```
import SwiftUI
import MapKit
import EventKit
import EventKitUI
import WebKit

private extension Color {
    static let brandPink = Color(red: 1.0, green: 38/255, blue:
138/255)
}

private struct PillTag: View {
    let text: String
    var body: some View {
        Text(text.uppercased())
            .font(.system(size: 16, weight: .semibold,
design: .rounded))
            .kerning(0.2)
            .foregroundColor(.white)
            .padding(.horizontal, 15)                // pill
width
            .frame(height: 30)                       // pill
height like screenshot
            .background(
                Capsule()
                    .fill(Color.brandPink.opacity(0.7))
                    .overlay(                            // subtle
bevel
                        Capsule().stroke(.white.opacity(0.18),
lineWidth: 1)
                    )
            )
            .shadow(color: .black.opacity(0.35), radius: 10, x:
0, y: 6)
    }
}

struct InfoRow: View {
    let icon: String
    let title: String
    var subtitle: String = ""

    var body: some View {
        HStack(alignment: .top, spacing: 16) {
            Image(systemName: icon)
                .font(.system(size: 22, weight: .bold))
                .frame(width: 36, height: 36)
                .foregroundColor(.white)
                .background(.white.opacity(0.08))
                .clipShape(Circle())
            VStack(alignment: .leading, spacing: 6) {
                Text(title).foregroundColor(.white)
                    .font(.system(size: 20, weight: .semibold))

Text(subtitle).foregroundColor(.white.opacity(0.82))
                    .font(.system(size: 17))
            }
            Spacer()
        }
        .padding(.vertical, 6)
    }
}

struct EventEditSheet: UIViewControllerRepresentable {
    let eventStore: EKEventStore
    let start: Date
    let end: Date
    let title: String
    let location: String
    @Environment(\.dismiss) private var dismiss

    func makeUIViewController(context: Context) ->
EKEventEditViewController {
        let vc = EKEventEditViewController()
        vc.eventStore = eventStore
        let event = EKEvent(eventStore: eventStore)
        event.title = title
        event.location = location
        event.startDate = start
        event.endDate = end
        event.calendar = eventStore.defaultCalendarForNewEvents
        vc.event = event
        vc.editViewDelegate = context.coordinator
        return vc
    }
    func updateUIViewController(_ uiViewController:
EKEventEditViewController, context: Context) {}
    func makeCoordinator() -> Coordinator
{ Coordinator(dismiss: dismiss) }

    final class Coordinator: NSObject, EKEventEditViewDelegate
{
        let dismiss: DismissAction
        init(dismiss: DismissAction) { self.dismiss = dismiss }
        func eventEditViewController(_ controller:
EKEventEditViewController,
                                     didCompleteWith action:
EKEventEditViewAction) {
            controller.dismiss(animated: true)
{ self.dismiss() }
        }
    }
}

// MARK: - Web View Components
struct WKWebViewWrapper: UIViewRepresentable {
    let url: URL

    func makeUIView(context: Context) -> WKWebView {
        let webView = WKWebView()
        let request = URLRequest(url: url)
        webView.load(request)
        return webView
    }

    func updateUIView(_ uiView: WKWebView, context: Context) {
        // Not needed for static URL
    }
}

struct WebViewContainer: View {
    let url: URL
    @Environment(\.dismiss) var dismiss

    var body: some View {
        VStack(spacing: 0) {
            // Custom black header with back button
            HStack {
                Button(action: {
                    dismiss()
                }) {
                    HStack(spacing: 4) {
                        Image(systemName: "chevron.left")
                        Text("UniCube")
                    }
                    .foregroundColor(.white)
                    .font(.headline)
                }
                .padding()

                Spacer()
            }
            .background(Color.black)

            // Embedded WKWebView
            WKWebViewWrapper(url: url)
                .edgesIgnoringSafeArea(.bottom)
        }
        .background(Color.black)
    }
}

struct EventViewPage: View {
    // nav
    @Environment(\.dismiss) private var dismiss
    @State private var liked = false
    @State private var showShare = false

    // calendar
    @State private var showCalendar = false
    @State private var eventStore = EKEventStore()

    // web view
    @State private var showWebView = false

    // map
    @State private var region = MKCoordinateRegion(
        center: CLLocationCoordinate2D(latitude: -37.8075,
longitude: 144.9640),
        span: MKCoordinateSpan(latitudeDelta: 0.004,
longitudeDelta: 0.004)
    )

    // event details
    private let eventTitle = "THERAPY DOGS"
    private let eventDayLine = "Monday 9 September 2025"
    private let timeLine = "11:00pm â 1:00pm"
    private let startDate: Date = {
        var c = DateComponents(); c.year = 2025; c.month = 9;
c.day = 9; c.hour = 23; c.minute = 0
        return Calendar.current.date(from: c) ?? .now
    }()
    private let endDate: Date = {
        var c = DateComponents(); c.year = 2025; c.month = 9;
```

```
c.day = 10; c.hour = 1; c.minute = 0
        return Calendar.current.date(from: c) ?? .now
    }()
    private let placeTitle = "RMIT Campus Building 10"
    private let placeDetail = "Level 4, Room 3"

    // Change this URL to your desired destination
    private let joinURL = URL(string: "https://www.rmit.edu.au/
students/news/2022/november/therapy-puppies-business-
students")!

    // Open Apple Maps to this event location with directions
    private func openInAppleMaps() {
        let coordinate = CLLocationCoordinate2D(latitude:
-37.8075, longitude: 144.9640) // RMIT area
        let placemark = MKPlacemark(coordinate: coordinate)
        let item = MKMapItem(placemark: placemark)
        item.name = "RMIT Campus Building 10, Level 4, Room 3"

        // Choose one: walking / driving / transit
        let options = [MKLaunchOptionsDirectionsModeKey:
MKLaunchOptionsDirectionsModeWalking]
        item.openInMaps(launchOptions: options)
    }

    var body: some View {
        NavigationStack {
            ZStack {
                Color.black.ignoresSafeArea()

                ScrollView(showsIndicators: false) {
                    VStack(alignment: .leading, spacing: 20) {
                        // Header image + FREE
                        Spacer()
                        ZStack(alignment: .topLeading) {
                            Image("doggy")
                                .resizable()
                                .scaledToFill()
                                .frame(height: 240)
                                .clipShape(RoundedRectangle(cor
nerRadius: 18, style: .continuous))
                                .clipped()
                                .overlay(

RoundedRectangle(cornerRadius: 18)
                                        .stroke(Color.white.opa
city(0.08), lineWidth: 1)
                                )
                            PillTag(text: "FREE")
                                .padding(.leading, 8)
                                .padding(.top, 16)
                        }
                        .padding(.horizontal, 24)

                        // Title + underline
                        VStack(alignment: .leading, spacing:
12) {
                            Text(eventTitle)
                                .foregroundColor(.white)
                                .font(.system(size: 32,
weight: .bold))

Rectangle().fill(Color.brandPink).frame(height: 2)
                        }
                        .padding(.horizontal, 24)

                        // Date / location
                        VStack(spacing: 10) {
                            InfoRow(icon: "calendar", title:
eventDayLine, subtitle: timeLine)
                                .padding(.horizontal, 24)
                            InfoRow(icon: "mappin.and.ellipse",
title: placeTitle, subtitle: placeDetail)
                                .padding(.horizontal, 24)
                        }


Divider().overlay(Color.white.opacity(0.14))
                            .padding(.horizontal, 24)

                        // Description
                        VStack(alignment: .leading, spacing:
10) {
                            Text("Description")
                                .font(.system(size: 22,
weight: .bold))
                                .foregroundColor(.white)
                            Text("Take a pup-powered study
break! Meet certified therapy dogs on campus and leave with
lighter shoulders.")
                                .foregroundColor(.white.opacity
(0.82))
```

```
                                .font(.system(size: 17))
                                .lineSpacing(4)
                        }
                        .padding(.horizontal, 24)

                        // Map
                        VStack(alignment: .leading, spacing:
10) {
                            Text("Location")
                                .font(.system(size: 22,
weight: .bold))
                                .foregroundColor(.white)


Map(position: .constant(.region(region)))
                                .mapControls
{ MapUserLocationButton(); MapCompass() }
                                .overlay(alignment: .center) {
                                    Image(systemName: "mappin")
                                        .font(.system(size:
28))
                                        .foregroundStyle(.red,
.white)
                                        .shadow(radius: 4)
                                }
                                .frame(height: 250)
                                .clipShape(RoundedRectangle(cor
nerRadius: 18, style: .continuous))
                                .overlay(

RoundedRectangle(cornerRadius: 18)
                                        .stroke(Color.white.opa
city(0.12), lineWidth: 1)
                                )
                        }
                        .padding(.horizontal, 24)

                        Spacer(minLength: 140)
                    }
                    .padding(.bottom, 12)
                }

                // Bottom actions (HIG: primary + secondary)
                VStack {
                    Spacer()
                    HStack(spacing: 24) {
                        Button { requestCalendarAndShow() }
label: {
                            ZStack {

Circle().fill(Color.brandPink.opacity(0.22)).frame(width: 72,
height: 72)
                                Image(systemName:
"calendar.badge.plus")
                                    .font(.system(size: 26,
weight: .bold))
                                    .foregroundColor(.brandPink
)
                            }
                        }

                        Button { showWebView = true } label: {
                            HStack {
                                Spacer(minLength: 0)
                                Text("JOIN")
                                    .font(.system(size: 24,
weight: .heavy, design: .rounded))
                                Spacer(minLength: 0)
                                Image(systemName:
"chevron.right")
                                    .font(.system(size: 20,
weight: .heavy))
                            }
                            .foregroundColor(.white)
                            .padding(.vertical, 16)
                            .padding(.horizontal, 22)
                            .background(Color.brandPink)
                            .clipShape(RoundedRectangle(cornerR
adius: 30, style: .continuous))
                        }
                        .accessibilityHint("Opens registration
page for the event")
                    }
                    .padding(.horizontal, 28)
                    .padding(.vertical, 18)
                    .background()
                }
            }
            .navigationBarTitleDisplayMode(.inline)
            .toolbar {
                ToolbarItem(placement: .navigationBarLeading) {
                    Button { dismiss() } label: {
                        HStack(spacing: 6) {
                            Image(systemName: "chevron.left")
                                .font(.system(size: 16,
weight: .medium))
                            Text("Categories List")
                                .font(.system(size: 17,
weight: .medium))
                        }
                        .foregroundColor(.white)
                    }
                }

ToolbarItemGroup(placement: .navigationBarTrailing) {
                        Button { showShare = true } label: {
                            Image(systemName:
"square.and.arrow.up")
                                .font(.system(size: 22,
weight: .bold))
                                .foregroundColor(.white)
                        }
                        Button {
                            withAnimation(.spring(response:
0.25, dampingFraction: 0.6)) {
                                liked.toggle()
                            }
                        } label: {
                            Image(systemName: liked ?
"heart.fill" : "heart")
                                .font(.system(size: 22,
weight: .bold))
                                .foregroundColor(liked ? .brand
Pink : .white)
                        }
                    }
            }
            .tint(.white) // toolbar icons
            .sheet(isPresented: $showCalendar) {
                EventEditSheet(
                    eventStore: eventStore,
                    start: startDate,
                    end: endDate,
                    title: "Therapy Dogs",
                    location: "\(placeTitle), \(placeDetail)"
                )
                .ignoresSafeArea()
            }
            .sheet(isPresented: $showShare) {
                let text = "Therapy Dogs â \(eventDayLine), \
(timeLine) at \(placeTitle) \(placeDetail)"
                ActivityView(activityItems: [text])
            }
            .fullScreenCover(isPresented: $showWebView) {
                WebViewContainer(url: joinURL)
            }
        }
        .navigationBarBackButtonHidden(true) // Hide the
default back button
        .preferredColorScheme(.dark)
    }

    // MARK: Calendar (iOS 17+)
    private func requestCalendarAndShow() {
        switch EKEventStore.authorizationStatus(for: .event) {
        case .authorized, .fullAccess, .writeOnly:
            showCalendar = true
        case .notDetermined:
            eventStore.requestFullAccessToEvents { granted, _
in
                DispatchQueue.main.async { if granted
{ showCalendar = true } }
            }
        case .denied, .restricted:
            break
        @unknown default:
            break
        }
    }
}

private struct ActivityView: UIViewControllerRepresentable {
    var activityItems: [Any]
    func makeUIViewController(context: Context) ->
UIActivityViewController {
        UIActivityViewController(activityItems: activityItems,
applicationActivities: nil)
    }
    func updateUIViewController(_ uiViewController:
UIActivityViewController, context: Context) {}
}

private struct PriceTag: View {
    let isFree: Bool
    var body: some View {
        Text(isFree ? "FREE" : "$")
            .font(.system(size: 14, weight: .regular))     // 
Figma spec
            .foregroundColor(.white)
            .frame(width: 83, height: 30)                   // 
Figma size
            .background(
                RoundedRectangle(cornerRadius: 7, 
style: .continuous)
                    .fill(Color.brandPink.opacity(0.70))    // 
70% opacity pink
            )
    }
}

#Preview {
    EventViewPage()
}
```