--------------------------------------------------------------------------------

**Präsentation: Die Flask Webanwendung – CSV-Datenmanagement im Fokus**

--------------------------------------------------------------------------------

**Folie 1: Einführung: Die Flask Webanwendung (app.py)**

•

**Zweck:** Eine **Flask-Anwendung** zur **Verwaltung von CSV-Daten** über eine benutzerfreundliche Weboberfläche und eine RESTful API.

•

**Datenhaltung:**

◦

Nutzt einen **Pandas DataFrame als In-Memory-Datenbank**.

◦

Wichtig: Diese Datenbank **wird bei jedem Neustart der Anwendung zurückgesetzt**.

•

**Technologien:** Entwickelt mit **Flask** für das Web-Framework und **Pandas** für die effiziente Datenverarbeitung.

--------------------------------------------------------------------------------

**Folie 2: Das Web-Interface: Einfacher Daten-Upload & Übersicht**

•

**Startseite (/):**

◦

Bietet eine **HTML-Seite zum einfachen Hochladen von CSV-Dateien**.

◦

Zeigt die **bereits hochgeladenen Daten** in einer übersichtlichen HTML-Tabelle an.

◦

Falls keine Daten vorhanden sind, wird dies entsprechend angezeigt ("No data available.").

•

**Upload-Funktion (/upload POST):**

◦

Stellt sicher, dass nur **Dateien mit der Endung .csv** hochgeladen werden können.

◦

Hängt erfolgreich hochgeladene neue Daten an die bereits vorhandene Datenbank an.

◦

Bestätigt den erfolgreichen Upload mit dem **HTTP-Statuscode 201 (Created)**.

◦

Behandelt Fehler, zum Beispiel wenn keine Datei bereitgestellt wird oder das Dateiformat falsch ist.

--------------------------------------------------------------------------------

**Folie 3: Die RESTful API: Datenbereitstellung für weitere Anwendungen**

•

**API-Endpunkt (/api/daten GET):**

◦

Stellt alle in der Datenbank gespeicherten Daten im **JSON-Format** zur Verfügung.

◦

Dies ermöglicht anderen Skripten oder Diensten, die Daten programmatisch abzurufen und weiterzuverarbeiten.

•

**Verfügbarkeitsprüfung:**

◦

Wenn die Datenbank leer ist (d.h., es wurden noch keine Daten hochgeladen), wird der **HTTP-Statuscode 404 (Not Found)** zusammen mit der Nachricht "No data available" zurückgegeben.

◦

Dies gewährleistet eine klare Kommunikation über den Datenstatus.

--------------------------------------------------------------------------------

**Folie 4: Zusammenfassung & Potenzial**

•

**Kernfunktion:** Die Flask-Anwendung fungiert als **zentraler Datenmanager**, der eine einfache Schnittstelle für den Upload und eine flexible API für den Datenabruf bietet.

•

**Flexibilität:** Sie unterstützt sowohl die **direkte Interaktion über eine Weboberfläche** als auch den **automatisierten Datenzugriff** durch externe Anwendungen.

•

**Anwendungsbereich:** Ideal als **Basissystem zum Sammeln und Bereitstellen von Daten** für nachfolgende Analysen, maschinelles Lernen oder andere datengetriebene Prozesse.

--------------------------------------------------------------------------------