เอกสารระบบ: AI Digital Product Factory

เอกสารสำหรับตรวจสอบระบบ — n8n Workflow Suite เวอร์ชัน `v1.0.0` · 6 workflow · 159 node

---

---

0. บริบทและข้อจำกัด

หัวข้อ	รายละเอียด
แพลตฟอร์ม	n8n (แนะนำ self-hosted)
AI provider	**OpenAI เท่านั้น** — ห้ามมี Claude / Gemini / local LLM / custom model
Marketplace หลัก	Etsy (Open API v3)
Marketplace รอง	Gumroad, MiriCanvas — **เตรียมแพ็กเท่านั้น ห้ามอัปโหลดอัตโนมัติ**
Storage	Google Drive (OAuth2)
Database	PostgreSQL (หลัก) / SQLite (สำรอง)
หน่วยการทำงาน	**1 run = สินค้า 1 ชิ้น ครบวงจร**
ขนาดระบบ	6 workflow, 159 node, ทุก node มี comment
ความปลอดภัย	ไม่มี API key ใน JSON เลย — ใช้ n8n credential + environment variable

---

---

1. สถาปัตยกรรมรวม

```
                    กด 1 ครั้ง (Schedule / Webhook / Manual)
                                 |
                      +---------v---------+
                      | 01 MASTER         |  34 nodes
                      | CONTROLLER        |  ตัวเดียวที่รู้ลำดับงาน
                      +---------+---------+
                                 | Execute Workflow x 7 จุด
     +------------+------------+--+---------+------------+
     v            v            v            v            v
 +---------+  +---------+  +---------+  +---------+  +---------+
 | 02      |->| 03      |->| 04      |->| 05      |->| 06      |
 | RESEARCH|  | PRODUCT |  | PUBLISH |  | BACKUP  |  | DATABASE|
 | 15 nodes|  | 48 nodes|  | 29 nodes|  | 24 nodes|  |  9 nodes|
 +---------+  +---------+  +---------+  +---------+  +---------+
```

หลักการ: 02–06 ไม่รู้จักกันเอง แต่ละตัวมี `Execute Workflow Trigger` (`inputSource: passthrough`) รับ JSON เข้า แล้วคืน JSON ออกเท่านั้น ทำให้เทสแยกทีละตัวได้ และเปลี่ยนตัวใดตัวหนึ่งไม่กระทบตัวอื่น

Workflow IDs (คงที่ เพื่อให้ node Execute Workflow หาปลายทางเจอ)

```
ADPF01MASTERCONTROL   ADPF02RESEARCHENGN   ADPF03PRODUCTENGIN
ADPF04PUBLISHENGIN    ADPF05DRIVEBACKUPX   ADPF06DATABASELOGX
```

---

2. Data Contract ระหว่าง Workflow

01 → 02

```
{ "run_id": "adpf-20260726-a3f9x2", "factory": "planner", "seed_keyword": "",
  "dry_run": false, "prompt_version": "v1.0.0", "etsy_api_base": "...",
  "etsy_shop_id": "...", "db_type": "postgres", "etsy_listing_state": "draft" }
```

02 → 01 → 03

```
{ "ok": true, "run_id": "...", "factory": "planner", "used_fallback": false,
  "research": {
    "product_type": "string", "category": "string", "target_customer": "string",
    "keyword": "string", "sub_keywords": ["..."],
    "difficulty": "easy|medium|hard", "selling_points": ["..."],
    "price_suggestion_usd": 6.5, "market_gap": "string", "long_tail": ["..."] },
  "market": { "competitor_count": 100,
    "price_usd": {"min":2,"p25":4,"median":6.5,"p75":12,"max":24},
    "average_favourites": 340, "top_tags": [{"value":"...","count":0}],
    "top_title_words": [], "etsy_categories": [], "sample_titles": [],
    "signal_quality": "good|thin|unavailable" } }
```

`sub_keywords` ไม่เกิน 15 ตัว แต่ละตัวยาวไม่เกิน 20 ตัวอักษร (เพื่อใช้เป็น Etsy tag ต่อได้)

03 → 01 → 04

```
{ "ok": true, "run_id": "...", "factory": "...", "product_name": "...", "slug": "...",
  "profile": {}, "metadata": {}, "seo": {}, "qa": {}, "research": {},
  "files": [ { "key": "product_pdf", "file_name": "x.pdf", "format": "PDF",
               "mime_type": "application/pdf", "purpose": "...", "bytes": 0,
               "b64": "<base64>" } ],
  "manifest": [], "formats": ["PDF","JPG","PNG","JSON","TXT"],
  "stats": { "pdf_pages": 31, "pdf_bytes": 0, "content_pages": 24,
             "images": {"requested":9,"generated":9,"failed":[],"placeholder":false},
             "total_bytes": 0 } }
```

กรณี QA ไม่ผ่าน จะคืน `{ "ok": false, "failure_reason": "QA gate rejected...", "qa": {} }`

04 → 01 → 05

```
{ "ok": true, "etsy_listing_id": 123456, "etsy_listing_state": "draft",
  "etsy_listing_url": "...", "uploaded_image_count": 5, "digital_file_id": 789,
  "gumroad_package_ready": true, "gumroad_package": {},
  "miricanvas_package_ready": false, "miricanvas_package": {},
  "extra_files": [] }
```

05 → 01 → 06

```
{ "ok": true, "drive_folder_id": "1AbC...",
  "drive_folder_path": "AI Digital Product Factory/Etsy/Calendars & Planners/Minimal Weekly Planner",
  "drive_folder_ids": {"root":"...","marketplace":"...","category":"...","product":"..."},
  "uploaded_files": [{"id":"...","name":"...","web_link":"..."}],
  "uploaded_file_count": 13 }
```

→ 06 (4 operations)

```
{ "op": "run_start | product_upsert | status_update | error_log",
  "run_id": "", "factory": "", "product_name": "", "category": "", "keywords": [],
  "marketplace": "", "drive_folder_id": "", "etsy_listing_id": "",
  "prompt_version": "", "status": "", "error_log": "" }
```

> **ข้อสังเกตการออกแบบ:** ไฟล์ถูกส่งเป็น `b64` ใน JSON ไม่ใช่ n8n binary เพราะ binary จะหายทุกครั้งที่ Code node คืน item ใหม่โดยไม่ copy ฟิลด์ binary มาด้วย ในสายที่มี Code node มากกว่า 20 ตัว โอกาสพลาดสูงมาก จึงแปลงเป็น binary เฉพาะ 3 จุดที่จะอัปโหลดจริง ต้นทุนคือ RAM ประมาณ 15–25 MB ต่อ run

---

---

3. รายละเอียดแต่ละ Workflow

3.1 — 01 Master Controller (34 nodes)

Trigger 3 ทาง รวมเข้าที่ `Init Run Context`

- `Trigger: Daily Schedule` — cron `0 3 * * *` (Asia/Bangkok)
- `Trigger: Webhook Run` — POST `/webhook/ai-digital-product-factory`
- `Trigger: Manual Run`

สายหลัก

```
Init Run Context       สร้าง run_id, normalize payload ทั้ง 3 trigger
Validate Environment   throw ทันทีถ้าขาด ETSY_API_BASE / ETSY_API_KEY / ETSY_SHOP_ID
Select Product Factory ระบุมา = ใช้ตามนั้น
                       "auto"  = pool[ floor(Date.now()/86400000) % pool.length ]
Prepare Run Start Log  payload op=run_start
Log: Run Started       -> 06
Restore Run Context    ดึง context กลับ (Execute Workflow แทนที่ item เดิม)
Run: Research Engine   -> 02   +
Check: Research OK             |
Compose Product Request        | ทุกตัวเปิด error output
Run: Product Engine    -> 03   | -> Stage Failed: X
Check: Product OK              |
Compose Publish Request        |
Run: Publish Engine    -> 04   |
Check: Publish OK              |
Compose Backup Request         |
Run: Drive Backup      -> 05   +
Prepare Product Record
Save: Product Record   -> 06 (op=product_upsert)
Build Success Summary
Respond: Run Finished
```

สาย error

```
Stage Failed: Research / Product / Publish / Backup   (Set node ติดป้าย stage)
  -> Build Error Payload      normalize ทุก failure เป็นรูปเดียว
  -> Log: Error To Database   -> 06 (op=error_log)
  -> Notify: Alert Webhook    Slack/Discord, ข้ามเงียบถ้าไม่ตั้ง env
  -> Stop On Fatal Error      mark execution ว่า failed

Trigger: Workflow Error Handler -> Format Global Error -> Log: Global Error
  (ต้องตั้ง 01 เป็น Error Workflow ของ 02–06 เพื่อจับ crash ที่หลุดออกมา)
```

3.2 — 02 Research Engine (15 nodes)

```
Receive Research Request
Prompt Library                  research_ai, seo_ai (versioned)
Normalize Research Input        seed keyword หรือ default ต่อ factory
Etsy: Search Active Listings    GET /v3/application/listings/active?limit=100
Etsy: Fetch Taxonomy Nodes      GET /v3/application/seller-taxonomy/nodes
Aggregate Etsy Market Signals   ย่อ 100 listing เหลือ "ตัวเลขสรุป" ก่อนส่ง AI
                                price percentile, tag freq x25, title words x25
Build Research Prompt
OpenAI: Research AI             temp 0.6, max_tokens 2000, json_object
Parse Research JSON             validate 7 ฟิลด์ + sub_keywords>=5 + selling_points>=3
Check: Research Valid  --false-> Research Fallback Concept
                                (ปั้น concept จากตัวเลข Etsy ล้วน ไม่ใช้ AI)
Build Keyword Expansion Prompt
OpenAI: SEO AI Keyword Expansion temp 0.5
Build Research Output           contract 7 ฟิลด์ + market
Return: Research Package
```

Etsy node ทั้ง 2 ตัวตั้ง `onError: continueRegularOutput` — API ล่มก็ยังทำงานต่อ ด้วยสัญญาณเท่าที่มี

3.3 — 03 Product Engine (48 nodes) — หัวใจของระบบ

ช่วง A: เลือกโรงงาน

```
Route: Factory Type (switch 8 ทาง + fallback)
  -> Factory Profile: Planner / Printable / Canva / Wall Art /
                      Resume / Spreadsheet / Kids / SVG / Fallback
  -> Merge: Factory Profiles     <- จุดบรรจบ ทุกอย่างหลังจากนี้ใช้ร่วมกัน
```

Set node แต่ละตัวยัด object `profile` เข้าไป — ความต่างระหว่างโรงงานเป็น **ข้อมูล ไม่ใช่โครงสร้าง** เพิ่มโรงงานที่ 9 = เพิ่ม 1 rule + 1 Set node

ช่วง B: เนื้อหา

```
Build Idea Prompt -> OpenAI: Idea AI (temp 0.8) -> Parse Idea JSON
    clamp ชื่อ <=60 ตัวอักษร, สร้าง slug, clamp page count
Build Content Prompt -> OpenAI: Writer AI (temp 0.7, max_tokens 8000)
    -> Parse Content JSON
    ตัดงบภาพ: หน้าที่ขอภาพเกิน profile.image_pages ถูกปฏิเสธ
Build Designer Prompt -> OpenAI: Designer AI (temp 0.9) -> Parse Image Plan
    sanitize SVG: ตัด <script>, <foreignObject>, on*= handler, external href
```

ช่วง C: ภาพ (loop ทีละใบ)

```
Split: Image Jobs -> N items
Loop: Image Jobs (batchSize 1)
   done(0) -> Collect Generated Images
   loop(1) -> Check: Generate Image
                true  -> OpenAI: Image API -> Store Generated Image -> กลับ loop
                false -> Use Placeholder Image (1x1 JPEG, dry_run)  -> กลับ loop

job list = cover(jpeg) + thumbnail(png) + preview x3(jpeg) + page images xN(jpeg)
Collect: throw เฉพาะเมื่อ cover หรือ thumbnail หาย
         ใบอื่นพัง -> บันทึกไว้แล้วไปต่อ
```

เหตุผลที่ภาพในเล่มเป็น JPEG: byte stream ของ JPEG ฝังลง PDF ด้วย `/DCTDecode` ได้ตรง ๆ ไม่ต้อง decode หรือ compress ใหม่ — ถ้าใช้ PNG ต้อง inflate และจัดการ alpha channel ซึ่งทำใน sandbox ของ Code node ไม่ได้

ช่วง D: SEO และการตรวจ

```
SEO Title -> SEO Description -> SEO Tags   (3 call, agent เดียวกันคนละ task mode)
Validate SEO Package
    ไม่เชื่อ AI: ตัด title 140, บังคับ tag = 13 ตัวพอดี แต่ละตัว <=20 อักษร
    ขาดก็เติมจาก research.sub_keywords, บังคับใส่ประโยค digital download
OpenAI: Metadata AI (temp 0.2) -> Parse Metadata JSON
    clamp ราคาเข้า profile.price_range_usd
OpenAI: QA AI (temp 0.1) -> Parse QA Report
    ตรวจซ้ำด้วยโค้ด: title<=140, tags=13, ทุก tag<=20
    passed = report.passed && blockers=0 && คะแนนต่ำสุด>=7
Check: QA Passed --false-> Build QA Failure Result -> คืน ok:false (ไม่ throw)
```

ช่วง E: ประกอบไฟล์

```
Build PDF Document   PDF 1.4 writer เขียนเอง ไม่มี npm หรือ service ภายนอก
                     รองรับ Helvetica + Helvetica-Bold, WinAnsi, word wrap,
                     ขึ้นหน้าใหม่อัตโนมัติ, หน้าภาพ fit/full-bleed, caption,
                     footer, document info, xref table + trailer
Export Files         manifest 11–13 ไฟล์ พร้อม b64
Return: Product Package
```

3.4 — 04 Publish Engine (29 nodes)

```
Normalize Publish Input      throw ถ้า product.ok !== true
OpenAI: Publisher AI (temp 0.2)
Parse Publisher Verdict      listing state มาจาก env ไม่ใช่จากโมเดล
Check: Publisher Approved --false-> Build Publish Failure
Build Etsy Listing Payload   clamp ทุกฟิลด์อีกรอบ
Etsy: Create Draft Listing   POST /v3/application/shops/{shop_id}/listings
                             form-urlencoded, state=draft, type=download,
                             who_made=i_did, when_made=made_to_order,
                             is_supply=false
Check: Listing Created
Enumerate Listing Images     cover, preview x3, thumbnail (5 ใบ เรียง rank)
Loop: Listing Images -> Prepare Image Upload
                     -> Etsy: Upload Listing Image (multipart)
Collect Image Uploads
Prepare Digital File Upload -> Etsy: Upload Listing File (multipart, PDF)
Check Gumroad Compatibility -> Check: Gumroad Compatible
     true  -> Prepare Gumroad Package    สร้าง gumroad-package.json
     false -> Skip Gumroad Package       บันทึกเหตุผล
Check MiriCanvas Compatibility -> เหมือนกัน (ต้องมี SVG source เท่านั้น)
Merge: Package Preparation -> Build Publish Result -> Return
```

เกณฑ์ compatibility เป็นโค้ดแบบ deterministic — Publisher AI **veto ได้อย่างเดียว อนุมัติไม่ได้**

3.5 — 05 Google Drive Backup (24 nodes)

```
Build Backup Plan   sanitize ชื่อโฟลเดอร์ (ตัด \ / : * ? " < > | ' และ control char)

x 4 ชั้น (root -> marketplace -> category -> product) แต่ละชั้นมี 4 node:
   Drive: Find X       GET files.list q="name='..' and mimeType=folder
                       and '<parent>' in parents and trashed=false"
   Check: X Exists
      true  -> Resolve X Folder            ใช้ id เดิม
      false -> Drive: Create X -> Resolve X Folder
   หมายเหตุ: Find/Create อ้าง plan ผ่าน $('<node>') เพราะ $json ถูกแทนที่
             ด้วย response ของ Drive API แล้ว

Enumerate Backup Files  product.files + publish.extra_files + workflow-log.txt
Loop: Upload Files -> Prepare Upload Binary -> Drive: Upload File (native node)
Collect Drive File Ids -> Return
```

Idempotent — รันซ้ำจะใช้โฟลเดอร์เดิม ไม่สร้างซ้ำ

3.6 — 06 Database Logger (9 nodes)

```
Build SQL Statement   validate op, normalize record, สร้าง 2 dialect พร้อมกัน
                      pg_params  : array 14 ค่า ($1..$14)
                      sqlite_sql : SQL ที่ escape เอง (' -> '')
Route: Database Type (switch on $env.DB_TYPE, fallback -> postgres)
   postgres -> Postgres: Ensure Schema -> Postgres: Write Record
   sqlite   -> SQLite: Ensure Schema   -> SQLite: Write Record
Build DB Result -> Return
```

PostgreSQL — statement เดียวครอบทั้ง 4 operation

```
WITH product AS (
  INSERT INTO adpf_products (...) VALUES ($1..$11)
  ON CONFLICT (run_id) DO UPDATE SET
    factory = COALESCE(EXCLUDED.factory, adpf_products.factory),
    -- ...  NULL ไม่ทับค่าเดิม
    status = EXCLUDED.status, updated_at = now()
  RETURNING id
), event AS (
  INSERT INTO adpf_events (run_id, op, level, message) VALUES ($1,$12,$13,$14)
  RETURNING id
)
SELECT (SELECT id FROM product) AS product_id, (SELECT id FROM event) AS event_id;
```

Schema

```
adpf_products(id, run_id UNIQUE, factory, product_name, category, keywords,
              marketplace, drive_folder_id, etsy_listing_id, prompt_version,
              status, error_log, created_at, updated_at)

adpf_events(id, run_id, op, level, message, created_at)
```

---

4. OpenAI Agents (8 ตัว)

#	Agent	Workflow	temp	max_tokens	หน้าที่
1	Research AI	02	0.6	2000	market signals → product concept
2	Idea AI	03	0.8	2500	concept → production brief
3	Writer AI	03	0.7	8000	เนื้อหาทุกหน้าที่พิมพ์ลง PDF
4	Designer AI	03	0.9	4000	image prompt set + SVG markup
5	SEO AI	02, 03	0.5–0.6	600–1500	keywords / title / description / tags
6	Metadata AI	03	0.2	1500	canonical metadata record
7	QA AI	03	0.1	2000	compliance gate
8	Publisher AI	04	0.2	2000	marketplace payload + compatibility

- ทุกตัวเรียก `POST /v1/chat/completions` พร้อม `response_format: {type:"json_object"}`
- Auth: `HTTP Request` + `authentication: predefinedCredentialType`,

`nodeCredentialType: openAiApi`

- Prompt เก็บใน node `Prompt Library` (Code node) ของแต่ละ workflow

และ mirror เป็นไฟล์ `prompts/*.md`

- Image: `POST /v1/images/generations` model `gpt-image-1`

5. Factory Profiles (8 โรงงาน)

factory	page size	pages	img ในเล่ม	ภาพรวม	formats	svg	gumroad	miricanvas	ราคา
planner	A4 portrait	24	4	**9**	PDF/PNG/JPG/JSON/TXT	✗	✓	✗	$4–15
printable	Letter portrait	12	3	**8**	PDF/PNG/JPG/JSON/TXT	✗	✓	✗	$3–12
canva	A4 portrait	10	3	**8**	+ SVG	✓	✓	✓	$6–24
wallart	2:3 portrait	8	6	**11**	PDF/PNG/JPG/JSON/TXT	✗	✓	✗	$5–20
resume	Letter portrait	8	1	**6**	+ SVG	✓	✓	✓	$5–18
spreadsheet	A4 landscape	10	2	**7**	PDF/PNG/JPG/JSON/TXT	✗	✓	✗	$5–22
kids	Letter portrait	16	8	**13**	+ SVG	✓	✓	✓	$3–12
svg	square	6	4	**9**	SVG/PDF/PNG/JPG/JSON/TXT	✓	✓	✓	$3–14

ภาพรวม = cover 1 + thumbnail 1 + preview 3 + ภาพในเล่ม N

แต่ละ profile ควบคุม: `page_width_pt` / `page_height_pt`, `target_pages`, `image_pages`, `image_size` / `cover_size` / `thumbnail_size` / `preview_size`, `preview_count`, `content_schema` (โครงเนื้อหาที่ Writer AI ต้องทำตาม), `art_direction`, `svg_required`, `output_formats`, `etsy_taxonomy_hint`, `price_range_usd`, `gumroad_ready`, `miricanvas_ready`

6. ผลผลิตต่อ run

```
AI Digital Product Factory/Etsy/<Category>/<Product Name>/
  |-- <slug>.pdf                    ตัวสินค้า ประมาณ 30 หน้า
  |-- <slug>-cover.jpg
  |-- <slug>-thumbnail.png
  |-- <slug>-preview-1..3.jpg
  |-- <slug>.svg                    เฉพาะ canva / resume / kids / svg
  |-- metadata.json / SEO.json / content.json
  |-- description.txt / Prompt.md
  |-- gumroad-package.json / miricanvas-package.json   ถ้า compatible
  +-- workflow-log.txt
```

รวมถึง Etsy draft listing 1 รายการ, 1 แถวใน `adpf_products` และ N แถวใน `adpf_events`

---

7. งบต่อ run

รายการ	จำนวน
Chat API	**11 ครั้ง** (02: 2 · 03: 8 · 04: 1) เท่ากันทุกโรงงาน
Image API	**6–13 ครั้ง** ← ตัวแปรเดียวที่ทำให้ต้นทุนต่างกัน
Etsy API	7–9 ครั้ง
Drive API	ประมาณ 14 ครั้ง
เวลา	6–11 นาที (ภาพกินเวลาประมาณ 70%)

`dry_run: true` จะข้าม Image API ทั้งหมด ใช้ JPEG ขนาด 1x1 แทน เหลือประมาณ 90 วินาที และไม่มีค่าใช้จ่ายภาพ

---

8. Error handling / Retry / Logging

กลไก	รายละเอียด
Retry	ทุก node ที่ออกเน็ต 3 ครั้ง เว้น 5 วินาที (Image API เว้น 8 วินาที)
Error output	`Execute Workflow` 4 จุดใน 01 และ `Etsy: Create Draft Listing`
Graceful degrade	Etsy search ล่ม → ใช้สัญญาณเท่าที่มี · Research AI เสีย → fallback · ภาพใบเดียวพัง → ข้าม
Fail closed	QA ไม่ผ่าน → `ok:false` หยุดก่อนถึง Etsy · Publisher veto → หยุด
Global catch	`Trigger: Workflow Error Handler` (ต้องตั้ง 01 เป็น Error Workflow ของ 02–06)
Logging	ทุก run เขียน `adpf_products` + `adpf_events` · `workflow-log.txt` ขึ้น Drive · alert webhook ตอนพัง

9. Environment Variables

```
N8N_BLOCK_ENV_ACCESS_IN_NODE=false      ต้องตั้ง ไม่งั้น $env อ่านไม่ได้

OPENAI_MODEL_TEXT=gpt-4.1
OPENAI_MODEL_IMAGE=gpt-image-1
OPENAI_IMAGE_QUALITY=high

ETSY_API_BASE=https://openapi.etsy.com
ETSY_API_KEY=...
ETSY_SHOP_ID=...
ETSY_LISTING_STATE=draft
ETSY_TAXONOMY_ID=2078

GDRIVE_ROOT_FOLDER_ID=root

DB_TYPE=postgres
DB_SQLITE_PATH=/home/node/.n8n/adpf.db

PROMPT_VERSION=v1.0.0
FACTORY_ROTATION=planner,printable,canva,wallart,resume,spreadsheet,kids,svg
ALERT_WEBHOOK_URL=                      (optional)
```

Credentials 4 ตัว

Credential	ชนิดใน n8n	ใช้ที่
OpenAI	OpenAI API	ทุก node `OpenAI: ...`
Etsy	OAuth2 API (generic)	ทุก node `Etsy: ...`
Google Drive	Google Drive OAuth2 API	ทุก node `Drive: ...`
PostgreSQL	Postgres	node `Postgres: ...`

Etsy OAuth2 scope: `listings_r listings_w listings_d shops_r shops_w email_r`

10. ข้อจำกัดที่รู้อยู่แล้ว (ไม่ใช่บั๊ก)

1. **PDF ไม่รองรับภาษาไทย** — base-14 font ไม่มี glyph ไทย ต้อง embed TrueType ถึงจะได้
2. **PDF layout เรียบ** — ข้อความ + ภาพเต็มหน้า ไม่มี multi-column / ตาราง / กริดจริง
3. `factory=auto` หมุน**ตามวัน** ไม่ใช่ตามการกด (กด 3 ครั้งในวันเดียว = โรงงานเดิม)
4. Gumroad / MiriCanvas เตรียม manifest เท่านั้น ไม่อัปโหลด (ตามข้อกำหนดของโจทย์)

5. SQLite ใช้ได้เฉพาะ self-hosted (ต้องมี `sqlite3` CLI และ node Execute Command)
6. ไม่มี feedback loop จากยอดขายกลับมาปรับ prompt
7. ไม่มีการ dedupe สินค้าข้าม run (กันซ้ำได้แค่ระดับ keyword)
8. ภาพสร้างทีละใบ ไม่ขนาน (เจตนา: กัน rate limit และใบเดียวพังไม่ล้มทั้งชุด)

---

11. จุดที่อยากให้ตรวจเป็นพิเศษ

1. **Etsy API v3** — `createDraftListing` ใช้ form-urlencoded ถูกไหม field ครบไหม

`taxonomy_id` จำเป็นต้องตรงหมวดจริงแค่ไหน endpoint upload image/file ถูกต้องไหม

1. **PDF writer** — xref offset, `/DCTDecode` stream length, `/MediaBox`,

การ escape string ใน content stream

1. **n8n expression** — `$('NodeName').first().json` ใช้ถูกจุดไหม

โดยเฉพาะใน 05 ที่ `$json` ถูกแทนที่ด้วย response ของ Drive

1. **Loop pattern** — `splitInBatches` output 0=done / 1=loop ถูกต้องไหม

การอ้าง `$('Loop: Image Jobs').first()` ได้ item ปัจจุบันจริงไหม

1. **SQL** — CTE ที่ `RETURNING` ผ่าน sub-select ทำงานถูกไหม

`ON CONFLICT` + `COALESCE` กันค่าหายจริงไหม

1. **SQLite heredoc** — การ escape `'` เป็น `''` ครอบคลุมไหม มีช่องโหว่ injection ตรงไหน
2. **SVG sanitize** — regex ตัด `<script>` / `on*=` / external `href` รัดกุมพอไหม
3. **ความปลอดภัย** — มี secret หลุดใน JSON ไหม (ตรวจแล้วไม่พบ แต่ช่วยยืนยัน)
4. **Etsy rate limit** — 10 req/s, 10,000 req/day ออกแบบพอไหมถ้ารันวันละหลายชิ้น
5. **นโยบาย Etsy เรื่องสินค้า AI** — ต้องเปิดเผยอะไรบ้าง ระบบทำครบไหม

---

12. สถานะการทดสอบ

การทดสอบ	ผล
Code node ทั้ง 70 ตัวผ่าน `node --check`	ผ่าน
สร้าง PDF จริงและเดิน xref table ทีละ entry	ผ่าน (9 หน้า, JPEG ฝัง 2 ใบ, offset ตรงทุกตัว)
ตรวจ secret ในไฟล์ JSON	ไม่พบ
ตรวจโครงสร้าง workflow (ชื่อซ้ำ / connection / orphan / comment)	ผ่านทั้ง 6 ไฟล์
ทดสอบเชื่อมต่อ Etsy / Drive / OpenAI จริง	**ยังไม่ได้ทดสอบ** (ไม่มี credential ใน environment ที่สร้าง)

เอกสารนี้ generate จาก `docs/SYSTEM_SPEC_TH.md` · repo `metasitwute021/wute` branch `claude/ai-product-factory-workflow-9ywkwg`