Sprint 7: Branding - Erweiterungen & Polishing – detaillierte Roadmap

Zeitraum: 22.–26. Sep 2025 (5 Arbeitstage)

Hauptziel: Nach dem Abschluss der Kernfeatures (Feedback, Offline Sync, 3D Heatmaps, Dashboard, Affiliate, Lizenzierung) konzentriert sich Sprint 7 auf den finalen Feinschliff: White-Label-Optionen ausbauen (Schriftarten, Icon-Sets, App-Name), Live Aktualisierungen per Firebase Remote Config implementieren, UI optimieren, Plan-Editor verbessern, umfassende End-to-end-Tests durchführen, Performance profilieren, letzte Bugs fixen und Materialien für das Pilotstudio und die App-Stores vorbereiten. Am Ende dieses Sprints soll die App bereit für den finalen Feinschliff in Sprint 8 sein.

1. Ausgangslage und Voraussetzungen

- **Bisherige Features:** Die App besitzt ein funktionsfähiges NFC-Tracking mit Offline-Sync, Gamification, Feedback & Surveys, 2D/3D-Heatmaps, ein Flatscreen-Dashboard, einen Affiliate-Shop und ein Lizenzsystem. Die Firestore-Regeln sichern Multi-Tenancy; alle wichtigen Flows sind implementiert.
- **Branding:** Der BrandingProvider verwaltet aktuell Logo, Primär- und Sekundär-Farben. White-Label-Funktionen für Schriftarten, App-Name und Icon-Set fehlen noch.
- **Remote Config:** Noch nicht genutzt – relevante UI-Parameter (Feature-Flags, Branding, Text) sollen zur Laufzeit aktualisierbar sein.
- **Plan-Editor:** Kann Pläne erstellen, kopieren, importieren und löschen, aber kein Undo/Wiederherstellen gelöschter Pläne.
- **Test-Infrastructure:** CI/CD ist vorhanden, Unit- und Widget-Tests für Kernprovider bestehen. End-to-End-Tests decken bereits NFC-Sync und Lizenzprüfung ab.

2. Tagesplanung (Tag → Aufgaben)

Tag 1 (Mo 22. Sep 2025) – Branding-Erweiterungen implementieren

Konzeption der neuen Branding-Optionen (Vormittag):

- **Schriftarten:** Liste kompatibler Google Fonts auswählen (z. B. Roboto, Open Sans, Montserrat) und als `fontFamily` in der `ThemeData` definierbar machen.
- **App-Name:** Struktur anpassen, damit der anzuzeigende Titel (Splash Screen, App-Bar, Info Dialog) aus dem Branding geladen wird. Fallback bleibt Tap-em.
- **Icon-Sets:** Zwei bis drei Icon-Varianten (z. B. Material Design, Line Icons, Flat Icons) als Asset-Bundles vorbereiten. Jeder Satz muss alle derzeit genutzten Icons abdecken.
- **Datenmodell:** Branding erweitern: neue Felder `fontFamily, appName, iconSet`. In der Firestore-Collection `gyms/{gymId}/branding` diese Felder optional anlegen.

Anpassung des BrandingProvider (Nachmittag):

- **Laden & Speichern:** Methoden `loadBranding` anpassen, um neue Felder zu lesen. Standardwerte setzen, falls Felder fehlen.
- **Update:** Cloud Function `updateBranding` erweitern, damit Admins neue Felder übermitteln können und im Storage hochgeladene Icon Sets speichern (z. B. Zip Archive mit Icons).

UI-Integration:

- **Branding-Screen:** Zusätzliche Formfelder für Schriftart (Dropdown), App-Name (Textfeld) und Icon-Set (Dateiupload oder Auswahl). Validierung und Speichern via `BrandingProvider.updateBranding`
- **App-Themes:** `main.dart` so anpassen, dass das Theme dynamisch die gewählte `fontFamily` und die Icons aus dem passenden Asset-Bundle nutzt.
- **Persistente Auswahl & Default:** Sicherstellen, dass beim Start der App die Branding-Einstellungen aus Firestore geladen und angewendet werden, bevor der Home-Screen angezeigt wird (z. B. Splash Screen, der auf Branding lädt).

Tag 2 (Di 23. Sep 2025) – Remote Config & UI-Polishing

Einrichtung von Firebase Remote Config (Vormittag):

- **Parameterdefinition:** Feature-Flags anlegen (`enableAffiliate, enableHeatmap3D, maxPlanDays, maintenanceMode`). Standardwerte definieren.
- **Client-Integration:** Abhängigkeit `firebase_remote_config` installieren und im `main.dart` initialisieren. Eine Methode `loadRemoteConfig` erstellen, die Config einmalig lädt und danach periodisch (alle 12 Stunden) aktualisiert.
- **Verwendung im Code:** Wichtige Stellen (Affiliate Button, 3D Heatmap Toggle, Plan Editor) so umbauen, dass sie sich an Remote Config orientieren.

UI-Feinschliff (Nachmittag):

- **Design Polish:** Vereinheitlichung von Abständen, Rand-Abschluss und Elevation; Buttons in `plan_editor.dart`, `report_screen_new.dart`, `challenge_screens` nach Material 3 Design.
- Farben aus dem Branding konsequent anwenden; 3D Heatmap verlinken mit Primär-/Akzentfarben.
- **Plan-Editor-Verbesserungen:**
  - Undo/Redo gelöschter Pläne: Beim Löschen eines Plans ein Archiv anlegen (`deletedPlans`) und Option zur Wiederherstellung in `PlanOverviewScreen` anzeigen.
  - RIR-Feld (Reps in Reserve) optional machen: Checkbox in UI, die das RIR-Eingabefeld ein- oder ausblendet.
  - Bessere Multitap-Prävention: In der `saveSession`-Methode boolean `isSaving` setzen, um doppeltes Speichern zu verhindern.
- **Persistente Nutzerpräferenzen:** `SharedPreferences` nutzen, um Einstellungen wie 2D vs. 3D Heatmap, Sprache, Dark-Mode und Plan-Editor-Optionen zu speichern und beim App-Start zu laden.

Tag 3 (Mi 24. Sep 2025) – Umfassende End-to-End-Tests

Testplanung & Skripterstellung (Vormittag):

- **Flows abdecken:** Registrierung/Onboarding, NFC-Scan & Session Erfassung, Offline Modus & Sync, XP Anstieg & Leaderboard, Feedback senden, Umfrage erstellen & abstimmen, Challenge starten & abschließen, Bestellung im Shop (Stripe Test), Lizenzprüfung, Dashboard View.
- **Testwerkzeug:** `integration_test` Package für Flutter; ggf. Cypress für das Web-Dashboard.

- **Vorlagen** für Setup und Tear Down erstellen (Test-User registrieren, Gym anlegen, Lizenz einrichten).

Implementierung der Tests (Nachmittag):

- Schrittweise Scripts schreiben, die jeden Flow simulieren:
  - **Reg./Login:** Füllen der Felder, Submission, Prüfen von role und Dashboard Zugang.
  - **NFC-Workflow:** GlobalNfcListener per Mock triggern; Sets erfassen, Offline Abbruch simulieren, späteres Reconnect testen.
  - **Feedback & Umfragen:** Buttons anklicken, Text/Bilder hochladen, Kategorien auswählen, Ergebnis Page prüfen.
  - **Affiliate:** Produkt zum Warenkorb hinzufügen, Checkout mit Testkartennummer (Stripe 4242424242), Bestellstatus prüfen.
  - **Dashboard & Lizenz:** Mit superadmin einloggen, Dashboard öffnen, Lizenzen einsehen.
- **Verwendung von** `mockito`/`fake_async` für Timer-Abhängigkeiten (Sync Queue, Rest Timer).

Automatisierung im CI:

- GitHub Actions anpassen, um `integration_test` mit `flutter test integration_test` und einem Headless Emulator/Driver auszuführen.
- Abbrecher definieren: Wenn ein Flow fehlschlägt, Ausgabe loggen und Workflow fehlschlagen lassen.

Tag 4 (Do 25. Sep 2025) – Performance Profiling & Bugfixing

Messung der App Performance (Vormittag):

- **Geräteauswahl:** Realistische Low-End-Geräte (z. B. Samsung A50, iPhone 7) + Mid-Range + High-End.
- **Profile & DevTools:** Flutter DevTools verwenden, um FPS, Memory Usage und CPU Spikes während intensiver Operationen zu messen (Heatmap 3D Rendering, Plan-Editor Scroll, Shop Checkout).
- **Netzwerk Analyser:** `Dio` oder das HTTP-Package mit Logging ausstatten; prüfen, ob Requests gebündelt oder unnötig oft ausgeführt werden (besonders für Remote Config, Surveys, XP Leaderboard).

Optimierungen implementieren (Nachmittag):

- **Rendering:** Lazy loading in Listen, Verwendung von `RepaintBoundary`, Vermeidung übermäßiger rebuilds.
- **Caching:** Bild Caching für Shop Produktbilder, Heatmaps und Icons verbessern.
- **Asynchrone Operationen:** Sicherstellen, dass teure Berechnungen (z. B. Heatmap Colors) im `compute` isolate ausgeführt werden.
- **Offline-Sync Queue:** Verhindern, dass die Queue im Hintergrund die UI blockiert; ggf. `WorkManager` verwenden.
- **Bugfixing:** Offene Bugs aus Issue Tracker (Crash bei Plan Import, ruckelnder Timer, falsche XP Anzeige) beheben.
- **Regressionstests:** Nach jeder Optimierung alle Unit-, Widget- und Integrationstests ausführen. Ein Profiling Report im Repository hinterlegen (z. B. `docs/performance_report_sep2025.md`).

Tag 5 (Fr 26. Sep 2025) – Onboarding & App-Store-Vorbereitung

Onboarding-Material (Vormittag):

- **Videos:** Kurze Screencasts (max. 2–3 Minuten) zu den wichtigsten Funktionen:
  - Einrichten des NFC-Kits im Studio.
  - Login & Onboarding für Mitglieder.
  - Anlegen eines Plans & Verwenden des Plan-Editors.
  - Nutzen des Dashboards (TV-Leaderboard).
- **PDF-Guides:** Schritt für Schritt (Setup Studio, Trainer Admin, Nutzung der App); in einfacher Sprache, mit Screenshots und QR-Links zu den Videos.

App-Store-Vorbereitung (Nachmittag):

- **Assets & Screenshots:** Mit allen Branding-Optionen (Dark Mode, 3D Heatmaps) schöne Screenshots erstellen (verschiedene iOS & Android-Auflösungen).
- **Beschreibung & Keywords:** App-Store-Texte entwerfen (Deutsch & Englisch), USP hervorheben (NFC-Tracking, Gamification, White-Label).
- **Datenschutz & Lizenzen:** Datenschutzhinweise auf den neuesten Stand bringen (Einbindung von Stripe, Remote Config).
- **Beta-Test via TestFlight & Google Play Console:** Neue Build hochladen, interne Tester einladen; Feedback sammeln.

Sprint-Abschluss & Planning für Sprint 8:

- **Review Meeting:** Alle neuen Branding-Optionen präsentieren, Ergebnisse der Performance-Tests teilen.
- **Retro:** Was hat funktioniert, was kann verbessert werden? Sammeln offener Bugs.
- **Backlog Grooming:** Nächste Schritte für Sprint 8 definieren (Restpolitur, Marketing Push, Pilotstudio Onboarding).

3. Definition of Done

- **Branding-Provider erweitert:** Felder für Schriftart, App-Name und Icon-Set in Firestore; Admins können diese über das Branding-Screen hochladen und ändern. Die App lädt die Einstellungen beim Start und zeigt sie korrekt an.
- **Remote Config aktiv:** Feature-Flags und Parameter werden via Firebase Remote Config verwaltet; das App-Verhalten passt sich ohne Update an.
- **UI-Feinschliff umgesetzt:** Vereinheitlichte Abstände, Farben, Buttons; Plan-Editor kann gelöschte Pläne wiederherstellen und RIR optional ein-/ausblenden. Multitap-Prävention funktionsfähig.
- **End-to-End-Tests laufen:** Vollständiger Test-Suite deckt alle Hauptflüsse ab; in CI integriert und stabil.
- **Performance optimiert:** Profiling auf mehreren Geräteklassen zeigt akzeptable FPS (>45 fps), niedrigen Speicherverbrauch und schnelle Ladezeiten; identifizierte Bottlenecks wurden behoben.

- **Bugs gefixt:** Alle kritischen Issues aus dem Tracker sind geschlossen; Regressionstests bestehen.
- **Onboarding-Material erstellt:** Videos und PDF-Guides sind im Repository und für Pilotstudios abrufbar.
- **App-Store-Assets vorbereitet:** Screenshots, Beschreibung, Datenschutzerklärung; Beta-Testing Build hochgeladen.
- **Sprint-Review durchgeführt:** Stakeholder bestätigen, dass die App bereit für letzte Politur in Sprint 8 ist.

4. Ausblick auf Sprint 8 (Finale Politur & Pilot-Launch)

Nach diesem Sprint wird Sprint 8 (29. Sep – 3. Okt 2025) die letzten offenen Punkte adressieren: letzte Bugs, Finalisierung der Marketing Assets, finales Beta-Feedback einarbeiten, Schulung des Pilotstudios durchführen, Datenschutz Review und Start der Pilotphase. Danach soll Tap-em offiziell in die ersten Studios ausgerollt werden.