

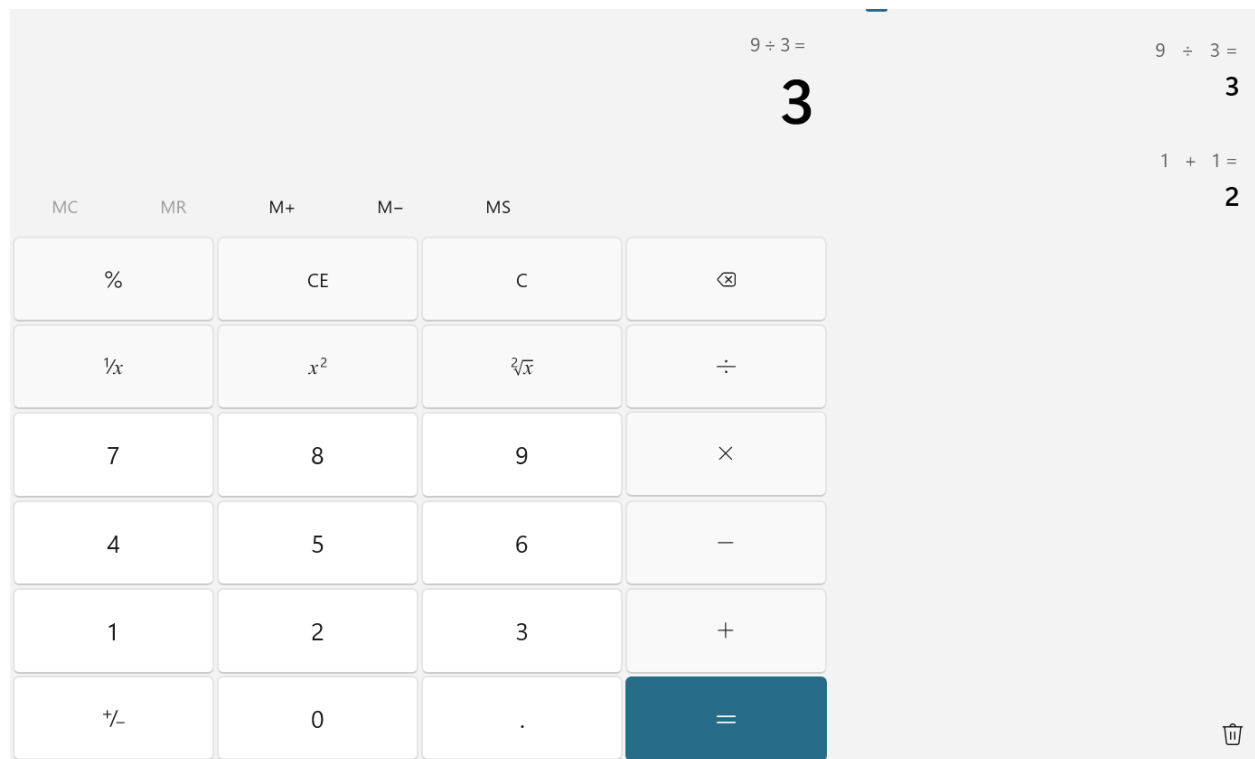
## Contents

Grundidee Taschenrechner: .....	1
Arbeitsprotokoll: .....	2

## Grundidee Taschenrechner:

Ich möchte eine Frontend-Seite entwickeln, die als Taschenrechner dient. Dafür soll es Zahl-Buttons von 0 bis 9 geben, mit denen sich auch mehrstellige Zahlen bilden lassen (z. B. 1 und 2 ergeben 12). Der Taschenrechner soll die grundlegenden Rechenoperationen unterstützen: Addition, Subtraktion, Multiplikation und Division. Der Rechnungsverlauf wird mithilfe einer Backend-API gespeichert und anschließend im Frontend angezeigt.

Beispiel Bild wie es am ende etwa aussehen könnte:



# Arbeitsprotokol:

## 5.12.2025 – Projektstart (Projektstruktur aufsetzen)

Heute habe ich mit dem Befehl `npm create vite@latest` das Projekt *Taschenrechner\_Projekt* erstellt und bereits leicht begonnen. Das Backend mit MongoDB habe ich auf meinem Rechner installiert und erfolgreich zum Laufen gebracht. Außerdem habe ich ein Dokument für die Arbeitsdokumentation angelegt. Auf GitHub habe ich ein Repository für das Projekt erstellt und es auf meiner VM initialisiert, sodass `git add` und `git push` nun funktionieren.

## 11.12.2025 – Programm Überarbeitung (Code funktional finalisieren)

Ich habe das Projekt vollständig umgesetzt. Der Taschenrechner ist funktionsfähig und verfügt über Buttons für Zahlen und Rechenoperationen. Mithilfe der vorgegebenen REST-API und der Datenbank habe ich einen Verlauf erstellt, der jede ausgeführte Rechnung speichert. Auf der Verlaufsseite gibt es für jeden Eintrag einen Bearbeiten-Button. Auf der Taschenrechnerseite habe ich zusätzlich einen Button eingebaut, mit dem sich der gesamte Verlauf löschen lässt. Beide Seiten sind über eine Navigation miteinander verbunden, sodass man bequem zwischen Rechner- und Verlaufsseite wechseln kann.

## 12.12.2025 – Projektende (Dokumentationen gemacht, Code verbessert)

Heute habe ich alle wichtigen Dokumente erstellt und im neuen Ordner *Dokumente* auf GitHub hochgeladen. Dazu gehören unter anderem die *Installationsanleitung.pdf*, das *Projekt\_UML.drawio.png*, der *Testplan.pdf* sowie die vollständige *Projektdokumentation*. Im Code habe ich die Calculator-Funktion aus der *App.jsx* in eine eigene Datei *Calculator.jsx* ausgelagert, um die Übersichtlichkeit zu erhöhen. Außerdem habe ich die Unit-Tests finalisiert, sodass sie nun fehlerfrei laufen und alle mit *Passed* zurückgegeben werden.