eShaman™ Full-Stack Mobile App & Landing Page – Premium Build Brief (v1.0)

Audience: AI code agents (Blackbox/MCP), senior engineers, designers, PMs.

Goal: Deliver a production-grade, premium mobile app (iOS/Android) + a high-impact landing page to begin list-building immediately. Theme: **prismatic crystals over mystic water**—glass, glow, and depth. Architecture is modern, secure, scalable, and friendly to autonomous code agents.

---

0) Executive Summary

- **App:** React Native (Expo + EAS) with a crystalline/glassmorphic UI, Oracle chat (OpenAI), tarot & astrology ritual flows, push notifications, subscriptions, content CMS, analytics, and secure backend with Firebase + Cloud Functions + Stripe/RevenueCat.
- **Landing Page:** Next.js (App Router) + Tailwind + Framer Motion (or Framer site) with crystal/water hero, email capture to Firebase/ConvertKit/SendGrid, scroll-jacking micro-interactions, social proof, and SEO.
- **Monorepo:** Turborepo with `apps/mobile`, `apps/landing`, `packages/ui`, `packages/config`, `packages/functions`, `packages/prompts`.
- **Dev Experience:** VS Code + MCP/Blackbox, strong ESLint/Prettier, Vitest/Jest, Detox, Playwright, GitHub Actions CI/CD, EAS builds, Vercel deploys. Secrets via `.env` + Secret Manager.

---

1) Tech Stack (Chosen for speed, quality, & agent-friendliness)

Mobile: Expo SDK 51+, React Native Reanimated 3, React Navigation 7, Gesture Handler, Skia, Lottie, Tamagui (or NativeWind), React Hook Form, Zod.

Backend: Firebase (Auth, Firestore, Cloud Functions, Cloud Storage, FCM/Notifications), Firebase App Check, Firebase Security Rules; optional GraphQL gateway (Apollo Server on Cloud Run) if needed for strict schema contracts.

AI: OpenAI API (chat/completions/image), prompt routers in Cloud Functions, safety guardrails.

Payments:

- **In-App:** RevenueCat (abstracts Apple/Google) for subscriptions.
- **Web:** Stripe Checkout + Customer Portal for landing page purchases; Stripe webhooks -> Firestore.

CMS: Sanity or Contentful for marketing/ritual copy & images.

Analytics & Quality: Firebase Analytics + Sentry (app & web); Crashlytics (optional); LogRocket (web).

Landing Page: Next.js 14+, Tailwind, Framer Motion (or Framer project), React Email for transactional templates.

CI/CD: GitHub Actions, Vercel (web), EAS (mobile), Firebase deploy (functions), Managed previews.

MCP/VSCode: Devcontainer, typed scripts, task runner, agent prompts (see §16).

---

2) Monorepo Layout (Turborepo)

```
/esh-monorepo
  /apps
    /mobile            # Expo app
    /landing           # Next.js landing
  /packages
    /ui                # Shared UI kit (Tamagui/NativeWind + RNW)
    /config            # tsconfig/eslint/prettier/tailwind configs
    /functions         # Firebase Cloud Functions (TypeScript)
    /prompts           # System & few-shot prompts (Oracle, Tarot, Astrology)
    /schemas           # Zod models, Firebase rule helpers
  /tools
    /scripts           # codegen, asset pipeline, CI helpers
  package.json
  turbo.json
  pnpm-workspace.yaml
```

---

3) Design System – “Crystal Over Mystic Water”

Visual Language:

- Glassmorphism panels on deep cosmic gradients + soft caustic highlights.
- Subtle volumetric fog, refractive glints, bokeh sparkle.
- Depth via parallax layers (water → mist → crystal shards → UI glass cards).

Color Tokens:

- `abyss` **#0A0F1F**, `indigo` **#1A1446**, `crystalBlue` **#8FD3FF**
- `auroraTeal` **#00D4FF**, `etherPurple` **#6A00F4**, `moonstone` **#DFE7FF**
- `roseQuartz` **#FFC1E3**, `amberBloom` **#FFD58A**, `emeraldGlint` **#00FFA3**
- Surfaces use `rgba(255,255,255,0.06–0.14)` with **1px** hairline borders `rgba(255,255,255,0.16)` and **20–36px** backdrop-blur.

Type:

- Headings: **Playfair Display** or **Cormorant Garamond** (serif elegance).
- UI Text: **Inter** or **General Sans**; Numerics: **IBMPlexMono** (accents).

Iconography: Lucide or Phosphor (thin/stroked), crystal-line accents.

Components:

- Glass Card, Orb Button (round with inner glow), Mystic Pill (segmented control with shimmer), Crystal Tabs, Chat Bubbles (right: auroraTeal glow; left: moonstone glow).

Motion & Micro-interactions:

- **Page transitions:** depth-fade + parallax (100ms translation + 200ms cubic Bezier ease), Reanimated shared elements between cards & details.
- **Water ripple shader:** Skia fragment shader on hero backgrounds; intensity responsive to touch/scroll.
- **Sparkle:** Lottie confetti shards; rate limited for perf.

---

4) Core Features (MVP++)

1. **Onboarding:** name, DOB, time, place (for natal), intentions, notification opt-in.
2. **Home:** Daily tarot card, lunar phase, affirmation, CTA to “Begin Ritual.”
3. **Oracle Chat:** AI guidance (OpenAI) with ritual-aware system prompt; saves transcripts; typing ellipsis + glow.
4. **Ritual Player:** step-by-step breathwork, visualization, soundscape; progress persists; timers & haptics.
5. **Library:** tarot spreads, meditations, lessons (from CMS).
6. **Profile & Subscriptions:** RevenueCat paywall, restore purchases; Stripe link account for web buyers.
7. **Notifications:** daily card, moon events, streaks; deep links to rituals.
8. **Admin (internal):** content push, feature flags, broadcast message.

---

5) Data Model (Firestore)

Collections:

- `users/{uid}`: profile (displayName, dob, birthTime, birthPlace, intents[], createdAt, marketingOptIn)
- `subscriptions/{uid}`: status (active, productId, platform, rcEntitlements, renewedAt)
- `readings/{uid}/{readingId}`: type (tarot/astro/oracle), prompt, result, tokens, createdAt
- `ritualRuns/{uid}/{runId}`: ritualId, stepsProgress, duration, completedAt
- `events/{eventId}`: broadcast/admin announcements
- `waitlist/{emailHash}`: email, source, createdAt

- `payments/{uid}/{paymentId}`: stripe metadata, amount, currency, status

Indexes: composite on `(uid, createdAt desc)` for readings, ritualRuns.

Security (sketch):

- Users read/write only their own docs; waitlist public write restricted via Cloud Function token or ReCAPTCHA/App Check. Admin via custom claims.

---

6) Optional GraphQL Gateway (Apollo)

Why: strict types for external agents; deterministic contracts.

SDL (excerpt):

```
scalar DateTime

type User { id: ID!, displayName: String, dob: String, birthTime: String,
birthPlace: String, intents: [String!] }

type Reading { id: ID!, userId: ID!, kind: String!, prompt: String!, result:
String!, createdAt: DateTime! }

type RitualRun { id: ID!, userId: ID!, ritualId: ID!, stepsProgress: [Int!]!,
durationSec: Int!, completedAt: DateTime }

type Query { me: User, myReadings(limit: Int=20, cursor: String): [Reading!]! }

type Mutation { createReading(kind: String!, prompt: String!): Reading!,
startRitual(ritualId: ID!): RitualRun! }
```

Resolvers talk to Firestore Admin SDK.

---

7) Cloud Functions (TypeScript)

- `ai/generateReading`: validates entitlement, builds system prompt (from `/packages/prompts`), calls OpenAI, stores transcript.
- `payments/createCheckoutSession`: Stripe Checkout for web; returns URL.
- `payments/stripeWebhook`: upserts `subscriptions` + `payments`.
- `notify/scheduleDailyCard`: cron; writes Notification tasks; sends via FCM.
- `waitlist/add`: verifies email + App Check; writes `waitlist`.
- `cms/revalidate`: Vercel/Sanity revalidation on publish.

---

8) Mobile App – Screens & Flows

Stack: Expo Router, Reanimated, Gesture Handler, Skia, Lottie, Tamagui/NativeWind.

Screens:

1. **Splash** (1.0s): crystal growth animation over water; logo subtle refract.
2. **Onboarding** (3–5 steps):
3. 1: Welcome (glass card, CTA)
4. 2: Personal data (DOB/time/place pickers with confetti accent)
5. 3: Intentions (chips with Mystic Pill interactions)
6. 4: Notifications opt-in; sample daily card preview
7. 5: Paywall (if gated features)
8. **Home:** Daily card (3D hover on tilt), moon phase ring, quick actions.
9. **Oracle Chat:** gradient background; chat bubbles; input with sigils; message streaming; save/ bookmark.
10. **Ritual Player:** full-bleed backdrop; steps with shared-element transitions; breaths synced to haptics.
11. **Library:** list → detail; shared glass card → full article; audio mini-player.
12. **Profile:** account, subscription state (RevenueCat), data export.

Transitions: depth-fade (200–300ms), glass blur ramp, parallax layers. Shared elements: card → details; avatar → profile. Gesture: swipe back with dampened spring.

Premium FX:

- **Skia water shader** on Home & Chat backgrounds (touch reactive).
- **Crystal bloom** (Lottie) on completion states.
- **Aurora sweep** shimmer over CTAs (infinite, 14s cycle).

---

9) Landing Page (Next.js) – “Crystal × Water”

Goals: Collect emails, tell the story, preview rituals, show credibility.

Sections:

1. **Hero:** full viewport water gradient (Canvas or video), foreground crystal PNG/GLTF with gentle drift; headline + sub; email capture (Firebase/ConvertKit/SendGrid); privacy note.
2. **Teaser video** (8–12s) crystal intro; autoplay muted inline.
3. **What you’ll feel:** three cards (Calm, Clarity, Synchronicity) with micro parallax.
4. **Oracle Chat demo:** animated chat bubble sequence.
5. **Rituals & Cycles:** lunar calendar strip + ritual tiles.
6. **Founder note / Ethos:** short para, tasteful portrait glow.
7. **FAQ, Footer, Social**.

Email Capture (Firebase): POST to `waitlist/add` with App Check token. Deduplicate via emailHash.

SEO: title/desc, OpenGraph, sitemap, robots, JSON-LD.

Accessibility: high contrast mode; prefers-reduced-motion respects.

---

10) Dependencies to Install

Root: `pnpm`, `turbo`, `typescript`, `eslint`, `prettier`, `lint-staged`, `husky`

Mobile app:

- `expo`, `react-native-reanimated`, `react-native-gesture-handler`, `@shopify/react-native-skia`, `lottie-react-native`, `expo-router`, `expo-notifications`, `expo-haptics`, `expo-secure-store`, `@react-native-community/netinfo`, `react-hook-form`, `zod`, `@tanstack/react-query`, `tamagui` (or `nativewind` + `tailwindcss`), `revenuecat/react-native-purchases`, `sentry-expo`, `@react-native-async-storage/async-storage`

Backend (functions): `firebase-admin`, `firebase-functions`, `openai`, `stripe`, `zod`, `dotenv`, `node-fetch`, `@google-cloud/secret-manager`, `pino`

Landing: `next`, `react`, `react-dom`, `tailwindcss`, `framer-motion`, `@vercel/analytics`, `@headlessui/react`, `zod`, `react-hook-form`, `@react-email/components`

Testing: `vitest` / `jest`, `@testing-library/react`, `detox` (mobile), `playwright` (web)

---

11) Env Vars (.env.*)

```
# OpenAI
OPENAI_API_KEY=

# Firebase client
NEXT_PUBLIC_FIREBASE_API_KEY=
NEXT_PUBLIC_FIREBASE_AUTH_DOMAIN=
NEXT_PUBLIC_FIREBASE_PROJECT_ID=
NEXT_PUBLIC_FIREBASE_STORAGE_BUCKET=
NEXT_PUBLIC_FIREBASE_MESSAGING_SENDER_ID=
NEXT_PUBLIC_FIREBASE_APP_ID=
NEXT_PUBLIC_FIREBASE_MEASUREMENT_ID=

# Firebase server
GOOGLE_APPLICATION_CREDENTIALS=/path/to/serviceAccount.json

# Stripe
```

```
STRIPE_SECRET_KEY=
STRIPE_WEBHOOK_SECRET=
NEXT_PUBLIC_STRIPE_PRICE_MONTHLY=
NEXT_PUBLIC_STRIPE_PRICE_YEARLY=

# RevenueCat
RC_PUBLIC_SDK_KEY_IOS=
RC_PUBLIC_SDK_KEY_ANDROID=

# Sentry
SENTRY_DSN=

# App
APP_URL_PROD=https://eshaman.io
APP_URL_STAGING=https://staging.eshaman.io
```

---

12) Build & Run (Local + CI)

Prereqs: Node 20+, pnpm, Java 17 (Android), Xcode (iOS, on macOS), Firebase CLI, Git.

Clone & bootstrap:

```
pnpm i -g turbo pnpm
pnpm install
pnpm -w dlx husky install
```

Mobile:

```
cd apps/mobile
pnpm expo prebuild # only if using native modules not handled by Expo
pnpm expo start
```

Landing:

```
cd apps/landing
pnpm dev
```

Functions:

```
cd packages/functions
pnpm build && firebase emulators:start
```

CI/CD: GitHub Actions: lint, typecheck, test; deploy `apps/landing` to Vercel; `packages/functions` via `firebase deploy`; EAS for mobile store builds.

---

13) Security & Compliance

- **App Check** for web & mobile; Firebase Security Rules (deny by default, allow own UID).
- **PII:** encrypt sensitive payloads at rest when possible; minimize retention.
- **Payments:** handle via Stripe/RevenueCat; never store card data.
- **AI safety:** content filters, disclaimers; allow user to report a reply.
- **Privacy:** GDPR/CCPA—data export & delete endpoints.

---

14) Analytics & Telemetry

- **Events:** onboarding_complete, reading_created(kind), ritual_started/finished(id), notification_opened, subscription_activated, churn, share_clicked.
- **Attribution:** UTM capture on waitlist; pass to user profile.
- **Sentry releases:** source maps, commit SHA tagging.

---

15) Content Pipeline (CMS)

- Schemas: `ritual`, `article`, `faq`, `mediaAsset`.
- Webhook → `cms/revalidate` → ISR pages update.
- Asset pipeline: compress PNG/WebP, generate LQIP blur data URLs.

---

16) Agent Brief (Paste into MCP/Blackbox)

Role: You are a senior full-stack engineer + designer. Build per the spec in this document. Ask no questions; choose sensible defaults.

Milestone-1 (Repo & Landing):

1. Initialize Turborepo per §2. Configure pnpm workspaces.
2. Scaffold `apps/landing` (Next.js) with Tailwind + Framer Motion + ESLint/Prettier.
3. Implement Landing per §9 with email capture hitting `waitlist/add` Cloud Function.
4. Add SEO basics, sitemap, OG images (crystal/water theme placeholders).
5. Vercel deploy + preview URL output.

Milestone-2 (Backend & Functions):

1. Initialize Firebase; set up Auth, Firestore, Storage, FCM; write base Security Rules.
2. Scaffold `packages/functions` with TS; implement endpoints from §7.
3. Configure Stripe keys + webhook; test with `stripe listen`.
4. Set up App Check; issue web token for landing.

Milestone-3 (Mobile App):

1. Scaffold Expo app; add libraries per §10; set Sentry and RevenueCat.
2. Implement screens per §8 with the transitions & effects described.
3. Wire Auth (Apple, Google, Email link). Sync subscriptions via RevenueCat.
4. Oracle Chat UI → calls `ai/generateReading`; persist to Firestore.
5. Notifications scheduling and deep links.

Milestone-4 (Polish & QA):

1. Accessibility pass; reduced-motion variants; dynamic type; RTL safe.
2. Performance: memoization, list virtualization, image caching, Reanimated on UI thread.
3. Tests: unit (Vitest), E2E (Detox/Playwright). Add GitHub Actions CI per §12.

Deliverables: compiling repos, deployed landing URL, Firebase project IDs, EAS profiles, store metadata JSON, test credentials.

---

17) Acceptance Criteria (MVP)

- Landing collects emails; entries visible in Firestore `waitlist`, deduped; 95+ Lighthouse perf.
- App boots in <2.5s on mid-tier Android; 60fps interactions; no redboxes.
- Oracle Chat produces responses in <4s median; transcripts saved.
- Subscriptions: purchase, restore, entitlement gating verified in sandbox.
- Notifications: daily card arrives within 5min of configured time; deep link works.
- CI/CD green; versioned releases; crash-free sessions ≥ 99.5% in pilot.

---

18) Example Code Stubs

Cloud Function handler (TS):

```
export const generateReading = onCall(async (req) => {
  const { kind, prompt } = z.object({ kind: z.string(), prompt:
z.string().min(8) }).parse(req.data)
  const uid = assertAuthed(req)
  await assertEntitled(uid, kind)
  const system = await loadPrompt(`system/${kind}.md`)
  const result = await openai.chat.completions.create({
```

```
    model: process.env.OPENAI_MODEL ?? 'gpt-4o-mini',
    messages: [{ role: 'system', content: system }, { role: 'user', content:
prompt }],
    temperature: 0.8,
  })
  const text = result.choices[0]?.message?.content ?? ''
  const doc = await
firestore.collection('readings').doc(uid).collection(uid).add({ kind, prompt,
result: text, createdAt: Date.now() })
  return { id: doc.id, result: text }
})
```

Landing – waitlist form action (Next.js, server action):

```
'use server'
import { z } from 'zod'

export async function joinWaitlist(prev: any, form: FormData) {
  const email = String(form.get('email') || '').trim()
  z.string().email().parse(email)
  const res = await fetch(process.env.FUNCTION_WAITLIST_URL!, {
    method: 'POST', headers: { 'content-type': 'application/json' },
    body: JSON.stringify({ email, source: 'landing' }),
  })
  if (!res.ok) return { ok: false, message: 'Try again.' }
  return { ok: true }
}
```

Expo Router – screen transition (Reanimated):

```
const config = { animation: 'timing', duration: 260 }
<Stack screenOptions={{ presentation: 'transparentModal', contentStyle: {
backgroundColor: 'transparent' }, animation: 'fade', animationDuration: 260 }}/>
```

---

19) Prompt Kits (packages/prompts)

/oracle/system.md (excerpt):

- Persona: Calm, luminous, grounded. Crystal-clear guidance. Avoid medical/financial absolutes; offer reflective prompts and rituals.
- Style: short paragraphs, present-tense invitations, gentle cadence.

/tarot/system.md:

- Use RWS deck references; upright/reversed; 1–3 card spreads; actionable insights.

/astrology/system.md:

- Use natal data if present; lunar/libration notes; practical advice windows.

---

20) Store Metadata (draft)

- **Name:** eShaman – Oracle & Rituals
- **Subtitle:** Clarity in your pocket.
- **Short desc:** Daily cards, lunar rituals, and an Oracle that listens.
- **Privacy URL:** https://eshaman.io/privacy
- **Marketing URL:** https://eshaman.io

---

21) Future Enhancements

- Social “circles” for shared rituals.
- Sound engine with spatial audio & binaural beats.
- Offline pack downloads.
- 3D crystals (GLTF) with Three.js/Expo-GL.

---

22) Quick Start Commands (copy/paste)

```
# 1) Create repo
npx create-turbo@latest esh-monorepo --package-manager pnpm

# 2) Add apps & packages (then install deps per §10)
# 3) Bootstrap Firebase + Functions, Stripe webhook
# 4) Run landing and mobile locally (see §12)
```

---

23) Visual Direction Notes

- Use **large glass cards** floating over deep water; soft noise, faint caustics.
- Crystal rim-light in etherPurple→auroraTeal gradient; specular streaks.
- Motion timing: 180–260ms transitions; 600–1200ms ambient loops.
- Respect **prefers-reduced-motion**; provide static background fallback.

End of Brief – build to this spec.

---

v1.1 Addendum — Tabs, Firestore Client, Framer Motion Hero

What changed:

- **Mobile:** Expo Router **Tabs** (`Home / Oracle / Rituals / Profile`) with a glassmorphic baseline and ready wiring for Firebase client init.
- **Shared:** `@esh/firebase-client` package (typed Firebase init for Auth + Firestore) to keep agents consistent across apps.
- **Landing:** Prebuilt **Framer Motion Crystal Hero** (parallax tilt, layered "crystal" shapes, email CTA block) ready to swap in assets.

Monorepo deltas:

```
/packages
  /firebase-client        # NEW: shared Firebase web/native init
/apps/mobile/app
  /(tabs)/_layout.tsx     # NEW Tabs container
  /(tabs)/home.tsx        # NEW Home stub
  /(tabs)/oracle.tsx      # NEW Oracle chat stub
  /(tabs)/rituals.tsx     # NEW Rituals stub
  /(tabs)/profile.tsx     # NEW Profile stub
  /config/firebase.ts     # NEW: uses @esh/firebase-client
/apps/landing/app/(components)/Hero.tsx  # NEW Framer Motion hero
```

New dependencies:

- Mobile: `firebase@^10`
- Packages: `@esh/firebase-client` (local), `@esh/schemas` already present
- Landing: already had `framer-motion`; v1.1 hero uses it.

Mobile env (Expo): Add these to `apps/mobile/app.json > expo.extra` (dev values shown):

```
{
  "firebaseApiKey": "YOUR_API_KEY",
  "firebaseAuthDomain": "YOUR_PROJECT.firebaseapp.com",
  "firebaseProjectId": "YOUR_PROJECT_ID",
  "firebaseStorageBucket": "YOUR_PROJECT.appspot.com",
  "firebaseSenderId": "000000000000",
```

```
  "firebaseAppId": "1:000000000000:web:abc123"
}
```

The client is initialized in `app/config/firebase.ts` and can be imported as:

```
import { auth, db } from "../config/firebase"
```

Expo Router tabs:

```
// app/(tabs)/_layout.tsx
<Tabs screenOptions={{ headerShown:false, tabBarActiveTintColor:'#00D4FF',
tabBarStyle:{ backgroundColor:'#0A0F1F' }}}>
  <Tabs.Screen name="home" options={{ title:'Home' }} />
  <Tabs.Screen name="oracle" options={{ title:'Oracle' }} />
  <Tabs.Screen name="rituals" options={{ title:'Rituals' }} />
  <Tabs.Screen name="profile" options={{ title:'Profile' }} />
</Tabs>
```

Landing hero usage: Import and render `Hero` in `app/page.tsx`. Replace the input/button block to post to your `waitlist/add` function or keep dev fallback.

Agent hand-off checklist (v1.1):

1. Install: `pnpm i` at repo root.
2. **Landing:** `cd apps/landing && pnpm dev`.
3. **Functions:** `cd packages/functions && pnpm build && firebase emulators:start`.
4. **Mobile:** `cd apps/mobile && pnpm start` (Expo).
5. Set Firebase keys in **Expo** `` ` and deploy Functions to get a real `` waitlist_add` URL.
6. Replace hero shapes with crystal renders/video and wire the form to `waitlist_add`.

Performance notes:

- Keep motion effects under 60fps by avoiding layout thrash (use `will-change: transform` equivalents and Framer Motion transforms).
- Respect `prefers-reduced-motion` on the landing hero.
