PROJECT 047

AI Client Operations & Service Delivery System

Make.com flagship portfolio - detailed workflow notes

PLATFORM	DATABASE	COMMUNICATION
Make.com	**Airtable**	**Slack + Gmail**
AI LAYER	INTAKE	FILE OPERATIONS
Groq - openai/gpt-oss-120b	**Google Forms + Sheets**	**Google Drive**

System Operating Model

The project is organized as six independent Make.com scenarios connected by Airtable state. Each scenario owns one stage of the service lifecycle. This keeps the automations modular, easier to troubleshoot, and safer than placing intake, AI, reminders, file setup, escalation, and closeout logic inside one oversized scenario.

Clients	One client record per person/company. Email-based lookup is used before creation so existing clients are reused rather than duplicated.
Service Requests	Primary lifecycle record. Holds request details, AI classification, team assignment, request status, timestamps, reminder tracking, linked tasks, and activity-log relationships.
Tasks	Generated execution work items linked back to a Service Request. Task Type, Assigned Team, Task Status, and Due Date support operational handoff.
Activity Log	Audit layer for success/warning events. Stores scenario, action, status, message, timestamp, and optional error metadata.

WF01 - Service Request Intake & AI Triage

Google Forms -> Google Sheets -> Groq AI -> Parse JSON -> Airtable client lookup -> Array Aggregator -> Router -> create/reuse Client -> create Service Request

Trigger and raw intake. Google Forms Watch Responses captures one submission bundle. Google Sheets Add a Row writes an immutable intake trace before any AI or CRM logic executes. Mapped fields include client name, company, email, phone, request title, description, preferred deadline, budget, links, and notes.

AI Request Triage. Groq returns constrained JSON rather than prose. Allowed Service Type values are Graphic Design, Video Editing, Web Development, Marketing Support, and Other. Priority is Low/Medium/High/Urgent; Complexity is Low/Medium/High; Assigned Team is Design Team, Video Team, Web Team, Marketing Team, or Operations.

Expected JSON structure:

```
{ "service_type": "", "priority": "", "complexity": "", "ai_summary": "", "requirements_complete": true,
"missing_requirements": [], "assigned_team": "" }
```

- **Consistency guardrail:** requirements_complete=true requires an empty missing_requirements array; false requires at least one necessary missing item.
- **Missing Requirements mapping:** the parsed array is joined into Airtable long text so empty arrays remain blank while incomplete requests retain readable requirement notes.
- **Request Status on creation:** New.

Duplicate Client Safeguard

Airtable Search Records checks the Clients table using the submitted email. The result is aggregated before a Router so the flow can distinguish an existing record from a no-match result without stalling.

Existing client route	Skip client creation. Create the Service Request and map the linked Client field to the Airtable record ID returned by the lookup.
New client fallback	Create a Client record first, then create the Service Request and link it using the newly created Airtable record ID.
Design note	Service-request duplicate protection was intentionally deferred as a future reliability enhancement; client deduplication was prioritized because repeated submissions from the same customer are operationally common.

Task Planning, Team Assignment & Onboarding

WF02 - Task Generation & Team Assignment

Airtable Watch -> Groq AI Task Planner -> Parse JSON -> Slack -> Activity Log -> Service Request Status Update -> Iterator -> Airtable Task Creation

Trigger. Airtable Watch Records monitors Service Requests using Record Created Time and Request ID as the label. Limit is 1 during controlled testing. The scenario is intended to process newly created requests once they are available for task planning.

AI Task Planner. The prompt converts a classified request into a task array. Each task contains task_name, task_type, assigned_team, priority, and notes. The representative Lumina campaign produced multiple production and review tasks across design, video, and marketing responsibilities.

JSON Parse + Iterator. Parse JSON exposes Tasks[] as an actual array. The Iterator is placed immediately before the task-creation path so one AI result can fan out into any number of Airtable task records. This is the key pattern that avoids hard-coding Task 1, Task 2, Task 3, etc.

Task ID	TASK- + Request ID + bundle order position. Bundle position keeps each generated task distinct within a request.
Service Request	Linked using the Airtable Service Request record ID, not the visible Request ID text.
Task Name	Iterator task_name.
Task Type	Iterator task_type mapped to Airtable single-select values such as Internal Task, Review, Delivery, Requirement, Client Follow-Up, or Other.
Assigned Team	Iterator assigned_team.
Task Status	Static Pending on creation.
Created At	now.

Status transition. After the plan is generated, the Service Request is updated to **Ready for Onboarding**. This state is the handoff condition consumed by Workflow 03. During build testing, stale Airtable field references caused update errors after schema changes; refreshing/reselecting the affected Airtable fields resolved the mismatch.

Internal visibility. Slack posts a task-plan summary containing Request ID, Request Title, Service Type, Priority, Complexity, Assigned Team, and total generated tasks. A separate Activity Log record stores Action = Task Plan Generated and Status = Success.

WF03 - Client Onboarding & Project Setup

Airtable Watch Ready Service Request -> Create Project Folder -> 01 Brief -> 02 Working Files -> 03 Review -> 04 Final Delivery -> Slack -> Activity Log

Trigger. Airtable Watch Records uses Last Modified Time and filters Request Status = "Ready for Onboarding". This allows the scenario to react to WF02 rather than to initial record creation.

Dynamic Google Drive structure. The parent project folder is created inside **Northstar - Client Projects**. Folder name is mapped from Request Title. Every subfolder then uses **Create Project Folder [3] -> Folder ID** as New Folder Location. This dynamic parent-ID mapping is essential: it guarantees each new project receives its own child folders instead of writing into a manually selected prior project.

Project folder	Request Title
01 Brief	Briefs, scope notes, references, client-supplied source material.
02 Working Files	Active design/video/web/marketing working assets.
03 Review	Internal/client review versions and approval-stage deliverables.
04 Final Delivery	Approved final files and handoff-ready assets.

Slack onboarding notice. The message maps Request Title, Request ID, Assigned Team, Priority, current Request Status, and Create Project Folder -> Web View Link so operations can open the new workspace directly from Slack.

Activity log. Log Onboarding Activity writes Workflow 03 - Client Onboarding & Project Setup, Action = Project Onboarding Started, Status = Success, the linked Service Request record, and a timestamp. The message confirms that the Drive structure and Slack notification completed.

FOLLOW-UP, ESCALATION & CLOSEOUT

State-Driven Service Reliability Notes

WF04 - Reminder & No-Response Follow-Up

Airtable Search -> Get Client Details -> Gmail Reminder -> Update Follow-Up Status -> Activity Log

Search condition. Service Requests are eligible when Request Status = Waiting for Client and the elapsed time threshold has been reached. During testing, the threshold was reduced to one minute; production timing should be changed to the real SLA window (for example 24 hours).

Test formula:

```
AND( {Request Status} = "Waiting for Client", DATETIME_DIFF(NOW(), {Updated At}, 'minutes') >= 1 )
```

Client lookup. Search Records returns Client[] as a linked-record array. Get Client Details must use **Client[1]** (the linked Airtable client record ID), not the top-level Service Request ID. The client record supplies the recipient email and name for Gmail.

Reminder tracking fields. Update Follow-Up Status retains Waiting for Client, sets Updated At = now, sets Last Reminder Sent At = now, and increments Reminder Count. The working Make expression uses numeric math rather than the array-oriented add() function:

```
sum(ifempty([2.Reminder Count]; 0); 1)
```

- Initial blank Reminder Count becomes 1.
- The next reminder reads the current value and produces 2, then 3, and so on.
- The dedicated Last Reminder Sent At field makes follow-up cadence explicit and prevents generic record edits from being mistaken for reminder activity.

Gmail reminder. Recipient = Get Client Details -> Email Address. Subject references Request Title. The body uses the client name and explains that execution is waiting on the client before proceeding. Test records should use a deliverable email address rather than example.com.

Activity log. Action = Client Reminder Sent, Status = Success, with message noting that the reminder, counter, and follow-up timestamp were recorded.

WF05 - Re-engagement & Escalation

Airtable Search -> Slack Escalation -> Mark Request Escalated -> Activity Log

Purpose. WF05 deliberately stops repeated automated client messaging after a threshold and hands the case to a human. This creates a clear automation-to-human escalation boundary instead of an endless reminder loop.

Representative test logic:

```
AND( {Request Status} = "Waiting for Client", {Reminder Count} >= 2, DATETIME_DIFF(NOW(), {Last Reminder Sent At},
'minutes') >= 1 )
```

- Slack message includes Request Title, Request ID, Assigned Team, Priority, Reminder Count, and current status.
- Airtable update sets Request Status = Escalated and Updated At = now.
- Activity Log uses Status = Warning to distinguish a business exception/escalation from a normal success event.

WF06 - Delivery Completion & Operations Closeout

Airtable Search Completed -> Slack Completion Notice -> Finalize Service Request -> Activity Log

Search condition. Request Status = Completed. The workflow is intentionally concise: completion is already a human/business state, so the automation only performs closeout bookkeeping and visibility.

- Slack announces the completed Request, Request ID, Assigned Team, Priority, and status.
- Finalize Service Request keeps Status = Completed, writes Updated At = now, and can stamp a dedicated Completed At field when present.
- Activity Log records Action = Service Request Completed, Status = Success, and confirms the operational closeout.

Production Notes & Known Enhancements

Scheduling	Testing used short Make schedules and minute-based thresholds. Production should use SLA-aware intervals and realistic 24/48-hour windows.
Idempotency	Client deduplication is implemented. Service-request duplicate protection can be added later using the Google Forms response ID or an idempotency key.
Error handling	Future version can add dedicated retry/error-handler routes, Slack failure alerts, and Activity Log entries with Error Code and Error Details.
Response detection	A future inbound-email or form-response listener can automatically move Waiting for Client back to In Progress when the client replies.
Reporting	Airtable interfaces or BI dashboards can track turnaround time, reminder rate, escalation rate, task completion, and service-type workload.