Electrolysis-Agent-Report

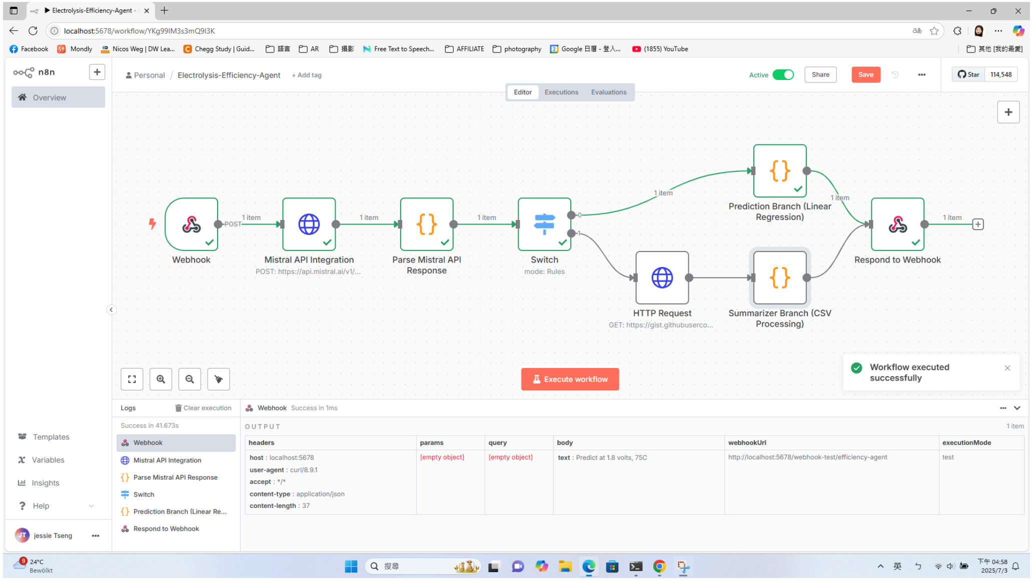
System Architecture

The system is built using **n8n** and consists of the following components:

1. **Webhook**– Accepts free-text requests from users.
2. **Mistral API Integration** – Sends the user's input to a hosted **LLM model** (e.g., Mistral) that interprets the request.
3. **Parse Mistral API Response** – Extracts structured information from the LLM's response, including:
   - `intent` (either `"predict_run"` or `"summarize_best"`)
   - relevant numeric parameters such as `voltage` and `temperature`
4. **Switch** – Directs the workflow based on the extracted `intent.`
5. **1.1 Prediction Branch** – Implements a pre-trained linear regression model with fixed coefficients.

   2.1 HTTP Requestre – trieves a public CSV file of 50 experimental runs.

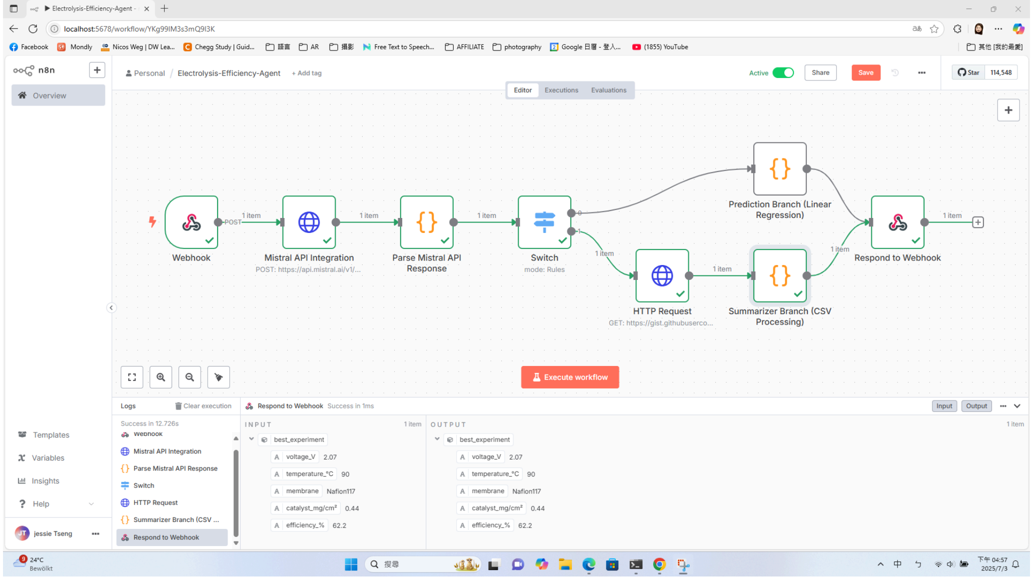
   2.2 Summarizer Branch – parses the CSV content by splitting the string into rows and columns, converting each row into a JSON object, identifying the row with the **highest efficiency_%.**
6. **Respond to Webhook** – Returns JSON output to the user.


Electrolysis-Efficiency-Agent
localhost:5678/workflow/YKg99IM3s3mQ9I3K
n8n
Overview
Personal / Electrolysis-Efficiency-Agent
+ Add tag
Active
Share
Save
Star 114,548
Editor
Executions
Evaluations
Webhook
POST
1 item
Mistral API Integration
POST: https://api.mistral.ai/v1/...
1 item
Parse Mistral API Response
1 item
Switch
mode: Rules
1 item
Prediction Branch (Linear Regression)
1 item
HTTP Request
GET: https://gist.githubuserco...
Summarizer Branch (CSV Processing)
Respond to Webhook
1 item
Execute workflow
Workflow executed successfully
Templates
Variables
Insights
Help
Jessie Tseng
Logs
Clear execution
Success in 41.673s
Webhook
Mistral API Integration
Parse Mistral API Response
Switch
Prediction Branch (Linear Re...
Respond to Webhook

Webhook Success in 1ms

OUTPUT 1 item

headers	params	query	body	webhookUrl	executionMode
host : localhost:5678 user-agent : curl/8.9.1 accept : */* content-type : application/json content-length : 37	[empty object]	[empty object]	text : Predict at 1.8 volts, 75C	http://localhost:5678/webhook-test/efficiency-agent	test


localhost:5678/workflow/YKg99IM3s3mQ9I3K
n8n
Overview
Personal / Electrolysis-Efficiency-Agent
+ Add tag
Active
Share
Save
Star 114,548
Editor
Executions
Evaluations
Webhook
POST
1 item
Mistral API Integration
POST: https://api.mistral.ai/v1/...
Parse Mistral API Response
Switch
mode: Rules
Prediction Branch (Linear Regression)
HTTP Request
GET: https://gist.githubuserco...
Summarizer Branch (CSV Processing)
Respond to Webhook
Execute workflow
Templates
Variables
Insights
Help
Jessie Tseng

```
Logs                    Clear execution
Success in 12.726s
Webhook
Mistral API Integration
Parse Mistral API Response
Switch
HTTP Request
Summarizer Branch (CSV ...
Respond to Webhook

Respond to Webhook  Success in 1ms

INPUT                                  1 item
best_experiment
  voltage_V         2.07
  temperature_°C    90
  membrane          Nafion117
  catalyst_mg/cm²   0.44
  efficiency_%      62.2

OUTPUT                                 1 item
best_experiment
  voltage_V         2.07
  temperature_°C    90
  membrane          Nafion117
  catalyst_mg/cm²   0.44
  efficiency_%      62.2
```

Linear Regression Model

📌 Data Preprocessing Steps

1. **Source**: Dataset imported from external CSV URL.
2. **Column Standardization**: Renamed the following columns for consistency:
   - `voltage_V` → `voltage`
   - `temperature_°C` → `temperature`
   - `efficiency_%` → `efficiency`
   - `catalyst_mg/cm²` → `catalyst`
3. **Missing Data Handling**: Dropped all rows with missing values using `dropna()`.
4. **Feature Selection**:
   - Features used: `voltage`, `temperature`
   - Target variable: `efficiency`
5. **Train-Test Split**:
   - 80% training data, 20% test data
   - `random_state=42` for reproducibility
6. **Standardization**:

- Input features were standardized using `StandardScaler` to ensure zero mean and unit variance.

7. **Regression Model**:
    - Applied `LinearRegression` from `scikit-learn` wrapped in a pipeline.

📈 Regression Model Performance

- **Model**: StandardScaler + LinearRegression
- **Test Set RMSE**:

  ✅ **{rmse:.2f}**

  (Root Mean Squared Error on hold-out test set)
- **Cross-Validation RMSE (5-Fold)**:

  ✅ **{cv_rmse.mean():.2f}**

  (Mean RMSE across 5 shuffled splits using K-Fold CV)
- Model 1 (There is another model I try)

```
Test RMSE: 1.04
5-Fold CV RMSE: 0.87
```

- **Coefficient Summary (Real Scale)**:

Term	Coefficient
Intercept	22.1581
Voltage	10.294
Temperature	0.1918

Example Inputs & Outputs

Prompt	Response
`Predict run at 1.85V and 80C`	`{"prediction": "58.53%", "parameters": {"voltage": 1.85, "temperature": 80}}`
`Summarize the best efficiency run`	`{"best_experiment": {"voltage_V": 2.07, "efficiency_%": 62.2, ...}}`
What is the expected efficiency at 1.7V and	[{"prediction": "51.17%","parameters": {"voltage": 1.7,"temperature": 60},"model_info": "Linear Regression

Prompt	Response
60°C?	(voltage & temp)"}]