

**Pflichtenheft**

**Einheitenumrechner**

***Kadrnoschka André***

***11.06.2025***

Inhalt

[Projektbeschreibung 2](#_Toc200495667)

[Ziele 2](#_Toc200495668)

[Ziel des Projektes 2](#_Toc200495669)

[Programmeinsatz 2](#_Toc200495670)

[Anwendungsbereich 2](#_Toc200495671)

[Systemumgebung 3](#_Toc200495672)

[Funktionale Anforderungen 3](#_Toc200495673)

[Views 3](#_Toc200495674)

[HistoryView 3](#_Toc200495675)

[SettingView 3](#_Toc200495676)

[UnitView 3](#_Toc200495677)

[Datenverarbeitung 3](#_Toc200495678)

[Eigene Klassen 3](#_Toc200495679)

[Events 3](#_Toc200495680)

[Datenerfassung 3](#_Toc200495681)

[Nicht-funktionale Anforderungen 4](#_Toc200495682)

[Benutzeroberfläche 4](#_Toc200495683)

[Benutzerfreundlichkeit 4](#_Toc200495684)

[Zuverlässigkeit 4](#_Toc200495685)

[Datenhaltung & Architektur 4](#_Toc200495686)

[Speicherung 4](#_Toc200495687)

[User Stories 5](#_Toc200495688)

# Projektbeschreibung

────────────────────────────────────────────────────────

Die Software ist ein vielseitiger Einheitenumrechner, der eine schnelle und benutzerfreundliche Umwandlung verschiedener Maßeinheiten ermöglicht. Sie unterstützt Umrechnungen von Längen-, Gewichts-, Volumen- sowie weiteren physikalischen Einheiten und eignet sich sowohl für den alltäglichen Gebrauch als auch für schulische oder berufliche Anwendungen. Zusätzlich zur Umrechnungsfunktion bietet die Anwendung eine Historie, in der alle bisherigen Umrechnungen dokumentiert werden. Dadurch können Nutzer ihre vergangenen Eingaben nachvollziehen und bei Bedarf erneut verwenden.

# Ziele

────────────────────────────────────────────────────────

## Ziel des Projektes

Das Ziel des Projekts ist die Entwicklung einer benutzerfreundlichen Software zur effizienten Umrechnung verschiedener Maßeinheiten. Dabei soll die Anwendung sowohl im Alltag als auch im schulischen und beruflichen Umfeld eine zuverlässige Unterstützung bieten. Durch die Implementierung einer Verlaufsfunktion wird die Nachvollziehbarkeit und Wiederverwendung früherer Umrechnungen gewährleistet. Langfristig soll die Funktionalität durch einen Quiz-Modus ergänzt werden, der das Verständnis und Wissen der Nutzer im Umgang mit Einheiten fördert.

# Programmeinsatz

────────────────────────────────────────────────────────

Anwendungsbereich  
Die Software ist eine Desktop-Anwendung zur Umrechnung von physikalischen Einheiten wie Länge, Masse oder Volumen. Sie richtet sich an Schülerinnen und Schüler sowie an Lehrkräfte und kann im schulischen Kontext oder zur individuellen Vorbereitung auf Prüfungen genutzt werden. Die Anwendung funktioniert vollständig offline und speichert Umrechnungsverläufe lokal, um Datenschutz und Zugänglichkeit auch ohne Internetverbindung zu gewährleisten. Eine optionale Quiz-Funktion zum selbstständigen Üben von Umrechnungen ist konzeptionell enthalten und wird bei entsprechender Zeit umgesetzt.

Systemumgebung  
Die Anwendung wurde für Windows 10 (oder höher) entwickelt und basiert auf .NET mit WPF unter Anwendung des MVVM-Architekturmusters. Zur Datenverwaltung werden Entity Framework Core sowie SQL oder JSON-Dateien verwendet.  
Empfohlene Mindestanforderungen:

* Betriebssystem: Windows 10 oder neuer
* Prozessor: 64-Bit, mind. 1.8 GHz
* Arbeitsspeicher: mindestens 4 GB (empfohlen 8 GB)

Die Entwicklung erfolgt mit Visual Studio 2022.

# Funktionale Anforderungen

────────────────────────────────────────────────────────

## Views

## HistoryView

WPF-Oberfläche zur Anzeige der gespeicherten Umrechnungen.

## SettingView

WPF-Oberfläche zum Eingeben und Speichern neuer Einheiten und Kategorien.

## UnitView

Stellt eine einfache Benutzeroberfläche zur Navigation zu den Hauptfunktionen dar.

## Datenverarbeitung

### Eigene Klassen

Das Projekt nutzt eigene Klassen wie ConversionEntry, Unit, und UnitCategory zur Verwaltung von Umrechnungen. Vererbung kommt bei der Strukturierung von Einheitenarten zum Einsatz.

Events

Zur Kommunikation zwischen ViewModels werden Events eingesetzt, etwa beim Ansichtswechsel oder bei neuen Umrechnungsergebnissen.

Datenerfassung

Alle Eingaben erfolgen über die grafische Benutzeroberfläche – inklusive Werteingabe, Einheitsauswahl und Navigation. Eine Dateieinbindung ist nicht notwendig.

Datenerhalt ist während der Laufzeit frei wählbar:

* SQL-Server (Datenbank) oder JSON-Dateien (Lokale Serialisierung)

# Nicht-funktionale Anforderungen

────────────────────────────────────────────────────────

## Benutzeroberfläche

Die Anwendung besitzt ein responsives und benutzerfreundliches Design, das sich an die Fenstergröße anpasst. ToolTips und beschriftete Buttons unterstützen die intuitive Bedienung. Farblich abgestimmte Elemente und Icons sorgen für ein ansprechendes Erscheinungsbild.

## Benutzerfreundlichkeit

* Dynamische Anpassung der Benutzeroberfläche an verschiedene Fenstergrößen
* Unterstützung durch ToolTips, klare Beschriftungen und strukturierte Layouts
* Eingabefelder werden auf Gültigkeit geprüft (z. B. Zahlenformate, Pflichtfelder)
* Fehlerbehandlung bei Datenzugriffen und Navigation durch try-catch-Blöcke

## Zuverlässigkeit

* Validierung aller Eingaben zur Vermeidung fehlerhafter Umrechnungen
* Anwendung bleibt auch bei fehlerhaften Eingaben oder Dateizugriffsproblemen stabil

# Datenhaltung & Architektur

────────────────────────────────────────────────────────

## Speicherung

Die Anwendung bietet zwei alternative Möglichkeiten zur Datenspeicherung:

* **JSON-Datei:** Umrechnungen und deren Historie werden lokal in einer JSON-Datei gespeichert.
* **SQLite-Datenbank:** Alternativ erfolgt die Speicherung strukturiert in einer lokalen SQLite-Datenbank über Entity Framework Core.

# User Stories

────────────────────────────────────────────────────────

* **Ein Benutzer kann über einen Button zur Umrechnungsansicht gelangen, um dort Einheiten wie Kilometer in Meter umzuwandeln.**
* **Ein Benutzer soll eine Eingabe machen können, z. B. eine Längeneinheit, und direkt das umgerechnete Ergebnis in einer anderen Einheit sehen.**
* **Ein Benutzer soll seine bisherigen Umrechnungen automatisch gespeichert bekommen, um sie später erneut aufrufen oder überprüfen zu können.**
* **Ein Benutzer kann im Menü oder über ein UI-Element (z. B. Slider) zwischen Dateispeicherung (JSON) und Datenbankspeicherung (SQLite) wechseln, um seine bevorzugte Speicherart auszuwählen.**
* **Ein Benutzer soll im Fehlerfall (z. B. ungültige Eingabe) eine klare Fehlermeldung erhalten, damit er seine Eingaben korrigieren kann.**
* **Ein Benutzer sieht Tooltips oder Hilfetexte bei Eingabefeldern und Buttons, um sich schnell in der Anwendung zurechtzufinden.**
* **Ein Benutzer profitiert von einer modernen, visuell ansprechenden Oberfläche mit Hintergrundbildern und Icons, um die Nutzung angenehmer zu gestalten.**