

A screenshot of a terminal window titled 'C:\Users\Dave\source/repos\'. The window displays a menu for a system named 'Lagerverwaltung' (Warehouse Management). The menu options are:

- 1. Login
- 2. Beenden

The prompt 'Bitte wählen Sie eine Option:' (Please select an option:) is visible at the bottom of the menu.

# Lagerverwaltungssystem

Entwicklung eines lokalen, konsolenbasierten Systems zur Materialverwaltung  
in C# (.NET)

# Projektziele & Kernanforderungen



## Benutzerverwaltung

Admin- und Mitarbeiter-Rollen mit Login-System und Passwortschutz



## Materialverwaltung

CRUD-Operationen für Artikel, Bestandsführung in Echtzeit



## Warenkorb-Funktion

Materialentnahmen mit temporärem Container und Checkout-Prozess



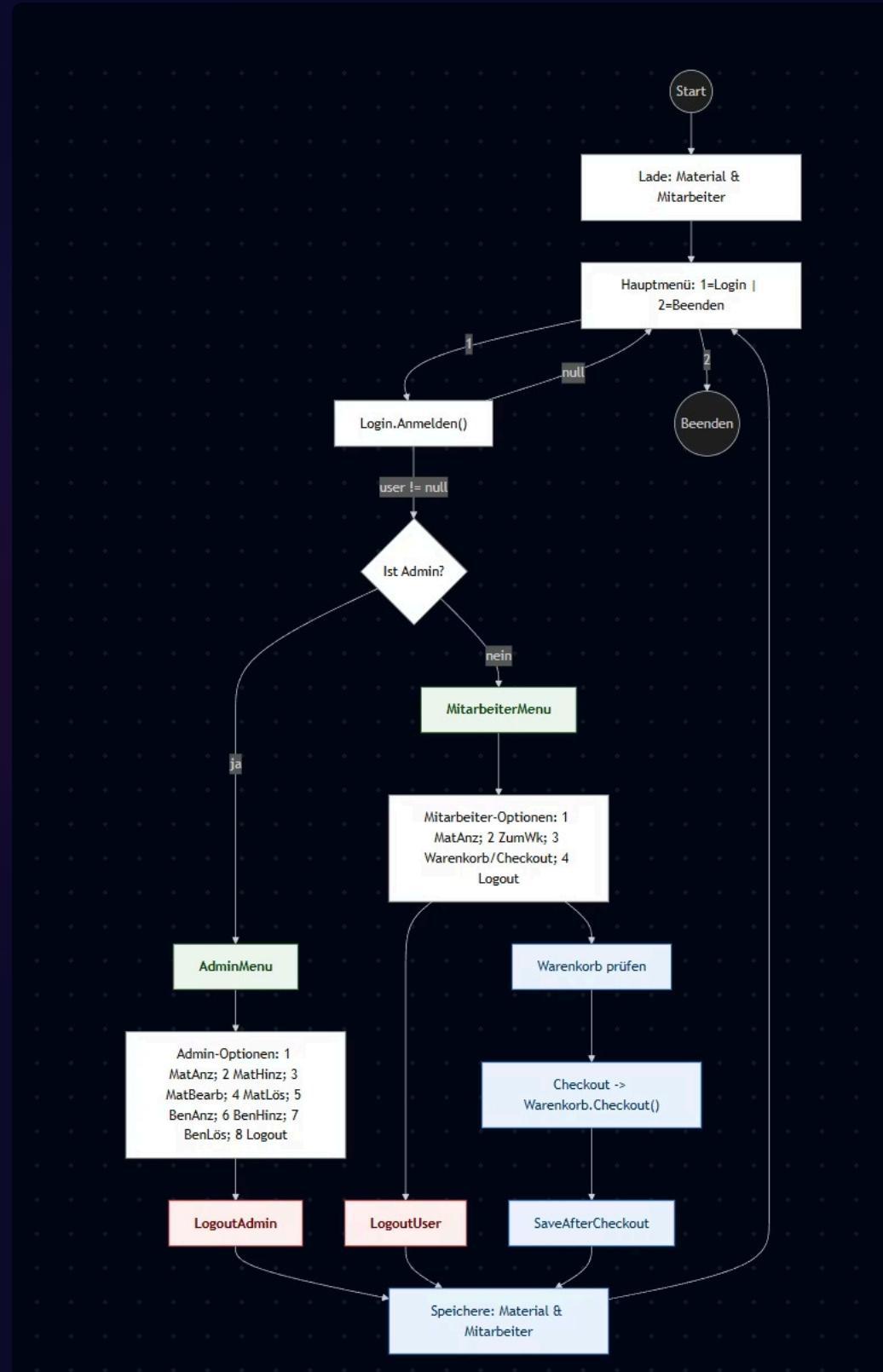
## JSON-Persistenz

Lokale Datenspeicherung ohne externe Datenbank-Abhängigkeiten

# Systemarchitektur

## Hauptkomponenten

- **Programm:** Hauptprogramm mit Menüführung
- **Login:** Authentifizierungs-Logik
- **AlleMitarbeiter:** Benutzerverwaltung
- **Admin/Mitarbeiter:** Rollenspezifische Zugriffe
- **Lager:** Bestandsverwaltung
- **Material:** Datenmodell mit JSON-Anbindung
- **Warenkorb:** Entnahme-Container



Steuerung erfolgt über intuitives Konsolenmenü mit klarer Rollenlogik

# Rollen & Berechtigungen

# Administrator

- Material anzeigen, hinzufügen, bearbeiten, löschen
  - Benutzer anzeigen, hinzufügen, löschen
  - Vollständiger Systemzugriff
  - Bestandskorrekturen durchführen

# Mitarbeiter

- Material einsehen und durchsuchen
  - Artikel zum Warenkorb hinzufügen
  - Warenkorb verwalten und Checkout
  - Eingeschränkter, prozessorientierter Zugriff

Login-System prüft Benutzerrechte aus **Benutzer.json** und leitet zum rollenspezifischen Menü

```
Lagerverwaltung

    class Login

        public Login()
        {
            AlleMitarbeiter.LadeDaten();
        }

        public AlleMitarbeiter Anmelden()
        {
            Console.Clear();
            Console.ForegroundColor = ConsoleColor.Green;
            Console.WriteLine("  <<<<><><><>>>> Login <<<<><><><>>>> ");
            Console.WriteLine("-----");
            Console.ResetColor();

            Console.Write("Username: ");
            string username = Console.ReadLine();

            Console.Write("Passwort: ");
            string password = Console.ReadLine();

            var user = AlleMitarbeiter.BenutzerListe
                .FirstOrDefault(u => u.Username == username && u.Password == password);

            if (user == null)
            {
                Console.WriteLine("Login fehlgeschlagen!");
                Console.ReadKey();
            }
            else
            {
                Console.WriteLine($"Login erfolgreich! Willkommen {user.Username}");
                Console.ReadKey();
            }
        }

        return user;
    }
}
```

# Datenhaltung mit JSON

1

## Material.json

Zentrale Materialliste mit allen Artikelinformationen und Bestandsdaten

2

## Warenkorb.json

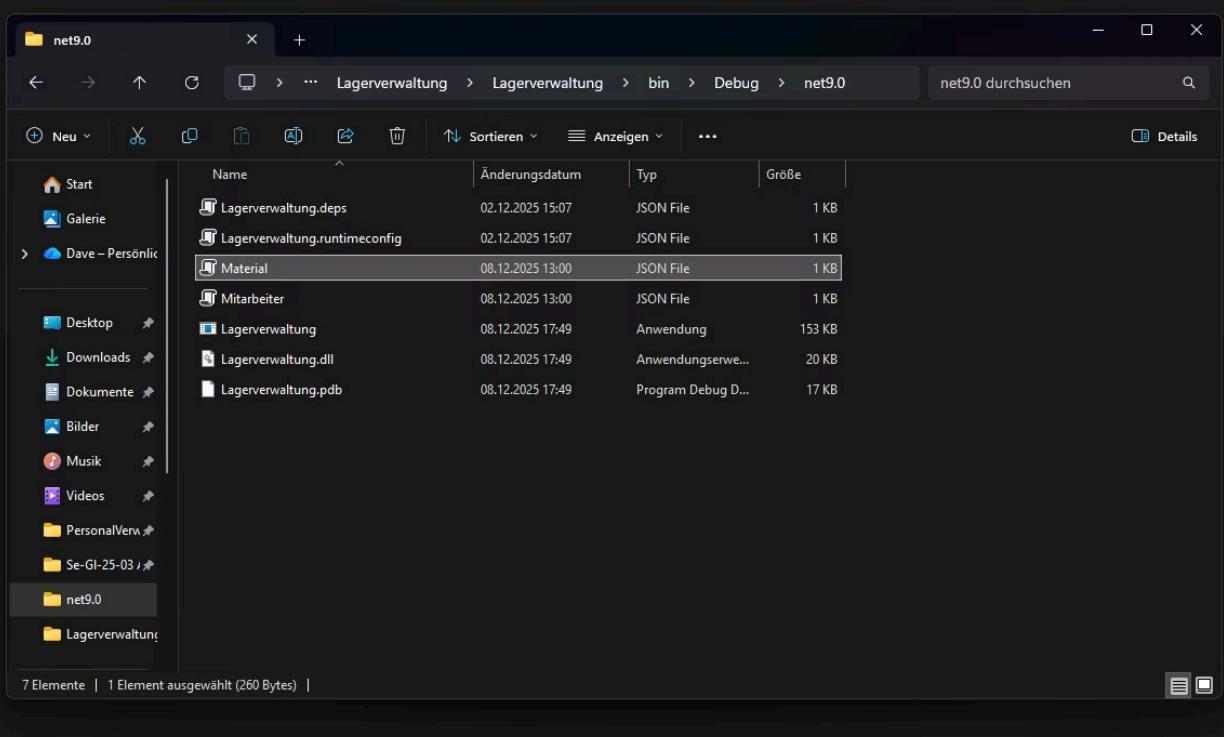
Individueller Warenkorb pro Mitarbeiter mit temporären Entnahmepositionen

3

## Benutzer.json

Benutzeraccounts mit Anmeldeinformationen und Rolleninformationen

- Keine Datenbank erforderlich – vollständig lokal lauffähig, ideal für kleine bis mittlere Anwendungsfälle



```
Material.json
Schema: <Kein Schema ausgewählt>
1 [
2   {
3     "Bezeichnung": "Schrauben",
4     "Artikelnummer": "11111",
5     "Lagerplatz": "12-12-12-12",
6     "Bestand": 3400
7   },
8   {
9     "Bezeichnung": "Paletten",
10    "Artikelnummer": "2222",
11    "Lagerplatz": "Blocklager",
12    "Bestand": 3456
13  },
14  {
15    "Bezeichnung": "Holzbrett 120cm",
16    "Artikelnummer": "33333",
17    "Lagerplatz": "A-01-04",
18    "Bestand": 120
19  },
20  {
21    "Bezeichnung": "Elektrokabel 10m",
22    "Artikelnummer": "44444",
23    "Lagerplatz": "B-02-08",
24    "Bestand": 85
25  },
26  {
27    "Bezeichnung": "Akuschrauber",
28    "Artikelnummer": "55555",
29    "Lagerplatz": "C-15-03",
30    "Bestand": 25
31  },
32  {
33    "Bezeichnung": "Metallrohr 2m",
34    "Artikelnummer": "66666",
35    "Lagerplatz": "D-09-06",
36    "Bestand": 200
37  },
38  {
39    "Bezeichnung": "Lagerbox klein",
40    "Artikelnummer": "77777",
41    "Lagerplatz": "Regal H",
42    "Bestand": 870
43  },
44  {
45    "Bezeichnung": "Schutzhelm gelb",
46    "Artikelnummer": "88888",
47    "Lagerplatz": "Safety-01",
48    "Bestand": 42
49  },
```

```
{  
    while (true)  
    {  
        Console.Clear();  
        Console.ForegroundColor = ConsoleColor.Green;  
        Console.WriteLine($"<><><> Admin Bereich ({user.Username}) <><><>");  
        Console.ResetColor();  
        Console.WriteLine("\n1. Material anzeigen");  
        Console.WriteLine("2. Material hinzufügen");  
        Console.WriteLine("3. Material bearbeiten");  
        Console.WriteLine("4. Material löschen");  
        Console.WriteLine("5. Benutzer anzeigen");  
        Console.WriteLine("6. Benutzer hinzufügen");  
        Console.WriteLine("7. Benutzer löschen");  
        Console.WriteLine("8. Logout");  
        Console.Write("Auswahl: ");  
  
        var input = Console.ReadLine();  
        switch (input)  
        {  
            case "1":  
                lager.ZeigeMaterial();  
                break;  
            case "2":  
                Console.Write("Name: ");  
                var name = Console.ReadLine();  
                Console.Write("Lagerplatz: ");  
                var lp = Console.ReadLine();  
                Console.Write("Artikelnummer: ");  
                var an = Console.ReadLine();  
                Console.Write("Bestand: ");  
                var bestand = int.Parse(Console.ReadLine() ?? "0");  
                lager.NeuesMaterialHinzu(name, lp, an, bestand);  
                break;  
            case "3":  
                lager.BearbeiteMaterial();  
                break;  
            case "4":  
                lager.LoescheMaterial();  
                break;  
            case "5":  
                Console.Clear();  
                foreach (var b in AlleMitarbeiter.BenutzerListe)  
                    Console.WriteLine($"{b.Username} (Admin: {b.IstAdmin})");  
                Console.ReadKey();  
                break;  
            case "6":  
                Console.Write("Username: ");  
                var u = Console.ReadLine();  
                Console.Write("Passwort: ");  
                var p = Console.ReadLine();  
                AlleMitarbeiter.BenutzerListe.Add(new AlleMitarbeiter(u, p, false));  
                AlleMitarbeiter.SpeichereDaten();  
                break;  
            case "7":  
                Console.Clear();  
                for (int i = 0; i < AlleMitarbeiter.BenutzerListe.Count; i++)  
                    Console.WriteLine($"{i + 1}. {AlleMitarbeiter.BenutzerListe[i].U  
                Console.Write("Nummer: ");  
                if (int.TryParse(Console.ReadLine(), out int idx) && idx > 0 && idx <  
                {  
                    AlleMitarbeiter.BenutzerListe.RemoveAt(idx - 1);  
                    AlleMitarbeiter.SpeichereDaten();  
                }  
                break;  
            case "8":  
                return;  
            default:  
                Console.ForegroundColor = ConsoleColor.Red;  
                Console.WriteLine("\u262f Ung\u00fcltige Eingabe!");  
                Console.ResetColor();  
                Console.ReadKey();  
                break;  
        }  
    }  
}
```

# Materialverwaltung im Detail

01

## Material hinzufügen

Name, Artikelnummer, Lagerplatz, Bestand erfassen – automatische JSON-Aktualisierung

02

## Material bearbeiten

Alle Felder \u00e4nderbar durch Admin, Eingabevalidierung verhindert Fehler

03

## Material löschen

Auswahl per Listenposition, Sicherheitsabfrage vor endg\u00fcltiger L\u00f6schung

04

## Bestand anzeigen

\u00d6bersichtliche Darstellung aller Artikel mit Echtzeitbest\u00e4nden

AlleMitarbeiter.cs Admin.cs Mitarbeiter.cs Lager.cs Material.cs Warenkorb.cs

```
g System;
g System.Collections.Generic;
g System.Linq;
g System.Text.Json;

namespace Lagerverwaltung

internal class WarenkorbMitarbeiter
{
    public string Artikelnummer { get; set; }
    public int Menge { get; set; }
}

internal class Warenkorb
{
    public List<WarenkorbMitarbeiter> Items { get; set; } = new List<WarenkorbMitarbeiter>();
    public static string FilePath { get; } = "Warenkorb.json";

    public void Add(Material m, int menge = 1)
    {
        if (m == null || menge <= 0) return;
        var pos = Items.FirstOrDefault(i => i.Artikelnummer == m.Artikelnummer);
        if (pos == null)
            Items.Add(new WarenkorbMitarbeiter { Artikelnummer = m.Artikelnummer, Menge = menge });
        else
            pos.Menge += menge;
    }

    public void Checkout(List<Material> materialien)
    {
        foreach (var item in Items)
        {
            var mat = materialien.FirstOrDefault(m => m.Artikelnummer == item.Artikelnummer);
            if (mat == null)
            {
                Console.WriteLine($"Material mit Id {item.Artikelnummer} nicht gefunden");
                continue;
            }

            if (mat.Bestand >= item.Menge)
            {
                mat.Bestand -= item.Menge;
            }
            else
            {
                Console.WriteLine($"Nicht genügend Bestand von {mat.Bezeichnung}. Gehen Sie auf Lagerbestand ein.");
            }
        }

        Items.Clear();
        Console.WriteLine("Checkout abgeschlossen. Bestand wurde aktualisiert.");
        Console.WriteLine("\nBitte weiter mit Enter");
        Console.ReadKey();
    }
}
```

# Warenkorb-Prozess

## Hinzufügen

- Artikelauswahl aus Materialliste
- Mengenangabe durch Mitarbeiter
- Speicherung als WarenkorbPosition
- Automatische Mengenerhöhung bei doppeltem Artikel

## Checkout

- Bestandsprüfung vor Entnahme
- Materialbestand wird reduziert
- Warnung bei unzureichendem Bestand
- Warenkorb-Leerung nach Abschluss
- JSON-Dateien werden aktualisiert

## &lt;&gt;&lt;&gt;&lt;&gt;&gt;&gt; Admin Bereich (Dave) &lt;&lt;&lt;&lt;&gt;&lt;&gt;&lt;&gt;&gt;&gt;

Material anzeigen  
Material hinzufügen  
Material bearbeiten  
Material löschen  
Benutzer anzeigen  
Benutzer hinzufügen  
Benutzer löschen

Logout  
nl: |

# Bedienungsablauf

## Admin-Menü

1. Material anzeigen
2. Material hinzufügen
3. Material bearbeiten
4. Material löschen
5. Benutzer anzeigen
6. Benutzer hinzufügen
7. Benutzer löschen
8. Logout

## Mitarbeiter-Menü

1. Material anzeigen
2. Zum Warenkorb hinzufügen
3. Warenkorb anzeigen / Checkout
4. Logout

# Herausforderungen / Schwierigkeiten

1. Logik im Blick behalten
2. Sich nicht von der KI verleiten lassen
3. genaueres lesen der ausgegebenen Vorschläge

The screenshot shows a Microsoft Visual Studio interface with the following details:

- Code Editor:** The main window displays the file `Lagerverwaltung\Login.cs`. The code implements a login system with a list of users. A tooltip "Tabstop zum Akzeptieren (2/2) Alt+Entf zu verwerfen F8 zum nächsten wechseln" is visible over some code.
- GitHub Copilot Chat:** A sidebar window titled "GitHub Copilot-Chat" contains a conversation:
  - Question: "Wie kann ich # verbessern? Code für Verbesserungen vorschlagen"
  - Answer: "Rückgabewert: Rückgabe des Basistyps `AlleMitarbeiter?` statt konkretem `Mitarbeiter`, damit Admin korrekt zurückgegeben wird."
- Code Completion:** Below the answer, a completion suggestion for the code is shown:

```
C# Lagerverwaltung>Login.cs Anwenden

using System;
using System.Collections.Generic;
using System.Linq;
namespace Lagerverwaltung
{
    internal class Login
    {
        private readonly List<AlleMitarbeiter> _benutzer = new List<AlleMitarbeiter>();
```
- Output Window:** The bottom-left window shows build logs:

```
Ausgabe
Ausgabe anzeigen von: Build
1>C:\Users\Dave\source\repos\AbschlussProjekt\Lagerverwaltung\Lagerverwaltung\Lagerverwaltung\Program.cs(21,18,21,25): warning CS0219: Die Variable "laufend" ist zugewiesen, ihr Wert wird aber nie verwendet.
1>C:\Users\Dave\source\repos\AbschlussProjekt\Lagerverwaltung\Lagerverwaltung\Lagerverwaltung\Admin.cs(69,25,69,32): warning CS0162: Unerreichter Code wurde entdeckt.
1> Lagerverwaltung -> C:\Users\Dave\source\repos\AbschlussProjekt\Lagerverwaltung\Lagerverwaltung\Lagerverwaltung\bin\Debug\net9.0\Lagerverwaltung.dll
===== Build: 1 erfolgreich, 0 Fehler, 0 aktuell, 0 überprüfung =====
===== Erstellen abgeschlossen um 10:52 und dauerte 04,968 Sekunden =====
```
- Bottom Bar:** The bottom bar includes tabs for "Aktives Dokument" and "Fragen Sie Copilot, oder verwenden Sie @workspace". It also features a "Fragen" button and a "GPT-5 mini" dropdown.

# Löschen/ Verschieben ganzer Zeilen

dadurch schnell immer wieder den Logiküberblick verloren

The screenshot shows a code editor window with a dark theme. The main pane displays a C# file named 'Lagerverwaltung.cs'. A large portion of the code, specifically from line 20 to line 27, is highlighted in red, indicating it is selected for deletion. Above this selection, there is a toolbar with buttons for 'Änderungen' (Changes), '-23' (Delete 23 lines), '+21' (Insert 21 lines), and options to 'bereitstellen' (Prepare) or 'rückgängig machen' (Undo). To the right of the main pane, a preview area shows the state of the code after the changes are applied. The bottom status bar indicates the current line and character position (Zeile: 23, Zeichen: 1) and the file encoding (UTF-8 with BOM).

```
1 iem;
2 em.Linq;
3
4 Lagerverwaltung
5
6 al class Lager
7
8 blic void FuegeMaterialHinzu(string name, int menge, string lagerplatz, string artikelnummer, int bestand)
9
10 if (string.IsNullOrWhiteSpace(name) || bestand < 0)
11 {
12     Console.WriteLine("Ungültige Eingabe. Bitte Name und Bestand überprüfen.");
13     Console.WriteLine("\nBitte weiter mit Enter");
14     Console.ReadKey();
15 }
16
17
18 Material.MaterialListe.Add(new Material(name, artikelnummer, lagerplatz, menge, bestand));
19 Material.SpeichereDaten();
20 Console.WriteLine("Material wurde zum Bestand hinzugefügt.");
21 Console.WriteLine("\nBitte weiter mit Enter");
22 Console.ReadKey();
23
24 else
25 {
26     Console.WriteLine("Ungültige Eingabe. Bitte überprüfen Sie den Namen u");
27 }
28
29
30 Material.MaterialListe.Add(new Material(name, artikelnummer, lagerplatz, menge, bestand));
31 Material.SpeichereDaten();
32 Console.WriteLine("Material wurde zum Bestand hinzugefügt.");
33 Console.WriteLine("\nBitte weiter mit Enter");
34 Console.ReadKey();
35
36 blic void ZeigeMaterial()
```

# Sicherheit & Verbesserungspotenzial

## Aktuelle Sicherheit

- Passwörter im Klartext (für Vorführzwecke)
- Keine SQL-Injection möglich
- Geschützte Dateizugriffe via File.Exists
- Nur für lokale, kontrollierte Umgebungen

## Verbesserungsmöglichkeiten

- **Passwort-Hashing:** BCrypt-Implementierung
- **Datenbank:** Migration zu SQLite
- **UI-Modernisierung:** WPF oder Weboberfläche
- **Rollenerweiterung:** Lagerleiter, Qualitätskontrolle
- **Logging:** Transaktionsprotokoll für Nachverfolgbarkeit

# Fazit & Ausblick



Funktionales System für kleine bis mittlere Anwendungsfälle

## Erreicht

Alle Kernprozesse implementiert,  
wartungsfreundliche Architektur,  
einfache JSON-Handhabung

## Limitation

Eingeschränkte Skalierbarkeit  
durch JSON, nur für kontrollierte  
Umgebungen geeignet

## Empfehlung

Ideal für Prototypen,  
Schulungszwecke und kleinere  
Lager mit überschaubarem  
Materialbestand

## Reflektion

mehr Verständnis C#  
Logik im Auge behalten  
genauere Ausarbeit bei der  
Projektplanung

Vielen Dank für`s Zuhören.