Roadmap zur Umsetzung des CSV-Export/Import Features

Ihr Entwicklungsteam

22. April 2025

Zielsetzung

Ziel dieser Roadmap ist die schrittweise Planung und Durchführung der Entwicklung eines robusten CSV-Export/Import-Features für Trainingspläne in der *Tap n' Track*-App. Coaches sollen ihre Pläne als CSV exportieren, extern bearbeiten und anschliessend wieder importieren können, um ihren Klienten angepasste Pläne bereitzustellen.

1 Phasenübersicht

Phase 1: Vorbereitung & Analyse

Phase 2: Architektur & Design

Phase 3: Implementierung Export

Phase 4: Implementierung Import

Phase 5: UI/UX Integration

Phase 6: Testing & Qualitätssicherung

Phase 7: Dokumentation & Deployment

2 Detaillierte Aufgaben

Phase 1: Vorbereitung & Analyse

- **Anforderungsaufnahme**
  - Genaue Felddefinitionen (Übung, Satz, Wiederholungen, Gewicht).
  - CSV-Format (Spaltentrennzeichen, Header).
- **Technologieauswahl**
  - Dart-Packages: `csv`, `file_picker`, `path_provider`, `share_plus`.
  - Festlegen, wo CSV-Dateien temporär abgelegt werden.

- **API-Spezifikation**
  - Endpunkte /services/api_services.dart:
    - `getTrainingPlan(clientId)` (bestehende Pläne abrufen).
    - `updateTrainingPlan(clientId, planData)` (importierte Pläne speichern).

Phase 2: Architektur & Design

- **Seiten- und Komponentenstruktur**
  - Export/Import-Buttons in `CoachDashboardScreen`.
  - Separate Helper-Klasse: `CsvPlanService`.
- **Datenmodell**
  - Definition eines DTOs für Planzeilen: `class PlanEntry { String exercise; int sets; int reps; double weight; }`
- **Error-Handling-Konzept**
  - File-Picker-Abbrüche.
  - CSV-Parsing-Fehler (fehlende Spalten, ungültige Werte).
  - Netzwerkfehler beim Speichern.

Phase 3: Implementierung Export

1) **CsvPlanService.exportPlanToCsv(clientId)**
   - Abruf Plandaten via API.
   - Aufbau `List<List<String>>` mit Header + Zeilen.
   - Nutzung `ListToCsvConverter(fieldDelimiter: ' ')` für CSV-String.
   - Schreiben in Datei mit `getTemporaryDirectory()`.
   - Teilen über `Share.shareFiles()`.
2) **UI: Export-Button**
   - Button Plan als CSV exportieren im `CoachDashboardScreen`.
   - Snackbar-Bestätigung bei Erfolg/Fehler.

Phase 4: Implementierung Import

1) **CsvPlanService.importCsvToPlan(clientId)**
   - Datei-Auswahl via `FilePicker.platform.pickFiles()`.
   - Einlesen des Inhalts.
   - Parsen mit `CsvToListConverter(fieldDelimiter: ' ')`.
   - Validierung: Spaltenanzahl, Datentypen.

- Mappen auf `List<PlanEntry>`.
- Senden an API-Endpunkt `updateTrainingPlan`.

2) **UI: Import-Button**

- Button CSV Plan importieren im `CoachDashboardScreen`.
- Fortschrittsindikator während des Imports.
- Snackbar bei Erfolg/Fehler.

Phase 5: UI/UX Integration

- **Layoutüberprüfung**
  - Export-/Import-Buttons an prominenter Position.
  - Konsistente Farb- und Stilführung mit restlicher App.
- **Responsives Design**
  - Test auf verschiedenen Bildschirmgrößen.
  - Scrollbarkeit bei langen Listen.

Phase 6: Testing & Qualitätssicherung

- **Unit Tests**
  - `CsvPlanService.exportPlanToCsv` mit Dummy-Daten.
  - `CsvPlanService.importCsvToPlan` mit validen/inkorrekten CSVs.
- **Integrationstests**
  - Mock-API: Dateiaustausch, Upload/Download end-to-end.
- **Usability-Tests**
  - Coach testet Export, Bearbeitung in Excel, Re-Import.
  - Feedback zur Fehlerbehandlung (ungültige CSV).

Phase 7: Dokumentation & Deployment

- **Entwicklerdokumentation**
  - Architekturdiagramm.
  - API-Endpunkt-Beschreibung.
  - CSV-Format-Spezifikation.
- **Endanwender-Dokumentation**
  - Kurzanleitung im Wiki/README.
  - Screenshots: Export, Bearbeitung, Import.

- **Release**
  - Versions-Tag im Git.
  - Aufbau CI/CD-Pipeline für automatisches Testing.
  - Deployment auf App Stores.

Zeitplanung (Beispiel)

Woche 1	Analyse, Design, API-Spezifikation
Woche 2	Implementierung CSV-Export
Woche 3	Implementierung CSV-Import
Woche 4	UI-Integration, Tests
Woche 5	Dokumentation, Deployment