

RECHNUNGSMANAGER TOOL SELBST BAUEN: MIT EMERGENT IN VIER PROMPTS

4 PROMPTS

TOOL: EMERGENT

STAND 2026-09-14

In vier Prompts baust du dir mit Emergent eine eigene Web-App für deine Rechnungen, ohne Programmierkenntnisse. Sie schreibt und verschickt Rechnungen, prüft über Stripe die Zahlungseingänge und mahnt säumige Kunden von allein. Jeder Prompt baut auf dem vorigen auf: Login und Datenmodell, Kunden und Rechnungen, Dashboard mit Stripe, Mahnwesen. Dazu kommt die 40-Euro-Regel für Firmenkunden, die zu spät zahlen.

Die App aus diesem Guide baust du mit Emergent. [Hier geht's direkt zu Emergent](#)

WAS DU BRAUCHST

Ein Emergent-Konto. Registrieren kannst du dich auf [emergent.sh](#). Emergent ist eine Plattform, die aus einer Beschreibung in normaler Sprache eine fertige Web-App baut. Es gibt eine kostenlose Stufe mit einem monatlichen Credit-Kontingent.

Ein Stripe-Konto. Stripe wickelt die Kartenzahlungen ab. In diesem Guide bleiben wir durchgehend im **Testmodus**, du brauchst also kein verifiziertes Geschäftskonto und es fließt kein echtes Geld.

Ein Google-Konto mit Gmail, damit die App Rechnungen und Erinnerungen per Mail verschicken kann.

Etwa eine Stunde Zeit. Die vier Bau-Schritte laufen nacheinander, dazwischen arbeitet Emergent jeweils ein paar Minuten.

Zwei Begriffe, die im Guide immer wieder vorkommen: Ein **Prompt** ist der Text, den du Emergent gibst, damit es etwas baut. Ein **Connector** ist eine Verbindung zu einem anderen Dienst, hier zu Stripe und zu Google, damit deine App dort etwas auslösen darf.

WAS AM ENDE DASTEHT

Eine Web-App mit eigenem Login. Dahinter ein Dashboard, das dir zeigt, wie viel offen ist, wie viel überfällig und seit wann, und was in den letzten 30 Tagen reinkam.

Du legst Kunden an, stellst Rechnungen mit einzelnen Positionen zusammen, lädst sie als PDF herunter und verschickst sie. Der Kunde bekommt einen Zahlungslink. Sobald er bezahlt, springt die Rechnung von selbst auf **bezahlt**. Und wenn er nicht bezahlt, geht die Erinnerung automatisch raus.

01

EMERGENT ÖFFNEN UND EIN PROJEKT STARTEN

Geh auf emergent.sh und logge dich ein. Wenn du noch kein Konto hast, klick oben rechts auf **Sign up** und registrier dich mit deiner E-Mail-Adresse oder deinem Google-Konto.

Nach dem Login landest du auf einer Seite mit einem großen Eingabefeld in der Mitte. Da hinein kommt gleich der erste Prompt.

Wichtig für alles, was folgt: Alle vier Prompts gehen nacheinander in **denselben** Chat, nicht in vier getrennte Projekte. Nur so baut Emergent jeweils auf dem auf, was vorher schon entstanden ist.

01

PROMPT 1: ROHGERÜST, LOGIN UND DATENSTRUKTUR

Füge diesen Prompt in das Eingabefeld ein und schick ihn ab. Emergent stellt dir danach ein paar Rückfragen zum Aufbau, die du bestätigst.

Was hier passiert, klingt unspektakulär, ist aber der wichtigste Teil des ganzen Builds: Emergent legt die Datenbank an und verknüpft Kunden, Rechnungen und einzelne Rechnungspositionen miteinander, damit die App später weiß, welche Rechnung zu wem gehört. Und das Login sorgt dafür, dass jeder Nutzer nur seine eigenen Daten sieht.

PROMPT 1: SETUP, LOGIN & DATENMODELL

```
Baue eine Web-App namens "Marlis Invoice Chaser" für Unternehmer, um Rechnungen zu erstellen, Zahlungen zu verfolgen und säumige Kunden automatisch zu erinnern. Tech-Stack: React Frontend, Node/Express Backend, PostgreSQL Datenbank, mit Navigation (Dashboard, Kunden, Rechnungen, Einstellungen). Füge Nutzer-Authentifizierung hinzu (E-Mail/Passwort, sichere Session-Verwaltung), sodass jeder Nutzer nur seine eigenen Daten sieht. Erstelle die Datenmodelle Client (Name, E-Mail, Firma), Invoice (Kunde, Rechnungsnummer, Betrag, Fälligkeitsdatum, Status: offen/bezahlt/überfällig, erstellt am) und InvoiceItem (Beschreibung, Menge, Einzelpreis), korrekt über Relationen verknüpft.
```

WENN DAS NICHT KLAPPT

Falls nach dem ersten Durchlauf kein Login da ist oder eines der drei Datenmodelle fehlt, schick im selben Chat nach: Lege zusätzlich die Modelle Client, Invoice und InvoiceItem mit Relationen an und füge ein Login mit E-Mail und Passwort hinzu.

Alternativer Weg zum selben Ergebnis: Zerleg Prompt 1 in zwei Nachrichten. Erst das Login und den Tech-Stack anfordern, im zweiten Schritt die drei Datenmodelle. Kürzere Prompts setzt Emergent zuverlässiger um.

01

PROMPT 2: KUNDENVERWALTUNG UND RECHNUNGSFORMULAR

Warte, bis Emergent mit dem ersten Durchlauf fertig ist, und füge dann im selben Chat den zweiten Prompt ein.

Jetzt entstehen die Oberflächen: die Seite, auf der du Kunden anlegst, und das Formular, mit dem du eine Rechnung zusammenstellst. Positionen eintragen, Menge und Preis, die Summe rechnet sich von selbst, Fälligkeitsdatum setzen, PDF erzeugen.

Wenn du am Rechnungsformat etwas ändern willst, zum Beispiel deine Steuernummer, deine Bankverbindung oder ein Logo auf der Rechnung, dann schreib das direkt mit in diesen Prompt hinein, bevor du ihn abschickst.

PROMPT 2: KUNDEN- & RECHNUNGSVERWALTUNG

```
Baue eine Kundenseite, auf der ich Kunden anlegen, bearbeiten und löschen kann, mit einer Liste inklusive Anzahl offener Rechnungen pro Kunde. Baue außerdem ein Formular zum Erstellen neuer Rechnungen: Positionen hinzufügen, Menge, Preis, automatische Summenberechnung, Fälligkeitsdatum wählen, und die Möglichkeit, die fertige Rechnung als PDF zu generieren.
```

01

PROMPT 3: DASHBOARD UND STRIPE-ANBINDUNG

Dritter Prompt, wieder in denselben Chat. Hier kommt das Dashboard dazu, das dir offene, überfällige und bezahlte Beträge auf einen Blick zeigt, und die Stripe-Anbindung für die Bezahlung.

Der interessante Teil steckt im Webhook. Ein Webhook ist eine Rückmeldung von Stripe an deine App: Sobald ein Kunde über den Zahlungslink bezahlt, meldet Stripe das zurück, die Rechnung springt von selbst auf **bezahlt** und das Dashboard aktualisiert sich. Du musst nichts von Hand abhaken.

PROMPT 3: DASHBOARD & STRIPE-INTEGRATION

```
Baue ein Dashboard mit echten Daten aus der Datenbank: Gesamtbetrag offener Rechnungen, Gesamtbetrag überfälliger Rechnungen, Gesamtbetrag bezahlter Rechnungen (letzte 30 Tage), und eine Liste der 5 dringendsten überfälligen Rechnungen. Integriere Stripe im Testmodus: generiere für jede Rechnung einen Zahlungslink, und richte einen Webhook ein, der den Rechnungsstatus bei Zahlung automatisch auf "bezahlt" setzt und das Dashboard live aktualisiert.
```

01

PROMPT 4: AUTOMATISCHES MAHNWESEN

Der letzte Bau-Prompt. Er sorgt dafür, dass die App täglich selbst nachschaut, welche Rechnungen überfällig sind, den Status umstellt und eine Erinnerung mit Zahlungslink an den Kunden schickt.

Der letzte Satz im Prompt ist der, der die Mahngebühr dazurechnet. Ohne ihn erinnert die App nur, sie erhöht den offenen Betrag aber nicht.

PROMPT 4: MAHNWESEN & FEINSCHLIFF

```
Baue eine automatisierte Logik, die täglich überfällige Rechnungen erkennt, deren Status aktualisiert und automatisch eine Erinnerungs-E-Mail mit Zahlungslink an den Kunden verschickt. Verbessere abschließend das UI: modernes Design, Statusfarben für offen/überfällig/bezahlt, Ladeanimationen, und eine Erfolgsmeldung bei Zahlungseingang. Berechne bei überfälligen Rechnungen automatisch eine Mahngebühr und Verzugszinsen und weise den neuen Gesamtbetrag in der Erinnerung aus.
```

TIPP

Die 40-Euro-Regel. Wenn dein Kunde ein Unternehmen ist und zu spät zahlt, steht dir nach § 288 Abs. 5 BGB eine Verzugs pauschale von 40 Euro zu, zusätzlich zu den Verzugszinsen. Gegenüber Privatkunden gilt diese Pauschale nicht.

Wenn deine App mit genau diesem Betrag rechnen soll, schreib ihn in Prompt 4 hinein, statt die Höhe der KI zu überlassen.

01

CONNECTOREN HINZUFÜGEN

Auf der rechten Seite deines Projekts findest du den Bereich **Connectors**. Füge dort beide hinzu, die diese App braucht:

Stripe, damit die Zahlungslinks erzeugt werden und der Statuswechsel auf bezahlt bei deiner App ankommt.

Google, damit die Rechnungen und die Erinnerungsmails rausgehen.

Du wirst dabei jeweils einmal aufgefordert, dich bei dem Dienst anzumelden und die Verbindung zu bestätigen. Erst danach laufen die Automatisierungen.

TIPP

Falls dir der Stripe-Connector eine Auswahl zwischen Test und Live anbietet: nimm Test. Damit arbeitest du mit Testkarten und bewegst kein echtes Geld.

WENN DAS NICHT KLAPPT

Wenn kein Zahlungslink erzeugt wird oder keine Mail rausgeht, sind fast immer die Connectoren noch nicht verbunden. Füge Stripe und Google rechts im Connector-Bereich hinzu und schick danach Prompt 3 und Prompt 4 noch einmal im selben Chat ab, mit dem Zusatz: Die Connectoren sind jetzt verbunden, bitte richte die Zahlungslinks und den Mailversand darüber ein.

Alternativer Weg zum selben Ergebnis: Verbinde Stripe und Google gleich zu Beginn, bevor du Prompt 1 abschickst. Dann stehen die Verbindungen schon bereit, sobald Emergent sie das erste Mal braucht. Findest du den Connector-Bereich nicht, schau in den Projekteinstellungen unter Integrationen.

01 EINMAL KOMPLETT DURCHTESTEN

Leg einen Testkunden an, schreib ihm eine Rechnung über einen kleinen Betrag und schick sie raus. Öffne dann den Zahlungslink und bezahle mit einer Stripe-Testkarte: Kartennummer **4242 4242 4242 4242**, ein beliebiges Ablaufdatum in der Zukunft, eine beliebige dreistellige Prüfnummer.

Geh danach zurück ins Dashboard. Die Rechnung muss jetzt auf **bezahlt** stehen und der offene Betrag entsprechend gesunken sein. Wenn das passiert, läuft die ganze Kette: Rechnung raus, Zahlung rein, Status um, Dashboard aktuell.

Für das Mahnwesen setzt du bei einer zweiten Testrechnung das Fälligkeitsdatum auf gestern. Beim nächsten täglichen Durchlauf muss sie auf überfällig springen und die Erinnerung rausgehen.

DER EHRliche TEIL

Im Testmodus fließt kein echtes Geld. Für echte Zahlungen brauchst du ein verifiziertes Stripe-Konto mit deinen Geschäftsdaten und musst in der App von den Test-Schlüsseln auf die Live-Schlüssel umstellen.

Jeder Bau-Durchlauf in Emergent verbraucht Credits, und die sind je nach Plan im Monat begrenzt. Vier Prompts plus Nachbesserungen sind kein einzelner Klick.

Und das Wichtigste: Das hier ist eine App, die du dir selbst gebaut hast, kein geprüftes Buchhaltungsprogramm. Für Rechnungen, die wirklich an Kunden rausgehen, gelten in Deutschland gesetzliche Pflichtangaben. Lass einmal deinen Steuerberater draufschauen, bevor du damit echte Kunden fakturierst.

HÄUFIGE FRAGEN

EMERGENT STELLT MIR NACH DEM ERSTEN PROMPT RÜCKFRAGEN, DIE ICH NICHT VERSTEHE. WAS ANTWORTE ICH?

Bestätige die Vorschläge, die Emergent selbst macht. Die Rückfragen drehen sich um technische Details, die für den weiteren Aufbau keine Rolle spielen, solange Kunden, Rechnungen und Positionen so angelegt werden, wie es im Prompt steht.

MÜSSEN ALLE VIER PROMPTS WIRKLICH IN DENSELBEN CHAT?

Ja. Jeder Prompt baut auf dem auf, was vorher entstanden ist. In einem neuen Projekt fängt Emergent bei null an, und die Teile passen dann nicht zusammen.

DIE ERINNERUNGSMAILS KOMMEN BEIM KUNDEN NICHT AN.

Schau zuerst im Spam-Ordner des Empfängers nach. Liegt dort nichts, ist meistens der Google-Connector nicht sauber bestätigt. Geh rechts in den Connector-Bereich, entferne die Verbindung und richte sie neu ein.

KANN ICH DIE APP AUCH ANDEREN ANBIETEN?

Technisch ja, sie ist von Anfang an mehrbenutzerfähig, weil jeder Nutzer nur seine eigenen Daten sieht. Sobald du damit Geld verdienst, kommen allerdings Themen wie Auftragsverarbeitung, Datenschutzerklärung und Impressum dazu.

ACHTUNG

Hinweis: Dieser Guide dient zu Informations- und Bildungszwecken. Alle Inhalte spiegeln den Stand bei Erstellung wider. Oberflächen, Funktionen und Preise der genannten Tools können sich ändern. Prüfe Ergebnisse von KI-Tools immer selbst nach. @GPTMarlon übernimmt keine Haftung für Entscheidungen auf Grundlage dieser Inhalte. Dieser Guide stellt keine Anlage-, Steuer-, Rechts- oder Gesundheitsberatung dar. Lass Ergebnisse bei Bedarf fachlich prüfen.

In dieser App verarbeitest du Namen, E-Mail-Adressen und Rechnungsdaten deiner Kunden. Sobald du sie über den eigenen Test hinaus einsetzt, brauchst du eine Datenschutzerklärung und musst prüfen, wo diese Daten gespeichert werden.