Sprint 4: Flatscreen-Dashboard – detaillierte Roadmap

Zeitraum: 1.–5. Sep 2025 (5 Arbeitstage)

Ziel: Ein separates Dashboard entwickeln, das auf TVs/Browsern angezeigt werden kann, sich über ein Gym-Token authentifiziert, die Top-Rankings und Challenges live darstellt, regelmäßig aktualisiert wird und per QR-Code in der App erreichbar ist. Nach diesem Sprint ist das Team bereit, den Affiliate-Marktplatz (Sprint 5) zu realisieren.

1. Voraussetzungen und Kontext

- **Vorangegangene Sprints:** 3D-Heatmap und Gamification (Sprint 3) sind implementiert; Offline-Sync funktioniert stabil (Sprint 2); Feedback-Workflows abgeschlossen (Sprint 1).
- **Datenquellen:** Leaderboard-Daten stammen aus `gyms/{gymId}/users/{userId}/rank/stats` (XP pro Kategorie); Challenges liegen unter `gyms/{gymId}/challenges/{weekly|monthly}/items/{challen`
- **Sicherheitsanforderungen:** Dashboard darf nur Lesezugriff haben; muss per Token oder Signatur auf das betreffende Gym beschränkt sein (Firestore-Regeln berücksichtigen).
- **Team:** Frontend-Web-Entwickler (Flutter Web oder React), Backend-Entwickler für Token-Generation, Mobile-Entwickler für QR-Code-Integration, QA-Engineer.

2. Aufgabenplan (Tag-für-Tag)

Tag 1 (Mo 1. Sep 2025) – Technologiewahl & Architekturplanung

Flutter Web vs. React (AM):

- **Flutter Web Vorteile:** Code-Wiederverwendung, einheitliches Styling mit Mobile-App, weniger Kontextwechsel; Animationen und Charts lassen sich mit vorhandenen Flutter-Widgets (z. B. `charts_flutter` oder `CustomPainter`) implementieren.
- **React Vorteile:** Schlanker, natives Web-Ecosystem, große Auswahl an Chart-Bibliotheken (Chart.js, Recharts); einfache Integration von Service Workers für Offline-Caching.
- **Entscheidungskriterien:** Team-Erfahrung, Deployment-Strategie, Performance auf Smart-TVs.
- **Vorschlag:** Flutter Web einsetzen, da Komponenten (Icons, Farben, Charts) aus der Haupt-App wiederverwendet und Theming konsistent angewandt werden können; zudem reduziert es die Zahl der Technologien im Projekt.

High-Level-Architektur (PM):

- **App-Struktur:** Eine Single-Page-App mit drei Hauptansichten:
  - `LeaderboardPage` (Tabs für „Gerät“, „Trainingstag“, „Muskelgruppe“).
  - `ChallengesPage` (Liste aktiver und kommender Challenges mit Fortschritt).
  - (Optional) `SettingsPage` (Auto-Refresh-Intervall, Full-Screen-Toggle).
- **Routing:** URL-Schema `/dashboard?token=XYZ`; der Token identifiziert das Gym (z. B. signierter JWT).
- **Auth-Mechanismus:** Das Dashboard liest den Token aus der URL, validiert ihn via Firebase Auth oder einer Cloud-Function, ruft anschließend nur Daten dieses Gyms ab.

- **Auto-Refresh:** Shared Provider (z. B. `RefreshProvider`) mit `Timer.periodic(30 Sekunden)`, der Firestore-Abfragen auslöst und UI aktualisiert.

Token-Generierung (Design):

- Erstellen einer Cloud-Function `generateDashboardToken(gymId)`; sie gibt ein JWT zurück, das `gymId` und ein Ablaufdatum (`exp`) enthält.
- Token wird mit Firebase Admin SDK signiert und kann vom Dashboard verifiziert werden.
- Firestore-Regeln ergänzen: `request.auth.token.gym_id == gymId` und `request.auth.token.role == 'dashboard'` erlauben nur Lesezugriff.

Tag 2 (Di 2. Sep 2025) – Implementierung des Dashboards

Projektsetup (AM):

- Neues Flutter-Project `dashboard_app` im Repository anlegen (unter `apps/dashboard/`).
- `flutter create .` und anschließend Abhängigkeiten hinzufügen: `cloud_firestore`, `firebase_core`, `flutter_countdown_timer` (optional), `provider` (für State-Management), `qr_flutter` (für später).
- Firebase-Web-Konfiguration hinzufügen (`firebase_options_dashboard.dart`).

Auth & Token-Validierung (AM):

- Beim Starten der App: Query-Parameter `token` aus der URL lesen (`Uri.base.queryParameters['token']`).
- Token mit `firebase_auth` via `signInWithCustomToken()` validieren oder alternative JWT-Bibliothek verwenden, um die Signatur zu prüfen.
- Nach erfolgreicher Auth: Extrahierte `gymId` im Provider speichern und an alle Firestore-Abfragen übergeben.
- Bei Fehler (Token ungültig/abgelaufen): Fehler-Seite anzeigen.

Leaderboard-Ansicht (PM):

- **Datenabfrage:**
  - Für XP pro Gerät: Query an `gyms/{gymId}/users/*/rank/stats` und Sortierung nach `deviceXP` (analog für `dailyXP` und `muscleXP`).
  - Aggregation: Da Firestore keine serverseitige Sortierung über Subcollections unterstützt, Daten clientseitig zusammenführen (max. 50 User für Top 10).
  - Optional: Cloud Function `getTopLeaderboard(gymId, category, limit)` implementieren, um Ranking serverseitig zu berechnen und Traffic zu minimieren.
- **UI-Darstellung:**
  - TabBar mit drei Tabs („Gerät“, „Trainingstag“, „Muskelgruppe“).
  - In jedem Tab: Liste mit Top-10-Einträgen (Rang, Username, XP).
  - Graphische Balken oder Medaillen zur Visualisierung der Abstände.

Challenges-Ansicht (PM):

- Firestore-Abfragen für aktive ($\texttt{start} \leq now \leq end$)$und kommende Challenges. Fortschritt berechnen$: $Fr jede Challenge und jeden User$:
- • Abfrage `completedChallenges` und log-Zähler (analog Cloud Function aus Sprint 2).

- Darstellung als Prozentbalken (z. B. `minSets` vs. `logCount`).

UI: Sortierung nach Enddatum; Unterscheidung wöchentlich vs. monatlich.

Auto-Refresh-Mechanismus:

- Globaler Timer (`Timer.periodic(Duration(seconds: 30))`) in einem Provider; ruft `fetchLeaderboard()` und `fetchChallenges()` erneut auf.
- Laden-Indicator oder dezenter „Letzte Aktualisierung: HH:MM:SS"-Text anzeigen.
- Sicherstellen, dass die Firestore-Kosten (Lesekontingent) im Rahmen bleiben, ggf. Caching einbauen und nur Delta-Updates abrufen (z. B. über Snapshot-Listener).

Tag 3 (Mi 3. Sep 2025) – Fullscreen-Modus & QR-Code-Integration

Fullscreen-Funktion (AM):

- Ein Button oder Menü in der Dashboard-UI, der `html.document.documentElement?.requestFullscreen()` aufruft (nur Flutter Web).
- Esc-Taste oder „Vollbild verlassen"-Icon implementieren.
- Fallback für Browser, die kein Fullscreen unterstützen (Hinweis anzeigen).

QR-Code-Generierung (PM):

- **Token-Bereitstellung:**
  - Im Admin-Dashboard (Mobile-App, z. B. `AdminDashboardScreen`) einen neuen Menüpunkt „TV-Dashboard" hinzufügen.
  - Beim Öffnen: API-Call an Cloud-Function `generateDashboardToken(gymId)` ausführen; URL generieren (`https://dashboard.tapem.com/?token=TOKEN`).
- **QR-Code Anzeigen:**
  - Verwendung von `qr_flutter` oder `barcode_widget` im Flutter-Mobile-App, um QR-Code aus der generierten URL zu erzeugen.
  - Beschriftung: „Scanne diesen Code mit dem TV-Browser oder Mobile-Browser".
  - Möglichkeit zum Kopieren der URL.
- **Sicherheit:**
  - Token läuft nach definierter Zeit (z. B. 24 h) ab; im UI anzeigen, wie lange der Code gültig ist; Option „neuen Code generieren".

Firestore-Regeln für Dashboard:

- Erweiterung: Rolle `dashboard` erhält nur Lesezugriff auf Gym-Daten; keine Schreibrechte.
- Regelbeispiel:

```
match /gyms/{gymId}/{document=**} {
  allow read: if request.auth.token.gym_id == gymId && request.auth.token.role == '
    ↪ dashboard';
  allow write: if false;
}
```

- QA-Engineer testet den Zugriff im Emulator: Token mit falscher `gymId` sollte keinen Zugriff bekommen.

Tag 4 (Do 4. Sep 2025) – Quality Assurance & Optimierung

End-to-End-Tests des Dashboards (AM):

- Manuelles Aufrufen des Dashboards im Browser/TV mit gültigem und ungültigem Token.
- Test der Auto-Refresh-Funktion: Daten sollten sich nach 30 Sekunden ohne manuelles Reload aktualisieren.
- Test der Vollbild-Funktion: Wechsel in den Fullscreen-Modus und zurück.
- Test, dass beim Ablaufen des Tokens eine Fehlermeldung und kein weiterer Datenzugriff erfolgt.

Leistungstests (PM):

- Messung des Render-Verhaltens der Top-10-Listen auf Smart-TV-Browsern; Beobachtung von Latenz und speicherintensiven Operationen (z. B. durch Developer Tools).
- Optimierung der Datenabfragen (Verwendung von `limit(10)`, `orderBy`) und Reduzieren redundanter Re-Renders (Verwendung von `Consumer`/`Selector` in Flutter).

Feintuning der UI:

- Anpassung an verschiedene Bildschirmgrößen (1080p, 4K); Vergrößern der Schrift und Balken für TV-Distanz.
- Sicherstellen, dass Farben und Branding des Gyms übernommen werden (Branding-Provider).
- Platzieren von Gym-Logo und Last-Updated-Zeitpunkt dezent auf dem Bildschirm.

Tag 5 (Fr 5. Sep 2025) – Dokumentation, Review & Übergabe

Dokumentation ergänzen:

- Erstellung von `docs/dashboard_setup.md`: Beschreibung des neuen Projektes, wie das Token generiert wird, Aufbau der URL, Deployment-Anweisungen (Firebase Hosting oder alternativ AWS/S3 + CloudFront).
- Erklärung des Auth-Flows mit JWT, Ablaufdatum und Firestore-Regeln.
- Hinweise zur Anpassung von Farben/Branding im Dashboard (Integration mit BrandingProvider).

Code-Review & Merge:

- Frontend- und Backend-Entwickler führen einen ausführlichen Review des Dashboard-Projekts durch.
- QA-Engineer bestätigt, dass Tests bestehen; alle Sicherheitsaspekte sind berücksichtigt.
- Nach dem Review wird das Dashboard in `agent` gemerged; GitHub Actions erzeugt einen Build-Artifact für Hosting.

Deployment & internes Testing:

- Deployment auf eine Test-URL (`dashboard-dev.tapem.com`) über Firebase Hosting oder Vercel.
- Interne Stakeholder testen die Anwendung mit ihren Gym-Daten.
- Feedback sammeln und in Backlog für Sprint 5 aufnehmen.

Sprint-Review & Planung Sprint 5:

- Präsentation des Dashboards auf einem TV-Gerät; QR-Code in der App scannen und Dashboard starten.
- Diskussion offener Punkte: zusätzliche Statistiken, farbliche Darstellung von Challenges; Notizen für Affiliate-Marktplatz definieren.
- Festlegung, welche Features im Sprint 5 (Affiliate-Marketplace) integriert werden müssen und ob kleinere Verbesserungen am Dashboard noch notwendig sind.

3. Definition of Done

- **Dashboard implementiert:** Ein eigenständiges Flutter-Web-Dashboard, das sich per Gym-Token authentifiziert und nur Daten des jeweiligen Gyms liest.
- **Live-Leaderboards & Challenges:** Top-10-Rankings pro Kategorie und aktuelle/kommende Challenges inklusive Fortschritt werden korrekt angezeigt.
- **Auto-Refresh & Full-Screen:** Die Ansicht aktualisiert sich automatisch alle 30 Sekunden; der Nutzer kann in den Full-Screen-Modus wechseln.
- **QR-Code bereitgestellt:** In der Mobile-App generiert der Admin einen zeitlich limitierten Token; QR-Code leitet auf die Dashboard-URL.
- **Sicherheit gewährleistet:** Token sind signiert, Firestore-Regeln erlauben nur lesenden Zugriff, falsche oder abgelaufene Token werden abgewiesen.
- **Tests bestanden:** End-to-End- und Leistungstests zeigen keine kritischen Probleme; der Code ist reviewed, dokumentiert und in die Haupt-Branch gemerged.

4. Ausblick auf Sprint 5

Nach dem erfolgreichen Dashboard-Launch kann sich das Team im Sprint 5 (8.–12. Sep 2025) dem Affiliate-Marktplatz widmen. Der Marktplatz wird auf dem Grundgerüst des Dashboards aufbauen (z. B. Widgets für Produktlisten können in React/Flutter Web wiederverwendet werden) und sorgt für zusätzliche Einnahmequellen.