

Project: Invoice2Pay – Intelligent Invoice Data Extraction & Reconciliation Platform

Multi-Agent AI Workflow

Technologies: n8n, Python, LLM, Gen AI, AI Foundary

Agent 1: Extraction Specialist

- Extracts clean, structured invoice data from PDFs and scanned images.
- Uses **pdfplumber** for digital PDFs and **Tesseract OCR** for images.
- Sends extracted text to **GPT-4o** for schema-based JSON generation.
- Handles multi-page invoices, multiple line items, and inconsistent vendor formats.
- Normalizes values (dates, currency, tax, amounts) for downstream validation.
- Outputs a well-formatted JSON ready for rule checks.

Agent 2: Validation Agent

- Performs business-rule validation using SAP PO and GR master data.
- Executes vendor, customer, price, quantity, tax, and GR posting checks.
- Implements tolerance rules (e.g., $\pm 2\%$ price deviation).
- Detects mismatches such as over-billing, missing GR, incorrect vendor, etc.
- Returns:
 - `Validation_Status` (Pass/Fail)
 - List of discrepancies with clear reason codes.
- Routes failed cases to Human-in-the-Loop (HIL) review.

Agent 3: SAP Posting Agent

- Converts validated invoice JSON into SAP ODATA/BAPI-compatible payload.
- Handles creation of invoice documents in SAP FI/MM.
- Ensures duplicate prevention by checking existing invoice numbers.
- Returns the SAP Invoice Document Number after successful posting.
- Updates the frontend dashboard with final posting status and audit logs.
- Provides automatic retry logic in case SAP is temporarily unavailable.

Project: Generative-AI-Content-Creation

Technologies: N8N, LLM, Gen AI, Hugging Face, Groq AI, Data Modeling

Developed an automated end-to-end workflow in n8n that transforms a simple topic string into a complete LinkedIn-ready blog post and matching visual asset using Generative AI. The workflow integrates Groq Chat AI for text generation and Hugging Face AI for image creation, producing high-quality, ready-to-publish content in under 20 seconds.

Github Reference: <https://github.com/prithvirajpatil29/Generative-AI-Content-Creation>

- **Generates 250-word LinkedIn-style blog posts with hashtags and CTA.**
- Creates visual assets (images) matching the blog topic.

- Ensures style variety for both text and visuals (listicle, story, question, contrarian; minimalist, corporate, neon, infographic).
- Fully automated, end-to-end workflow (<20 seconds per topic).
- Stores outputs in a DataTable for tracking and retrieval.
- Returns structured JSON for easy integration with front-end apps or CMS.

Project: Portfolio Management System Automation

Technologies: Power Automate Desktop (PAD), Power Automate Cloud, Python, AWS S3, Outlook

- Designed and implemented a complete automation solution for daily portfolio file processing.
- Automated file retrieval from AWS S3, organizing them into a structured, date-wise folder hierarchy for daily operational tracking.
- Developed data validation workflows using Python to check Excel files for common issues such as #N/A, #VALUE!, or missing fields.
- Applied business rules shared by the Operations team to clean and process the input files, generating validated output files.
- Automated web uploads of the final output files to the client portal using PAD web automation.
- Integrated email notifications for each Asset Management Company (AMC).
Notifications included either:
 - Valid files → Triggered email with Approve/Reject buttons using Outlook Adaptive Cards.
 - Invalid files → Sent error summary reports for manual correction.
- Leveraged Power Automate Cloud to schedule PAD desktop flows on a daily basis, eliminating manual intervention.
- Achieved 90% reduction in manual effort and improved accuracy and turnaround time for daily operations.

Project: Automated Reconciliation System

Technologies: Power Automate Cloud, Power Automate Desktop (PAD), Python, Dataverse, AI (LLM-based Invoice Extraction)

- **Solution Design & Architecture:**
 - Designed the reconciliation workflow architecture integrating **UI, automation, and backend logic**.
 - Defined the data flow between **bank transaction sources, GL data, and reconciliation engine**, ensuring smooth communication between components.
- **Rule-Based Reconciliation Logic (Python):**

- Developed a **rule-driven reconciliation engine in Python** to match transactions between bank and ledger data.
- Designed a **configurable rule structure**, allowing dynamic creation of matching criteria such as:
 - Date tolerance
 - Amount difference thresholds
 - Description or reference-based matching
- Built logic to identify **matched items, unmatched (open) items, and partial matches**, with detailed output reports.
- Displayed reconciliation summary metrics such as **total processed, matched, and unmatched amounts** in the final report.
- **Dynamic Rule Configuration via JSON:**
 - Created a **user interface (UI)** for defining reconciliation rules dynamically.
 - These rules were automatically converted to **JSON format**, which acted as input for the Python-based reconciliation engine — enabling flexibility to handle new business requirements without changing the core code.
- **AI-Powered Invoice Data Extraction (AR/AP Integration):**
 - Integrated an **AI Large Language Model (LLM)**-based feature to extract key data from **Accounts Receivable (AR)** and **Accounts Payable (AP)** invoices.
 - The extracted data was automatically cleaned and dumped into the **ledger system**, eliminating manual entry and reducing reconciliation errors.
 - Improved data ingestion accuracy and processing speed for bulk invoice entries.

Project: Dynamic Scheduling of Processes (Time-Zone Based Triggering)

Technologies: Power Automate Cloud, PAD, Dataverse

- Designed a time-based scheduling system to trigger automation processes dynamically across multiple time zones.
- Maintained a Dataverse table containing execution times (in 15-minute intervals) for different regions.
- Configured a cloud flow running every 15 minutes to:
 - Fetch scheduling data from Dataverse.
 - Compare the current system time with scheduled time.
 - Trigger PAD flows only for regions matching the exact time slot.

- This approach allowed timezone-aware scheduling, ensuring automation tasks executed at the right local time for global users.
- Reduced dependency on static schedules, enabling flexibility and scalability for multiple processes and regions.

Project: Email Approval Workflow with Custom HTML Buttons

Technologies: Power Automate Desktop, Outlook Adaptive Cards

- Implemented customized approval workflows by embedding Approve/Reject HTML buttons in emails.
- Emails were auto-generated by PAD and sent to operations teams for decision-making.
- On clicking “Approve”: The workflow automatically resumed and executed the next set of processes.
- On clicking “Reject”: A separate workflow was triggered to log rejection reasons and notify stakeholders.
- This eliminated the need for manual confirmation emails, creating a seamless decision-making process and reducing approval cycle time.

Project: Chatbot Development using Microsoft Copilot

Technologies: Copilot Studio, PDF-based training, Prompt Engineering

- Designed and deployed a knowledge-based chatbot using Microsoft Copilot.
- Trained the chatbot on PDF documents containing domain-specific content.
- Implemented reasoning capabilities to answer user queries contextually.
- Enhanced operational efficiency by allowing business users to quickly retrieve information without manual document search.

Project: Large-Scale Data Extraction from Real Estate Platforms

Technologies: Power Automate Desktop, Excel

- Automated the extraction of 4,000+ property records from platforms such as 99acres.
- PAD flows simulated real-user interactions like login, navigation, scrolling, and pagination to handle dynamic and infinite-scroll content.
- Extracted structured property data (location, price, type, size, amenities, etc.).
- Exported the processed data into Excel files, formatted as per the schema provided.
- Delivered a clean dataset ready for business analysis and reporting, replacing hours of manual web data collection.