ANDROID DEVELOPMENT

Class 4: State & Lists

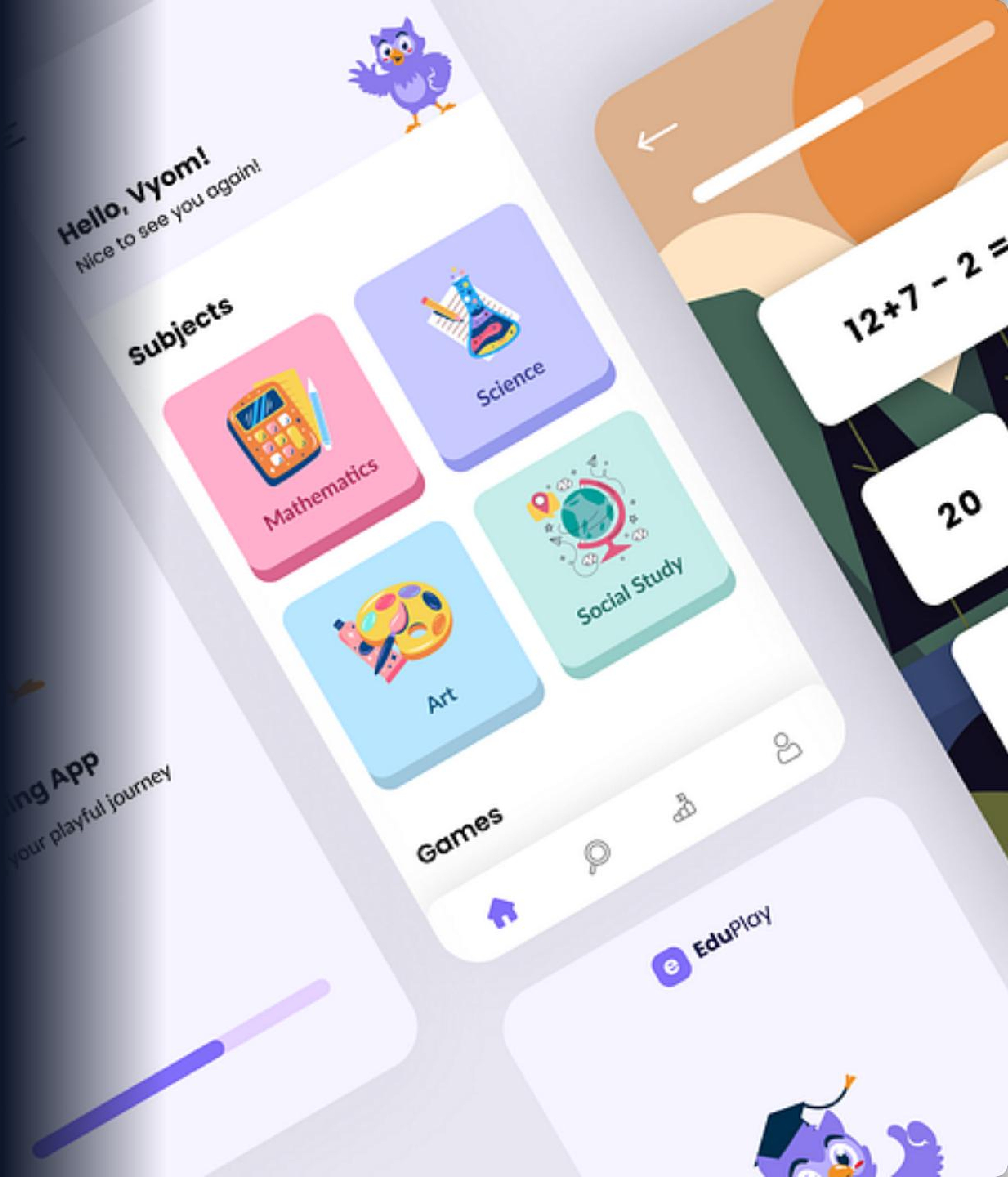
The Core Loop: Handling Interactivity and Data

Instructor: Mark Joseli

Today's Goal

Making apps interactive and handling dynamic lists.

- Understand Recomposition
- Manage State
- Build Scrollable Lists

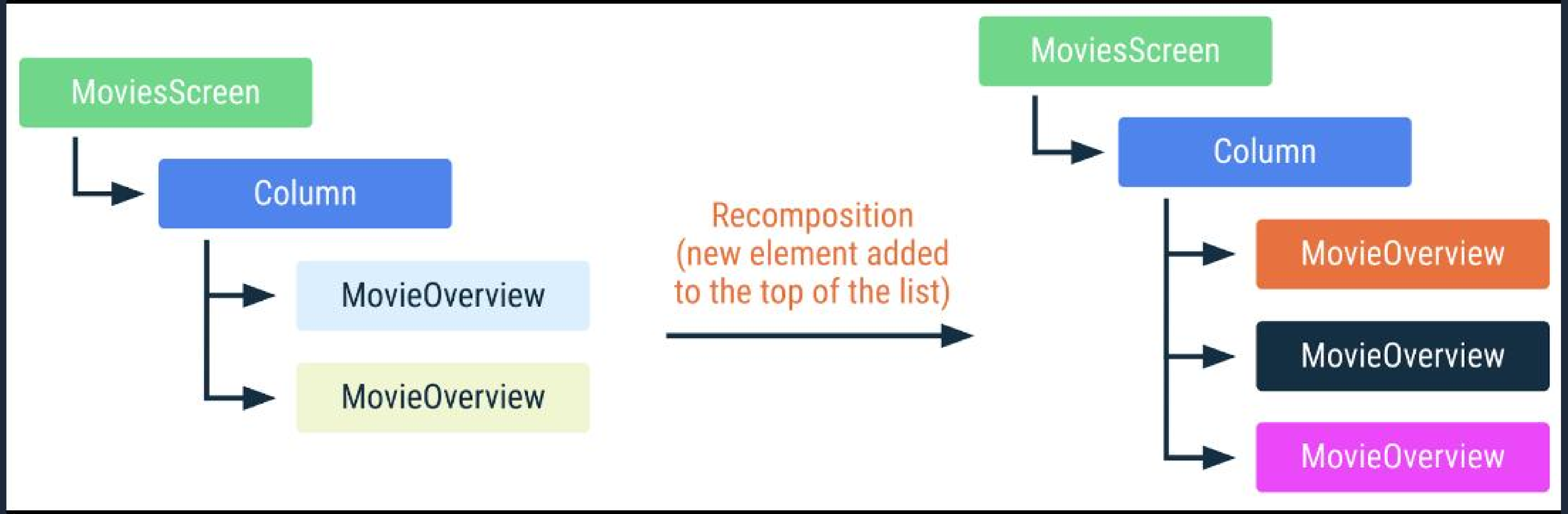


Theory: Recomposition

Compose functions can execute frequently and in any order.

> **Recomposition:** The process of re-running composable functions when their `State` changes.

> Compose is optimistic: it only updates the parts of the UI that actually changed.



State: `mutableStateOf`

To make Compose react to changes, you need **State**.

```
val count = mutableStateOf(0) // Later in UI...
Text(text = "Count: ${count.value}") Button(onClick
= { count.value++ })
```

Any Composable reading `count.value` will strictly "subscribe" to it and update automatically.

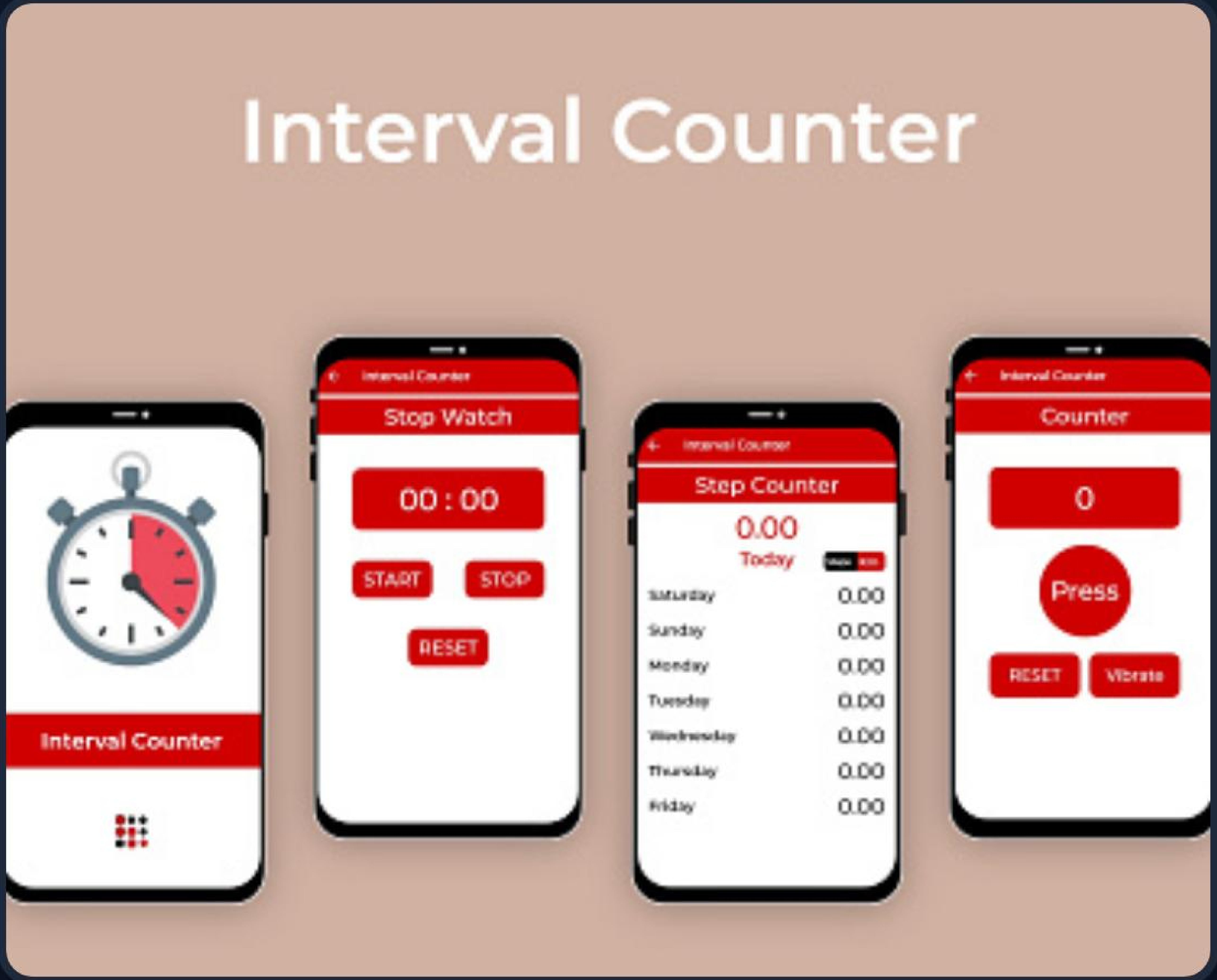
Observes changes automatically.

Memory: `remember`

Recomposition destroys and recreates local variables. We need to **remember** the value across updates.

```
var count by remember { mutableStateOf(0) }
Button(onClick = { count++ }) { Text("Clicks:
$count") }
```

Without `remember`, the count would reset to 0 every time the screen updates.



State Hoisting

Moving state up to the caller to make components reusable.

Stateful

A component that holds its own state. Hard to reuse or test.

The Pattern

Pass data **down** (parameters). Pass events **up** (lambdas).

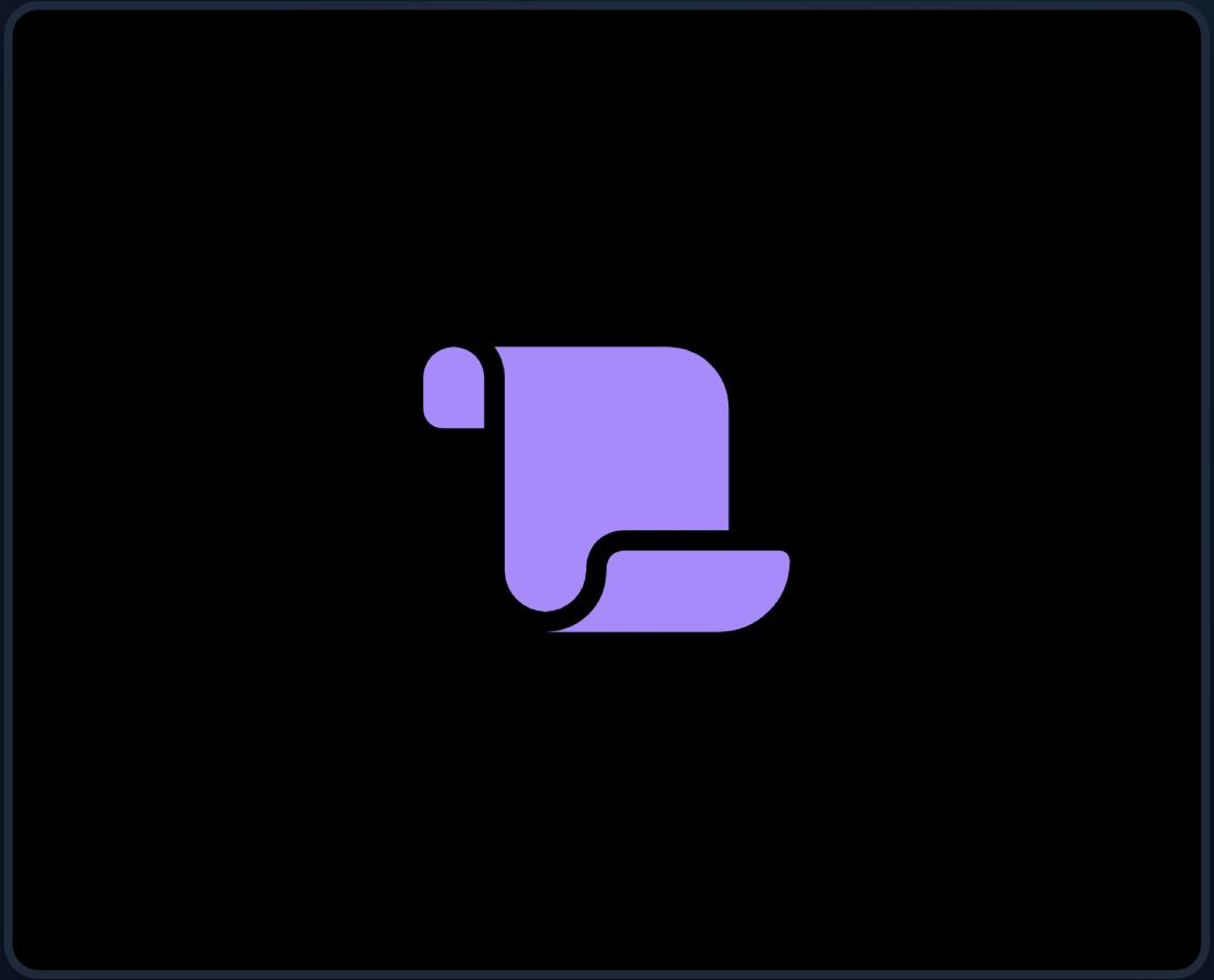
Stateless

A component that just displays what it's told. Pure and reusable.

Lists: `LazyColumn`

Why not just use `Column`?

- > `Column` renders **everything** at once. Bad for performance if you have 100+ items.
- > `LazyColumn` only renders what is currently visible on screen.
- > It is the modern replacement for `RecyclerView`.

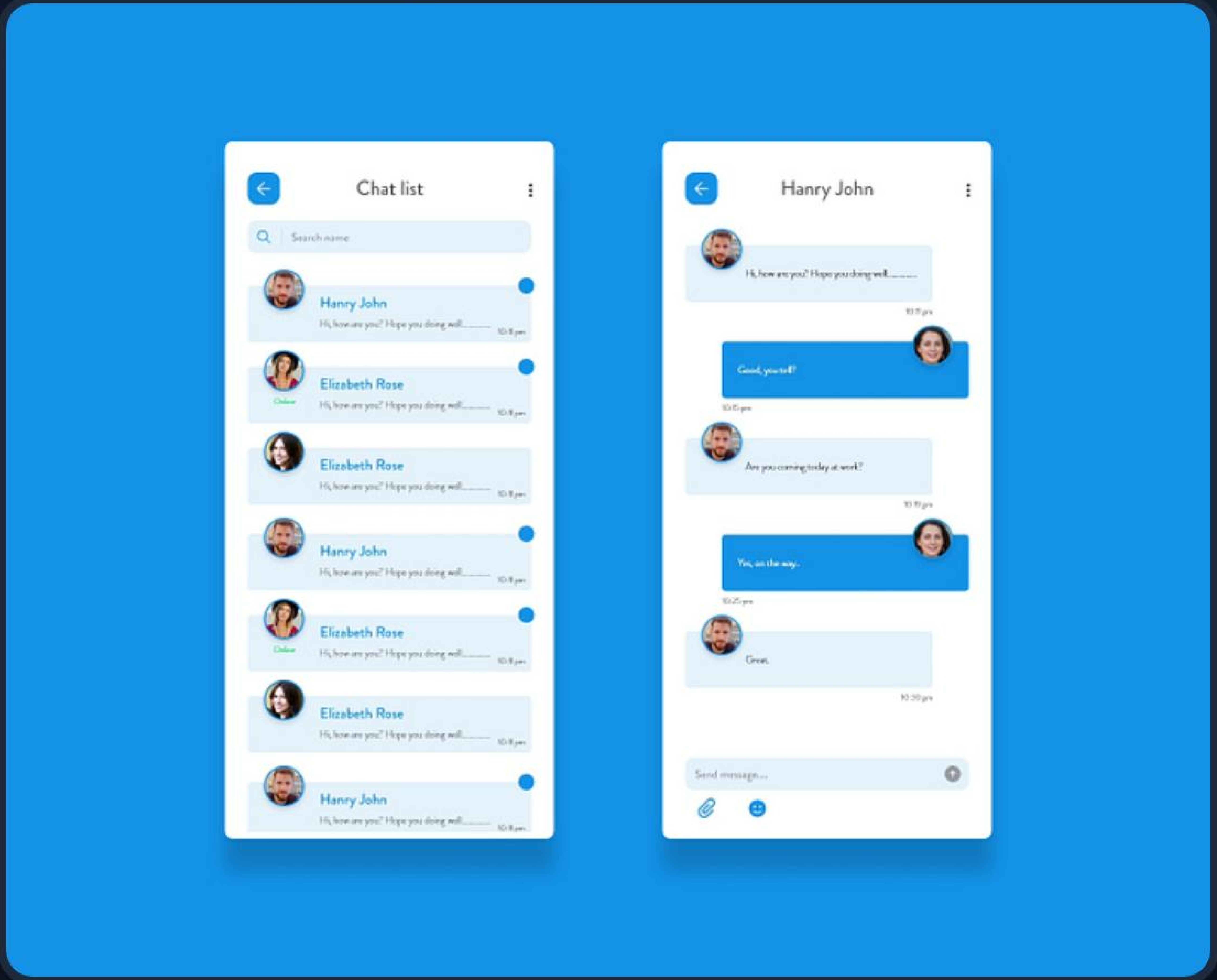


LazyColumn Code

Simple and concise syntax.

```
LazyColumn { items(messages) { message →
MessageRow(message) } }
```

We pass a list of data, and define how **one** single row should look. Compose handles the scrolling and recycling.



Hi, how are you? Hope you doing well.......

Good, you tell?

Are you coming today at work?

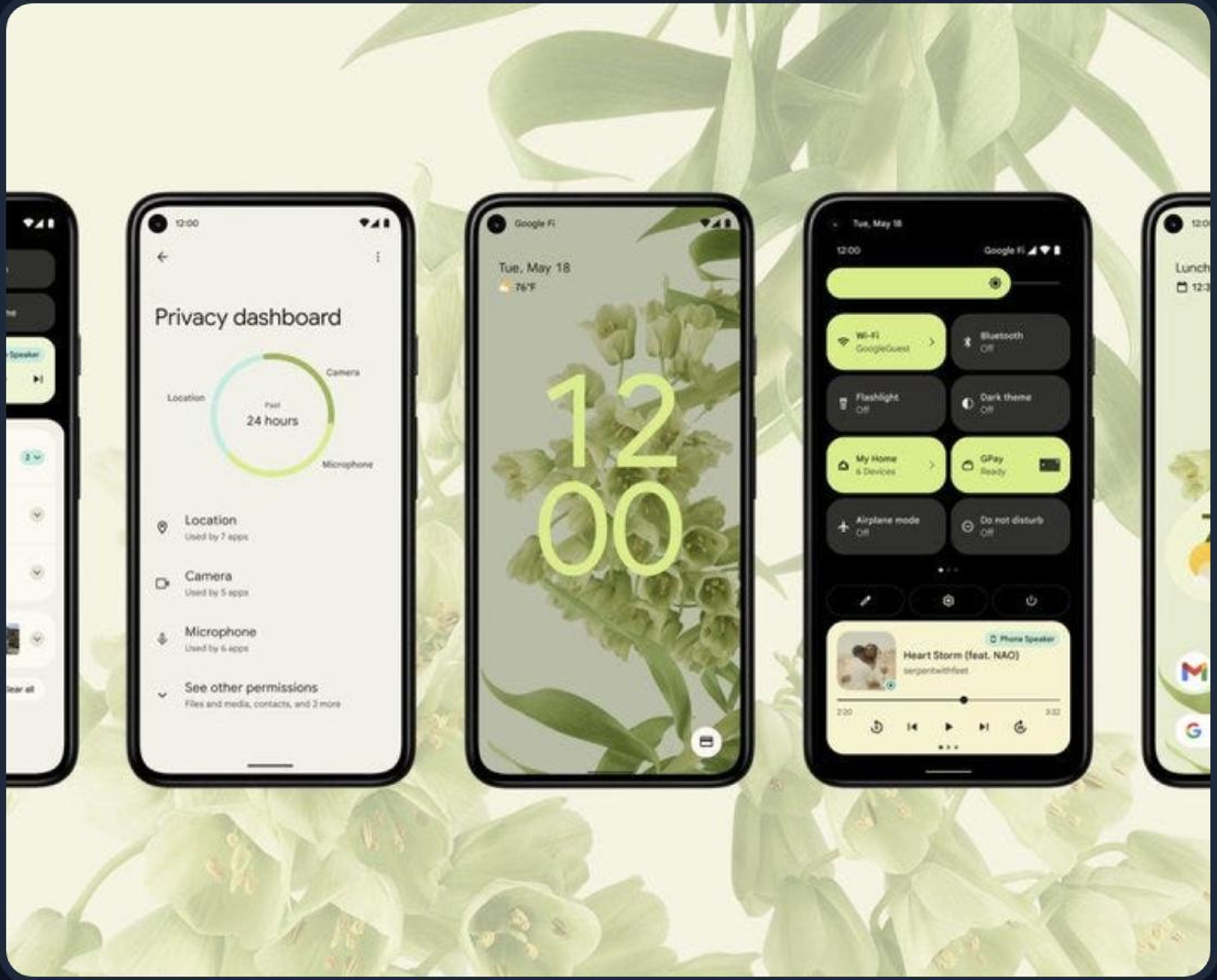
Yes, on the way.

Great.

Design: Material 3

Google's latest design system.

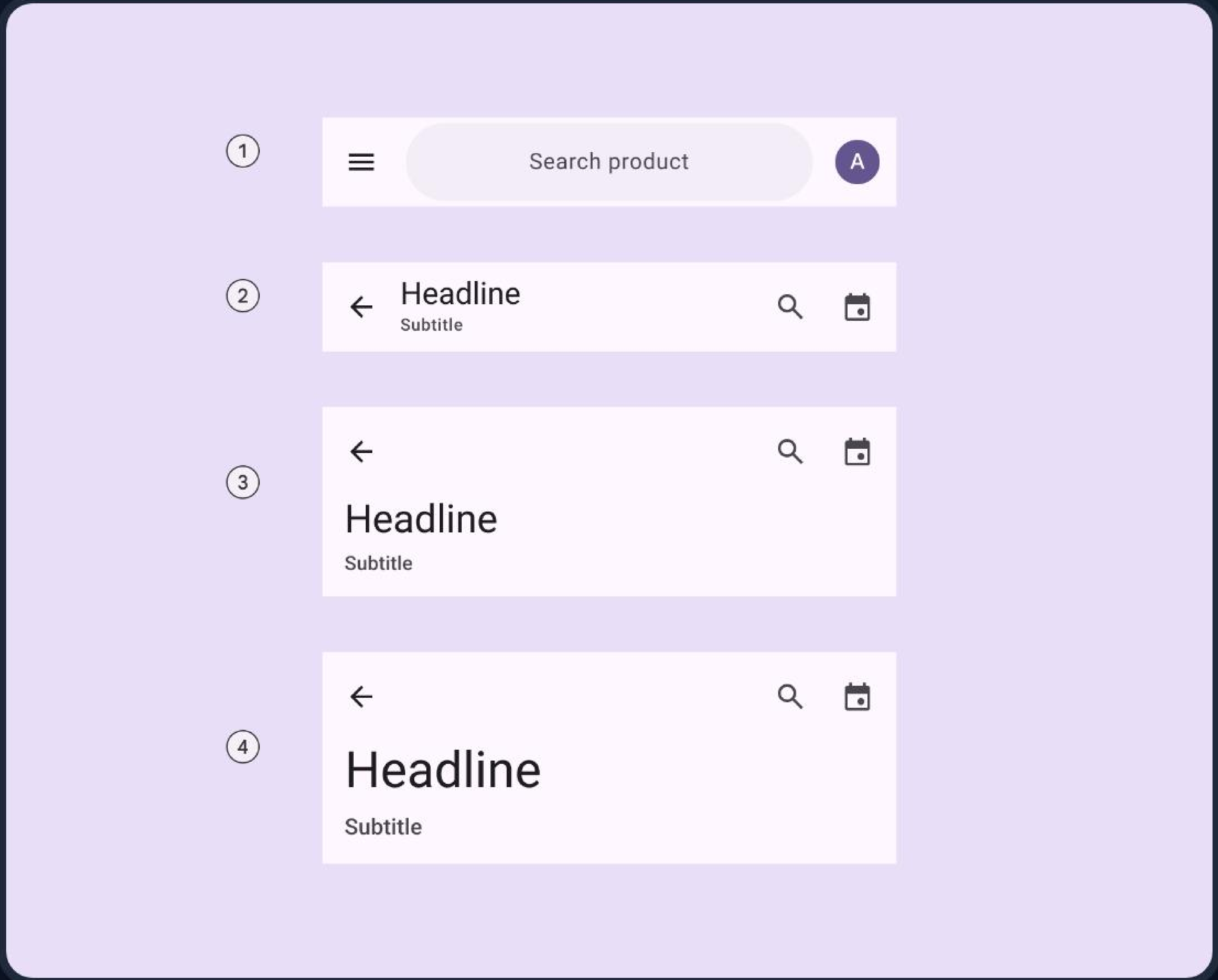
- **Dynamic Color:** UI adapts to the user's wallpaper.
- **Modern Components:** Updated Cards, Buttons, and Navigation bars.
- Implemented via the `MaterialTheme` composable.



Structure: `Scaffold`

Provides the standard structure for a screen (TopBar, FAB, BottomBar).

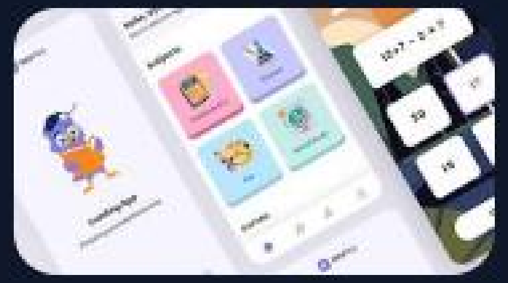
```
Scaffold( topBar = { TopAppBar(title = {
Text("Home") }) } ) { padding → // Screen content
goes here Box(Modifier.padding(padding)) }
```



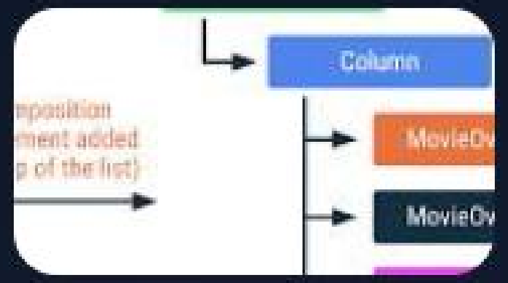
Questions?

Let's implement a dynamic list!

Image Sources


https://cdn.dribbble.com/userupload/6460022/file/original-9a54e848ba3bcf8a80f24b815130d338.png?resize=752x&vertical=center

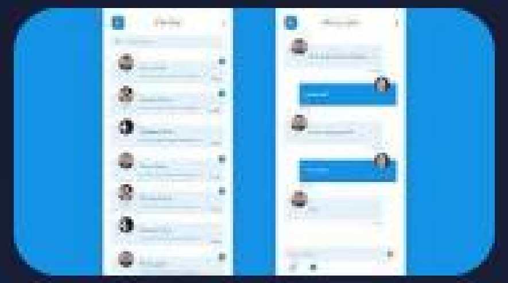
Source: dribbble.com


https://developer.android.com/static/develop/ui/compose/images/lifecycle-newelement-top-all-recompose.png

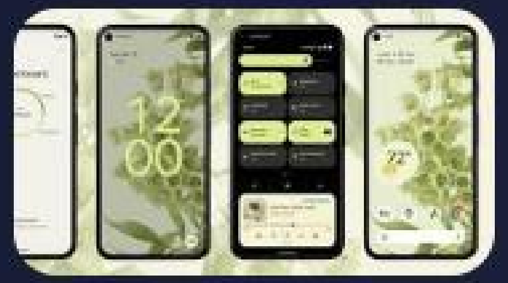
Source: developer.android.com

https://cdn.dribbble.com/userupload/32581025/file/original-3193faaac0052c28c989e5c59f61545e.jpg?resize=400x0

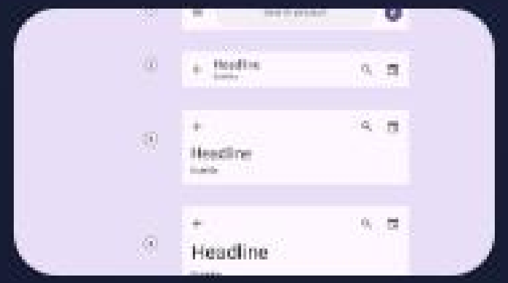
Source: dribbble.com


https://cdn.dribbble.com/userupload/5891622/file/original-2feb32f4b93befefb388bd449b9e206e.jpg?resize=752x&vertical=center

Source: dribbble.com


https://miro.medium.com/v2/resize:fit:1400/0*0TsUyrLKDw94DzVA

Source: medium.com


https://lh3.googleusercontent.com/R6NchWHyVf-clBxfBYoBw7-5iz6ymHQr5DUgW5NzI0ZRCArjukJnX0Mh5kQZVzEEauU64zRGfzAhXYDelTEGF_4mC-8UlJg44nnOaZFths4=s0

Source: m3.material.io