



Flutter Split-Screen Responsive Design

for Phone, Tablet, Desktop, and Web



Marco Napoli

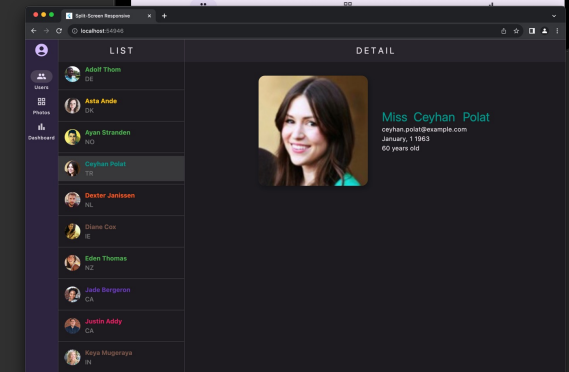
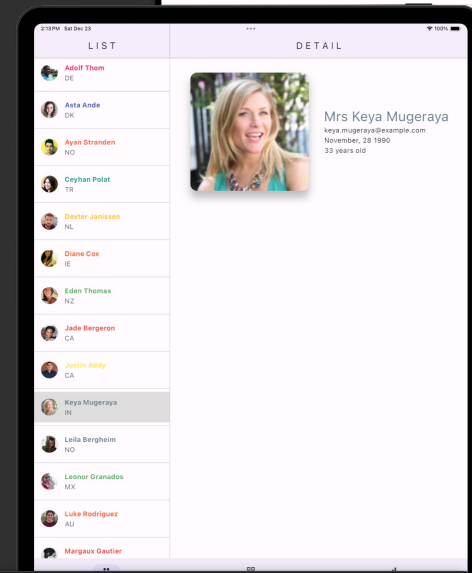
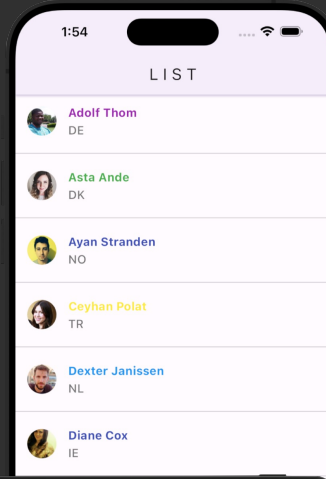
X @JediPixels

YouTube @JediPixels

LinkedIn In/marco-napoli-jedipixels

Overview: Tasks and Goals

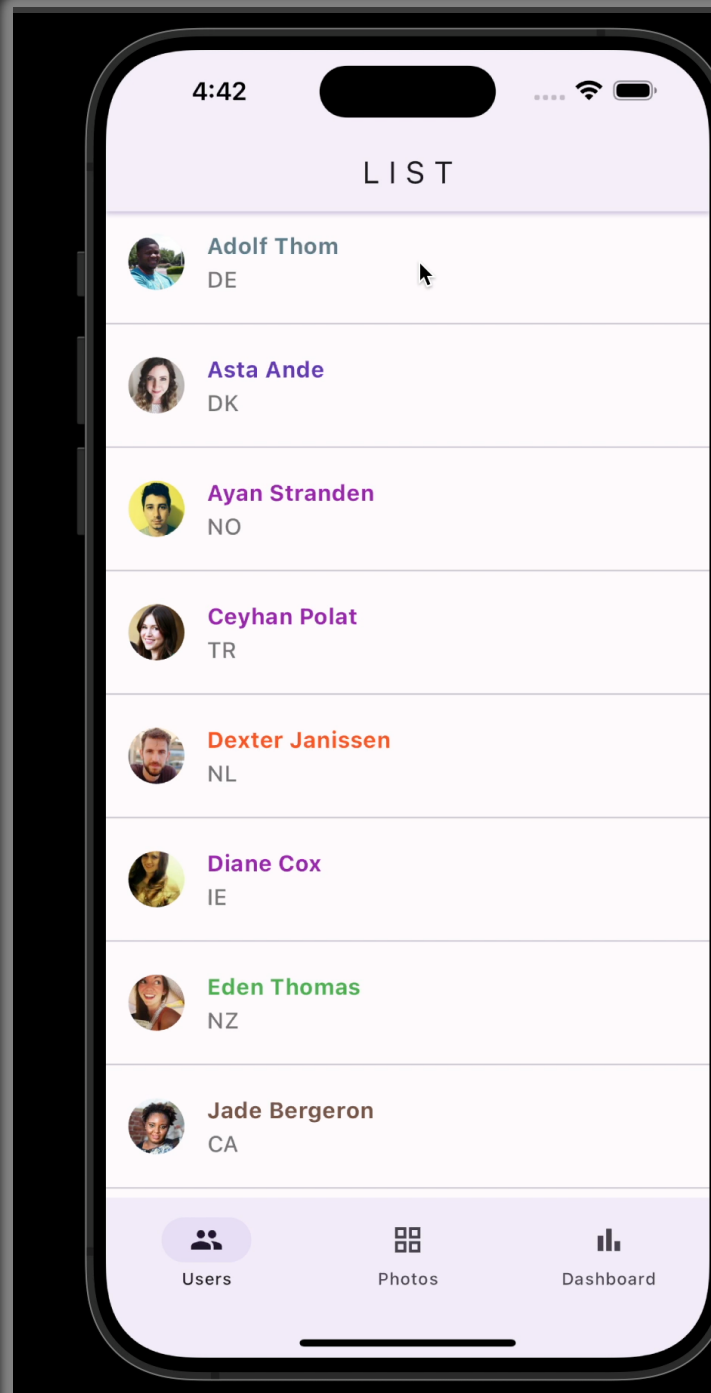
- Create a responsive app supporting a Split-Screen List/Detail layout from a single code base supporting multi-platform phone, tablet, desktop, and web.
- Split-Screen List/Detail on the same screen depending on device, and navigation from List to Detail page on phones.
- Desktop and Web requires to handle both mobile, desktop, and web responsive layout when user changes the size of the app's screen.
- Guess what? No third-party plugins or packages needed.



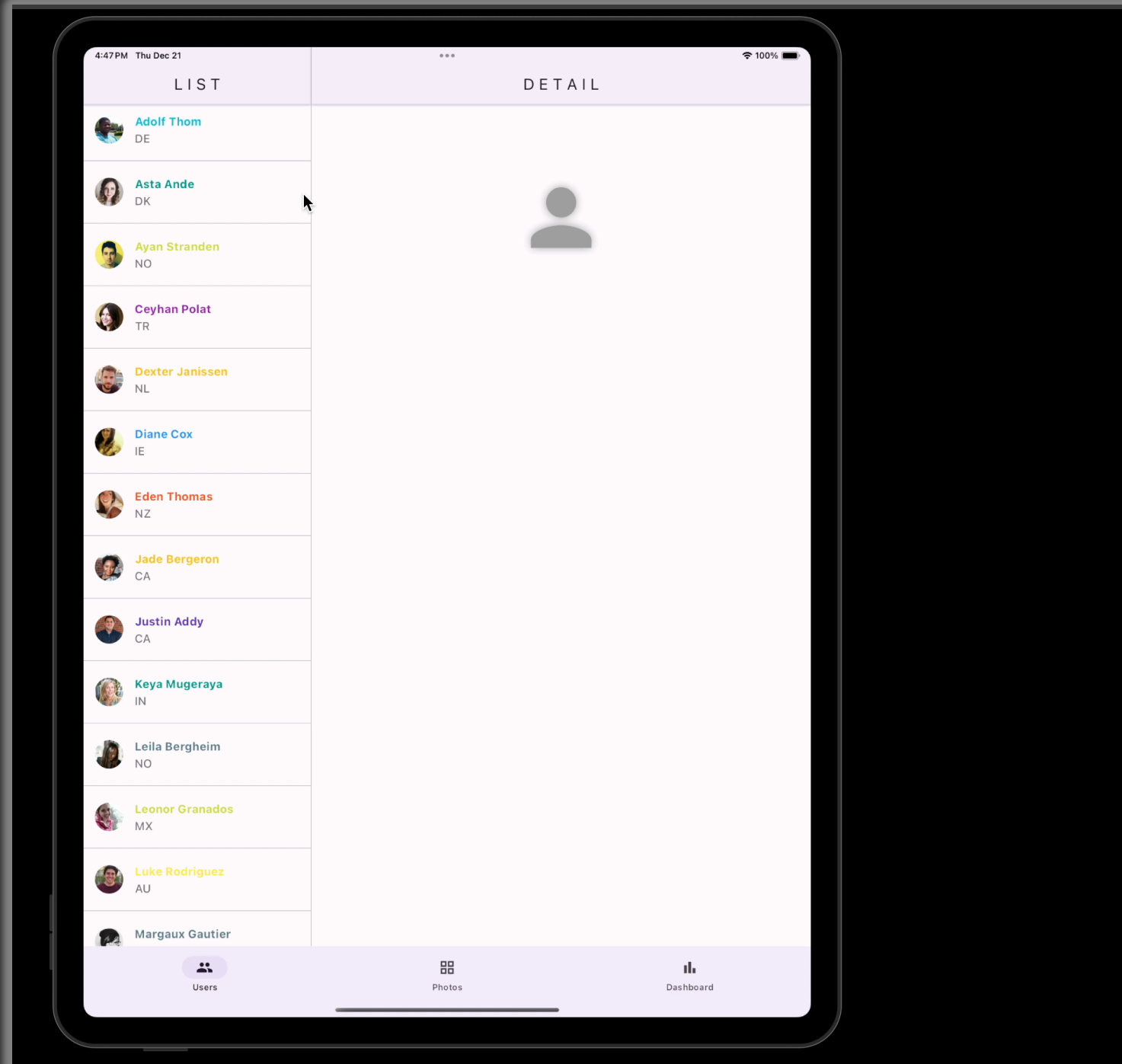
Overview: App's Demo

- You'll start by analyzing the app's common features, reusable logic, and widgets.
- You'll implement Dark and Light Mode, global State Management, and Responsive Layout Builder widget.
- Phone app navigates from the List page to the Details page.
- Tablet, Desktop, and Web app, you'll create a Split-Screen layout showing the List and Detail pages on the same screen.

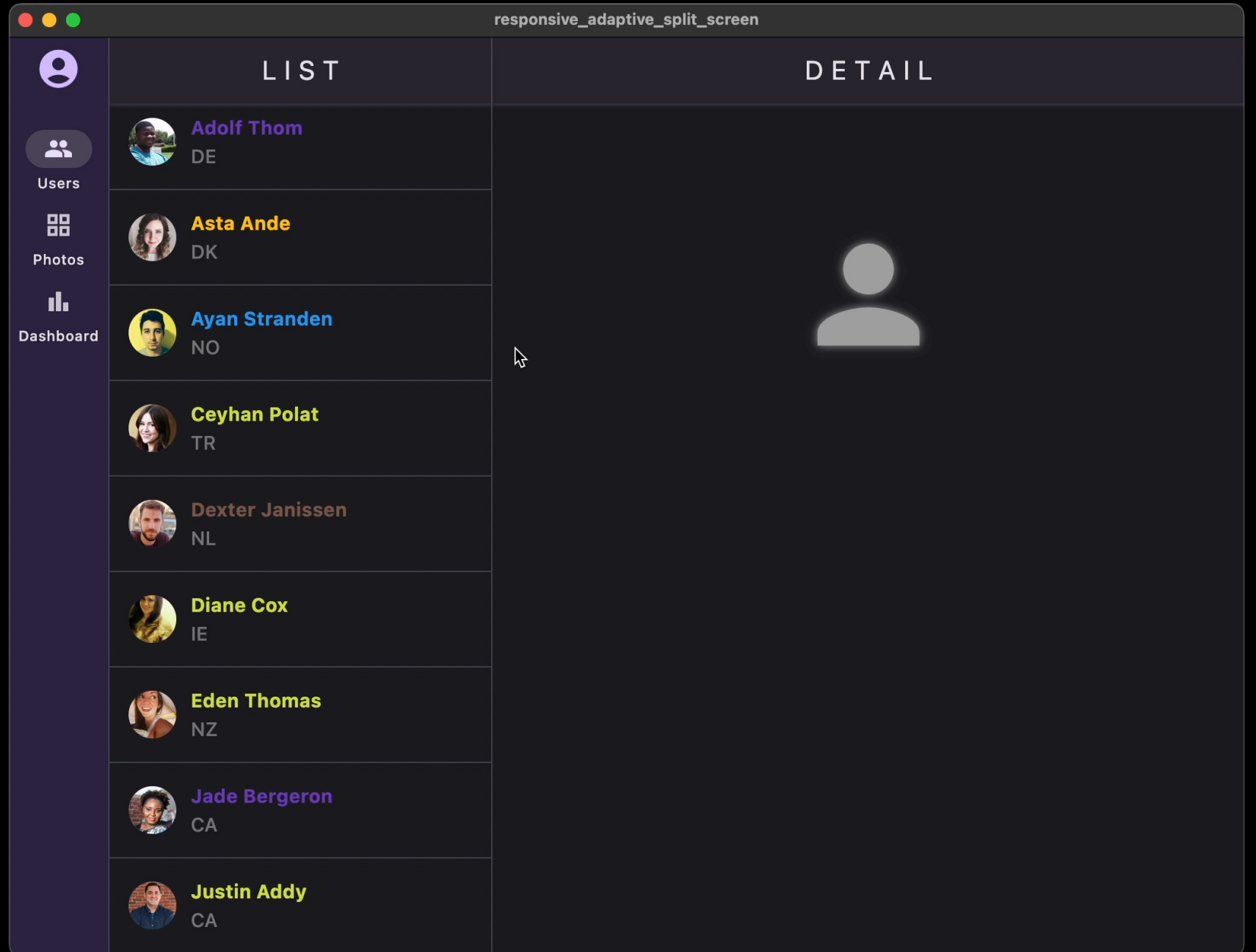
iPhone



iPad



macOS



Web

Responsive Adaptive Split-Screen


localhost:64072

Users

Photos


Dashboard

LIST




Adolf Thom

DE




Asta Ande

DK




Ayan Stranden

NO




Ceyhan Polat

TR




Dexter Janissen

NL




Diane Cox

IE



Eden Thomas


NZ



Jade Bergeron

CA

DETAIL



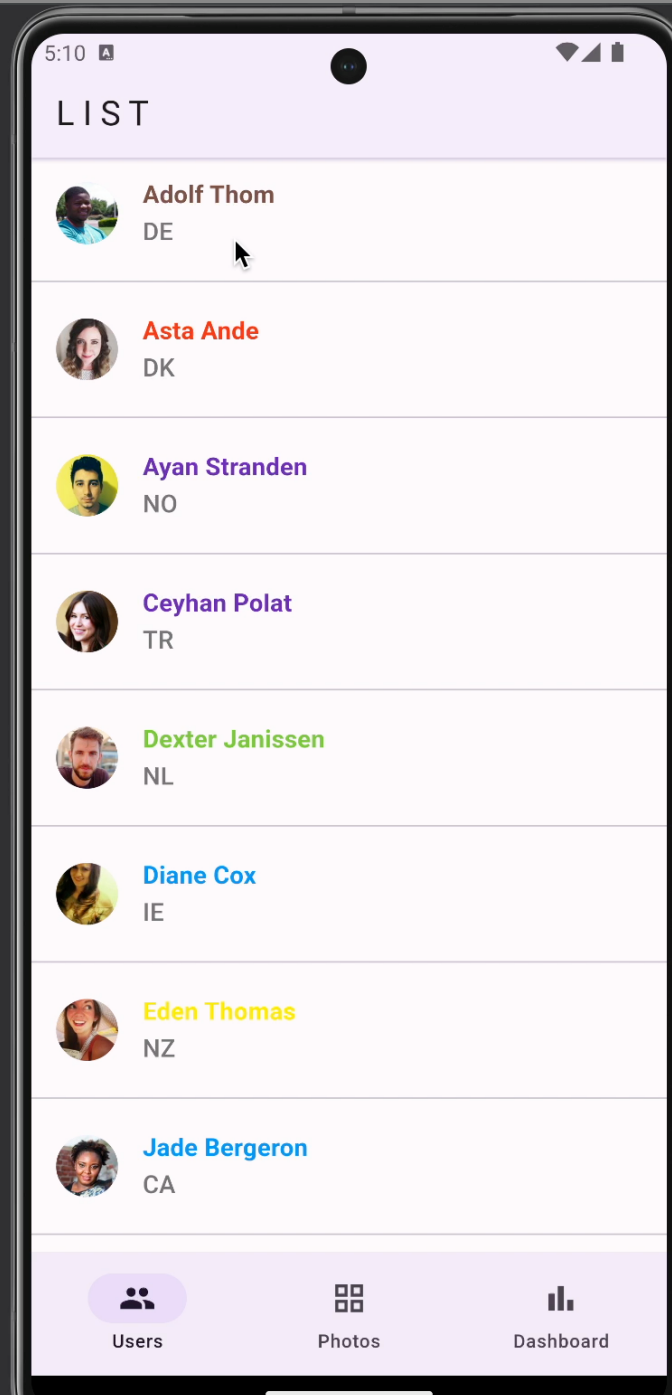
Mr Adolf Thom

adolf.thome@example.com

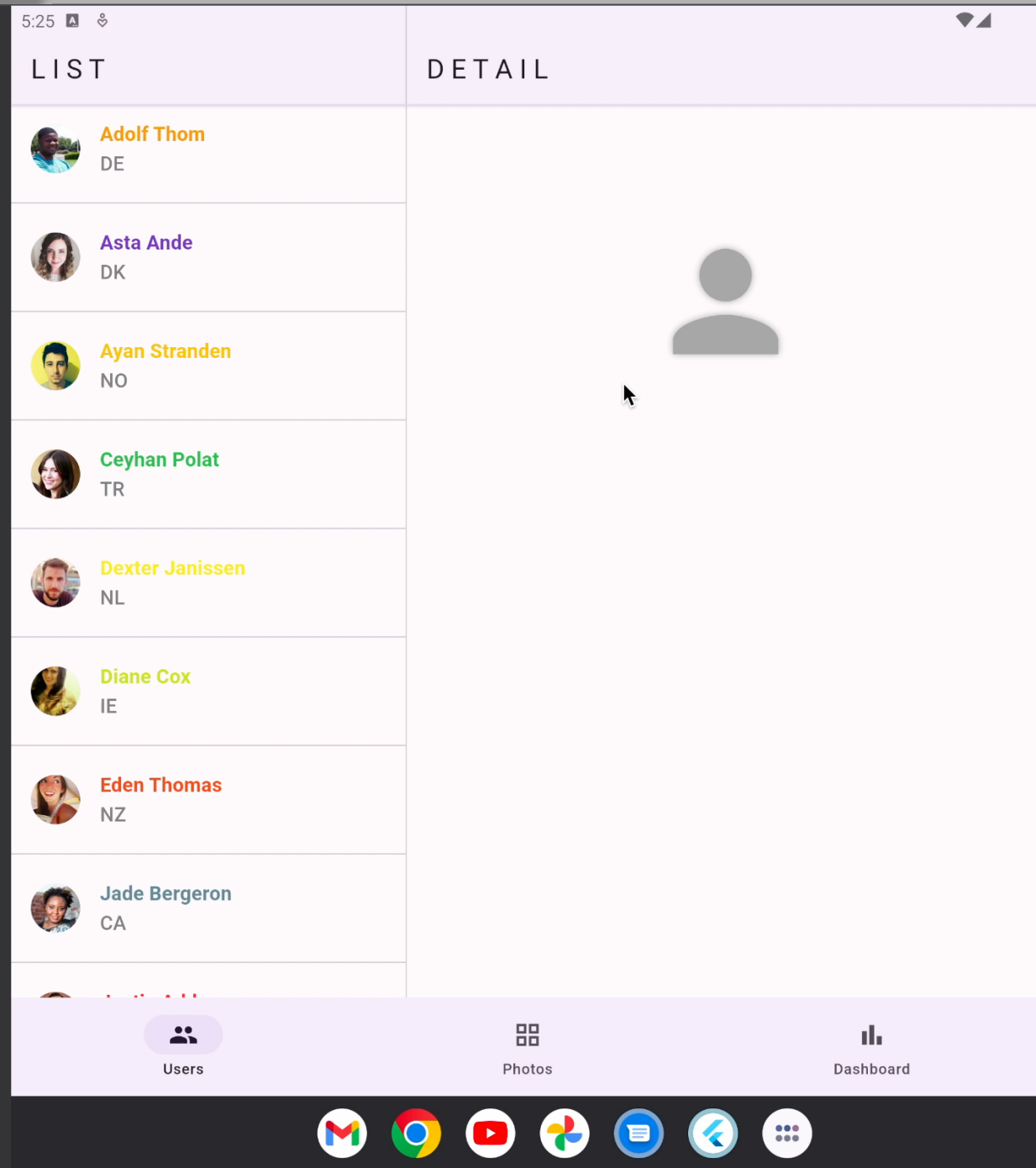
October, 4 1980

43 years old

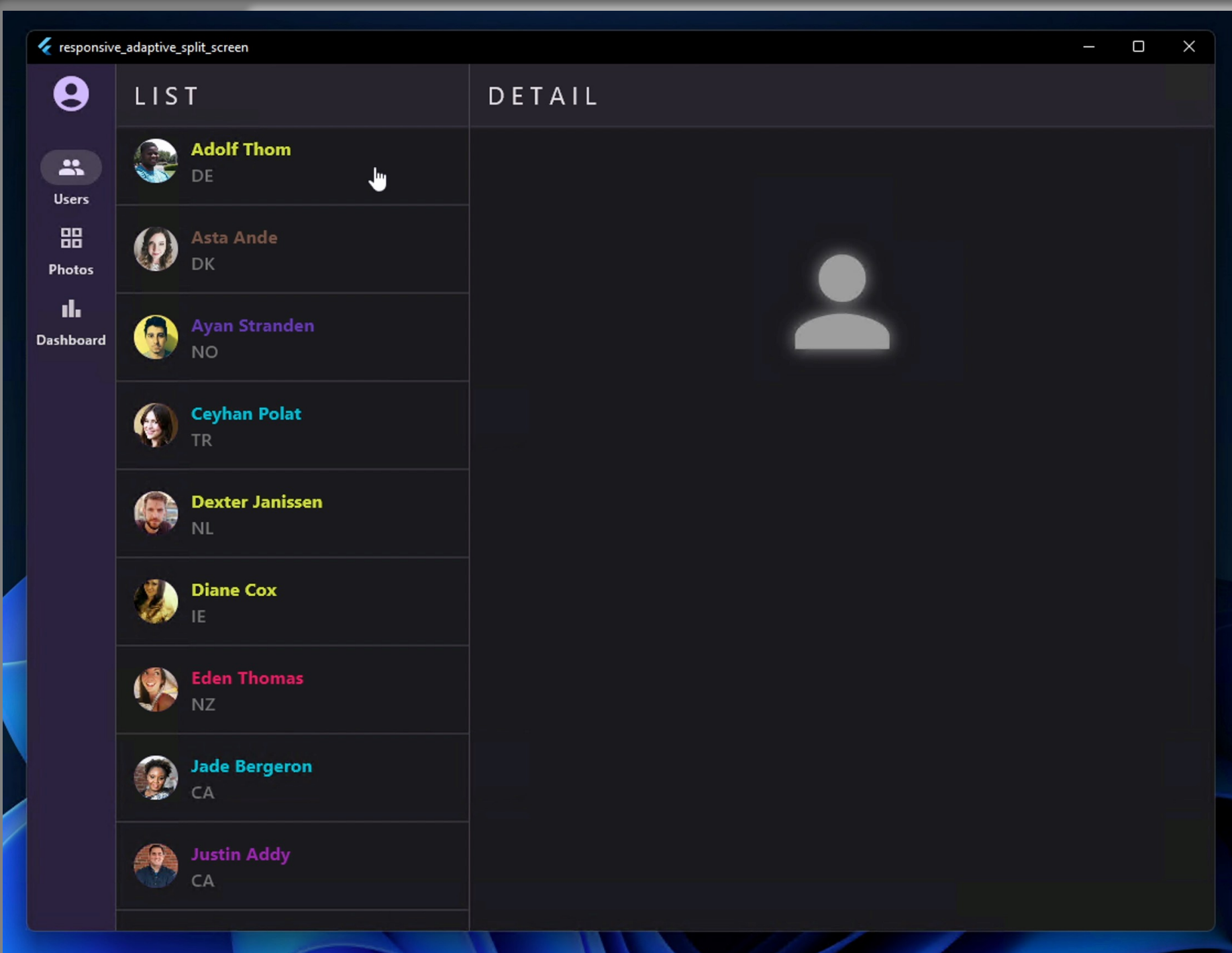
Pixel 7



Fold-Out 8 (foldable)



Windows



Base Structure: Opening and Reviewing the Starter Project

Next



Base Structure: Opening and Reviewing the Starter Project

Helpers and Models

Overview: Helpers and Models

- Analyzing App's Common Features
- Importing Starter Project
 - Reviewing Root
 - `main` page
 - Reviewing Helpers
 - `helpers/constants`
 - `helpers/formatters`
 - `helpers/nav_transition`
 - `helpers/themes`
 - Reviewing Models
 - `models/user_model`

Get the Starter Project

Next

Import **starter project** from **source_code/split_screen_responsive_starter** directory provided in **Resources** of this course

Summary: Helpers and Models

- Analyzed App's Common Features
- Imported Starter Project
 - Reviewed Root
 - **main** page
 - Reviewed Helpers
 - helpers/**constants**
 - helpers/**formatters**
 - helpers/**nav_transition**
 - helpers/**themes**
 - Reviewed Models
 - models/**user_model**

Base Structure: Reviewing the Starter Project State and Widgets

Next



Base Structure: Reviewing the Starter Project

State and Widgets

Overview: State and Widgets

- Analyzing App's Common Features
- Reviewing State
 - state/**app_state**
 - state/**app_state_notifier**
- Reviewing Widgets
 - widgets/**app_bar_elevated**
 - widgets/**graph_bar**
 - widgets/**nav_bar**
 - widgets/**nav_rail**
 - widgets/**title_gradient_bar**

Overview: State and Widgets

- Reviewing macOS -> Runner
 - DebugProfile.entitlements
 - `<key>com.apple.security.network.client</key>`
`<true/>`
`<key>com.apple.security.network.server</key>`
`<true/>`
- Reviewing Web – Index.html
 - Body -> `<script>`
 - `let config = {renderer: "html"}`
`engineInitializer.initializeEngine(config).then(function(appRunner) {}`

Continue Reviewing Starter Project State and Widgets

Next

Import **starter project** from **source_code/split_screen_responsive_starter** directory provided in **Resources** of this course

Summary: State and Widgets

- Analyzed App's Common Features
- Reviewed State
 - state/**app_state**
 - state/**app_state_notifier**
- Reviewed Widgets
 - widgets/**app_bar_elevated**
 - widgets/**graph_bar**
 - widgets/**nav_bar**
 - widgets/**nav_rail**
 - widgets/**title_gradient_bar**

Summary: State and Widgets

- Reviewed macOS -> Runner
 - DebugProfile.entitlements
 - `<key>com.apple.security.network.client</key>`
`<true/>`
`<key>com.apple.security.network.server</key>`
`<true/>`
- Reviewed Web – Index.html
 - Body -> `<script>`
 - `let config = {renderer: "html"}`
`engineInitializer.initializeEngine(config).then(function(appRunner) {}`

Creating Helpers and Pages

Next



Creating Helpers and Pages

Overview: Helpers and Pages

- Modifying Helpers
 - helpers/**constants** (partial)
- Creating Pages
 - pages/responsive/**split_screen**
 - pages/responsive/**responsive_layout_builder**
 - pages/responsive/**desktop_web**
 - pages/responsive/**mobile_tablet**
 - pages/**home**

Live Coding

Live Demo's

for iPhone, iPad, macOS, Web, Pixel 7, Fold-Out 8, and Windows...

Summary: Helpers and Pages

- Modified Helpers
 - helpers/**constants** (partial)
- Created Pages
 - pages/responsive/**split_screen**
 - pages/responsive/**responsive_layout_builder**
 - pages/responsive/**desktop_web**
 - pages/responsive/**mobile_tablet**
 - pages/**home**

Course Summary

Next

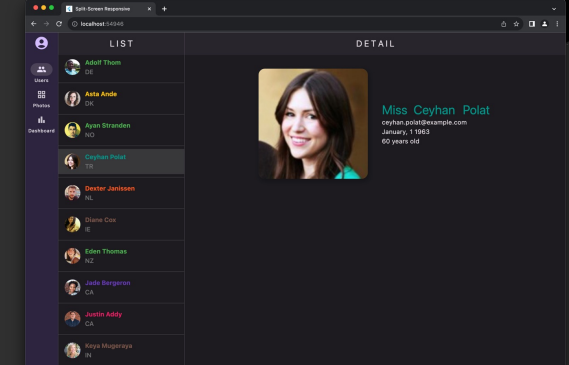
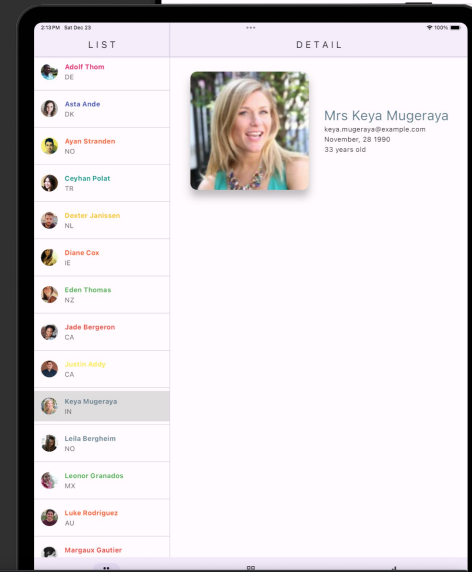
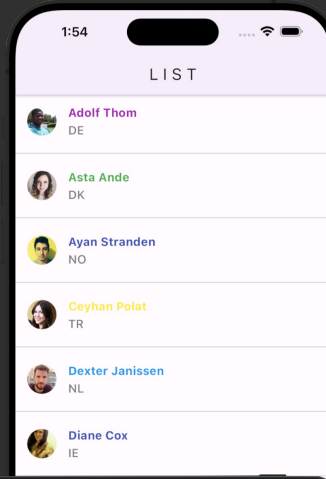


Split-Screen Responsive Design

Course Summary

Course Summary: Tasks and Goals

- Created a responsive app supporting a Split-Screen List/Detail layout from a single code base supporting multi-platform phone, tablet, desktop, and web.
- Split-Screen List/Detail on the same screen depending on device, and navigation from List to Detail page on phones.
- Desktop and Web requires to handle both mobile, desktop, and web responsive layout when user changes the size of the app's screen.
- Guess what? No third-party plugins or packages needed.



Course Summary: App's Demo

- You started by analyzing the app's common features, reusable logic, and widgets.
- You implemented Dark and Light Mode, global State Management, and Responsive Layout Builder widget.
- Phone app navigates from the List page to the Details page.
- Tablet, Desktop, and Web app, you created a Split-Screen layout showing the List and Detail pages on the same screen.



Thank You



Marco Napoli

X @JediPixels

YouTube @JediPixels

in In/marco-napoli-jedipixels