EVENTPASS UG

IMPLEMENTATION AUDIT REPORT

Audit Date: February 18, 2026

Platform: iOS Native (Swift/SwiftUI)

iOS Target: iOS 16+

Total Files Analyzed: 128 Swift files

Auditor: Claude Code (Senior Software Architect)

EXECUTIVE SUMMARY

EventPass UG is a well-architected iOS mobile application for event ticketing with dual-role support (Attendees & Organizers). The app features comprehensive UI/UX implementation with all backend services using mock implementations. The codebase follows clean architecture principles (MVVM + Repository Pattern) and is production-ready for backend integration.

MVP READINESS SCORE: 52/100

The app has exceptional UI/UX (95% complete) but lacks real backend integration, payment processing, and Firebase configuration. All repositories use mock implementations.

1. IMPLEMENTED FEATURES

1.1 User Authentication

Feature	Status	Evidence
Email/Password Sign In	MOCK	AuthRepository.swift:45-67
Email/Password Sign Up	MOCK	AuthRepository.swift:70-95
Google Sign In	MOCK/UI	AuthRepository.swift:98-115
Apple Sign In	MOCK/UI	AuthRepository.swift:118-135
Phone OTP Auth	MOCK	AuthRepository.swift:138-165
Guest Mode	REAL	ContentView.swift:35-48

1.2 Event Discovery & Browsing

Feature	Status	Evidence
Event List/Grid	REAL	AttendeeHomeView.swift
Category Filtering	REAL	AttendeeHomeViewModel.swift
Time-Based Filtering	REAL	Today/This Week/Month filters
Search Events	REAL	SearchView.swift
Recommendations	REAL	RecommendationRepository.swift

1.3 Ticket Purchase Flow

Feature	Status	Evidence
Ticket Type Selection	REAL	TicketPurchaseView.swift
Quantity Selector	REAL	Limited by availability
Payment Method Selection	REAL	MTN/Airtel/Card options
Payment Processing	MOCK	PaymentRepository.swift:110
Success Screen	REAL	TicketSuccessView.swift

1.4 QR Code System

Feature	Status	Evidence
QR Code Generation	REAL	CoreImage API
QR Code Display	REAL	QRCodeView.swift
QR Code Scanning	REAL	AVFoundation

Ticket Validation	LOCAL	TicketRepository.swift

2. ORGANIZER FEATURES

2.1 Event Management

Feature	Status	Evidence
Event Creation Wizard	REAL	CreateEventWizard.swift
Edit Event	REAL	Uses same wizard
Delete Event	REAL	EventRepository.swift
Ticket Tier Setup	REAL	ManageEventTicketsView.swift
Save as Draft	LOCAL	EventRepository.swift

2.2 Dashboard & Analytics

Feature	Status	Evidence
Total Revenue	LOCAL	OrganizerDashboardView.swift
Tickets Sold	LOCAL	Calculated from local data
Event Analytics	MOCK	EventAnalyticsViewModel.swift
Withdraw Funds	UI ONLY	Button disabled

3. NOT IMPLEMENTED FEATURES

3.1 Critical Missing Features

Feature	Impact	Priority
Real Backend API	Cannot go live	CRITICAL
Payment Gateway	No real payments	CRITICAL
Firebase Auth	No persistent auth	CRITICAL
Push Notifications (FCM)	No remote notifs	HIGH
Cloud Image Storage	Images local only	HIGH

3.2 Missing Business Features

- Refund System - No refund request flow
- Promo Codes - No discount/coupon system
- Attendee List Export - No CSV/PDF export
- Event Cancellation Flow - No mass refund handling
- Multi-Currency - Only UGX supported
- Ticket Transfer - Cannot transfer to others
- Waitlist - No sold-out waitlist

4. TECHNICAL GAPS & RISKS

4.1 Critical Risks

RISK 1: No Backend Integration

All 10+ repositories use mock implementations. No HTTP networking layer exists. No API endpoint configuration. Impact: App cannot function in production.

RISK 2: No Real Payment Processing

Payment success is simulated (90% random). No Flutterwave/Paystack SDK integrated. No payment verification webhooks. Impact: Cannot process real transactions.

RISK 3: No Firebase Configuration

No GoogleService-Info.plist found. No FCM setup. Impact: No push notifications, no real authentication.

RISK 4: Insecure Data Storage

TestDatabase stores passwords in UserDefaults. No Keychain integration for tokens. No encryption at rest. Impact: Security vulnerability.

5. MVP READINESS SCORE BREAKDOWN

Category	Weight	Score	Weighted
UI/UX Completeness	20%	95/100	19.0
User Authentication	15%	20/100	3.0
Event Management	15%	85/100	12.75
Ticket Purchase Flow	15%	25/100	3.75
Payment Integration	15%	10/100	1.5
QR Code System	10%	90/100	9.0
Notifications	5%	60/100	3.0
Backend Integration	5%	0/100	0.0

TOTAL: 52/100

6. PRIORITY NEXT BUILDS

Phase 1: Backend Foundation (Week 1-2)

- Create HTTP Networking Layer (URLSession wrapper, base URL config)
- Set Up Firebase Project (GoogleService-Info.plist, Auth, FCM)
- Implement Real Authentication (Replace MockAuthRepository)

Phase 2: Payment Integration (Week 2-3)

- Integrate Flutterwave SDK
- Implement MTN MoMo STK push
- Implement Airtel Money STK push
- Set up webhook handlers for payment verification

Phase 3: Core Backend APIs (Week 3-4)

- Event API Integration (Replace MockEventRepository)
- Ticket API Integration (Replace MockTicketRepository)
- Image upload to cloud storage

Phase 4: Notifications & Analytics (Week 4-5)

- Configure APNs certificates and FCM
- Integrate SendGrid for email
- Integrate AfricasTalking for SMS (Uganda)

Phase 5: Polish & Security (Week 5-6)

- Migrate sensitive data to Keychain
- Implement certificate pinning
- Apple Wallet integration
- Security audit

7. CONCLUSION

EventPass UG is a professionally architected iOS application with comprehensive UI/UX but zero production-ready backend integration.

Strengths:

- Clean MVVM + Repository architecture
- Protocol-based dependency injection
- Comprehensive dual-role implementation
- Production-quality UI components
- Real QR code generation/scanning
- Well-defined domain models (18 models)

Critical Gaps:

- No real backend API
- No payment gateway integration
- No Firebase/authentication services
- All data persisted locally only

Estimated Time to MVP: 5-6 weeks with dedicated team

Recommended Team: 1 iOS Developer, 1 Backend Developer, 1 DevOps Engineer

8. FILE INVENTORY SUMMARY

Feature Screens (43 files)

Auth: 7 screens | Attendee: 9 screens | Organizer: 14 screens | Common: 17+ shared screens

Domain Models (18 files)

User, Event, Ticket, TicketType, OrganizerProfile, NotificationModel, UserInterests, UserPreferences, Payment models, Support models

Repositories (14 files)

Auth, Event, Ticket, Payment, Notification, UserPreferences, Location, Calendar, Recommendation, Analytics, and more

Core Utilities

QRCodeGenerator.swift, PersistenceController.swift, ServiceContainer.swift, AppStorageManager.swift

Report Generated: February 18, 2026

Auditor: Claude Code (Senior Software Architect)