Deal Stage - Notifications (In-App Notification and Email)

1. Overview

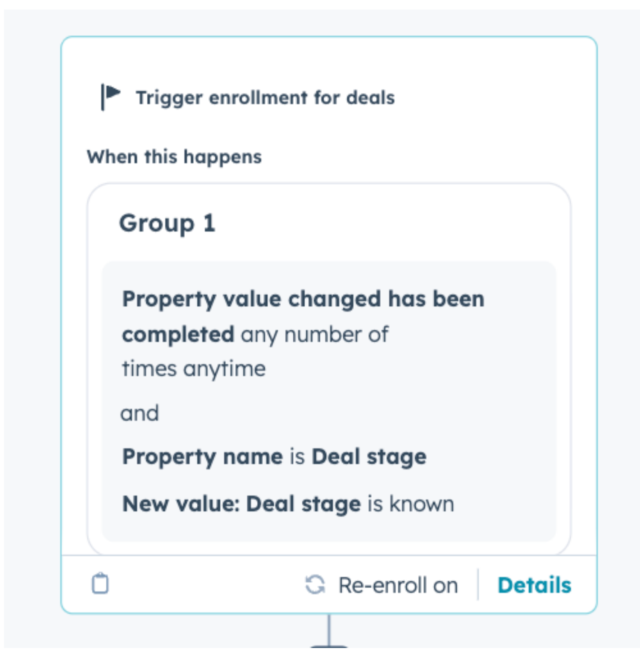
This workflow triggers whenever a deal's **Deal stage** property changes and routes the deal through a stage-based branch structure. Each branch sends an in-app notification (and optionally an email) to the responsible internal user, prompting timely next steps based on where the deal sits in the pipeline. This standardizes internal communication, reduces missed updates, and ensures accountability at every stage of the deal lifecycle, from early qualification through contract stages to final outcomes like Closed Won or Closed Lost.

2. Trigger

- **Workflow type:** Deal-based workflow (HubSpot)
- **Enroll when:** Deal property *Deal stage* changes

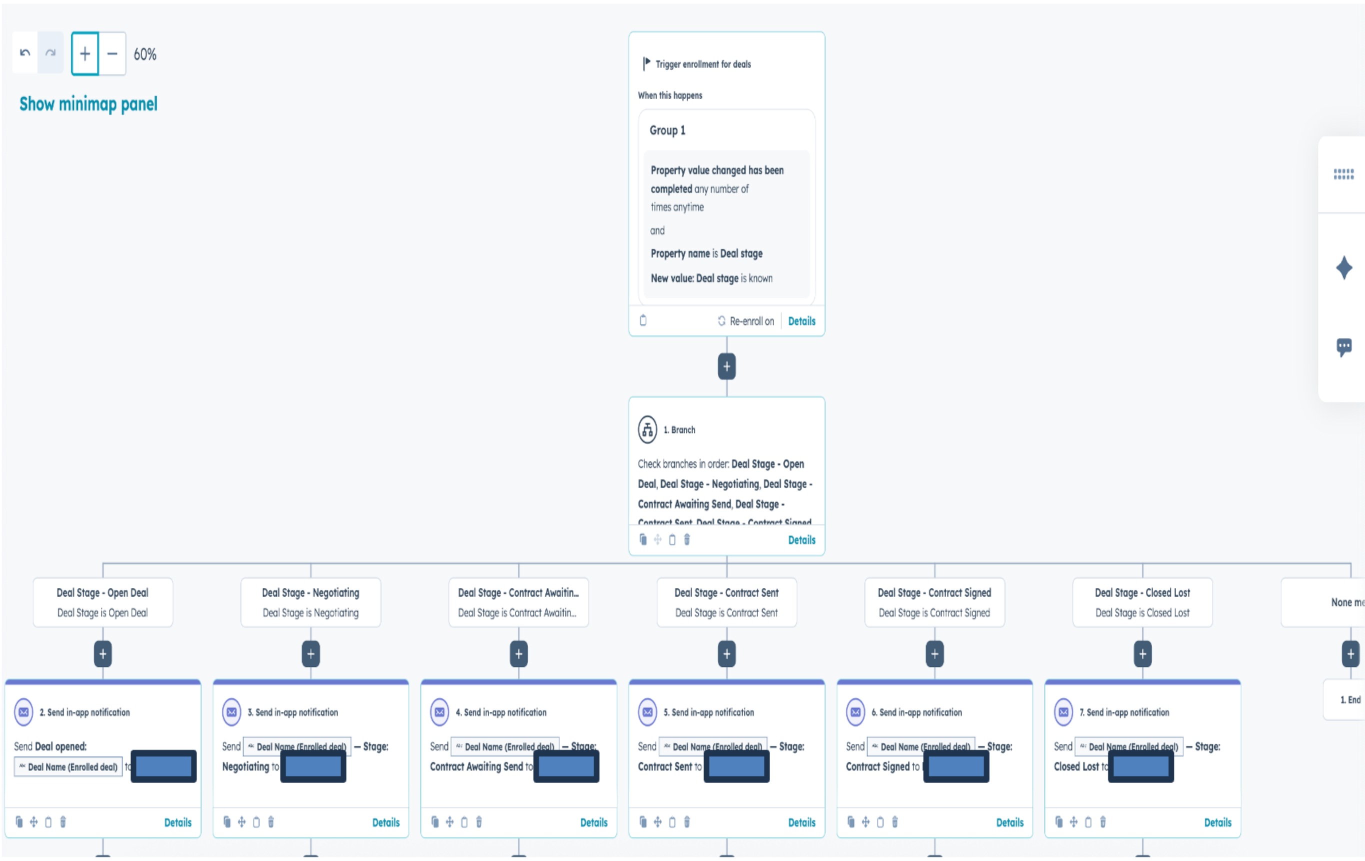
  This ensures notifications are tied to real pipeline movement rather than manual checks.
- **Re-enrollment:** Enabled

  Re-enrollment is turned on so the **same deal can trigger multiple times**, each time it enters a new stage. This is essential for stage-based routing workflows, without it, a deal would only notify once and ignore future stage changes.



3. Logic and routing

After a deal is enrolled, the workflow uses a **Branch** step to route the deal to the correct notification path based on its **current Deal stage**. If none match, the workflow ends.



- **Branch step purpose:**
  The Branch step checks the deal's stage and sends it down the matching route. Each route corresponds to a single stage (or a small group of closely related stages) and contains the notification actions.
- **Branch order matters:**
  Conditions are evaluated **top-to-bottom**. Once HubSpot finds the first matching branch, it executes that branch.
  To avoid incorrect routing, stage conditions should be **mutually exclusive** (a deal should match only one branch at a time).
- **One branch per run:**
  A deal should only match one branch for a given stage change—this prevents duplicate notifications.
- **None met / fallback branch:**
  Include a final **"None met"** branch that ends safely. This prevents the workflow from failing if:

a new stage is added later, a stage name is changed, or if the workflow is applied to a different pipeline unintentionally.

4. Stage mapping (template)

Deal stage (example)	Recipient (recommendation)	Notification intent
Open Deal	Deal owner	Notify that a new deal is active
Negotiating	Deal owner	Prompt follow-up / next steps
Contract Awaiting Send	Deal owner + Sales manager (optional)	Ensure contract is sent
Contract Sent	Deal owner	Track time since contract sent
Contract Signed	Deal owner + Onboarding (optional)	Handoff to onboarding
Closed Lost	Deal owner	Capture loss reason / learnings

5. Notification content

Use a consistent notification format across stages so recipients can scan quickly. Avoid including sensitive customer details in notification bodies.

Recommended in-app message template and an optional email subject:

```
Deal: {{ Deal Name }}
Stage: {{ Deal Stage }}
Action: Open the deal record, confirm the latest notes/tasks, and complete
the next required action for this stage (e.g., follow up, send documents,
or schedule the next step).
```

```
Deal moved to {{ Deal Stage }}: {{ Deal Name }}
```

Configuration steps (HubSpot)

- Create a **Deal-based workflow** named: *Deal Stage Notifications (In-App + Email)*.
- Set enrollment: *Deal stage* changes (enable re-enrollment).
- Add a **Branch**: one condition per stage (or per group of stages).
- Within each branch, add: **Send in-app notification** (and **Send internal email** if required).
- End each branch after sending the notification. Ensure a *None met* branch ends safely.

6. Testing checklist

- Move a test deal through each stage and confirm only one branch fires per stage change.
- Verify recipient logic (e.g., deal owner) works when the owner is present and when it is missing.
- Confirm messages include only safe fields (deal name and stage).
- Check workflow history for each run to confirm re-enrollment behaves as expected.

Conclusion

This deal-stage notification workflow improves pipeline visibility by automatically alerting the responsible internal owner whenever a deal moves to a new stage. By removing the need for manual pipeline monitoring, it supports faster follow-ups, reduces missed handoffs, and helps maintain consistent CRM hygiene. The workflow is easy to maintain (stage-based branches can be updated as the pipeline evolves) and provides a reliable internal prompt to take the next appropriate action at each step of the deal lifecycle.