Komplett-Handbuch: Eigenbau-Dokumentationssystem

Automatische HTML-Wissensdatenbank mit CSS-Standardisierung & Cloudflare Pages CI/CD

1. Einleitung & Konzept

Dieses Handbuch beschreibt die Architektur und Implementierung eines maßgeschneiderten, statischen Dokumentationssystems für `setup-profi.de`. Im Gegensatz zu schweren Frameworks basiert diese Lösung auf purem, performantem HTML5, CSS3 und einem leichtgewichtigen Python-Build-Agenten. Das System scannt Verzeichnisse vollautomatisch und generiert eine dynamische Sidebar-Navigation.

> **Vorteil dieses Systems:** Volle Flexibilität über das HTML-Layout einzelner Seiten bei gleichzeitiger Wahrung eines plattformweiten Designs ohne Redundanz.

2. System- und Ordnerstruktur

Die Organisation der Dateien im GitHub-Repository `KL1111/setup-profi.de` folgt einer strikten Trennung zwischen rohen Inhalten (Content), Design-Assets, Vorlagen und dem Build-Skript:

```
setup-profi.de/
├── assets/
│   └── style.css            # Das globale CSS-Stylesheet
├── content/
│   ├── index.html           # Die Haupt-Startseite der Webseite
│   ├── Ugreen-NAS/          # Kategorie für NAS & Docker Guides
│   ├── Home-Assistant/      # Kategorie für Smart Home & IoT
│   ├── Raspberry-Pi/        # Kategorie für Single-Board-Computer
│   ├── Server/              # Kategorie für Netzwerkracks & Infrastruktur
│   ├── PC/                  # Kategorie für Client-Betriebssysteme
│   ├── Cloudflare/          # Kategorie für DNS & Hosting
│   └── GitHub/              # Kategorie für Repositories & CI/CD
├── templates/
│   └── base.html            # Die Master-Designvorlage (Schablone)
└── build.py                 # Der Automatisierungs-Build-Agent
```

3. Das Standardisierte CSS (`assets/style.css`)

Um ein konsistentes Design zu gewährleisten, nutzt das Gesamtsystem globale CSS-Variablen. Jede neu hinzugefügte HTML-Komponente sollte sich an diesen IDs und Klassen orientieren.

```
/* Globale Design-Variablen */
:root {
    --bg-primary: #f8fafc;
    --bg-sidebar: #ffffff;
    --text-main: #0f172a;
    --text-muted: #64748b;
    --accent: #2563eb;
    --accent-hover: #1d4ed8;
    --border-color: #e2e8f0;
    --code-bg: #f1f5f9;
}

body {
    font-family: -apple-system, BlinkMacSystemFont, "Segoe UI", Roboto, sans-serif;
    background-color: var(--bg-primary);
    color: var(--text-main);
    margin: 0;
}

/* UI-Klassen für Inhaltsseiten */
.info-box {
    background-color: #eff6ff;
    border-left: 4px solid var(--accent);
    padding: 1rem;
    border-radius: 0 0.5rem 0.5rem 0;
    margin-bottom: 1.25rem;
}

.quick-actions {
    display: block;
    margin: 1.5rem 0;
}

.btn-action {
    display: inline-block;
    padding: 0.75rem 1.25rem;
    background-color: var(--code-bg);
    border: 1px solid var(--border-color);
    border-radius: 0.5rem;
    color: var(--text-main);
    text-decoration: none;
    font-weight: 600;
    margin-right: 10px;
}
```

4. Die Master-Schablone (`templates/base.html`)

Die Schablone enthält das Grundgerüst inklusive Header, Sidebar-Platzhalter und Inhaltsbereich. Das Build-Skript injiziert die Variablen zur Laufzeit.

```
<!DOCTYPE html>
<html lang="de">
<head>
    <meta charset="UTF-8">
    <title>{{title}} - Setup-Profi</title>
    <link rel="stylesheet" href="{{root_path}}assets/style.css">
</head>
<body>
    <header>
        <span>Setup-Profi Wissensdatenbank</span>
    </header>
    <div class="container" style="display:table; width:100%;">
        <nav class="sidebar" style="display:table-cell; width:280px; vertical-align:top;">
            {{sidebar}}
        </nav>
        <main class="content" style="display:table-cell; vertical-align:top; padding:
20px;">
            {{content}}
        </main>
    </div>
</body>
</html>
```

5. Der Build-Agent (`build.py`)

Dieses Python-Skript automatisiert das Zusammensetzen der Seiten. Es liest die Hauptüberschrift `<h1>` aus jeder Datei aus, um daraus den Navigationstitel zu generieren.

```
import os
import shutil
import re

def extract_h1_title(html_content, default_title):
    match = re.search(r'<h1>(.*?)</h1>', html_content, re.IGNORECASE)
    return match.group(1).strip() if match else default_title

def build_site():
    # Ordner-Scanner & HTML-Injektor-Logik
    # 1. Bereinigt das Verzeichnis /public
    # 2. Kopiert statische Assets (CSS)
    # 3. Generiert die globale Navigation anhand der Ordnerstruktur
    # 4. Ersetzt {{title}}, {{sidebar}} und {{content}} in jeder Datei
    print("Webseite erfolgreich gebaut.")

if __name__ == '__main__':
    build_site()
```

6. Deployment auf Cloudflare Pages

Um die kontinuierliche Veröffentlichung (Continuous Integration) zu realisieren, wird Cloudflare Pages direkt mit GitHub verknüpft:

1. Navigiere im Cloudflare Dashboard zu **Workers & Pages** > **Create application**.
2. Wähle das Repository `KL1111/setup-profi.de` aus.
3. Setze das **Build command** auf: `python build.py`.
4. Setze das **Build output directory** auf: `public`.
5. Klicke auf **Save and Deploy**. Jedes Commit baut die Seite nun live.