

**N8N Multi-Client Email Campaign System Blueprint**

Since you're using N8N, here's the complete blueprint specifically designed for your n8n workflow architecture:

**N8N Workflow Structure**

**Main Campaign Workflow**

[Postgres Trigger] → [Client Context Loader] → [Content Parser] → [AI Prompt Builder] → [Groq Chat Model] → [Email Template Builder] → [Client Email Router] → [SMTP/Resend Sender] → [Campaign Logger]

**Database Schema for N8N**

**Core Tables**

-- Clients table  
CREATE TABLE clients (  
 id SERIAL PRIMARY KEY,  
 name VARCHAR(255) NOT NULL,  
 domain VARCHAR(255),  
 status VARCHAR(50) DEFAULT 'active',  
 created\_at TIMESTAMP DEFAULT NOW()  
);  
  
-- Client email configurations  
CREATE TABLE client\_email\_config (  
 id SERIAL PRIMARY KEY,  
 client\_id INTEGER REFERENCES clients(id),  
 from\_email VARCHAR(255) NOT NULL,  
 from\_name VARCHAR(255),  
 smtp\_host VARCHAR(255) DEFAULT 'smtp.gmail.com',  
 smtp\_port INTEGER DEFAULT 587,  
 smtp\_username VARCHAR(255),  
 smtp\_password VARCHAR(500),  
 use\_resend BOOLEAN DEFAULT FALSE,  
 resend\_api\_key VARCHAR(500),  
 domain\_verified BOOLEAN DEFAULT FALSE  
);  
  
-- Client content storage  
CREATE TABLE client\_content (  
 id SERIAL PRIMARY KEY,  
 client\_id INTEGER REFERENCES clients(id),  
 content\_type VARCHAR(100), -- 'knowledge\_base', 'logo', 'brand\_guide'  
 file\_path VARCHAR(500),  
 content\_data JSONB,  
 updated\_at TIMESTAMP DEFAULT NOW()  
);  
  
-- Enhanced email campaigns table  
CREATE TABLE email\_campaigns (  
 id SERIAL PRIMARY KEY,  
 client\_id INTEGER REFERENCES clients(id),  
 subscriber\_email VARCHAR(255),  
 subscriber\_name VARCHAR(255),  
 campaign\_day INTEGER, -- 1, 2, 4, 5, 7, 9  
 status VARCHAR(50) DEFAULT 'pending',  
 scheduled\_at TIMESTAMP,  
 sent\_at TIMESTAMP,  
 created\_at TIMESTAMP DEFAULT NOW()  
);

**N8N Node Configuration**

**1. Postgres Trigger Node**

-- Trigger query (modify your existing one)  
SELECT   
 ec.\*,  
 c.name as client\_name,  
 c.domain as client\_domain  
FROM email\_campaigns ec  
JOIN clients c ON ec.client\_id = c.id  
WHERE ec.status = 'pending'  
ORDER BY ec.created\_at ASC

**2. Client Context Loader (Code Node)**

// Load all client-specific data  
const clientId = $('Postgres Trigger').item.json.client\_id;  
const subscriberName = $('Postgres Trigger').item.json.subscriber\_name;  
  
// Query client content  
const knowledgeBase = await $execution.getNodeParameter('postgres').query(  
 `SELECT content\_data FROM client\_content   
 WHERE client\_id = $1 AND content\_type = 'knowledge\_base'`,  
 [clientId]  
);  
  
const brandAssets = await $execution.getNodeParameter('postgres').query(  
 `SELECT content\_data FROM client\_content   
 WHERE client\_id = $1 AND content\_type = 'brand\_assets'`,  
 [clientId]  
);  
  
const emailConfig = await $execution.getNodeParameter('postgres').query(  
 `SELECT \* FROM client\_email\_config WHERE client\_id = $1`,  
 [clientId]  
);  
  
return [{  
 json: {  
 clientId,  
 subscriberName,  
 knowledgeBase: knowledgeBase[0]?.content\_data || {},  
 brandAssets: brandAssets[0]?.content\_data || {},  
 emailConfig: emailConfig[0] || {},  
 campaignDay: $('Postgres Trigger').item.json.campaign\_day  
 }  
}];

**3. AI Prompt Builder (Code Node)**

const clientData = $input.item.json;  
const { knowledgeBase, brandAssets, subscriberName, campaignDay } = clientData;  
  
// Dynamic prompt based on client data  
const systemPrompt = `  
You are an expert email copywriter for ${knowledgeBase.company\_name || 'the company'}.  
  
Company Information:  
- Industry: ${knowledgeBase.industry || 'Not specified'}  
- Target Audience: ${knowledgeBase.target\_audience || 'General audience'}  
- Brand Voice: ${knowledgeBase.brand\_voice || 'Professional and friendly'}  
- Key Products/Services: ${knowledgeBase.products || 'Various services'}  
  
Write email ${campaignDay} of a 6-email onboarding sequence for ${subscriberName}.  
  
Brand Guidelines:  
- Use the company's brand voice: ${knowledgeBase.brand\_voice}  
- Include relevant information from the knowledge base  
- Make it personal and engaging  
- Include a clear call-to-action  
  
Knowledge Base Context:  
${JSON.stringify(knowledgeBase, null, 2)}  
`;  
  
const userPrompt = `Create email ${campaignDay} for ${subscriberName}. Make it personalized and relevant to our company's offerings.`;  
  
return [{  
 json: {  
 systemPrompt,  
 userPrompt,  
 clientData  
 }  
}];

**4. Enhanced Groq Chat Model Node**

// Use the dynamic prompts from previous node  
{  
 "messages": [  
 {  
 "role": "system",  
 "content": "{{ $json.systemPrompt }}"  
 },  
 {  
 "role": "user",   
 "content": "{{ $json.userPrompt }}"  
 }  
 ],  
 "model": "llama3-8b-8192",  
 "temperature": 0.7  
}

**5. Email Template Builder (Code Node)**

const aiContent = $('Groq Chat Model').item.json.choices[0].message.content;  
const clientData = $('AI Prompt Builder').item.json.clientData;  
const { brandAssets, emailConfig } = clientData;  
  
// Build HTML email with client branding  
const htmlEmail = `  
<!DOCTYPE html>  
<html>  
<head>  
 <style>  
 body { font-family: Arial, sans-serif; }  
 .header { background-color: ${brandAssets.primary\_color || '#007bff'}; padding: 20px; text-align: center; }  
 .logo { max-width: 200px; height: auto; }  
 .content { padding: 20px; }  
 .footer { background-color: #f8f9fa; padding: 15px; text-align: center; font-size: 12px; }  
 </style>  
</head>  
<body>  
 <div class="header">  
 ${brandAssets.logo\_url ? `<img src="${brandAssets.logo\_url}" alt="Logo" class="logo">` : ''}  
 <h1 style="color: white; margin: 10px 0;">${emailConfig.from\_name || 'Company Name'}</h1>  
 </div>  
 <div class="content">  
 ${aiContent.replace(/\n/g, '<br>')}  
 </div>  
 <div class="footer">  
 <p>${emailConfig.from\_name || 'Company'} | ${brandAssets.address || ''}</p>  
 <p><a href="#unsubscribe">Unsubscribe</a></p>  
 </div>  
</body>  
</html>  
`;  
  
// Extract subject line from AI content (first line)  
const lines = aiContent.split('\n');  
const subject = lines.find(line => line.toLowerCase().includes('subject:')) ||   
 `Day ${clientData.campaignDay} - Welcome to ${emailConfig.from\_name}`;  
  
return [{  
 json: {  
 html: htmlEmail,  
 subject: subject.replace(/subject:\s\*/i, ''),  
 clientData  
 }  
}];

**6. Client Email Router (Switch Node)**

// Route based on client email configuration  
const emailConfig = $('Email Template Builder').item.json.clientData.emailConfig;  
  
if (emailConfig.use\_resend && emailConfig.resend\_api\_key) {  
 return [0]; // Route to Resend  
} else if (emailConfig.smtp\_username && emailConfig.smtp\_password) {  
 return [1]; // Route to SMTP  
} else {  
 return [2]; // Route to fallback  
}

**7A. Resend API Node (HTTP Request)**

{  
 "method": "POST",  
 "url": "https://api.resend.com/emails",  
 "headers": {  
 "Authorization": "Bearer {{ $('Email Template Builder').item.json.clientData.emailConfig.resend\_api\_key }}",  
 "Content-Type": "application/json"  
 },  
 "body": {  
 "from": "{{ $('Email Template Builder').item.json.clientData.emailConfig.from\_email }}",  
 "to": ["{{ $('Postgres Trigger').item.json.subscriber\_email }}"],  
 "subject": "{{ $('Email Template Builder').item.json.subject }}",  
 "html": "{{ $('Email Template Builder').item.json.html }}"  
 }  
}

**7B. SMTP Node (Enhanced)**

// Dynamic SMTP configuration  
const emailConfig = $('Email Template Builder').item.json.clientData.emailConfig;  
const emailData = $('Email Template Builder').item.json;  
  
return [{  
 json: {  
 sender: emailConfig.from\_email,  
 recipient: $('Postgres Trigger').item.json.subscriber\_email,  
 subject: emailData.subject,  
 html: emailData.html,  
 smtp\_host: emailConfig.smtp\_host,  
 smtp\_port: emailConfig.smtp\_port,  
 smtp\_user: emailConfig.smtp\_username,  
 smtp\_password: emailConfig.smtp\_password  
 }  
}];

**8. Campaign Logger (Postgres Node)**

UPDATE email\_campaigns   
SET   
 status = 'sent',  
 sent\_at = NOW()  
WHERE id = {{ $('Postgres Trigger').item.json.id }}

**Client Onboarding Workflow**

**Separate N8N Workflow for Client Setup**

[Webhook - Client Registration] → [Create Client Record] → [Process Uploaded Files] → [Store Content] → [Setup Email Config] → [Import Subscribers] → [Initialize Campaigns]

**File Processing Node (Code)**

// Process uploaded client files  
const files = $('Webhook').item.json.files;  
const clientId = $('Create Client Record').item.json.id;  
  
const processedContent = {};  
  
// Process knowledge base  
if (files.knowledge\_base) {  
 processedContent.knowledge\_base = JSON.parse(files.knowledge\_base);  
}  
  
// Process brand assets  
if (files.brand\_assets) {  
 processedContent.brand\_assets = {  
 logo\_url: files.logo\_url,  
 primary\_color: files.primary\_color,  
 secondary\_color: files.secondary\_color,  
 font\_family: files.font\_family  
 };  
}  
  
return [{  
 json: {  
 clientId,  
 processedContent  
 }  
}];

**Subscriber Import Workflow**

**CSV Import Handler**

[Webhook - CSV Upload] → [CSV Parser] → [Data Validation] → [Bulk Insert Subscribers] → [Initialize Campaign Sequences]

**Implementation Steps**

**Phase 1: Database Setup**

1. Create the enhanced database schema
2. Migrate existing single-client data
3. Add client management interface

**Phase 2: Workflow Modification**

1. Update your existing workflow with multi-client nodes
2. Add client context loading
3. Implement dynamic email routing

**Phase 3: Client Onboarding**

1. Create client registration workflow
2. Build file upload and processing system
3. Add subscriber import functionality

**Phase 4: Testing & Launch**

1. Test with 2-3 pilot clients
2. Monitor performance and deliverability
3. Scale to more clients

This N8N-specific blueprint transforms your existing single-client workflow into a fully scalable multi-tenant system while leveraging n8n's visual workflow capabilities and maintaining your current AI-powered email generation approach.