

# SaveUp-App

## Projektdokumentation

IBZ Aarau Applikationsentwicklung EFZ  
Modul 322

Abgabedatum: 12.05.2025

Verfasst von: Mohamed Gebeili & Majd Bayeh



---

## Inhaltsverzeichnis

1	Einleitung .....	3
2	Ausgangslage und Ziele .....	3
3	IPERKA-Gliederung.....	4
4	Mockups / Wireframes (Screen Design).....	4
4.1	MainPage .....	4
4.2	AddSavingPage .....	5
4.3	SavingsListPage .....	5
5	Systemarchitekturentwurf.....	6
5.1	Frontend (XAML / .NET MAUI): .....	6
5.2	Backend (C#) .....	6
5.3	Datenverwaltung (JSON / XML): .....	6
5.4	Systemübersicht .....	6
6	Gantt-Diagramm .....	7
6.1	Meilenstein .....	7
7	Testplan und Protokoll.....	8
8	Fazit.....	8
9	Quellenverzeichnis.....	8
10	Glossar .....	9

# 1 Einleitung

Die SaveUp-App ist eine .NET MAUI Anwendung zur Erfassung und Verwaltung von gesparten Ausgaben. Ziel der Anwendung ist es, Benutzern die Möglichkeit zu bieten, kleine Verzichtsausgaben festzuhalten, um langfristig Geld für größere Investitionen zu sparen. Diese Dokumentation beschreibt die Ausgangslage, Ziele, Planung, technische Umsetzung und abschließende Reflexion des Projekts.

## 2 Ausgangslage und Ziele

**Das Projekt wurde im Rahmen des Moduls LBV 322 entwickelt. Ziel war es, eine einfache und intuitive Anwendung zu erstellen, die mindestens drei Content Pages umfasst und es Nutzern ermöglicht, Ersparnisse mit Kurzbeschreibung, Preis und Datum/Uhrzeit zu erfassen und lokal zu speichern. Das Projekt sollte die folgenden Kernanforderungen erfüllen:**

- Erfassen von gesparten Artikeln (Kurzbeschreibung, Preis, Datum/Uhrzeit)
- Speicherung der Daten in einer lokalen Datei (XML oder JSON)
- Anzeige aller Einträge in einer Liste mit Gesamteinsparung
- Zwei Menüfunktionen (Eintrag speichern und Liste aufrufen)
- MVVM-Architektur
- Verwendung von XAML-Styling
- Eigenes App-Icon
- Dokumentation und Testing

### 1.2. Einleitung

- Sie möchten für eine grössere private Investition (z.B. Ferien etc.) Geld sparen und haben sich aus diesem Grund für den Verzicht von üblichen kleinen Ausgaben wie z.B. Kaffee, Süßigkeiten etc. entschieden.
- Alle gesparten Kleinkäufe möchten Sie in einer App festhalten, sodass Sie sich laufend über den angesparten Geldbetrag informieren können

### 1.3. Auftrag

- Entwickeln Sie eine .NET MAUI App aus min. drei Content Pages in welcher Sie eine Kurzbeschreibung und den Preis des gesparten Artikels erfassen und speichern können.
- Die gesparten Verzichtsprодукte sollen dann in einer Liste inkl. der Preissumme angezeigt werden.
- Folgende Pflichtenforderungen müssen eingehalten werden:
  - Name und Titel der App: SaveUp
  - App besteht min. aus 3 Content Pages
  - GUI Design (Mock-Up's)
  - Produkterfassung besteht aus min. 2 Eingaben (Kurzbeschreibung u. Preis)
  - Datum/Uhrzeit als weiteres Attribut, wann der Kaufverzicht erfolgte.
  - Bietet zwei Menüfunktionen (Action) zur Speicherung und Aufruf der Listendarstellung an
  - Anzeige der Gesamteinsparung (Total der Verzichtskäufe)
  - Einfache bzw. intuitive Bedienung, geeignetes Layout (Styles)
  - Codestrukturierung nach Model View ViewModel (MVVM) Entwurfsmuster
  - Verwendung von XAML-Styles der Steuerelemente
  - Speichern (Persistenz) der Einträge in einer lokalen Datei (XML, JSON)
  - Eigenes App-Icon
  - Dokumentation u. Testing

### 3 IPERKA-Gliederung

<b>I (Idee)</b>	<b>Ziel: Kleine Ausgaben (z.B. Kaffee) tracken und Einsparungen aufsummieren, um größere Investition (Urlaub) zu ermöglichen.</b>
<b>P (Plan)</b>	- Pflichtenheft mit 3 Content Pages - MVVM-Architektur - Lokale JSON-Persistenz - Optionale Features: Löschen, Charts
<b>E (Entscheidung)</b>	Framework: .NET MAUI (Cross-Platform) UI: XAML mit eigenen Styles Architektur: MVVM mit CommunityToolkit.Mvvm
<b>R (Realisierung)</b>	- Model: SavingItem (Description, Price, DateTime) - Views: MainPage, AddSavingPage, SavingsListPage - ViewModels: AddSavingVM, SavingsListVM - Service: SavingService für JSON I/O
<b>K (Kontrolle)</b>	- Testplan & Protokoll (Kapitel 5) - Gantt-Chart zur Terminüberwachung (siehe Tabelle unten bzw. Excel-Datei)
<b>A (Auswertung)</b>	Reflexion: XAML-Fehler debuggen, MVVM-Prinzipien vertieft, Spaß an cross-platform Entwicklung

## 4 Mockups / Wireframes (Screen Design)

### 4.1 MainPage

- **Oben: App-Icon (appicon.svg), Titel „SaveUp“**
- **Anzeige: Gesamtsumme (zentrales Label)**
- **Zwei Buttons: „Verzicht hinzufügen“, „Einsparungen anzeigen“**

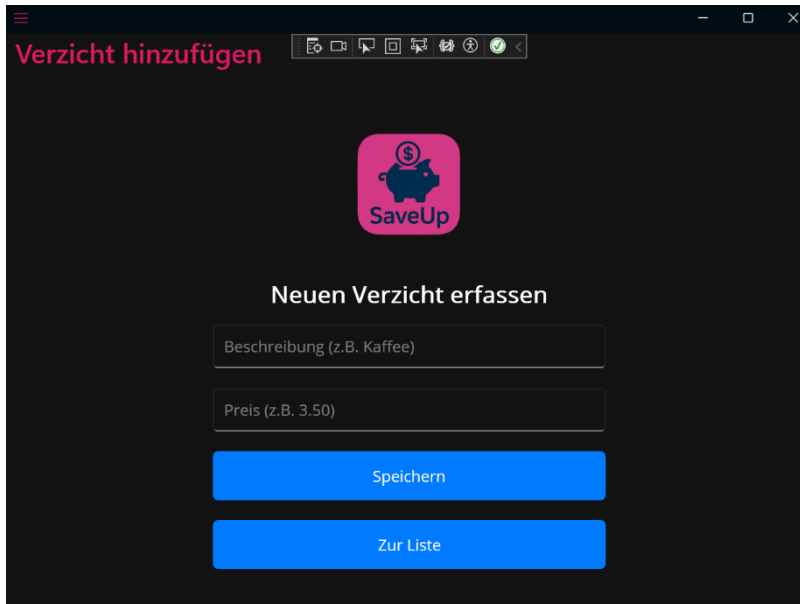


2.

## 4.2 AddSavingPage

- **Oben:** App-Icon, Titel „Neuen Verzicht erfassen“
- **Eingabefelder:** Beschreibung, Preis (numeric), Datum/Uhrzeit (automatisch)
- **Buttons:** „Speichern“, „Zur Liste“

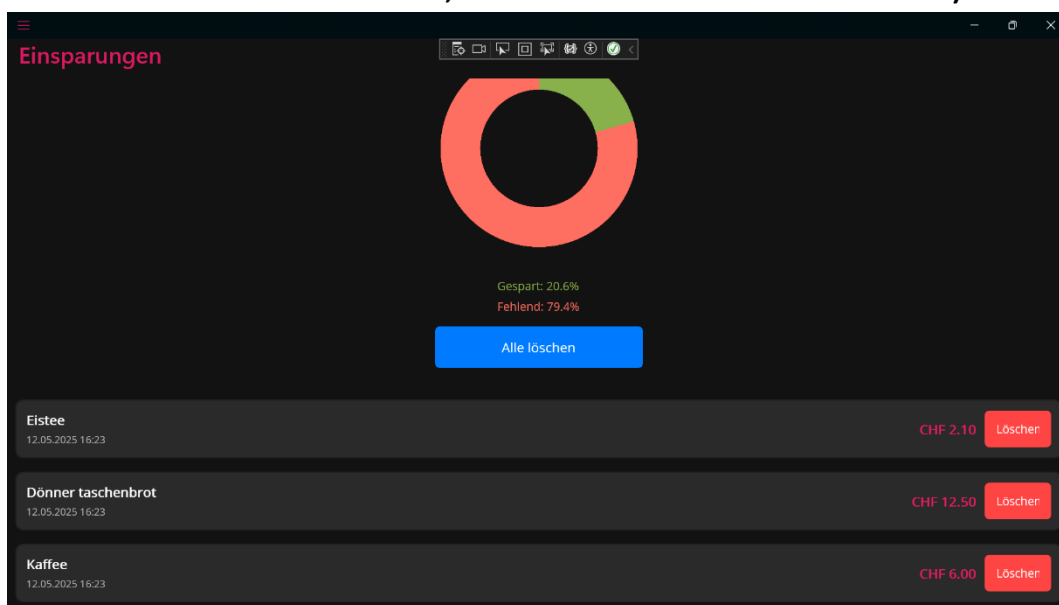
3.



## 4.3 SavingsListPage

- **Oben:** Titel „Einsparungen“
- **Darstellung:** Donut-Diagramm als Fortschrittsanzeige (Gespart: 20,6 %, Fehlend: 79,4 %) und darunter Button „Alle löschen“
- **Unten:** Liste der Einträge mit Beschreibung, Preis und Datum/Uhrzeit – jeweils mit „Löschen“-Button (z. B. Eistee – CHF 2.10 – 12.05.2025 16:23; Döner taschenbrot – CHF 12.50 – 12.05.2025 16:23; Kaffee – CHF 6.00 – 12.05.2025 16:23)

4.



---

## 5 Systemarchitekturentwurf

Die Systemarchitektur der SaveUp-App umfasst folgende Hauptkomponenten:

### 5.1 Frontend (XAML / .NET MAUI):

- UI-Design und Layout für die Content Pages.
- Benutzerinteraktionen wie Eingaben und Anzeigen von Daten.
- Implementierung von Animationen und Styles für eine moderne Benutzererfahrung.

### 5.2 Backend (C#)

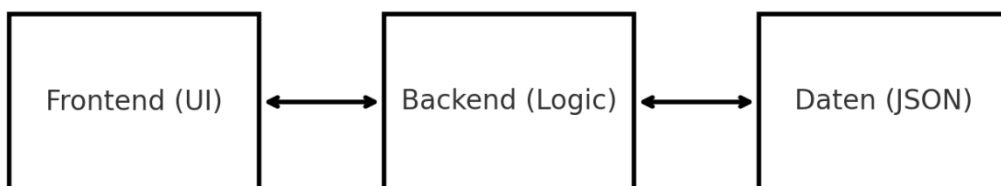
- Logik zur Datenverarbeitung.
- Speicherung und Verwaltung von Verzichtsartikeln.
- Anbindung an die lokale JSON-Datei zur dauerhaften Speicherung.

### 5.3 Datenverwaltung (JSON / XML):

- Speicherung der gesparten Einträge in einer lokalen JSON-Datei.
- Nutzung von Datenmodellen zur Serialisierung und Deserialisierung.
- Trennung von Logik (ViewModel), Daten (Model) und UI (View) für eine saubere und wartbare Codebasis.

### 5.4 Systemübersicht

Das folgende Diagramm zeigt die Systemarchitektur der SaveUp-App:



\_\_\_\_\_

[illegible]

## 7 Testplan und Protokoll

### Teststrategie

- Integrationstests zur Sicherstellung der korrekten Interaktion zwischen Frontend und Backend.
- Manuelle Tests zur Überprüfung der Benutzeroberfläche und Bedienbarkeit.

Test-ID	Testbeschreibung	Eingabedaten	Erwartetes Ergebnis	Tester	Ergebnis
T1	Speichern eines Verichtsartikels	Name: "Kaffee", Preis: 4.50	Artikel in Liste gespeichert	Mohamed	Erfolgreich
T2	Gesamteinsparung korrekt berechnet	Artikel: 3 Einträge, Gesamt: 20.00	Summe 20.00 angezeigt	Majd	Erfolgreich
T3	Löschen eines Eintrags	Artikel: "Kaffee"	Artikel entfernt, Summe angepasst	Mohamed	Erfolgreich

## 8 Fazit

Die Entwicklung der SaveUp-App hat gezeigt, dass auch kleinere Projekte eine detaillierte Planung und strukturierte Umsetzung erfordern. Durch die Arbeit mit .NET MAUI konnte ein tiefes Verständnis für plattformübergreifende Entwicklung gewonnen werden. Besonders die Implementierung der Datenmodelle und die Gestaltung der Benutzeroberfläche haben dazu beigetragen, die eigenen Fähigkeiten in der App-Entwicklung weiter auszubauen. Insgesamt konnte das Projekt erfolgreich abgeschlossen werden. Die App bietet alle geplanten Funktionen und erfüllt die Anforderungen der Aufgabenstellung. Durch das gemeinsame Arbeiten an der Dokumentation und Präsentation wurde auch die Teamfähigkeit gestärkt.

## 9 Quellenverzeichnis

Die Inhalte dieser Dokumentation basieren auf eigenen Entwürfen und entwickelten Code-Dateien. Alle verwendeten externen Quellen wurden im Rahmen der Projektentwicklung dokumentiert.



---

## 10 Glossar

### **.NET MAUI**

Eine plattformübergreifende App-Entwicklungsplattform, die es ermöglicht, native Anwendungen für iOS, Android, macOS und Windows mit einer gemeinsamen Codebasis zu erstellen.

### **App-Icon**

Ein kleines, grafisches Symbol, das eine mobile oder Desktop-Anwendung repräsentiert.

### **Backend**

Der Teil einer Anwendung, der die Datenverarbeitung, Geschäftslogik und Datenverwaltung übernimmt. Bei der SaveUp-App in C# implementiert.

### **Content Page**

Eine Seite in einer .NET MAUI App, die den Inhalt der Anwendung darstellt. Beispiele in der SaveUp-App sind MainPage, AddSavingPage und SavingsListPage.

### **CRUD**

Abkürzung für Create, Read, Update, Delete. Grundlegende Datenbankoperationen, die auch in der SaveUp-App verwendet werden.

### **Datenmodell**

Eine Datenstruktur, die die Art und Weise beschreibt, wie Daten innerhalb der Anwendung gespeichert und verarbeitet werden.

### **JSON (JavaScript Object Notation)**

Ein leichtgewichtiges Datenformat, das zum Speichern und Übertragen von Daten verwendet wird. In der SaveUp-App zur Speicherung von Verzichtseinträgen verwendet.

### **Frontend**

Der sichtbare Teil einer Anwendung, der die Benutzeroberfläche darstellt. Bei der SaveUp-App mit XAML implementiert.

### **Gantt-Diagramm**

Ein Balkendiagramm, das den Zeitplan eines Projekts darstellt, einschließlich Aufgaben, Meilensteinen und Abhängigkeiten.

### **IPERKA**

Eine Methode zur strukturierten Planung und Durchführung von Projekten. Steht für Informieren, Planen, Entscheiden, Realisieren, Kontrollieren und Auswerten.

### **Layout**

Die Anordnung der grafischen Elemente auf einer Benutzeroberfläche.

### **MVVM (Model-View-ViewModel)**

Ein Software-Architekturmuster, das die Trennung von Logik, Daten und Benutzeroberfläche ermöglicht. In der SaveUp-App zur Strukturierung des Codes verwendet.

### **Mockup**

Ein visuelles Designmodell, das die Struktur und den Inhalt einer Benutzeroberfläche zeigt.

### **Serialisierung**

Der Prozess, Datenstrukturen in ein Format umzuwandeln, das gespeichert oder übertragen werden kann. In der SaveUp-App wird JSON zur Serialisierung verwendet.

### **View**

Die visuelle Darstellung der Daten in einer Anwendung. In der SaveUp-App durch XAML definiert.

### **ViewModel**

Der Teil des MVVM-Musters, der die Logik und Daten der Benutzeroberfläche kapselt und die Datenbindung ermöglicht.

### **XAML (Extensible Application Markup Language)**

Eine Markup-Sprache, die zum Erstellen von Benutzeroberflächen in .NET MAUI-Apps verwendet wird.