Slack Configuration Guide for CX-Catalyst

Complete guide for configuring Slack integration with the CX-Catalyst AI Support System.

Table of Contents

---

1. Overview

Slack is a critical integration for CX-Catalyst, enabling:

Feature	Workflow	Purpose
Human Review Queue	Workflow 4	AI drafts posted for human approval
Proactive Alerts	Workflow 3	System anomaly notifications
Escalation Notifications	Workflow 4	Senior engineer alerts
Daily Reports	Workflow 5	Learning insights and metrics
Budget Alerts	Token Tracking	Usage threshold warnings

Architecture

```
┌─────────────────────────────────────────────────────────────┐
│                     n8n Workflows                            │
│                                                              │
│ Workflow 3     Workflow 4     Workflow 5     Token Tracking  │
│ (Proactive)    (Collab)       (Learning)     (Budgets)       │
└──────┬──────────────┬──────────────┬──────────────┬─────────┘
       │              │              │              │
       ▼              ▼              ▼              ▼
┌─────────────────────────────────────────────────────────────┐
│                    Slack Bot API                             │
│                 (OAuth2 / Bot Token)                         │
└──────┬──────────────┬──────────────┬──────────────┬─────────┘
       │              │              │              │
       ▼              ▼              ▼              ▼
┌────────────┐ ┌──────────────┐ ┌────────────┐ ┌─────────────┐
│ #support-  │ │ #support-    │ │ #support-  │ │ #support-   │
│ alerts       │  │ review        │  │ metrics      │  │ escalations │
└──────────────┘  └───────────────┘  └──────────────┘  └─────────────┘
```

2. Create Slack App

Step 2.1: Access Slack App Management

1. Go to api.slack.com/apps
2. Sign in with your Slack workspace credentials
3. Click **Create New App**

Step 2.2: Choose Creation Method

Select **From scratch** (recommended for full control)

Step 2.3: Configure Basic Information

Field	Value
App Name	`CX-Catalyst Support Bot`
Workspace	Select your target workspace

Click **Create App**

Step 2.4: Add App Icon & Description

Navigate to **Basic Information** > **Display Information**:

Field	Recommended Value
App icon	Upload a 512x512 PNG (optional)
Short description	`AI-powered support automation and notifications`
Background color	`#2563eb` (blue theme)

3. Configure OAuth & Permissions

Step 3.1: Navigate to OAuth Settings

Click **OAuth & Permissions** in the left sidebar.

Step 3.2: Add Bot Token Scopes

Scroll to **Scopes** > **Bot Token Scopes** and add:

Scope	Purpose
`chat:write`	Post messages to channels
`chat:write.public`	Post to channels bot isn't a member of
`channels:read`	List and view public channels

`channels:join`	Auto-join public channels
`reactions:write`	Add emoji reactions to messages
`reactions:read`	Read emoji reactions
`users:read`	Read user information
`users:read.email`	Read user email addresses
`files:write`	Upload files (for reports)
`im:write`	Send direct messages

Optional scopes for advanced features:

Scope	Purpose
`commands`	Slash commands (e.g., `/support`)
`incoming-webhook`	Legacy webhook support
`app_mentions:read`	Respond to @mentions

Step 3.3: Install to Workspace

1. Scroll up to **OAuth Tokens for Your Workspace**
2. Click **Install to Workspace**
3. Review permissions and click **Allow**
4. Copy the **Bot User OAuth Token** (starts with `xoxb-`)

Save this token securely! You'll need it for n8n configuration.

---

4. Create Required Channels

Step 4.1: Create Channels in Slack

Create these channels in your Slack workspace:

Channel	Purpose	Who Should Join
`#support-review`	Human review queue for AI drafts	Support team, CSMs
`#support-alerts`	Critical system alerts & anomalies	Support leads, SRE
`#support-escalations`	Senior engineer escalation requests	Senior engineers
`#support-metrics`	Daily reports & insights	Support managers, leadership
`#support-general`	Team discussions (optional)	Entire support team
`#support-budget-alerts`	Token usage warnings (optional)	Finance, engineering leads

Step 4.2: Invite the Bot

For each channel:

1. Open the channel
2. Click the channel name to open settings
3. Go to **Integrations** > **Add apps**
4. Search for `CX-Catalyst Support Bot`
5. Click **Add**

Or use `/invite @CX-Catalyst Support Bot` in each channel.

Step 4.3: Get Channel IDs (Optional but Recommended)

For more reliable message delivery, use channel IDs instead of names.

Method 1: Via Slack URL

1. Open channel in Slack web/desktop
2. Check URL: `https://app.slack.com/client/TXXXXXXXX/CXXXXXXXXX`
3. The `CXXXXXXXXX` part is the channel ID

Method 2: Via API

```
curl -H "Authorization: Bearer xoxb-your-token" \
  "https://slack.com/api/conversations.list?types=public_channel,private_channel" \
  | jq '.channels[] | {name: .name, id: .id}'
```

---

5. Configure n8n Credentials

Step 5.1: Create Slack Credential in n8n

1. In n8n, go to **Settings** > **Credentials**
2. Click **Add Credential**
3. Search for **Slack API**
4. Select **Slack API**

Step 5.2: Configure OAuth2 (Recommended Method)

For OAuth2 authentication:

Field	Value
Credential Name	`Slack API - CX Catalyst`
Authentication	OAuth2
Client ID	From Slack App > Basic Information
Client Secret	From Slack App > Basic Information

Click **Sign in with Slack** and authorize.

Step 5.3: Alternative: Access Token Method

For simpler setup using the bot token directly:

Field	Value

Credential Name	`Slack API - CX Catalyst`
Authentication	Access Token
Access Token	`xoxb-your-bot-token`

Step 5.4: Test the Connection

1. Create a test workflow with a **Slack** node
2. Configure to send a message to `#support-general`
3. Execute and verify the message appears

---

6. Workflow Integration Points

Workflow 3: Proactive Issue Detection

Purpose: Alert team to system anomalies

Channel: `#support-alerts`

When triggered:

- Critical health metrics detected
- Unusual case volume spikes
- Critical error log patterns

Message includes:

- Anomaly severity (Critical/High/Medium)
- AI analysis summary
- Affected metrics/systems
- Recommended actions
- Investigation links

Workflow 4: Collaborative Support Hub

Purpose: Human-in-the-loop review queue

Channels:

- `#support-review` - Primary review notifications
- `#support-escalations` - Rejected cases needing senior review

When triggered:

- New case enters review queue
- Case escalation after rejection

Message includes:

- Case ID and customer info
- AI-drafted solution
- Confidence score
- Review action buttons/links
- Deadline for review

Workflow 5: Continuous Learning

Purpose: Daily insights and improvement opportunities

Channel: `#support-metrics`

When triggered:

- Daily scheduled run (typically 6 AM)

Message includes:

- Daily resolution stats
- Top performing KB articles
- Identified knowledge gaps
- AI accuracy metrics
- Recommended improvements

Token Tracking Workflow

Purpose: Budget utilization alerts

Channel: `#support-budget-alerts` or `#support-alerts`

When triggered:

- Token usage crosses 80% (warning)
- Token usage crosses 95% (critical)
- Projected exhaustion within 3 days

Message includes:

- Current utilization percentage
- Provider (Anthropic/OpenAI)
- Budget period remaining
- Projected exhaustion date

---

7. Message Formatting & Templates

Review Queue Message Template

Used in Workflow 4 `Post to Slack Review` node:

```
:memo: *Human Review Required*

*Case ID:* {{ $json.case_id }}
*Review ID:* {{ $json.review_id }}
*Customer:* {{ $json.customer_context.name || 'Unknown' }} ({{
$json.customer_context.account_tier }})
*Priority:* {{ $json.classification.priority }} {{ $json.classification.priority ===
'high' ? ':large_orange_circle:' : ':large_yellow_circle:' }}
*Category:* {{ $json.classification.category }} / {{ $json.classification.subcategory
}}

---
```

```
*Customer Issue:*
{{ $json.description.substring(0, 500) }}{{ $json.description.length > 500 ? '...' : '' }}

---

*AI Draft Solution:*
{{ $json.ai_draft.solution.substring(0, 800) }}{{ $json.ai_draft.solution.length > 800 ? '...' : '' }}

---

*AI Confidence:* {{ Math.round($json.ai_draft.confidence * 100) }}%
*Escalation Recommendation:* {{ $json.ai_draft.escalation_recommendation }}

*Review Notes from AI:*
{{ $json.ai_draft.review_notes }}

---

*Actions:*
:white_check_mark: Approve: `{{ $env.N8N_WEBHOOK_BASE_URL }}/webhook/support/review/{{ $json.review_id }}/approve`
:pencil2: Edit & Approve: Reply with edits in thread
:x: Reject: `{{ $env.N8N_WEBHOOK_BASE_URL }}/webhook/support/review/{{ $json.review_id }}/reject`

_Please review within 2 hours._
```

Escalation Message Template

Used in Workflow 4 `Slack - Escalation` node:

```
:rotating_light: *Senior Engineer Review Needed*

*Case ID:* {{ $json.case_id }}
*Review ID:* {{ $json.review_id }}

The AI-drafted solution was rejected by the reviewer.

*Reviewer Comments:*
{{ $('Parse Review Action').first().json.reviewer_comments || 'No comments provided' }}

*Original Issue:*
{{ $json.description.substring(0, 300) }}...

Please assign a senior engineer to investigate this case.
```

Proactive Alert Template

For Workflow 3 anomaly alerts:

```
{{ $json.anomalies.overall_severity === 'critical' ? ':red_circle:' :
':large_yellow_circle:' }} *Proactive Alert - {{
$json.anomalies.overall_severity.toUpperCase() }}*

*Run ID:* {{ $json.run_id }}
*Detected At:* {{ $json.analysis_completed_at }}

---

*Summary:*
{{ $json.ai_analysis.summary }}

*Root Cause Hypothesis:*
{{ $json.ai_analysis.root_cause_hypothesis }}

*Customer Impact:*
{{ $json.ai_analysis.impact_assessment }}

---

*Metric Anomalies:*
{{ $json.anomalies.metric_anomalies.map(m => `• ${m.metric}: ${m.value} ${m.unit}
(${m.status})`).join('\n') || 'None' }}

*Case Volume Anomalies:*
{{ $json.anomalies.case_anomalies.map(a => `• ${a.category}/${a.subcategory}:
${a.current_count} cases (z-score: ${a.z_score})`).join('\n') || 'None' }}

---

*Recommended Actions:*
{{ $json.ai_analysis.recommendations.map((r, i) => `${i+1}. ${r}`).join('\n') }}

*Auto-remediation:* {{ $json.ai_analysis.auto_remediation_safe ? ':white_check_mark:
Safe to auto-remediate' : ':x: Manual intervention required' }}
```

Daily Metrics Template

For Workflow 5 daily reports:

```
:chart_with_upwards_trend: *Daily Support Intelligence Report*

*Date:* {{ new Date().toLocaleDateString('en-US', { weekday: 'long', year: 'numeric',
month: 'long', day: 'numeric' }) }}

---

*Case Resolution Summary:*
• Total Cases: {{ $json.daily_stats.total_cases }}
• Resolved: {{ $json.daily_stats.resolved }} ({{ Math.round($json.daily_stats.resolved
/ $json.daily_stats.total_cases * 100) }}%)
• Self-Service: {{ $json.daily_stats.self_service }} ({{
```

```
Math.round($json.daily_stats.self_service / $json.daily_stats.total_cases * 100) }}%)
• Escalated: {{ $json.daily_stats.escalated }}

*AI Performance:*
• Avg Confidence: {{ Math.round($json.ai_stats.avg_confidence * 100) }}%
• Approval Rate: {{ Math.round($json.ai_stats.approval_rate * 100) }}%
• Corrections Made: {{ $json.ai_stats.corrections }}

*Token Usage:*
• Anthropic: {{ $json.token_stats.anthropic.toLocaleString() }} tokens (${{
$json.token_stats.anthropic_cost.toFixed(2) }})
• OpenAI: {{ $json.token_stats.openai.toLocaleString() }} tokens (${{
$json.token_stats.openai_cost.toFixed(2) }})

---

*Top Performing KB Articles:*
{{ $json.top_articles.slice(0, 5).map((a, i) => `${i+1}. ${a.title}
(${a.success_rate}% success)`).join('\n') }}

*Identified Knowledge Gaps:*
{{ $json.knowledge_gaps.slice(0, 3).map((g, i) => `${i+1}. ${g.category}:
${g.description}`).join('\n') || 'No significant gaps identified' }}

---

*Improvement Recommendations:*
{{ $json.recommendations.map((r, i) => `${i+1}. ${r}`).join('\n') }}
```

Budget Alert Template

For token usage warnings:

```
{{ $json.status === 'critical' ? ':rotating_light:' : ':warning:' }} *Token Budget
Alert - {{ $json.status.toUpperCase() }}*

*Provider:* {{ $json.provider }}
*Budget:* {{ $json.budget_name }}

*Current Usage:*
• Tokens: {{ $json.tokens_used.toLocaleString() }} / {{
$json.token_limit.toLocaleString() }} ({{ $json.utilization_pct }}%)
• Cost: ${{ $json.cost_used.toFixed(2) }} / ${{ $json.cost_limit.toFixed(2) }}

*Period:* {{ $json.period_start }} to {{ $json.period_end }}
*Days Remaining:* {{ $json.days_remaining }}

{{ $json.projected_exhaustion ? `*Projected Exhaustion:*
${$json.projected_exhaustion}` : '' }}

{{ $json.status === 'critical' ?
  '*Action Required:* Consider reducing non-essential AI operations or increasing
budget.' :
  '*Recommendation:* Monitor usage patterns and prepare for potential budget
increase.' }}
```

---

8. Interactive Components

Option A: Webhook-Based Actions

The current implementation uses webhook URLs for actions:

```
:white_check_mark: Approve: `https://your-
n8n.com/webhook/support/review/{review_id}/approve`
```

Pros: Simple, no additional Slack app configuration **Cons:** Users must click link, then return to Slack

Option B: Slack Block Kit with Buttons

For a more native experience, configure Block Kit:

Enable Interactivity

1. Go to Slack App > **Interactivity & Shortcuts**
2. Turn on **Interactivity**
3. Set **Request URL**: `https://your-n8n.com/webhook/slack/interactions`

Block Kit Message Format

```
{
  "blocks": [
    {
      "type": "header",
      "text": {
        "type": "plain_text",
        "text": "Human Review Required"
      }
    },
    {
      "type": "section",
      "text": {
        "type": "mrkdwn",
        "text": "*Case ID:* {{ $json.case_id }}\n*Customer:* {{ $json.customer_name
}}"
      }
    },
    {
      "type": "section",
      "text": {
        "type": "mrkdwn",
        "text": "*AI Draft Solution:*\n{{ $json.ai_draft.solution.substring(0, 500)
}}"
      }
    },
    {
      "type": "actions",
      "elements": [
        {
          "type": "button",
          "text": {
            "type": "plain_text",
            "text": "Approve"
          },
          "style": "primary",
          "action_id": "approve_review",
          "value": "{{ $json.review_id }}"
        },
        {
          "type": "button",
          "text": {
            "type": "plain_text",
            "text": "Edit & Approve"
          },
          "action_id": "edit_review",
          "value": "{{ $json.review_id }}"
        },
        {
          "type": "button",
          "text": {
            "type": "plain_text",
            "text": "Reject"
          },
          "style": "danger",
          "action_id": "reject_review",
          "value": "{{ $json.review_id }}"
        }
      ]
    }
  ]
}
```

Create Interaction Handler Workflow

Create a new n8n workflow to handle button clicks:

```
[Webhook: POST /webhook/slack/interactions]
    ↓
[Parse Slack Payload]
    ↓
[Switch by action_id]
    ├── approve_review → [Process Approval]
    ├── edit_review → [Open Modal]
    └── reject_review → [Process Rejection]
```

Option C: Slash Commands

Add slash commands for quick actions:

1. Go to Slack App > **Slash Commands**
2. Click **Create New Command**

Command	Request URL	Description
`/support-status`	`https://your-n8n.com/webhook/slack/status`	Check queue status
`/support-metrics`	`https://your-n8n.com/webhook/slack/metrics`	Get today's metrics
`/support-search`	`https://your-n8n.com/webhook/slack/search`	Search KB

---

9. Advanced Configuration

Rate Limiting

Slack has rate limits. To avoid hitting them:

1. **Batch notifications** - Group multiple alerts into one message
2. **Use delays** - Add 1-second delays between messages in loops
3. **Deduplicate** - Don't send duplicate alerts for same issue

Message Threading

Keep conversations organized with threads:

```
// In n8n Slack node configuration
{
  "channel": "#support-review",
  "text": "Update on case...",
  "threadTs": "{{ $json.original_message_ts }}" // Reply in thread
}
```

User Mentions

Mention specific users or groups:

```
// Mention user by ID
<@U1234567890> Please review this urgent case.

// Mention channel
<!channel> Critical alert!

// Mention user group
<!subteam^S1234567890> On-call team needed.
```

Scheduled Messages

Send messages at specific times:

```
// In n8n Code node before Slack node
const targetTime = new Date();
```

```
targetTime.setHours(9, 0, 0, 0); // 9:00 AM

return {
  json: {
    post_at: Math.floor(targetTime.getTime() / 1000),
    // ... other fields
  }
};
```

Environment-Based Channels

Use different channels for dev/staging/production:

```
// In n8n expression
const env = $env.ENVIRONMENT || 'production';
const channelMap = {
  'development': '#support-dev',
  'staging': '#support-staging',
  'production': '#support-review'
};
return channelMap[env];
```

---

10. Troubleshooting

Common Issues

"channel_not_found" Error

Cause: Bot not in channel or wrong channel name/ID

Solutions:

1. Invite bot to channel: `/invite @CX-Catalyst Support Bot`
2. Use channel ID instead of name
3. Verify `chat:write.public` scope for public channels

"not_in_channel" Error

Cause: Bot lacks permission to post

Solutions:

1. Add `channels:join` scope
2. Manually invite bot to channel
3. For private channels, ensure bot is a member

"invalid_auth" Error

Cause: Token expired or invalid

Solutions:

1. Regenerate bot token
2. Re-authenticate OAuth in n8n

3. Verify token hasn't been rotated

"rate_limited" Error

Cause: Too many API calls

Solutions:

1. Add delays between messages: Wait node with 1s delay
2. Batch multiple items into single message
3. Implement exponential backoff

Messages Not Appearing

Possible causes:

1. Workflow execution failed (check n8n execution logs)
2. Bot not in channel
3. Message sent to wrong channel
4. Slack app not properly installed

Debug steps:

1. Check n8n execution output
2. Verify channel name/ID
3. Test with simple "Hello" message
4. Check Slack app permissions

Testing Your Setup

Test 1: Basic Message

```
// Create test workflow
// Slack node settings:
{
  "channel": "#support-general",
  "text": "Test message from CX-Catalyst at " + new Date().toISOString()
}
```

Test 2: Rich Formatting

```
{
  "channel": "#support-general",
  "text": "*Bold* _italic_ ~strikethrough~ `code`\n• Bullet point\n1. Numbered list"
}
```

Test 3: Mention User

```
{
  "channel": "#support-general",
  "text": "<@U1234567890> This is a test mention" // Replace with real user ID
}
```

Logging & Monitoring

Add logging to track Slack operations:

```
-- Create table for Slack message logging
CREATE TABLE slack_message_log (
    log_id UUID PRIMARY KEY DEFAULT gen_random_uuid(),
    channel VARCHAR(100),
    message_type VARCHAR(50),
    message_ts VARCHAR(50),
    workflow_name VARCHAR(100),
    success BOOLEAN DEFAULT TRUE,
    error_message TEXT,
    created_at TIMESTAMP DEFAULT NOW()
);

-- Query for failed messages
SELECT * FROM slack_message_log
WHERE success = FALSE
  AND created_at > NOW() - INTERVAL '24 hours'
ORDER BY created_at DESC;
```

Quick Reference

Required Scopes

```
chat:write
chat:write.public
channels:read
channels:join
reactions:write
reactions:read
users:read
```

Channel Summary

Channel	Purpose
`#support-review`	Human review queue
`#support-alerts`	System alerts
`#support-escalations`	Senior escalations
`#support-metrics`	Daily reports

Key Webhook Endpoints

Endpoint	Purpose
`/webhook/support/review/:id/approve`	Approve review

`/webhook/support/review/:id/reject`	Reject review
`/webhook/support/review/:id/edit`	Edit and approve
`/webhook/slack/interactions`	Button clicks (if using Block Kit)

Useful Links

- Slack API Documentation: https://api.slack.com/docs
- Block Kit Builder: https://app.slack.com/block-kit-builder
- Slack App Management: https://api.slack.com/apps
- n8n Slack Node Docs: https://docs.n8n.io/integrations/builtin/app-nodes/n8n-nodes-base.slack/

Slack Configuration Guide v1.0 - January 2026