TICKEZY

Event • Ticket • Moment

A Real-World Ticketing App Built with Swift & SwiftUI

UMURERWA Lisa Ornella
MUGABO André

SwiftUI Pitch Presentation
December 30, 2025

Contents

1 Introduction

Tickezy is a real-world iOS ticketing app built using Swift and SwiftUI. It focuses on **fast, smooth, and intuitive ticketing** for users.

Speaker Notes

Lisa: Good morning. We are UMURERWA Lisa Ornella and MUGABO André. Today we are presenting Tickezy, a real-world ticketing application built using Swift and SwiftUI.

1.1 Core Message

- Demonstrates real-world iOS development using SwiftUI.
- Emphasizes clean, scalable, and maintainable architecture.
- Focus on smooth user experience and modern SwiftUI practices.

Speaker Notes

Lisa: This presentation focuses on how we used SwiftUI to build a clean, modern, and scalable iOS application.

2 Problem Statement

- Buying event tickets can be slow, confusing, or unintuitive.
- Users expect **native mobile apps** that are responsive and visually appealing.

Opportunity: Build a simple and intuitive ticketing experience using SwiftUI.

Speaker Notes

Lisa: Many ticketing platforms feel complex. SwiftUI helped us focus on clarity and user experience.

3 SwiftUI Basics

3.1 App Lifecycle

- Modern SwiftUI apps use the `@main` entry point.
- Scene-based lifecycle replaces storyboards.
- Simplifies navigation and state management.

Speaker Notes

Lisa: Tickezy uses the modern SwiftUI lifecycle, which simplifies state and navigation management.

3.2 Views and UI Components

- `NavigationStack` for navigation
- `List` and `ScrollView` for content
- Reusable UI components for consistency

Declarative and state-driven user interface.

Speaker Notes

Lisa: SwiftUI allows us to describe the UI declaratively. The UI reacts automatically to state changes.

3.3 State Management

- `@State` for local view state
- `@Published` and `ObservableObject` for shared state
- UI automatically updates when state changes

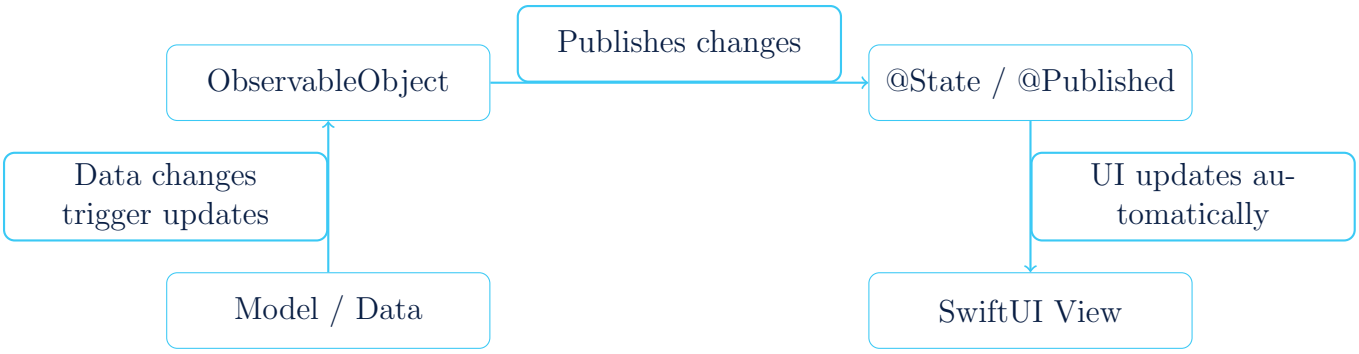

Figure 1: SwiftUI State Management Flow

Speaker Notes

André: SwiftUI listens to state changes in @State or ObservableObject and updates the UI automatically.

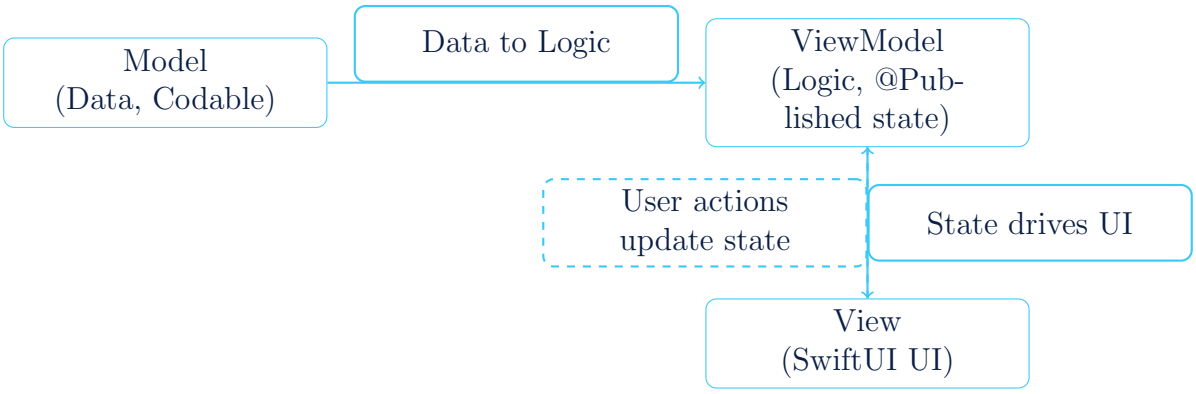

Figure 2: MVVM Architecture Flow

4 MVVM Architecture

Speaker Notes

André: MVVM separates UI from business logic, making the app scalable and easy to maintain.

5 UI Wireframes

5.1 Splash Screen Wireframe

5.2 Event List Wireframe

5.3 Ticket View Wireframe

6 Backend Integration

Feature	SwiftUI Impact
Authentication	Secure login flow
QR Code	Digital ticket display
PDF Ticket	Downloadable passes
Email	Confirmation feedback

Table 1: Backend Features Supporting UX

Speaker Notes

André: Backend services exist to support a smooth and responsive SwiftUI experience.

7 Challenges & Lessons

- Managing asynchronous data with SwiftUI
- Designing reactive, state-driven UI
- Maintaining clean architecture while scaling features

Speaker Notes

Lisa: SwiftUI taught us to think in terms of state rather than traditional screen-based UI.

8 Conclusion

Tickezy demonstrates the team's ability to build **scalable, real-world iOS applications** with SwiftUI. By combining MVVM, modern SwiftUI features, and backend support, Tickezy delivers a **smooth, intuitive, and professional user experience**.

Speaker Notes

Both: Thank the audience and invite questions.

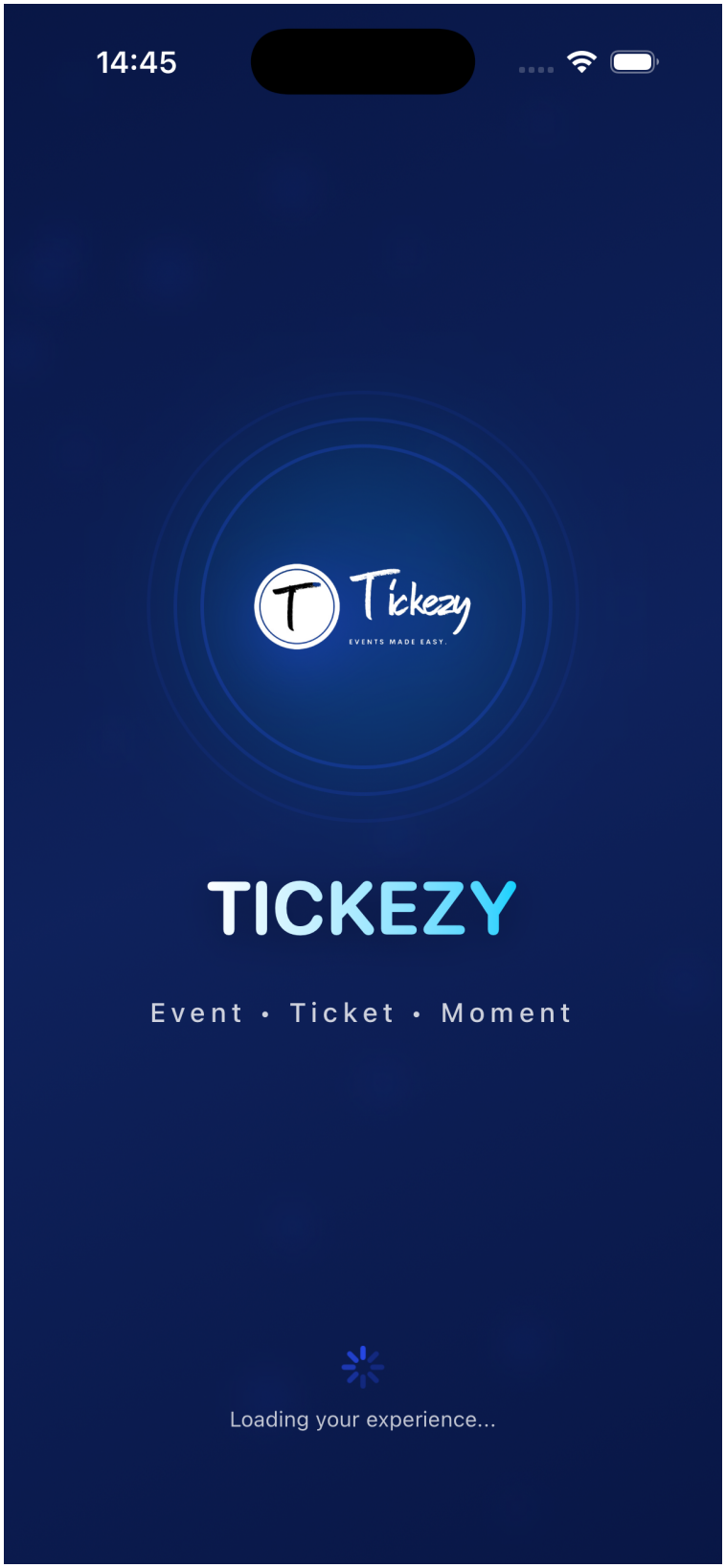

Figure 3: Splash Screen Wireframe

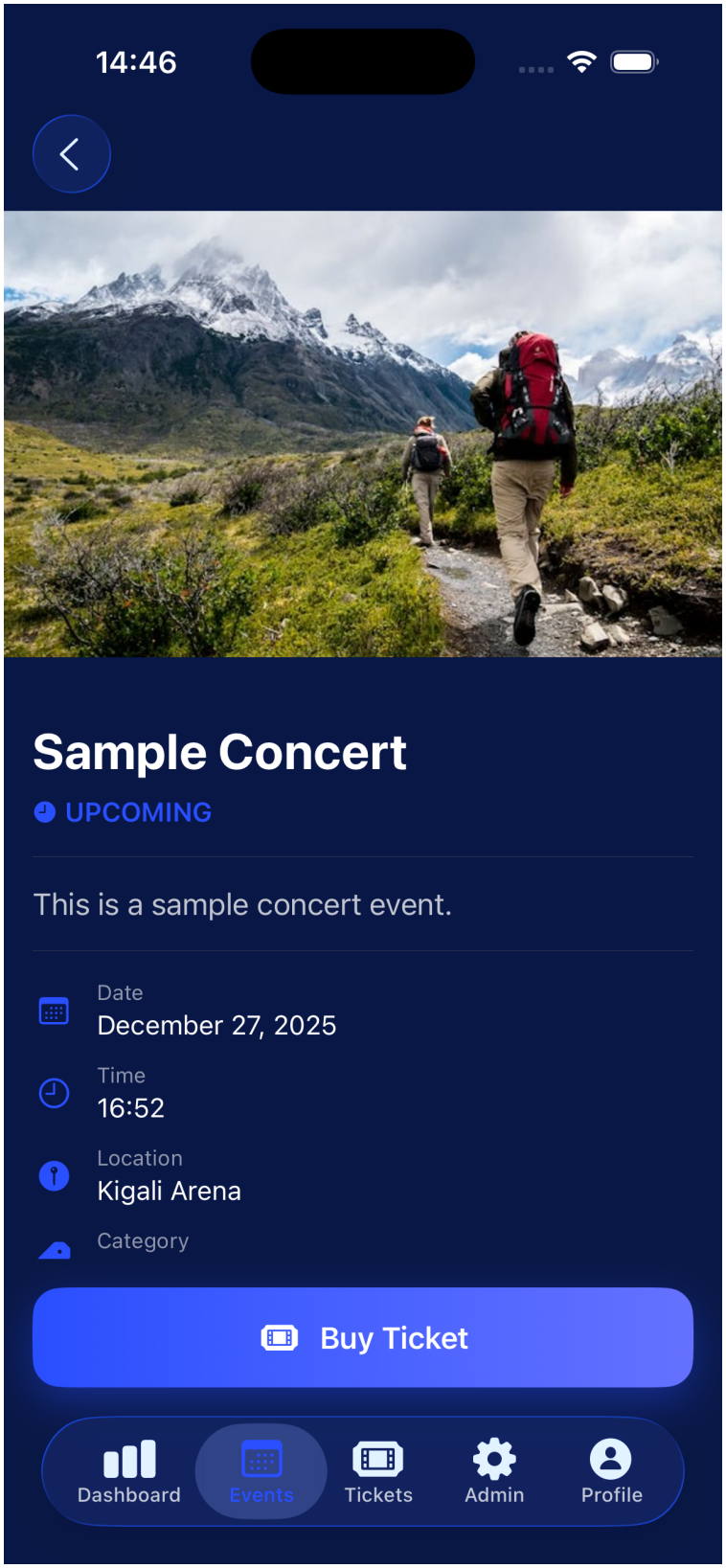

Figure 4: Event List Wireframe

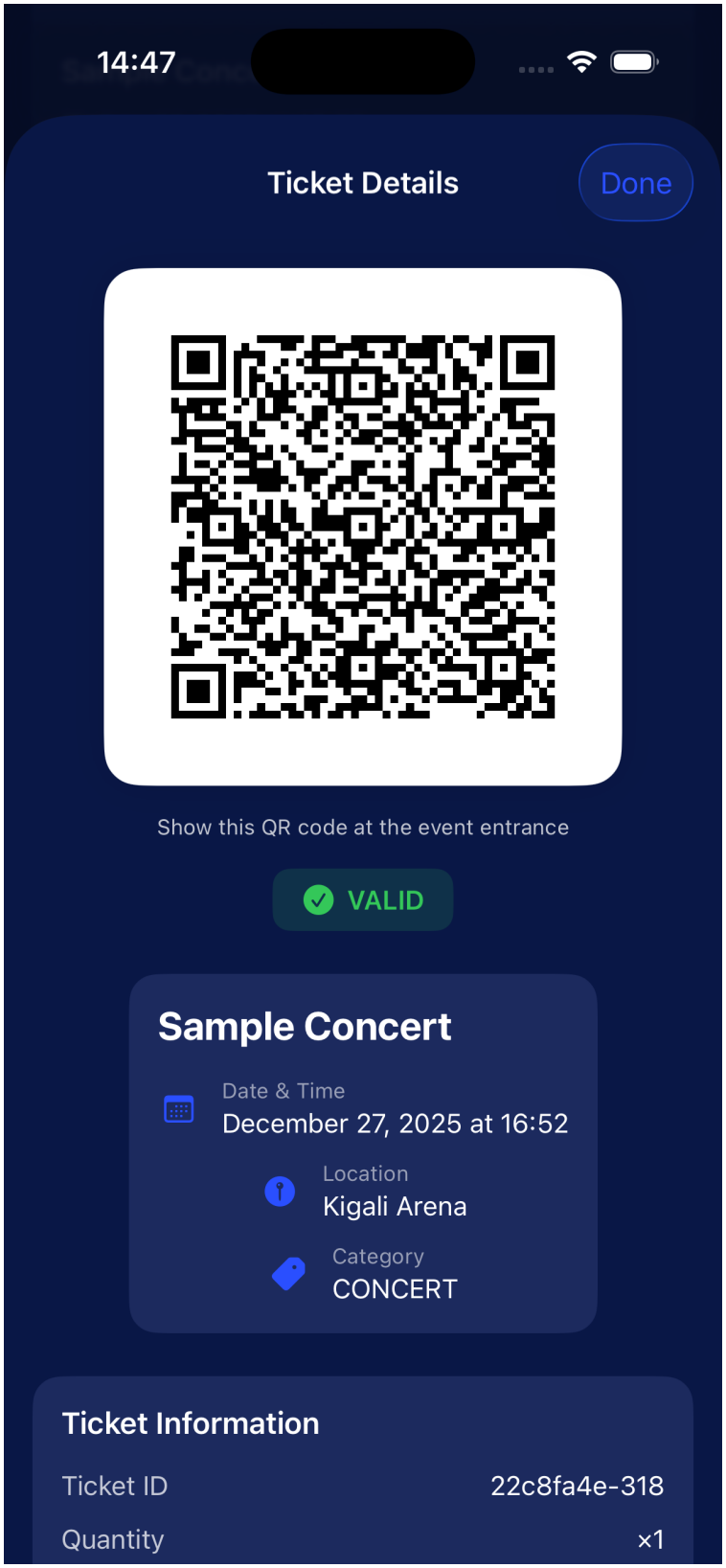

Figure 5: Ticket View Wireframe