Praktikum Prozessautomatisierung

Das Ziel der Prozessautomatisierung ist es, Geschäftsprozesse möglichst weitgehend zu automatisieren. Ein typischer Geschäftsprozess umfasst meist mehrere Systeme oder Services: Ein Kunde füllt ein Formular aus, die Daten müssen sicher in einer Datenbank abgelegt werden, und anschließend müssen Berechnungen (z.B. Preise order Rabatte) durchgeführt werden.

In diesem Praktikum lernen Sie, wie man verschiedene Anwendungen und Services miteinander verbindet. Nach Abschluss diese Praktikums haben Sie einen Prozess implementiert der folgende Schritte umfasst:

1. Erfassung: Daten werden über ein digitales Formular erfasst.
2. Speicherung: diese Daten werden in einer Datenbank gespeichert, um sie dauerhaft (persistent) verfügbar zu machen.
3. Verarbeitung: es wird eine eine Funktion aufgerufen, die auf Basis der Daten eine Berechnung durchführt.

Um diesen Prozess abzubilden, nutzen Sie einen modernen Software-Stack, wie er auch zur Prozessautomatisierung im Unternehmen zum Einsatz kommt.

- *n8n*: n8n ist die Prozess-Engine. Es steuert den Prozess und sorgt dafür, dass die Daten zur richtigen Zeit am richtigen Ort ankommen. Es ist ein mächtiges Werkzeug für die Prozessautomatisierung, das auch ohne tiefe Programmierkenntnisse bedienbar ist.
- *PostgreSQL*: Eine professionelle, relationale Datenbank, die im Prozess erzeugte Daten speichert.
- *Adminer*: Ein webbasiertes Werkzeug, mit dem Sie Ihre Datenbank verwalten können.
- *Docker*: Damit Sie sich nicht mit komplizierten Installationen auf verschiedenen Betriebssystemen aufhalten, nutzen wir Docker. Es stellt sicher, dass unsere Automatisierungsumgebung bei jedem von Ihnen exakt gleich aussieht und funktioniert.

Der Prozess im Detail

In den folgenden Schritten bauen Sie ein System, das eine Kundenbestellung simuliert. Dabei werden Sie sehen, wie n8n die Daten aus dem Formular liest, sie in die Datenbank schreibt und am Ende über eine Logik-Funktion den korrekten Validierungscode für Ihre Abgabe berechnet.

Vorbereitung

Bevor Sie mit der Implementierung des Prozesses starten, müssen Sie einen Ort definieren, an dem die Datenbank die Daten speichert. Gehen Sie hierzu wie folgt vor:

1. Erstellen Sie auf Ihrem Rechner (z. B. unter “Dokumente”) einen Ordner `praktikum-is`.

2. Erstellen Sie innerhalb dieses Ordners einen Unterordner namens `praktikum-01`.
3. Erstellen Sie in diesem Unterordner `praktikum-01` eine neue Textdatei und nennen Sie diese `docker-compose.yml`.
4. Kopieren Sie den folgenden Inhalt exakt in diese Datei und speichern Sie diese ab (auf korrekte Einrückung achten!):

```yaml
services:
  db:
    image: postgres:16-alpine
    restart: always
    environment:
      POSTGRES_USER: is_user
      POSTGRES_PASSWORD: is_pass
      POSTGRES_DB: praktikum_db
    volumes:
      - ./postgres_data:/var/lib/postgresql/data

  adminer:
    image: adminer
    restart: always
    ports:
      - "8080:8080"

  n8n:
    image: n8nio/n8n:latest
    restart: always
    ports:
      - "5678:5678"
    environment:
      - N8N_HOST=localhost
    volumes:
      - ./n8n_data:/home/node/.n8n
    depends_on:
      - db
```

Hinweis: Achten Sie auf die korrekte Bezeichnung der Ordner und Dateien. Insbesondere auch Groß- und Kleinschreibung beachten!

Schritt 1: n8n und Datenbank in Docker starten

Um die gesamte Umgebung (*n8n, Datenbank und Adminer*) zu aktivieren, nutzen wir einen kurzen Befehl über die Kommandozeile Ihres Rechners. Dies ist der sicherste Weg, um alle Dienste gleichzeitig zu starten und korrekt miteinander zu verknüpfen.

1. *Terminal öffnen*:

- *Windows*: Suchen Sie im Startmenü nach *PowerShell* oder *Eingabeaufforderung* und öffnen Sie diese.
- *macOS*: Suchen Sie über die Spotlight-Suche (Befehl + Leertaste) nach *Terminal* und öffnen Sie es.

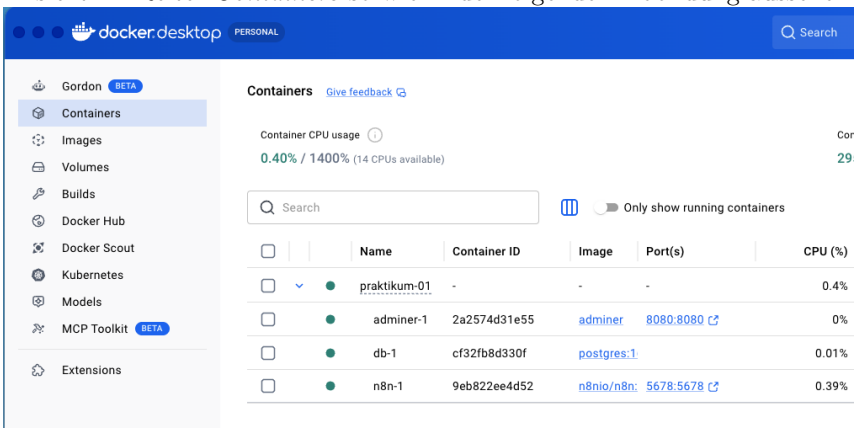
2. *In den Projektordner navigieren*: Geben Sie den Befehl `cd` (steht für *change directory*) gefolgt von einem Leerzeichen ein. Ziehen Sie nun Ihren Ordner `praktikum-01` aus dem Dateimanager (*Explorer* bei Windows oder *Finder* bei Mac) direkt in das Terminalfenster. Der Pfad zum Ordner wird automatisch eingefügt. Drücken Sie die *Enter-Taste*.

3. *Container starten*: Geben Sie nun exakt folgenden Befehl ein und bestätigen Sie erneut mit *Enter*:

   ```
   docker compose up -d
   ```

   > *Hinweis*: Docker lädt nun im Hintergrund die notwendigen Programmpakete herunter und startet die Dienste. Dies kann beim ersten Mal ein paar Minuten dauern.

4. *Verwaltung über Docker Desktop*: Sobald der Befehl abgeschlossen ist, können Sie das Terminal schließen. Ab jetzt müssen Sie keine Befehle mehr tippen. Öffnen Sie *Docker Desktop* und wechseln Sie in den Reiter *Containers*. Dort finden Sie Ihr Projekt (`praktikum-01`) und können die gesamte Umgebung in Zukunft ganz einfach per Mausklick *starten*, *stoppen* oder *neu starten*. Falls alles funktioniert hat sollte die Ansicht im Reiter *Containers* so wie in der folgenden Abbildung aussehen:



5. Sie erreichen die Automatisierungsoberfläche von n8n nun unter: http://localhost:5678. Adminer, das Tool zur Verwaltung der Datenbank ist unter http://localhost:8080 erreichbar.

Schritt 2: Datenbank erstellen

Bevor n8n im Prozess Daten speichern kann, müssen Sie die Tabelle in der Datenbank anlegen. Wir nutzen dafür Adminer direkt im Browser.

1. Öffnen Sie http://localhost:8080.
2. Loggen Sie sich mit folgenden Daten ein:
   - System: PostgreSQL
   - Server: db
   - Benutzer: is_user
   - Passwort: is_pass
   - Datenbank: praktikum_db
3. Klicken Sie links auf SQL-Kommando und fügen Sie diesen Befehl in das Textfeld ein:

```
CREATE TABLE orders (
  id INTEGER GENERATED ALWAYS AS IDENTITY PRIMARY KEY,
  kunde TEXT,
  produkt INTEGER,
  menge INTEGER,
  gesamtpreis FLOAT
);
```

   Klicken Sie auf den `Execute`-Button um Befehl auszuführen.

Schritt 3: Prozess in n8n automatisieren

Öffnen Sie n8n im Browser (http://localhost:5678). Wir bilden nun den Prozess bestehend aus den Schritten Erfassung -> Speicherung -> Verarbeitung ab.

1. Erfassung (Form Trigger)
   - Fügen Sie einen Knoten n8n Form Trigger hinzu.
   - Geben Sie der Form den Namen `Bestellformular`.
   - Erstellen Sie über Add Field drei Felder: Benutzername (Typ: Text Input), Produkt-ID (Typ: Number) und Menge (Typ: Number).
   - Klicken Sie auf *Test this trigger* und geben Sie Testdaten ein.
2. Externen Preis abfragen (HTTP Request)
   - Fügen Sie einen Knoten HTTP Request hinzu (Methode: GET).
   - URL: Klicken Sie auf `Expression` und fügen Sie folgende URL ein: `https://dummyjson.com/products/{{ }}`
   - Ziehen Sie dann die Produkt-ID aus dem Input zwischen die geschweiften Klammern. Die resultierende URL sieht wie folgt aus: `https://dummyjson.com/products/{{ $json['Produkt-ID'] }}`
   - Testen Sie den Schritt.

     Das Ausführen diese Schrittes simuliert den Aufruf eines externen Warenwirtschaftssystems mit den im Formular eingegebenen Daten, in diesem Fall die Produkt-ID.

3. Gesamtpreis berechnen
   - Fügen Sie einen Knoten *Edit Fields* hinzu.
   - Erstellen Sie ein Feld Gesamtpreis (Typ: Number).
   - Nutzen Sie als Expression:
     `{{ $json.price * $('On form submission').item.json.Menge }}`
     Diese können Sie auch durch Drag & Drop der Felder erstellen.
   - Testen Sie den Schritt. Als Ergebnis sollte der Gesamtpreis als Produkt aus Preis und Menge berechnet werden.
4. Speicherung
   - Fügen Sie einen Knoten Postgres hinzu (Typ: Insert rows in a table)
   - Erstellen Sie neue Credentials (Zugangsdaten) mit folgenden Parametern:
     - Host: db
     - Database: praktikum_db
     - User: is_user
     - Password: is_pass
   - Wählen Sie als Tabelle die Tabelle `order` aus.
   - Entfernen sie den Primärschlüssel (die Spalte `id`). Diese wird von der Datenbank automatisch erstellt.
     Hinweis: Wichtig! Wenn Sie die Spalte nicht entfernen kommt es bei der Ausführung des Prozesses zu Fehlern.
   - Mappen Sie die Spalten der Datenbank per Drag & Drop auf die entsprechenden Werte aus Ihrem Workflow. Das Feld `Benutzername` aus dem Workflow wird auf das Feld `kunde` in der Datenbank abgebildet.
5. Testen und Validieren des DB-Inhalts
   - Führen Sie Ihren Workflow nun einmal vollständig aus, indem Sie unten auf Execute Workflow klicken und die Daten im Formular absenden.
   - Um zu prüfen, ob die Daten tatsächlich dauerhaft gespeichert wurden, nutzen wir das Tool Adminer.
     - Öffnen Sie Ihren Browser unter http://localhost:8080.
     - Loggen Sie sich erneut ein (Server: db, Benutzer: is_user, Passwort: is_pass, Datenbank: praktikum_db).
     - Klicken Sie in der linken Spalte auf die Tabelle orders und wählen Sie dann den Punkt Daten auswählen.
     - Erfolgskontrolle: Wenn Ihre im Formular eingegebenen Testdaten nun als neue Zeile in der Tabelle erscheinen, war die systemübergreifende Speicherung erfolgreich.

Schritt 4: Abgabe

Um das Praktikum erfolgreich abzuschließen, müssen Sie einen individuellen Validierungscode generieren. Dieser Code dient als digitaler Nachweis, dass Ihr Prozess korrekt konfiguriert wurde und die Daten erfolgreich verarbeitet werden.

1. Fügen Sie einen neuen Knoten vom Type Edit Field ein.
2. Neues Feld für den Abgabecode hinzufügen:
   - Geben Sie als Namen für das Feld `ILIAS_ABGABE_CODE` und als Typ `String` ein.
   - Kopieren Sie folgende Formal als `Expression` in das entsprechende Feld.

```
{{ ($('On form submission').item.json.Benutzername +
  "IS-PRAKTIKUM-2026" +
  $('Edit Fields').item.json.Gesamtpreis)
  .hash('sha256').substring(0, 15).toUpperCase() }}
```

3. Der komplette Prozess sieht nun ähnlich dem in folgender Abbildung gezeigten Prozesse aus.

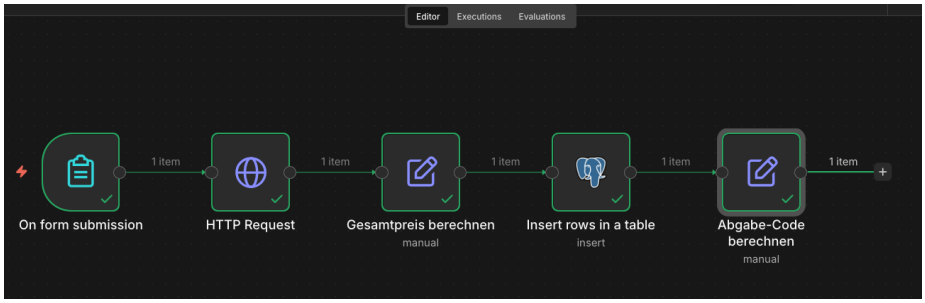


Figure 1: Prozess nach Abschluss des Praktikums

4. Durchführung der Abgabe:
   - Klicken Sie unten auf Execute Workflow.
   - Geben Sie im Formular Ihren Benutzername für Ilias, die Produkt_ID `124` und die Menge 5 ein.
   - Der Workflow läuft durch. Klicken Sie auf die letzte Knoten (Code), um Ihren `ILIAS_ABGABE_CODE` zu sehen.
   - Tragen Sie diesen Code im ILIAS-Test ein.