

## User Story: Neue Ausgabe (Expense) erfassen

<b>Priorität</b>	Hoch
<b>Rolle</b>	Nutzer der Expanse-Tracker-App
<b>Ziel</b>	Als Nutzer der Expanse-Tracker-App möchte ich eine neue Ausgabe(Expense) erfassen, indem ich Betrag, Datum, eine Kategorie, eine Zahlungsmethode und optional eine Notiz eingebe,
<b>Nutzen</b>	sicherzustellen, dass alle meine finanziellen Transaktionen lückenlos und korrekt im System dokumentiert sind und später für Berichte und Analysen zur Verfügung stehen.

### Akzeptanzkriterien

- Der Betrag muss als positiver numerischer Wert (Gleitkommazahl) validiert und gespeichert werden.
- Datum, Kategorie und Zahlungsmethode sind Pflichtfelder.
- Die eingegebene Kategorie-ID und Zahlungsmethoden-ID müssen in den jeweiligen Datenbanktabellen existieren (Foreign Key Constraint).
- Nach erfolgreicher Speicherung in der Datenbank (durch DBController.insert\_expense) wird die neue Ausgabe in der Hauptübersicht angezeigt.
- Bei fehlenden Pflichtfeldern oder ungültigen Daten erhält der Nutzer eine informative Fehlermeldung.

### Verwandte

<b>Komponenten</b>	Expense Klasse, Category Klasse, Payment_method Klasse, DBController.insert_expense()
--------------------	---

## User Story: Ausgabenliste anzeigen

<b>Priorität</b>	Hoch
<b>Rolle</b>	Benutzer der App
<b>Ziel</b>	Als Nutzer der Expense-Tracker App möchte ich alle meine Ausgaben (Expenses) in einer Liste sehen, um einen guten Überblick auf meine Ausgabeverhältnis zu erhalten.

### Akzeptanzkriterien

- Die Liste muss mindestens folgende Informationen für jede Ausgabe anzeigen: Datum, Betrag, Kategorie-Name, Zahlungsmethoden-Name und Notiz.
- Die Liste muss standardmäßig nach Datum (absteigend) sortiert sein (neueste zuerst).
- Die Anwendung muss eine entsprechende Datenbankabfrage (z.B. ein SELECT mit JOINS) verwenden, um alle benötigten Informationen (einschließlich der Namen von Kategorie und Zahlungsmethode) effizient abzurufen.
- Die Liste soll auch dann korrekt angezeigt werden, wenn keine Ausgaben erfasst wurden (leere Liste mit entsprechender Nachricht).

### Verwandte

<b>Komponenten</b>	Expense Klasse, Category Klasse, Payment_method Klasse, DBController
--------------------	--

## **User Story: Bestehende Ausgabe bearbeiten**

<b>Priorität</b>	Hoch
<b>Rolle</b>	Nutzer der App
<b>Ziel</b>	Als Nutzer der Expense-Tracker-App möchte ich eine bereits erfasste Ausgabe auswählen und alle ihre Felder (Betrag, Datum, Kategorie, Zahlungsmethode, Notiz) bearbeiten und aktualisieren, um (Nutzen)Fehler in den erfassten Daten (z.B. falsche Kategorie, Tippfehler im Betrag) korrigieren und somit die Genauigkeit meiner Finanzhistorie sicherstellen zu können.

### **Akzeptanzkriterien**

- Der Nutzer muss die Möglichkeit haben, aus der Ausgabenliste heraus in einen Bearbeitungsmodus für die gewählte Ausgabe zu gelangen.
- Alle Felder der Ausgabe müssen editierbar sein.
- Nach dem Speichern muss das System die Datenvalidierung (z.B. positiver Betrag, existierende Kategorie/Zahlungsmethode) erneut durchführen.
- Die Bearbeitung muss in der Datenbank persistent gespeichert werden (durch `DBController.update_expense()`).
- Nach erfolgreicher Bearbeitung wird der Nutzer auf die Ausgabenliste zurückgeführt, wo die aktualisierte Ausgabe sofort sichtbar ist.
- Das System muss sicherstellen, dass die aktualisierte Ausgabe in der Datenbank durch ihre eindeutige ID identifiziert wird.

### **Verwandte**

**Komponenten** Expense Klasse, `DBController.update_expense()`