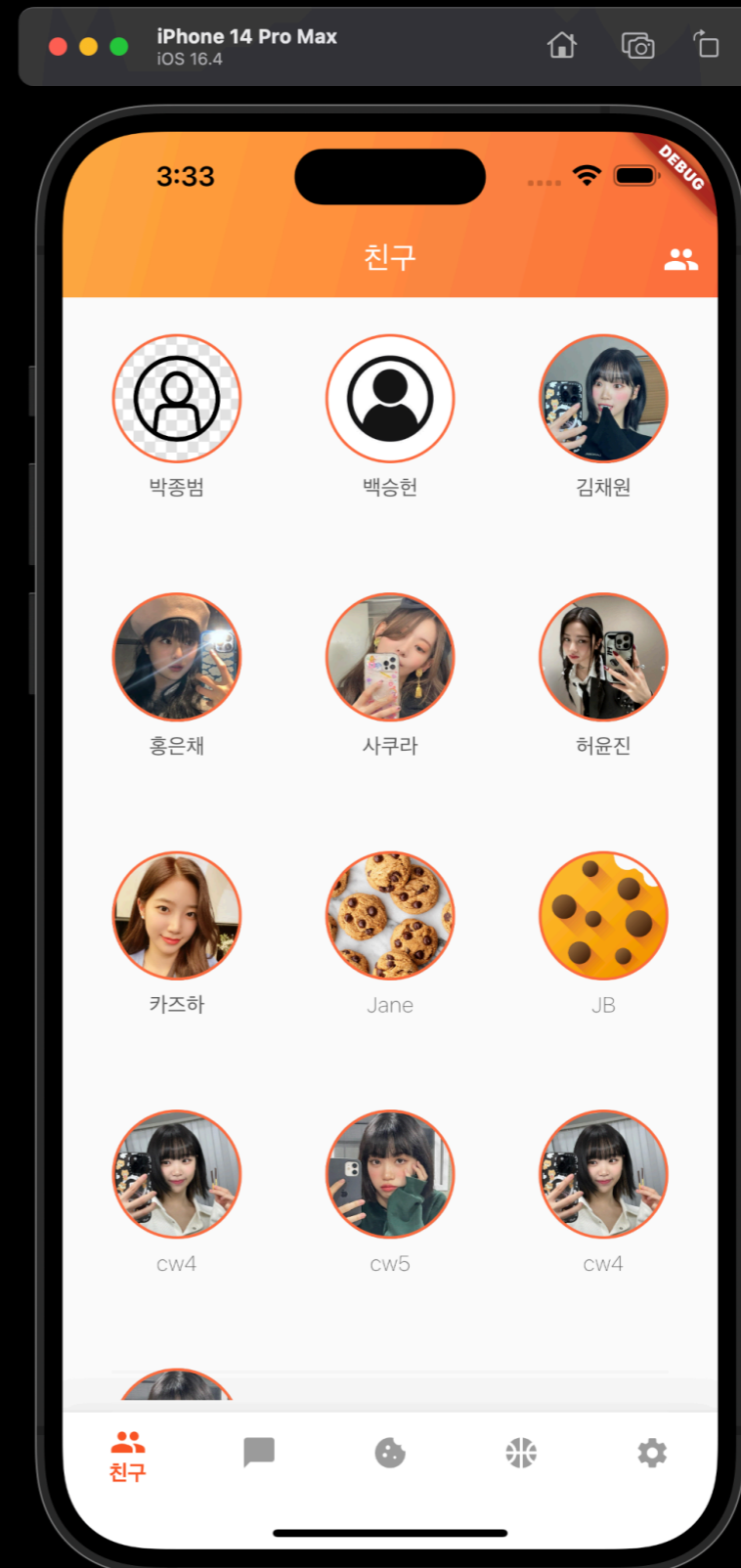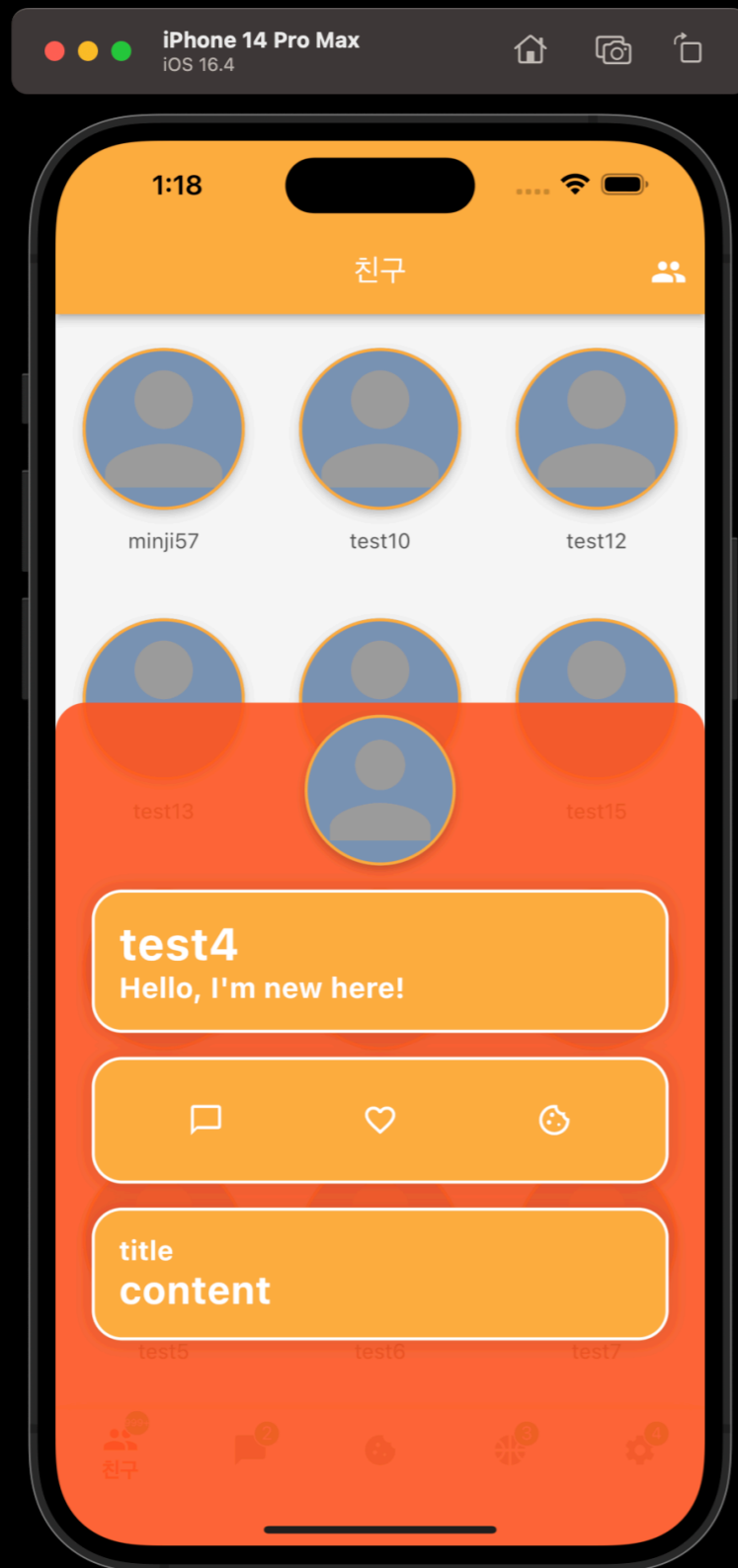
Chat Tab & Chat Room
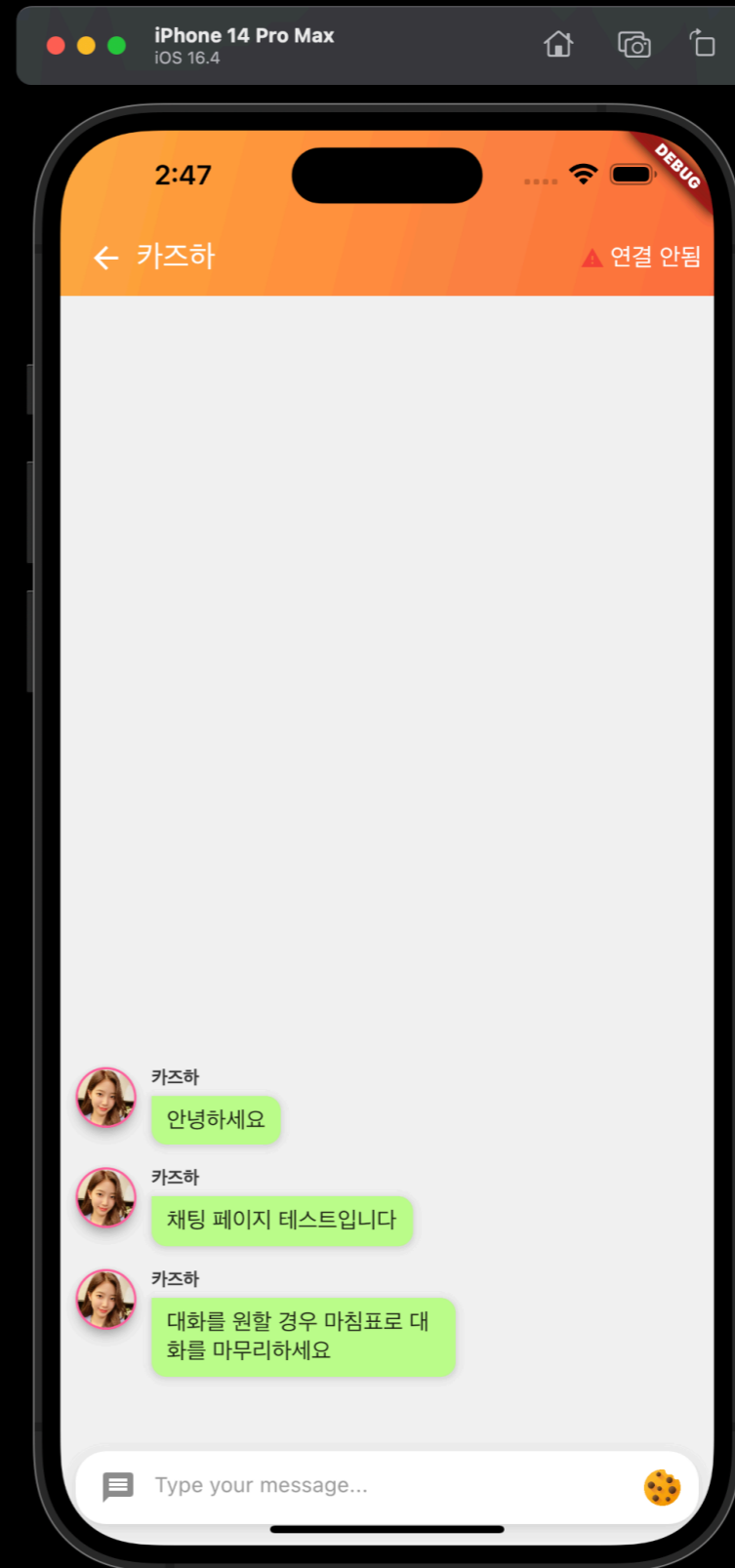
Maps Tab
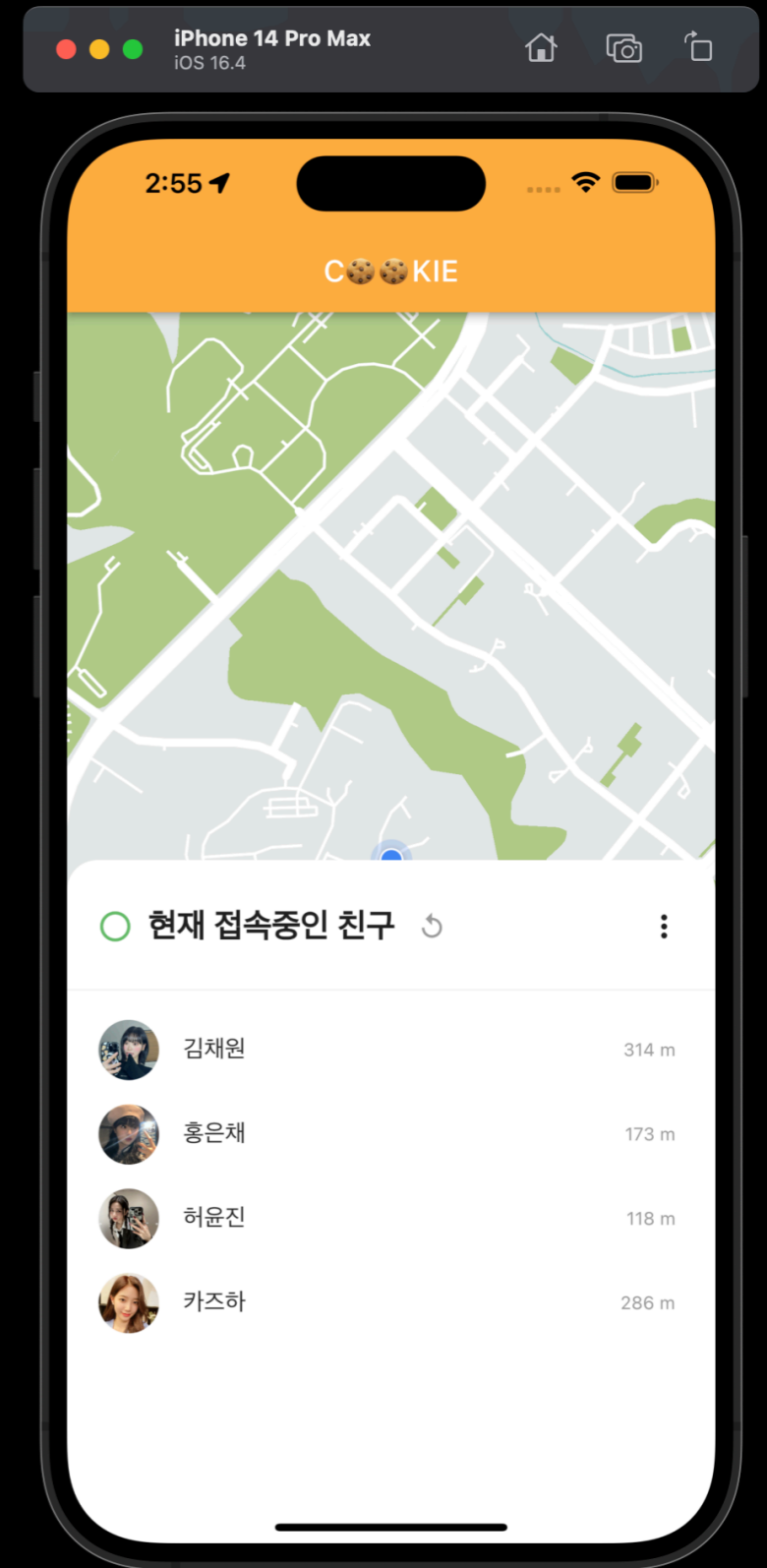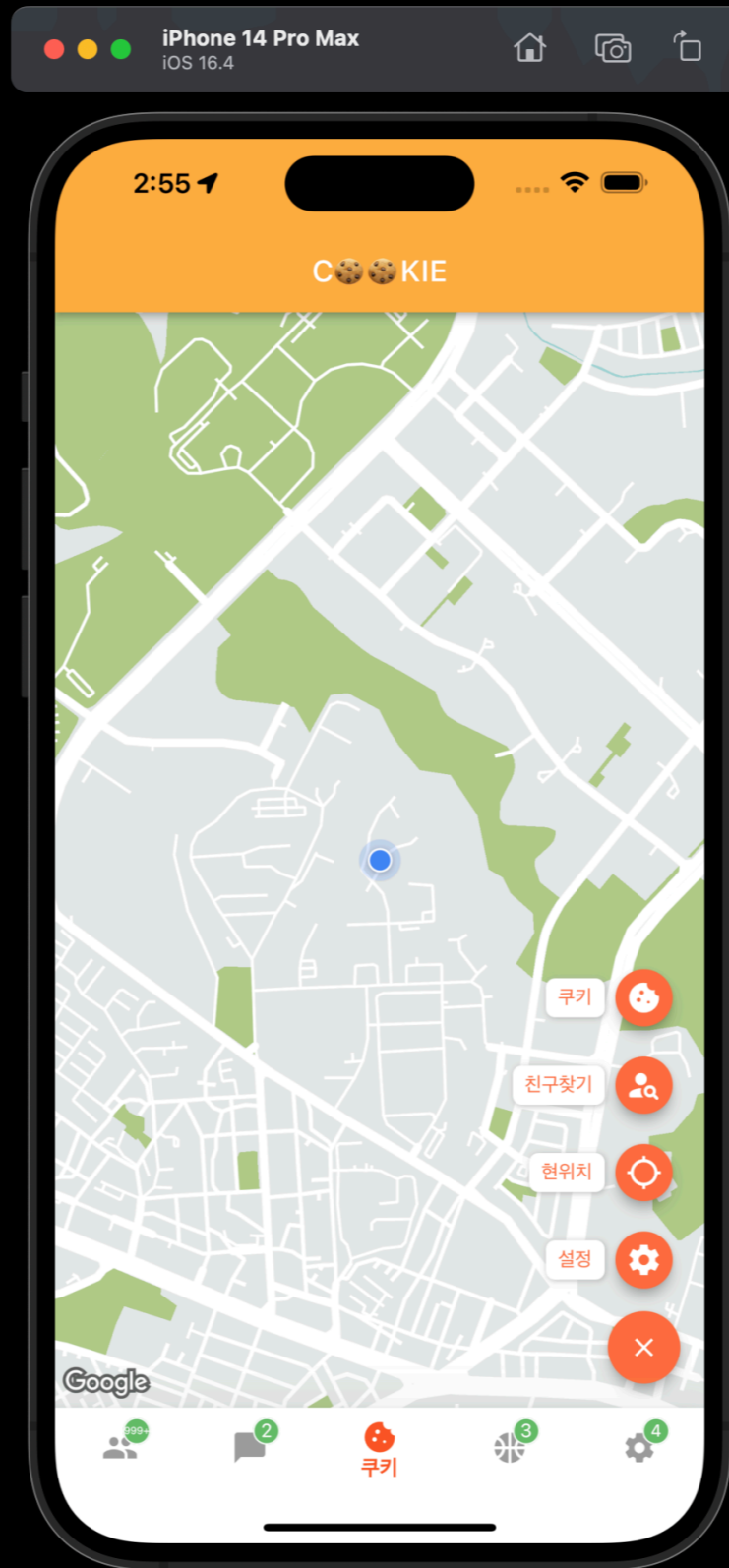
Settings Tab

Conclusion

What we've learned
1st Semester

- Flutter framework & Dart language

- Design Architecture Pattern (MVC, MVP, MVVM…)

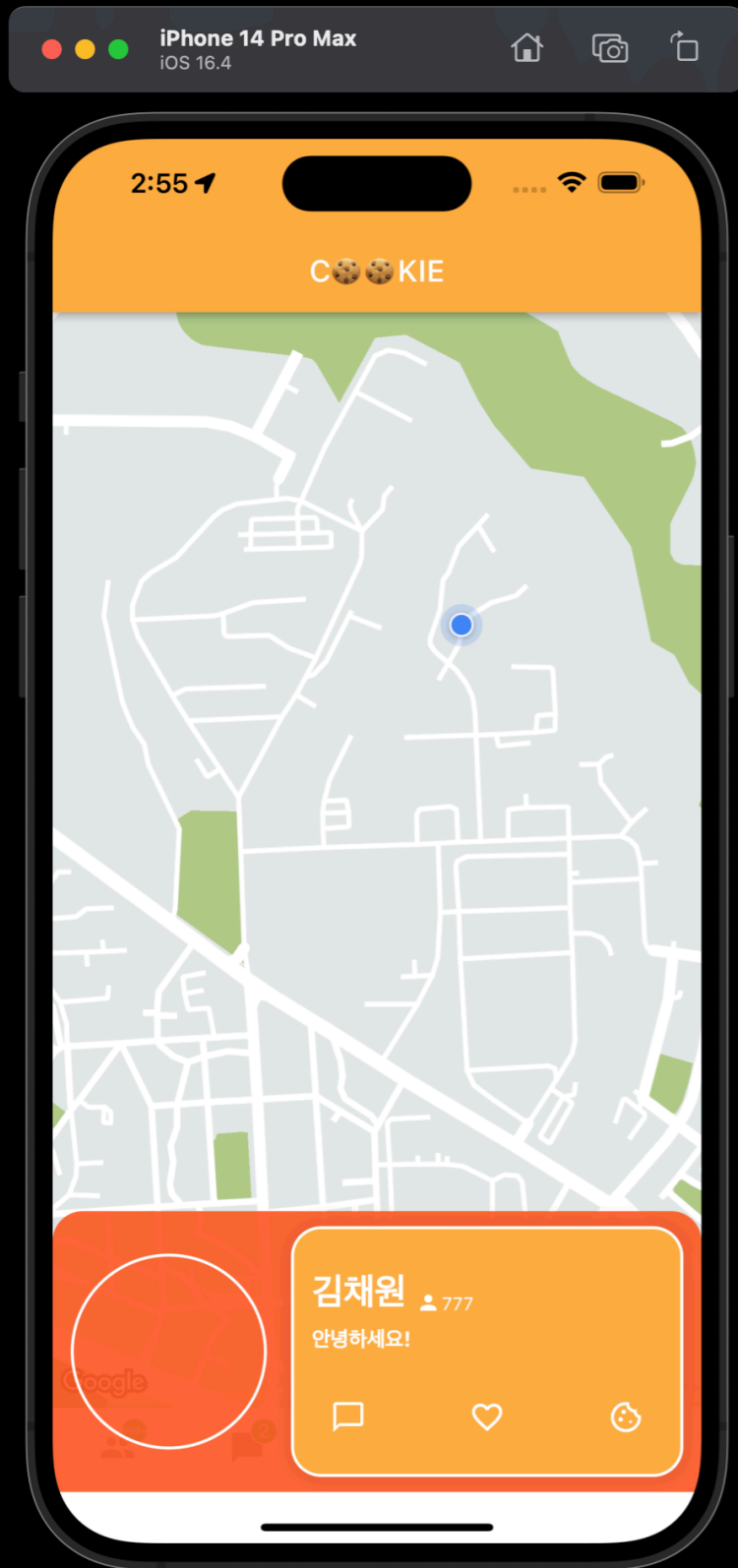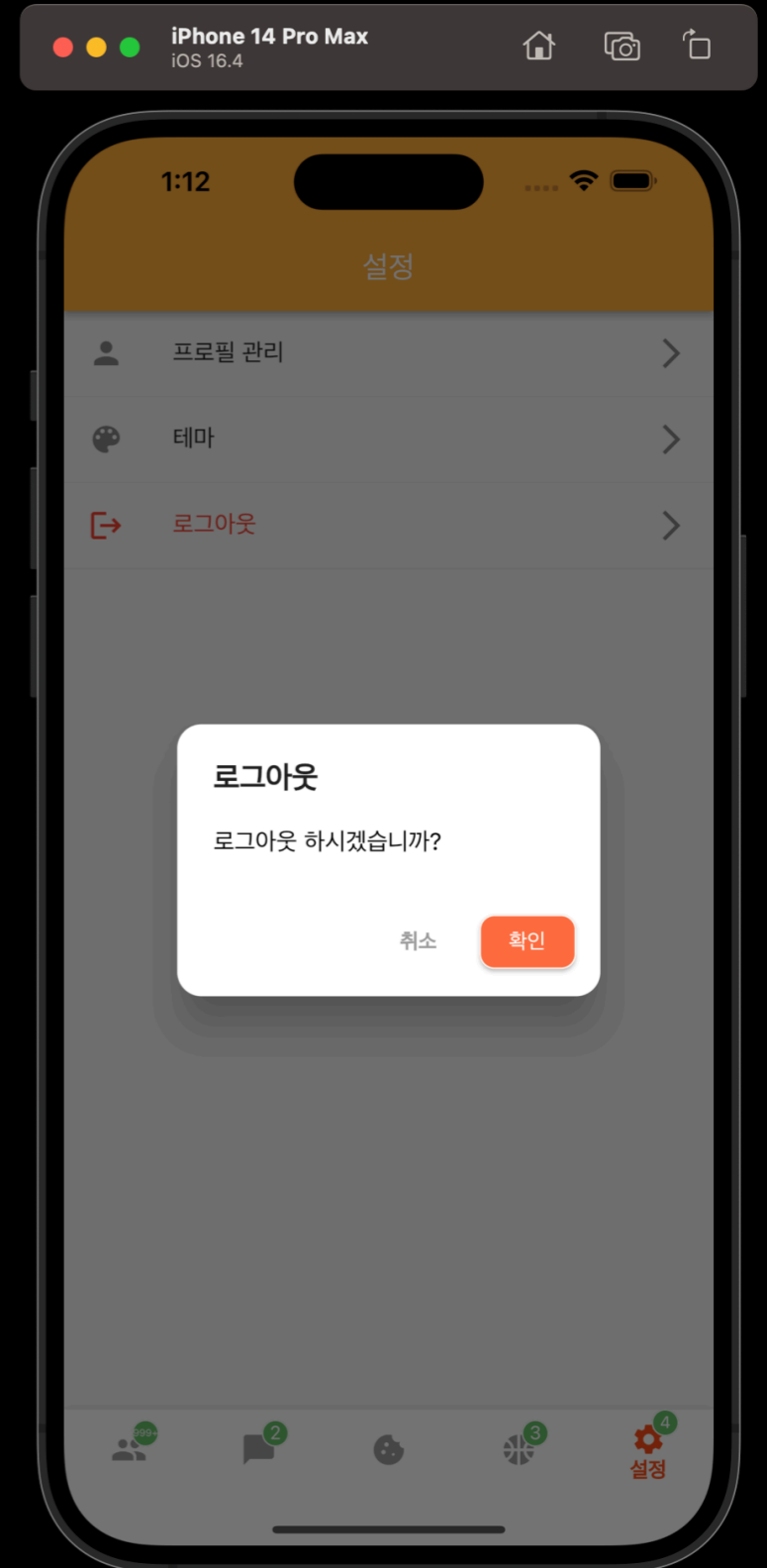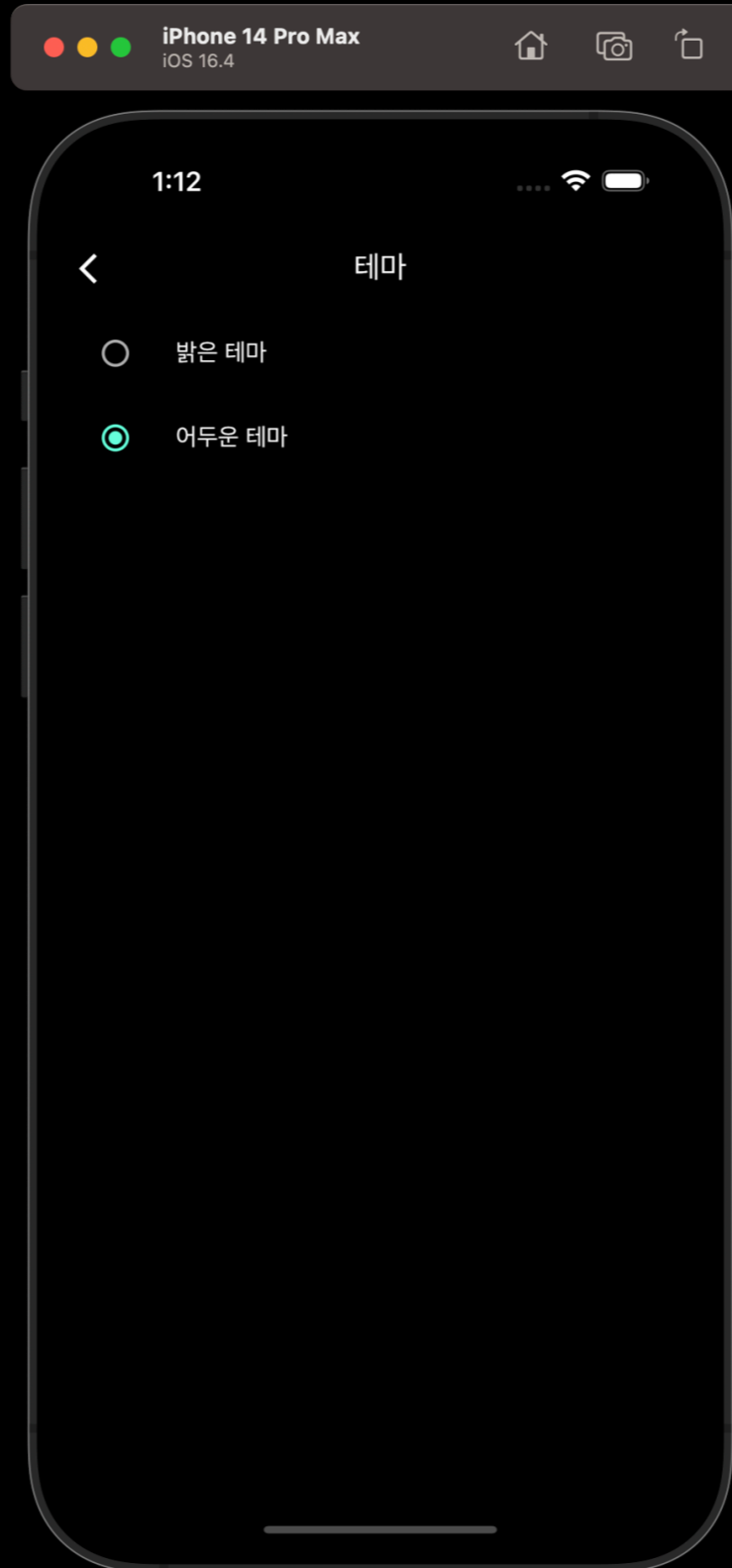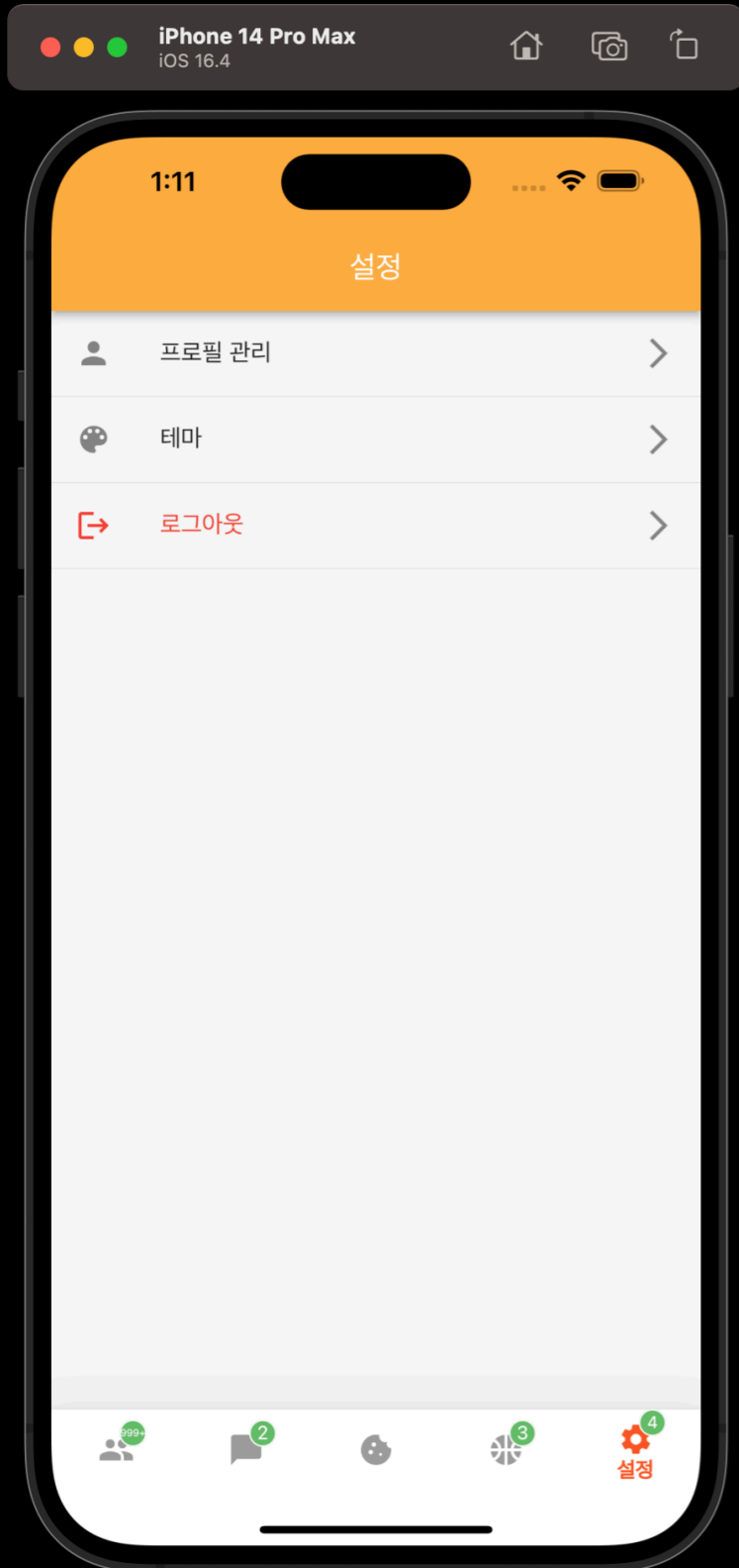- Git & Github Collaboration

- Backend API Implementations

- JWT token Authorization

- Importance of TYPE

What we've done
1st Semester

- Basic Structure

  - Took longer than our expectations

  - Since, we didn't know the importance of the Basic Structure

- UIs

  - Friends Tab, Chat Tab, Maps Tab, Setting Tab

  - More organized, better-looking changes needed

- Account Creation / Login

Benefit

What we expect

- Increased communication with real friends

- Visualize Friends' status via Maps tab, not just Chat

- Provide users with an easy-of-use, intuitive UI

- Share informations safely by encrypting user information

Future Plans

Future Plans
Summer Vacation & 2nd Semester

- Stabilized MVVM Structure

- Stabilized Chatting Service

- Alerts (FCM)

- Location Sharing

- (Schedule Sharing)

- Authority Management

- Stabilization before actual service

- PlayStore / AppStore Distribution

Thank You

Questions are welcome