DataTaker - Projektdokumentation
(A1) Kurzbeschreibung

Projektteam: Arno Kesper und Studierende der Universität Marburg

29. Januar 2025

1 Inhaltliche Beschreibung des Projekts

1.1 Überblick

Das Projekt *DataTaker* ist eine mobile Applikation, die darauf abzielt, Nutzerinnen und Nutzern ein intuitives Werkzeug zur Erfassung von Daten in frei konfigurierbaren Tabellen bereitzustellen. Durch den Einsatz moderner Technologien (React Native, TypeScript, Expo) soll eine plattformunabhängige Nutzung auf Android- und iOS-Geräten ermöglicht werden.

Die Kernidee besteht darin, dass Anwender eigenständig Tabellen erstellen können, in denen Spalten (bzw. Variablen) definierbar sind. Jede Spalte kann einen spezifischen Datentyp besitzen (z. B. Text, Zahl, Datum, Boolean, Bild, Enum-Auswahl), sodass die Eingabe an die individuellen Anforderungen der Nutzenden angepasst werden kann. Zudem sollen Import- und Exportfunktionen (z. B. als CSV, JSON sowie optional ZIP-Archive inklusive Bilddaten) den Datenaustausch vereinfachen. Darüber hinaus soll das Projekt im Nachhinein erweiterbar sein (Funktionen wie Graphen zu den Tabellen o.ä.).

1.2 Zielsetzung

Das Hauptziel besteht darin, einen flexiblen, leicht bedienbaren und erweiterbaren Datenerfassungs-Workflow zu schaffen. Dabei stehen folgende Punkte im Vordergrund:

- **Benutzerfreundlichkeit:** Einfache Bedienung ohne Vorwissen, klarer Fokus auf gute UX-Design-Patterns.
- **Flexibilität:** Unterstützung verschiedener Datentypen und die Möglichkeit, Tabellen nachträglich anzupassen.
- **Plattformunabhängigkeit:** Einsatz von React Native, um sowohl Android- als auch iOS-Nutzende anzusprechen.
- **Nachhaltige Datenhaltung:** Nutzung einer lokalen Datenbank (Expo-SQLite) für persistente Speicherung.
- **Datenaustausch:** Einfache Export- und ggf. Importfunktionen, um mit anderen Anwendungen oder Systemen zu interagieren.

2 Rahmenbedingungen

2.1 Projekteinbettung und Auftraggeber

Das Projekt *DataTaker* ist nicht in ein größeres Projekt eingebettet, sondern eine eigenständige Arbeit im universitären Kontext. Als **Auftraggeber** fungiert Arno Kesper und die Projektgruppe selbst, bestehend aus drei Studierenden einer Hochschule, die im Rahmen einer Lehrveranstaltung diese Anwendung konzipieren, um ihre Fähigkeiten in der App-Entwicklung, Softwarearchitektur sowie UX-Design zu demonstrieren. Externe Unternehmen oder Arbeitsgruppen sind nicht beteiligt.

2.2 Besondere Vorgaben

Es existieren keine speziellen formalen oder inhaltlichen Vorgaben seitens der Universität. Die einzigen Richtlinien ergeben sich aus den Lernzielen der Lehrveranstaltung sowie den Anforderungen, die sich die Studierenden selbst gesteckt haben. Qualität, Verständlichkeit und Praktikabilität stehen an oberster Stelle, ohne dass formale Zertifizierungen oder Industriestandards zwingend gefordert sind.

2.3 Zeitlicher Rahmen

Der zeitliche Umfang des Projekts ist begrenzt. Es begann zum Start des Semesters und soll bis ca. **März** abgeschlossen sein. Der Endtermin ist vor allem von Abgabeterminen innerhalb des universitären Curriculums bestimmt. Dies stellt einen klaren zeitlichen Rahmen für die Fertigstellung der Kernfunktionen, grundlegenden Tests sowie gegebenenfalls eine kurze Evaluationsphase.

2.4 Qualitätsanforderungen und Skalierbarkeit

Obwohl es sich um ein universitäres Projekt handelt, gelten folgende Qualitätsanforderungen:

- **Stabilität:** Die App sollte unter typischen Nutzungsbedingungen absturzfrei laufen.
- **Performance:** Reaktionszeiten sollten angenehm kurz sein, auch bei größeren Tabellen oder vielen Bilderfassungen.
- **Skalierbarkeit:** Das Grundgerüst soll so entworfen sein, dass zukünftige Datentypen oder neue Features leicht integrierbar sind.
- **Benutzererfahrung:** Eine einfache, intuitive Navigation, übersichtliche Darstellungen und Hilfestellungen bei der Eingabe.
- **Wartbarkeit:** Durch die Verwendung von TypeScript und einer klaren Architektur soll der Code gut wartbar, erweiterbar und dokumentiert sein.

2.5 Technische Rahmenbedingungen

Die technische Umsetzung basiert auf den folgenden Komponenten:

- **Frontend:** React Native (Expo), TypeScript für strikte Typisierung und Wartbarkeit.
- **Backend (lokal):** Expo-SQLite für persistente Datenspeicherung.
- **Bildverarbeitung:** Expo-Camera zum Erfassen von Bildern, Expo-File-System zum Speichern von Bilddateien.
- **Export:** JSON- und CSV-Funktionen sind geplant, potenzielle Erweiterung auf ZIP-Archive mit integriertem Bildmaterial.

3 Zusammenfassung der erreichten Ergebnisse

Im bisherigen Projektverlauf wurden wichtige Grundfunktionen erfolgreich implementiert:

- **Tabellenerstellung:** Nutzende können individuelle Tabellen mit frei definierbaren Spalten (Variablen) anlegen.
- **Datenerfassung:** Verschiedene Datentypen (Text, Zahl, Boolean, Bilder) stehen in der Erfassung zur Verfügung.
- **Persistenz:** Alle erfassten Daten werden lokal in einer SQLite-Datenbank gespeichert.
- **Exportfunktion:** Ein Export von Tabellen im CSV-Format wird unterstützt, sodass Daten einfach weiterverarbeitet werden können.
- **Benutzeroberfläche:** Ein ansprechendes, reaktives UI unter Verwendung von React Native und TypeScript wurde bereitgestellt.

Darüber hinaus wurden im Laufe der Implementierung wichtige Erkenntnisse zur Architektur, Datenhaltung und UX gesammelt, welche ein stabiles Fundament für den weiteren Ausbau der App legen.

3.1 Mögliche zukünftige Arbeiten

Obwohl ein Großteil der Kernfunktionen vorhanden ist, bestehen weitere Ausbaumöglichkeiten:

- **Neue Datentypen:** Unterstützung zusätzlicher Felder (z. B. Geo-Koordinaten, Audio, Video).
- **Erweiterte Auswertungsfunktionen:** Diagramme oder Statistiken direkt innerhalb der App, um Daten komfortabel zu visualisieren.
- **Datei-Import:** Neben dem Export wäre ein Import externer Daten (z. B. CSV, JSON) für einen durchgängigen Datenaustausch sinnvoll.

- **Online-Synchronisierung:** Eine Schnittstelle zu Servern oder Cloud-Diensten, um Daten auf mehreren Endgeräten aktuell zu halten.
- **Multi-User-Fähigkeit:** Rollen- und Berechtigungsverwaltung in größeren Arbeitsgruppen-Umgebungen.

4 Erfahrungsbericht

4.1 Entscheidungen und deren Auswirkungen

Im Entwicklungsprozess wurden verschiedene Entscheidungen getroffen, die sich im Nachhinein als hilfreich oder eher schwierig erwiesen haben:

- **Einsatz von React Native und TypeScript:**
  - *Vorteil:* Klare Typisierung, gute Wiederverwendbarkeit und verhältnismäßig kurze Einarbeitungszeit.
  - *Nachteil:* Gruppe hat keine Vorkenntnisse in TypeScript oder React Native gehabt.
- **Expo-SQLite für lokale Speicherung:**
  - *Vorteil:* Sehr schnell eingebunden, offline-fähig und genügend für die angestrebte Datenmenge.
  - *Nachteil:* Weniger flexibel als moderne serverseitige Lösungen (aktuell nur lokal aber mit Datenbank auf Server möglich), falls zukünftig eine Online-Synchronisierung gewünscht wird.
- **Fokus auf Benutzerfreundlichkeit:**
  - *Vorteil:* Schnelles Prototyping von UI/UX-Konzepten durch Paper-Komponenten, was zu einem intuitiven Layout führte.

4.2 Tipps für zukünftige Entwickler

- **Wiederverwendbare Komponenten erstellen:** Durch modularen Aufbau können künftige Funktionserweiterungen rasch eingebunden werden.
- **Dokumentation nicht vernachlässigen**