Actividad 7.5. Diagramas funcionales. Comportamiento, estados y eventos.

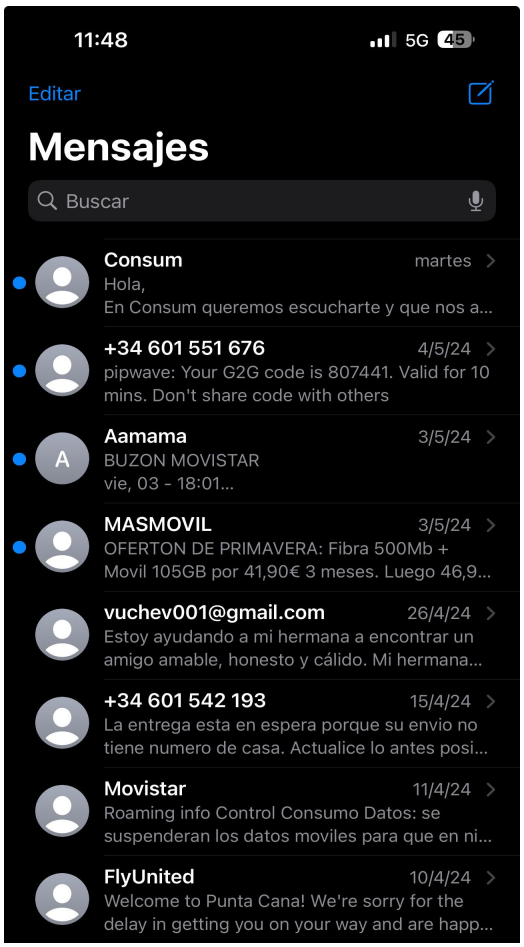
A partir de un escenario determinado en una aplicación de móvil o webapp, debes crear un diagrama funcional de estados.

Dicho diagrama mostrará los eventos generados a una o varias acciones Determinadas.

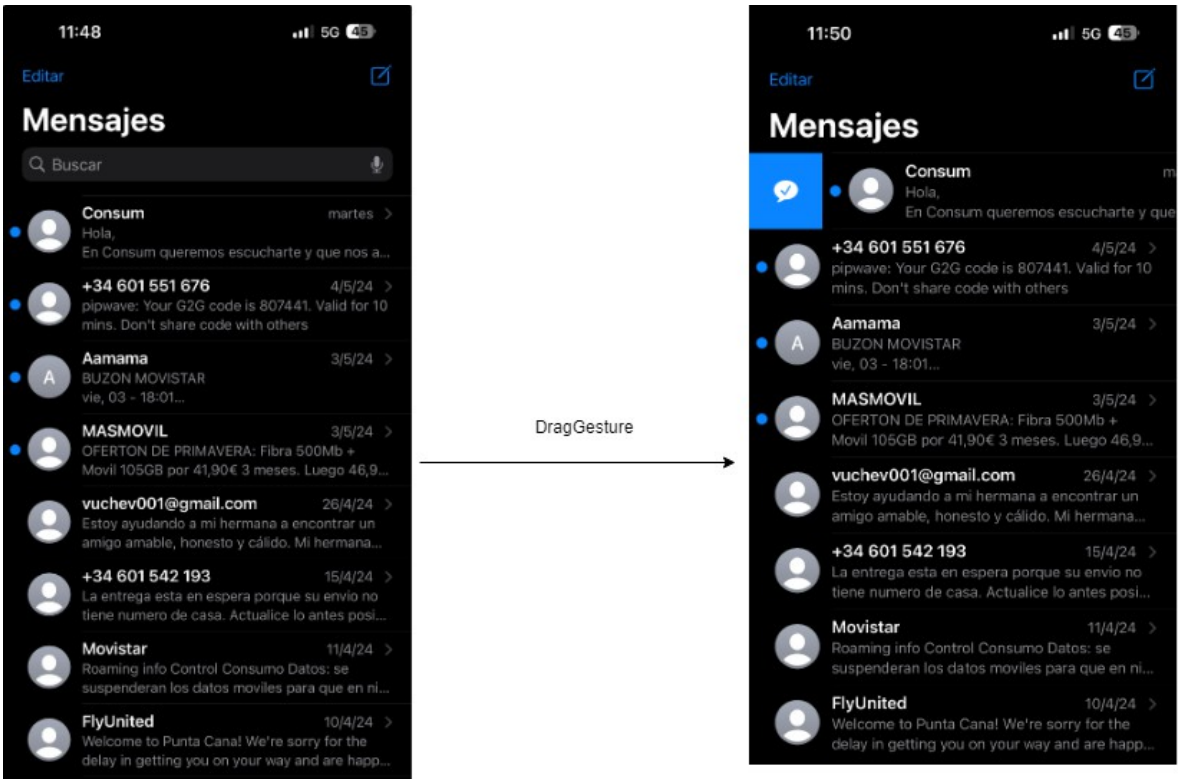
Además del diagrama de estados, también se listarán los eventos generados y la función a la que se llama en un lenguaje de programación determinado, SwiftIO / Android →Flutter, React, .js u otro que prefieras.

El escenario es la aplicacion mensajes de IOS.

Escenario Principal:



Para el primer objeto, vamos a gestionar-lo. En este caso, si arrastramos la interfaz hacia la derecha, ocurre el siguiente evento y cambia de estado:



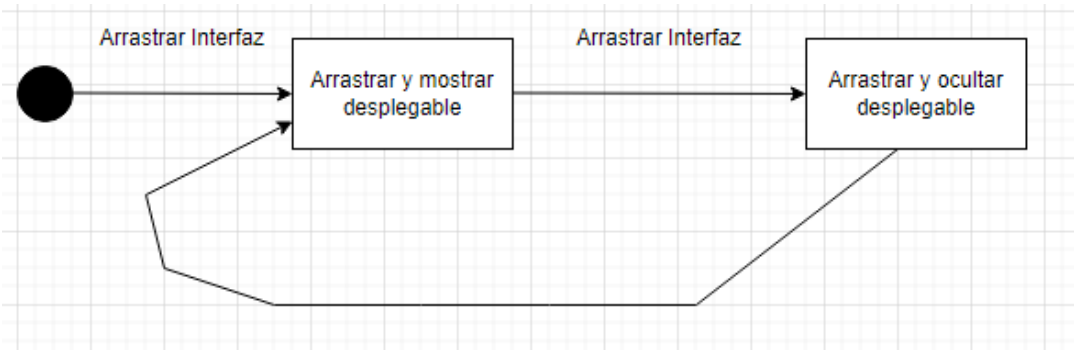
Al arrastrar hacia la derecha aparecera un desplegable que te permitira marcar como leido el mensaje o no.

Para que ocurra este evento, debemos de utilizar la funcion DragGesture de la plataforma SwiftUI.

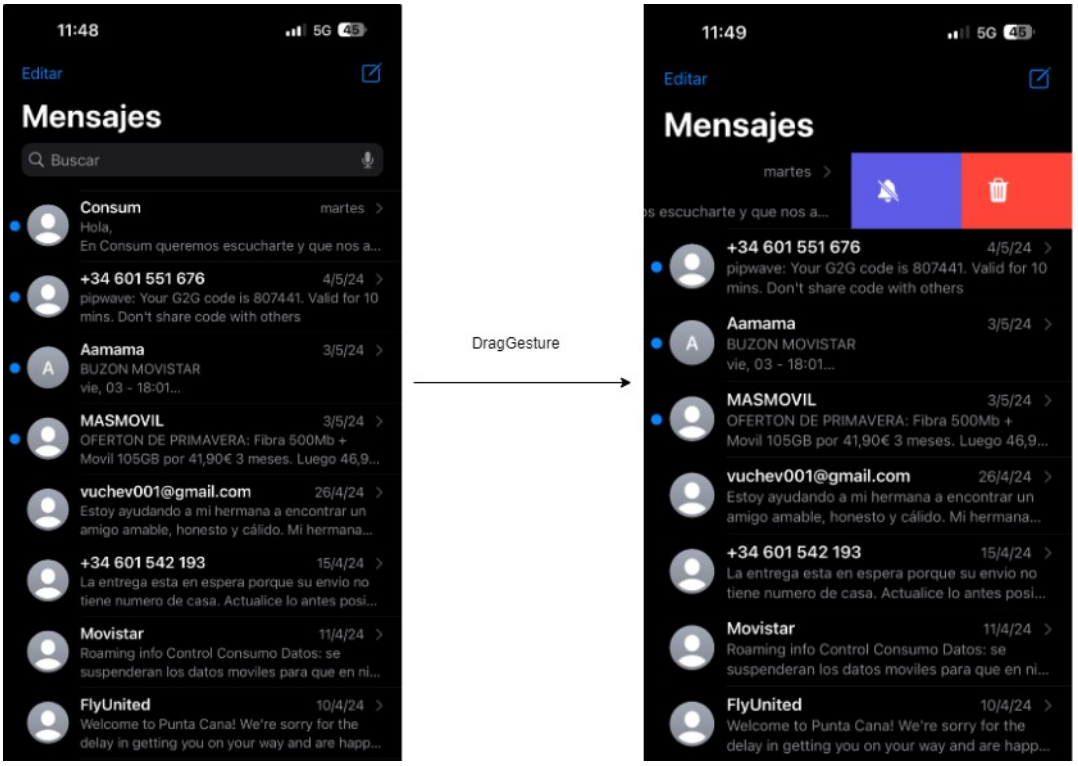
```
struct DragGestureView: View {

@State private var isDragging = false

var drag: some Gesture {

        DragGesture()

                .onChanged { _ in self.isDragging = true }

                .onEnded { _ in self.isDragging = false }
}

var body: some View {

        Circle()

        .fill(self.isDragging ? Color.red : Color.blue)

        .frame(width: 100, height: 100, alignment: .center)
        .gesture(drag)
        }
}
```

Al usar la funcion DragGesture, canviara del estado predeterminado a el estado con el desplegable, si queremos volver al estado inicial utilizaremos el DragGesture hacia el lado contraria.

El diagrama de estados:



Otro Evento, en este caso, si arrastramos la interfaz hacia la izquierda, ocurre el siguiente evento y cambia de estado:



Al arrastrar hacia la izquierda en este caso aparecera un desplegable que te permitira gestionar las notificaciones ademas de poder eliminar el objeto.

Para que ocurra este evento, debemos de utilizar la funcion al igual que antes DragGesture de la plataforma SwiftUI.

```
struct DragGestureView: View {

@State private var isDragging = false

var drag: some Gesture {

        DragGesture()

                .onChanged { _ in self.isDragging = true }

                .onEnded { _ in self.isDragging = false }
}

var body: some View {

        Circle()

        .fill(self.isDragging ? Color.red : Color.blue)
```

```
        .frame(width: 100, height: 100, alignment: .center)
        .gesture(drag)

        }
}
```

Al usar la funcion DragGesture, canviara del estado predeterminado a el estado con el desplegable, si queremos volver al estado inicial utilizaremos el DragGesture hacia el lado contraria.

El diagrama de estados: