Breem Advertising Screens System Guide

The Full Story from Four Perspectives

Release Date: 2025-09-27

Quick Introduction

This guide explains the workflow of the Breem Advertising Screens System (Back-End + Admin Dashboard + Android App for Screens) in a storytelling yet technically precise way. The goal is to enable any team member to understand both the big picture and the implementation details, as well as acceptance tests, from the very first read.

Roadmap

• Backend Perspective: What happens inside Laravel at each step (Handshake/Playlist/Heartbeat/Playbacks/Config).
• Admin Dashboard Perspective: How to upload, distribute, monitor, and report ads.
• Android Developer Perspective: How to implement API flow, caching, ETag, and offline handling.
• Android Screen User Perspective: The final uninterrupted experience.

At the end, checklists ensure production readiness for each party.

Backend Perspective (Laravel) — What happens behind the scenes?

Main Components:
• Models: Ad, AdSchedule, Screen, Place, PlaybackLog, ScreenLog.
• Services: AdSchedulerService (build playlist based on status/dates/linking + fallback + ETag + cache), ScreenApiService/HeartbeatService (Handshake/Heartbeat/Config), DeviceConfigService, PlaybackService.
• Observers: Clear playlist cache immediately after any impactful change (activate/schedule/link/unlink).
• Storage & URLs: Central URL resolver converting paths into public links (local via asset or S3 via Storage::url).
• ffprobe: If uploaded video duration = 0, attempt to read duration via ffprobe; otherwise return clear validation error.
• Scheduled Jobs: CheckScreenHealthJob (mark delayed devices as Offline + notify Mail/Slack webhook if configured), CheckExpiringAdsJob (update expired/expiring ads

status).
• Permissions: ads.*, screens.*, places.*, reports.*, settings.* tied to 'admin' guard.

Device API v1 Sequence

POST /api/v1/screens/handshake → Bind device, return screen_id + general config.
GET /api/v1/config (+ ETag) → Cacheable settings.
GET /api/v1/screens/{id}/playlist (+ If-None-Match) → Items + ETag or 304.
POST /api/v1/screens/heartbeat → Update last_heartbeat + log.
POST /api/v1/playbacks → Submit playbacks (single/batch).

Admin Dashboard Perspective — How campaigns are managed

1) Log in (with 2FA if required).
2) Create Ad: Upload file (video/image), enter title/description (EN/AR), set start and end dates. If video has no duration, system tries ffprobe; otherwise asks for manual input.
3) Send for approval → Approve/Reject. Once approved, activate/deactivate as needed.
4) Distribute Ad to specific screens with play_order and schedules per screen if required.
5) Monitor: Dashboard shows Online/Offline, last Heartbeat, alerts with Acknowledge option.
6) Reports: Performance (playbacks), Uptime reports exportable to CSV.

Android Developer Perspective (Android TV) — How the loop is built

Initialization:
• Generate stable device_uid (preferably ANDROID_ID + model suffix) and store.
• Retrofit + OkHttp Interceptor to add X-Device-UID and Accept: application/json.
• Room DB: Config, PlaylistItem, PlaybackQueue tables.
• ExoPlayer + SimpleCache: Play videos cached on disk; images/GIF handled via timer.
• Foreground Service to control loop, with Workers: Heartbeat/PlaylistRefresh/PlaybackFlush.

Playback Flow:
1) Handshake on boot (or when binding fails) → save screen_id/config.
2) Config (ETag) → update intervals.
3) Playlist (If-None-Match) → 200 (store+prefetch) or 304 (use cache).
4) Playback Loop: Video (onEnded) or Image/GIF (Timer), then send Playback (instant/batch).
5) Heartbeat every heartbeat_interval_sec (default 60s).
6) Offline: Keep playing from cache beyond TTL, queue playbacks, flush once network returns.
7) Smart Backoff: 5xx/Timeout → exponential; 429 → Retry-After; 401/403 → re-handshake.

Android Screen User Perspective — The final experience

• Continuous playback without interruptions, even if the network drops (cache).
• Quick reflection of new ads added/activated by admin.
• Fallback ensures no blank screens.

Quick Checklists

Backend:
• ETag changes on any impactful edit.
• Fallback always available.
• ffprobe duration checks clear.
• Health/alert jobs running; Slack optional.
• Permissions & roles active; Blade @can applied.

Dashboard:
• Upload/Approve/Activate/Distribute/Schedule flawless.
• Monitoring updates every minute + Acknowledge option.
• Reports export correctly (CSV).
• Bilingual + RTL fully functional.

Android:
• Interceptor always adds X-Device-UID.
• Playlist/Config handled with ETag/304.
• Prefetch + SimpleCache working.
• Playback queue + flush on network return.
• Stable Heartbeat; smart Backoff; UTC times.

Screen User:
• No visible interruptions.
• Updates appear automatically.
• Fallback prevents blank screens.