Sprint 3: 3D-Heatmap & Gamification-Enhancements – detaillierte Roadmap

Zeitraum: 25.–29. Aug 2025 (5 Arbeitstage)

Ziel: Die Muskelgruppen-Darstellung auf eine dreidimensionale Ebene heben und zusätzliche Anreize durch Gamification schaffen. Die Intensitätswerte sollen jetzt sowohl auf ein 3D-Modell übertragen als auch weiterhin in 2D angezeigt werden können. Das System soll auf älteren Geräten performant laufen und neue Erfolge (Badges) ausgeben.

1. Voraussetzungen und Rahmen

- **Abgeschlossene Vorarbeiten:** Offline-Sync funktioniert stabil (Sprint 2). Die 2D-Heatmap ist vorhanden und der `MuscleGroupProvider` kann Intensitätswerte berechnen. Ein Basismodell für die 3D-Heatmap existiert als Platzhalter mit TODO in der Codebasis.
- **Technische Basis:** Flutter 3.x, Dart 3.x; Assets-Ordner mit 3D-Modelldateien und Texture-Maps; `MuscleRegion` Enum definiert; intensitätsbasierte Werte pro Muskelgruppe werden vom Provider bereitgestellt.
- **Team:** Zwei Frontend-Entwickler, ein 3D-/Graphics-Spezialist, ein QA-Engineer.

2. Aufgabenplan (Tag-für-Tag)

Tag 1 (Mo 25. Aug 2025) – Bibliotheks-Evaluierung & 3D-Modell-Vorbereitung

Auswahl der 3D-Bibliothek (AM):

- Evaluierung gängiger Flutter-3D-Packages:
  - `flutter_cube`: einfache Import-Unterstützung für Wavefront (.obj) und GLTF; geringerer Overhead.
  - `flutter_gl` / `three_dart`: tiefere Kontrolle, besseres Shader-Management; allerdings höhere Komplexität.
- Kriterien: Performance auf mobilen Geräten, Unterstützung für dynamisches Färben von Mesh-Segmenten, Lizenz (MIT/BSD).
- Entscheidung dokumentieren und im Team abstimmen (z. B. `flutter_cube` als einfacher Startpunkt).

Vorbereitung des 3D-Modells (PM):

- 3D-Designer (oder offen lizenzierte Ressource) liefert ein menschliches Ganzkörper-Modell, das in einzelne Muskelgruppen (Segmente) unterteilt ist – vorzugsweise als GLTF.
- Erstellen einer Map `muscleRegion -> meshPartName`, die eindeutig den Mesh-Teilen zugewiesen wird (z. B. `"pectoralis_major> Region Chest`).
- Ablage der Modelldatei im Verzeichnis `assets/models/`; Sicherstellen, dass sie im `pubspec.yaml` korrekt eingebunden ist.
- Platzhalter & Datenstruktur anpassen:
  - Falls nötig, `AdvancedBodyHeatmap3D` bzw. `BodyHeatmap3D` Widget vorbereiten, das das 3D-Modell lädt, aber noch keine Farben setzt.
  - Datenstruktur im `MuscleGroupProvider` ergänzen, um 3D-Meshes referenzieren zu können (z. B. Map von `MuscleRegion` zu Farbwert).

Tag 2 (Di 26. Aug 2025) – Integration des 3D-Models & Farbzuordnung

Implementierung des 3D-Viewers:

- Im neuen Widget `BodyHeatmap3D` (in `features/muscle_group/presentation/widgets/`) die gewählte 3D-Bibliothek initialisieren.
- Laden des GLTF-Modells; Prüfen, ob das Modell korrekt gerendert wird.
- Kamera-Position und Zoom definieren (Front-Ansicht, optionale Rotation per Touch-Gesten).

Farbzuweisung basierend auf Intensität:

- Methode `updateColors(Map<MuscleRegion, double> intensities)` in `BodyHeatmap3D` implementieren.
- Zu jedem `meshPartName` die Intensität aus `MuscleGroupProvider` lesen und anhand definierter Schwellen (`>0.66 -> primaryColor`; `>0 -> secondaryColor`; sonst muted) die Farbe setzen (ähnlich 2D).
- Falls die verwendete Bibliothek Shader-Änderungen unterstützt: Änderung der Material-Farbe für das Mesh-Segment; ansonsten semitransparente Farb-Overlays.

Synchronisation zwischen 2D & 3D:

- `MuscleGroupScreenNew` so erweitern, dass es das 3D-Widget instanziiert und die Intensitätsdaten vom Provider an beide Widgets (2D und 3D) gleichzeitig weitergibt.
- Sicherstellen, dass beim Wechsel der Ansicht (siehe Tag 3) die aktuellen Intensitätswerte persistent bleiben und nicht neu berechnet werden müssen.

Tag 3 (Mi 27. Aug 2025) – Umschaltmechanismus & Benutzereinstellungen

UI-Schalter implementieren:

- In `MuscleGroupScreenNew` (oder Muskelgruppen-Tab) oberhalb des Heatmap-Bereichs einen Toggle (z. B. `ToggleButtons` oder `SegmentedControl`) mit den Optionen „2D“ und „3D“ anzeigen.
- Auswahl soll das jeweilige Widget (2D oder 3D) rendern, ohne die Navigationsebene zu verlassen.

Speichern der Nutzerpräferenz:

- Verwendung von `shared_preferences` oder `hive` für lokale Persistenz: Schlüssel `preferredHeatmapMode` mit Wert `2d` oder `3d`.
- Beim Öffnen der `MuscleGroupScreen` wird die Einstellung gelesen und der Toggle entsprechend gesetzt.
- Falls ein Gerät zu leistungsschwach ist (siehe Performance-Tests), Override auf 2D.

Gamification-Hook:

- Beim erstmaligen Wechseln auf die 3D-Ansicht eine Meldung „Neues Feature entdeckt – +100 XP!“ anzeigen und `ChallengeProvider`/`XpProvider` eine entsprechende XP-Belohnung (Badge „3D-Explorer“) hinzufügen.
- Ein neues Badge-Dokument im Nutzerkonto anlegen (z. B. `badges/{badgeId: '3d_explorer'}`).

Tag 4 (Do 28. Aug 2025) – Performance-Tests & Optimierungen

Messung der Render-Performance:

- Auf älteren Geräten/Emulatoren (z. B. iPhone 8, Android Nexus 5X) das 3D-Widget via `flutter drive` testen.
- FPS (Frames per Second) messen; Speicherverbrauch beobachten (z. B. mit `dart_devtools`).
- Ziel: $>$ 30 FPS bei Nutzung der 3D-Heatmap; wenn niedriger, alternative Strategien prüfen (Mesh-Simplification, Reduzieren der Polygone).

Optimierungen:

- Reduzieren der Polygonanzahl des 3D-Modells; eventuell Level-of-Detail (LOD) bereitstellen.
- Shader-Effekte abschalten oder vereinfachen (kein per-fragment Lighting, Nutzung von unlit Materialien).
- Lazy-Loading: Teile des Modells erst nach Bedarf laden (Front-/Back-Side).
- Preloading im Hintergrund, damit der UI-Thread nicht blockiert.

Fallback-Mechanismus:

- Wenn das Gerät die Performance-Grenzwerte nicht erfüllt, 3D-Option deaktivieren und den Nutzer informieren („3D-Heatmap nicht unterstützt, 2D-Ansicht wird verwendet“).

Gamification-Erweiterungen:

- Evaluieren weiterer Gamification-Ideen: z. B. „Heatmap-Master“ Badge für 7 Tage in Folge mit vollständiger Muskelabdeckung; diese Ideen notieren, aber primär implementieren wir nur den 3D-Explorer.

Tag 5 (Fr 29. Aug 2025) – Tests, QA & Abschluss

Unit-Tests für MuscleGroupProvider erweitern:

- Test, ob `computeIntensity` (oder gleichnamige Methode) weiterhin korrekt für jede `MuscleRegion` Werte zwischen 0 und 1 liefert.
- Test, ob die Farbzuordnung (z. B. $>$0.66) aus dem Provider an das 3D-Widget übergeben wird.
- Test, dass die Map `muscleRegion -> meshPartName` vollständig ist (alle Regions abgedeckt).

Widget-Tests für Umschaltfunktion:

- Mock `SharedPreferences` und prüfen, dass der Toggle den Modus wechselt und die Präferenz schreibt.
- Test, dass beim erstmaligen Wechsel in die 3D-Ansicht die Badge-Vergabe ausgelöst wird (z. B. `XpProvider.addBadge` wird aufgerufen).

Manual QA & Regressionstest:

- QA-Engineer testet die 3D-Heatmap mit verschiedenen Intensitätsszenarien (Leerer Tag, niedrige Intensität, volle Intensität) und vergleicht die Farben mit der 2D-Ansicht.
- Test der Gamification-Belohnung: Mehrere Nutzer wechseln zwischen den Ansichten; Badge darf pro Nutzer nur einmal vergeben werden.

- Performance-Test wiederholen, nachdem Optimierungen integriert wurden.

Code-Review & Dokumentation:

- Gemeinsamer Review von 3D-Integration, Prefs-Speicherung, Badge-Logik; Einhaltung der Architekturrichtlinien und Patterns.
- Aktualisierung der Dokumentation (`docs/heatmap.md`):
  - Beschreibung des 3D-Modells, der verwendeten Bibliothek und der Funktionsweise der Farbkodierung.
  - Hinweise zu Performance-Anforderungen und Fallback-Strategie.
- README mit einem Screenshot der neuen 3D-Heatmap ergänzen.

Sprint-Review & Übergabe an Sprint 4:

- Demo: 2D/3D-Toggle, Anzeige der Intensitäten, Gamification-Badge.
- Offene Punkte identifizieren: Nicht umgesetzte Gamification-Ideen, potenzielles Feintuning der Farbschwellen.
- Vorbereitung auf Sprint 4 (Flatscreen-Dashboard): Klären, welche Daten aus der Heatmap im Dashboard sinnvoll sind (nur XP oder auch Heatmap-Statistiken?).

3. Definition of Done

- **3D-Heatmap integriert:** Ein geladenes 3D-Modell mit segmentspezifischer Farbe, basierend auf aktuellen Intensitätswerten, wird in der App dargestellt.
- **Umschaltmechanismus vorhanden:** Nutzer können in der Muskelgruppen-Seite zwischen 2D- und 3D-Ansicht wechseln; ihre Präferenz wird gespeichert.
- **Performance gesichert:** Die 3D-Ansicht läuft auf den getesteten Geräten flüssig; auf leistungsschwachen Geräten wird die 3D-Option deaktiviert.
- **Gamification:** Erster 3D-Heatmap-Aufruf verleiht ein Badge („3D-Explorer“); erneute Aufrufe vergüten kein weiteres Mal.
- **Tests:** Unit-Tests und Widget-Tests decken neue Logik ab; Integrationstests bestätigen, dass Toggle und Badge-Logik funktionieren.
- **Dokumentation:** Aktualisierte Heatmap-Dokumentation mit Modellbeschreibung, Farbschwellen und Performance-Hinweisen.
- **Review bestanden:** Code wurde reviewed, in `agent` gemerged und ist für Sprint 4 bereit.

4. Ausblick auf Sprint 4

Nach der erfolgreichen Einführung der 3D-Heatmap liegt der Fokus im nächsten Sprint (1.–5. Sep 2025) auf dem Flatscreen-Dashboard: Eine Web- oder Flutter-Web-Anwendung soll das Leaderboard und die Challenges live anzeigen. Die im Sprint 3 gewonnenen Erfahrungen mit Performance und Daten-Mapping werden hilfreich sein, um die Darstellung im Browser zu optimieren.