n8n으로 RAG Agent 구축하기: Supabase와 AI로 만드는 나만의 업무 비서

목차

소개: 반복적인 문서 검색과 답변, AI에게 맡기세요

회사 생활을 하다 보면 누구나 한 번쯤은 방대한 양의 내부 문서 속에서 원하는 정보를 찾기 위해 애먹었던 경험이 있을 것입니다. 수백 페이지에 달하는 취업 규칙, 시시각각 업데이트되는 업무 매뉴얼, 복잡한 고객 응대 가이드라인 등. 필요한 정보는 분명 어딘가에 존재하지만, 그것을 찾아내기까지의 과정은 결코 간단하지 않습니다. 매번 담당자에게 문의하거나, 문서 전체를 뒤져가며 시간을 허비하는 것은 비효율의 전형입니다. (영상 0:01)

이러한 현실적인 문제를 해결하기 위해 등장한 기술이 바로 **RAG(Retrieval-Augmented Generation, 검색 증강 생성)**입니다. RAG는 우리가 제공한 특정 문서를 AI가 학습하고 이해하여, 그 내용을 기반으로 정확하고 신뢰도 높은 답변을 생성해주는 혁신적인 AI 아키텍처입니다. 마치 특정 분야의 전문가에게 질문하듯, 우리 회사의 문서에 대해 질문하고 즉각적인 답변을 얻을 수 있게 되는 것입니다.

이 가이드는 코딩에 익숙하지 않은 사람도 쉽게 따라 할 수 있는 로우코드(Low-code) 자동화 툴 **n8n**과 오픈소스 데이터베이스 **Supabase**를 활용하여, 단 15분 만에 나만의 RAG Agent를 구축하는 전 과정을 상세히 안내합니다. 이 글을 끝까지 따라오시면, 여러분은 더 이상 문서 더미 속에서 길을 잃지 않고, 반복적인 정보 검색 업무를 AI에게 맡김으로써 핵심 업무에 더욱 집중할 수 있는 강력한 'AI 업무 비서'를 갖게 될 것입니다. 이를 통해 개인의 업무 효율을 넘어 팀 전체의 생산성을 극대화하는 첫걸음을 내디딜 수 있습니다.

1. RAG란 무엇이고 어떻게 작동하는가? (핵심 개념 이해)

RAG Agent를 본격적으로 구축하기에 앞서, 그 기반이 되는 RAG 기술의 핵심 개념과 작동 원리를 이해하는 것은 매우 중요합니다. 이 섹션에서는 RAG가 무엇이며, 어떤 과정을 통해 질문에 대한 답변을 생성하는지 초보자의 눈높이에 맞춰 시각적인 흐름과 함께 설명합니다.

RAG(Retrieval-Augmented Generation)의 정의

RAG는 '검색 증강 생성(Retrieval-Augmented Generation)'의 줄임말로, 그 이름에 핵심 원리가 모두 담겨 있습니다. (영상 0:36) 이는 대규모 언어 모델(LLM)이 답변을 생성(Generation)할 때, 외부의 신뢰할 수 있는 지식 베이스에서 관련 정보를 **검색(Retrieval)**하여 그 내용을 **증강(Augmented)**, 즉 보강하는 기술적 접근 방식입니다.

기존의 LLM은 방대한 양의 데이터를 사전 학습하지만, 학습 시점 이후의 최신 정보나 특정 조직만이 보유한 내부 데이터(예: 회사의 재무 보고서, 법률 문서 등)에 대해서는 알지 못합니다. 이로 인해 부정확하거나 존재하지 않는 정보를 사실처럼 꾸며내는 '환각(Hallucination)' 현상이 발생하기도 합니다. RAG는 바로 이 지점에서 LLM의 한계를 보완합니다. LLM이 추측에 의존하

는 대신, 우리가 제공한 특정 문서라는 '사실 기반' 위에서 답변을 생성하도록 유도하여 답변의 정확성과 신뢰도를 획기적으로 높이는 것입니다.

RAG의 작동 원리 (2단계 프로세스)

RAG의 작동 방식은 크게 '데이터 준비 및 인덱싱'과 '검색 및 답변 생성'이라는 두 단계로 나눌 수 있습니다. 이는 마치 도서관에서 책을 정리하고 색인을 만드는 과정과, 독자가 질문했을 때 해당 색인을 통해 필요한 책을 찾아 답변해주는 과정에 비유할 수 있습니다.

1. 데이터 준비 및 인덱싱 (문서 라이브러리 구축)

이 단계는 AI가 참고할 지식의 기반을 마련하는 과정으로, 보통 오프라인에서 한 번 또는 주기적으로 수행됩니다.

- **문서 로딩 및 분할 (Chunking):** 먼저 AI에게 학습시킬 문서(PDF, DOCX, TXT 등)를 불러옵니다. 문서는 내용이 매우 길 수 있기 때문에, LLM이 한 번에 처리할 수 있는 단위로 잘게 쪼개는 과정이 필요합니다. 이를 '청킹(Chunking)'이라고 합니다. (영상 0:59) 문서를 쪼개는 이유는 LLM의 컨텍스트 길이 제한(한 번에 처리할 수 있는 텍스트의 양)을 넘지 않고, 질문과 가장 관련 있는 부분만 효율적으로 검색하기 위함입니다. 단순히 글자 수로 자르는 것보다, 문단이나 문장처럼 의미가 유지되는 단위로 지능적으로 분할하는 것이 답변의 품질을 높이는 데 유리합니다.
- **임베딩 (Embedding):** 컴퓨터는 인간의 언어를 직접 이해하지 못합니다. 따라서 쪼개진 텍스트 청크(chunk)들을 컴퓨터가 이해하고 계산할 수 있는 형태, 즉 숫자의 배열인 '벡터(Vector)'로 변환해야 합니다. 이 과정을 '임베딩'이라고 합니다. (영상 1:08) 임베딩을 통해 각 텍스트 조각은 고유한 숫자 좌표를 갖게 되며, 의미적으로 유사한 텍스트들은 이 벡터 공간에서 서로 가까운 위치에 자리하게 됩니다. 이 벡터 변환 과정은 나중에 사용자의 질문과 문서 내용 간의 관련성을 수학적으로 계산하는 데 필수적입니다.
- **벡터 데이터베이스 저장:** 임베딩을 통해 벡터로 변환된 문서 조각들은 검색이 용이하도록 특수한 데이터베이스에 저장됩니다. 이를 '벡터 데이터베이스(Vector Database)'라고 부릅니다. (영상 1:42) 이 데이터베이스는 수많은 벡터들 사이에서 특정 벡터와 가장 유사한 벡터들을 빠르고 효율적으로 찾아내는 데 최적화되어 있습니다.

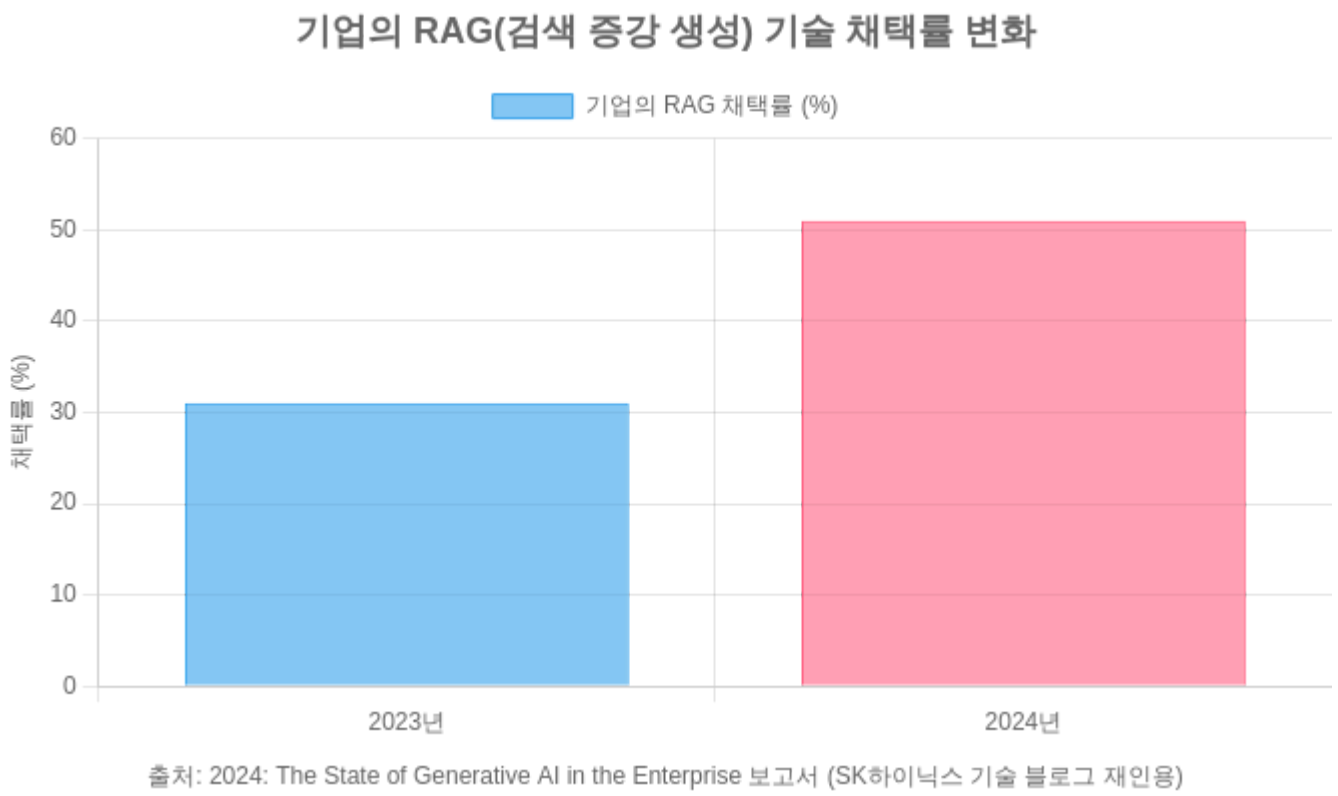
2. 검색 및 답변 생성 (실시간 질문 답변)

사용자가 실제로 질문을 던졌을 때 실시간으로 일어나는 과정입니다.

- **질문 임베딩:** 사용자가 입력한 질문 역시, 문서 데이터를 변환할 때 사용했던 것과 **동일한 임베딩 모델**을 통해 벡터로 변환됩니다. (영상 1:55) 동일한 모델을 사용해야 같은 벡터 공간 내에서 의미를 비교할 수 있기 때문입니다.
- **유사도 검색 (Similarity Search):** 시스템은 질문 벡터를 가지고 벡터 데이터베이스에 저장된 수많은 문서 벡터들과 비교합니다. 그리고 질문과 의미적으로 가장 유사한, 즉 벡터 공간에서 가장 가까운 거리에 있는 문서 청크들을 몇 개 찾아냅니다. 이를 '유사도 검색' 또는 '의미론적 검색(Semantic Search)'이라고 합니다.
- **프롬프트 증강 및 답변 생성:** 마지막으로, 검색된 관련성 높은 문서 청크들과 사용자의 원본 질문을 하나로 합쳐 LLM에게 전달합니다. (영상 2:16) 이때 LLM에게 주어지는 프롬프트는 "다음 [문서 내용]을 참고하여 이 [질문]에 답하세요"와 같은 형태가 됩니다. 이제 LLM은 막연한 지식에 의존하는 것이 아니라, 명확한 근거 자료를 바탕으로 맥락에 맞는 정확한 답변을 생성하게 됩니다.

RAG를 사용하는 이유

RAG 아키텍처는 기업과 개발자에게 여러 가지 명확한 이점을 제공하며, 이로 인해 최근 Generative AI 분야에서 가장 주목받는 기술 중 하나로 자리매김하고 있습니다. 실제로 한 보고서에 따르면, 기업의 RAG 채택률은 2023년 31%에서 2024년 51%로 크게 상승하며 지배적인 AI 설계 구조로 부상했습니다.



출처: 2024: The State of Generative AI in the Enterprise 보고서 (SK하이닉스 기술 블로그 재인용)

RAG의 핵심 장점

- **답변의 신뢰성 및 정확성 향상:** LLM이 외부의 검증된 지식 소스를 참조하도록 강제함으로써, 사실에 기반하지 않은 정보를 생성하는 '환각' 현상을 크게 줄일 수 있습니다.
- **최신 및 비공개 데이터 활용:** LLM을 재학습시키는 데는 막대한 비용과 시간이 소요됩니다. RAG를 사용하면 모델을 다시 학습시킬 필요 없이, 지식 베이스만 최신 정보나 기업 내부의 민감한 데이터로 업데이트하여 LLM이 이를 즉시 활용하게 할 수 있습니다. (영상 2:23)
- **비용 및 효율성:** 방대한 문서를 통째로 LLM에 전달하는 것은 컨텍스트 길이 제한에 걸리거나 API 비용을 급증시킬 수 있습니다. RAG는 질문과 관련된 핵심적인 부분만 선별하여 전달하므로, 더 빠르고 비용 효율적으로 정확한 답변을 얻을 수 있습니다. (영상 2:37)
- **출처 추적 가능:** RAG는 어떤 문서를 참고하여 답변을 생성했는지 출처를 제시할 수 있습니다. 이는 사용자가 답변의 신뢰도를 직접 검증할 수 있게 하여 투명성을 높입니다.

2. 실전! n8n과 Supabase로 RAG Agent 구축하기 (단계별 가이드)

이제 RAG의 개념을 이해했으니, 직접 우리만의 AI 업무 비서를 만들어 볼 차례입니다. 이 섹션에서는 로우코드 자동화 툴 n8n과 오픈소스 데이터베이스 Supabase를 사용하여 RAG Agent를 구축하는 전체 과정을 단계별로 상세하게 안내합니다. 코딩 경험이 거의 없어도 쉽게 따라 할 수 있도록 각 단계를 자세히 설명하겠습니다.

사전 준비: Supabase 벡터 데이터베이스 설정

RAG 시스템의 첫걸음은 문서 데이터를 저장하고 검색할 '도서관', 즉 벡터 데이터베이스를 마련하는 것입니다. 우리는 이를 위해 **Supabase**를 사용할 것입니다.

Supabase 소개

Supabase는 오픈소스 Firebase 대체재로 알려진 서비스로, 개발에 필요한 다양한 백엔드 기능을 제공합니다. 특히 초보자도 쉽게 다룰 수 있는 사용자 친화적인 인터페이스를 제공하며, 관계형 데이터베이스(PostgreSQL)와 벡터 데이터베이스(pgvector 확장 기능)를 통합하여 사용할

수 있는 유연성이 큰 장점입니다. (영상 3:10) 무료 플랜으로도 충분히 RAG 시스템을 구축하고 테스트해볼 수 있어 이번 실습에 매우 적합합니다.

프로젝트 생성

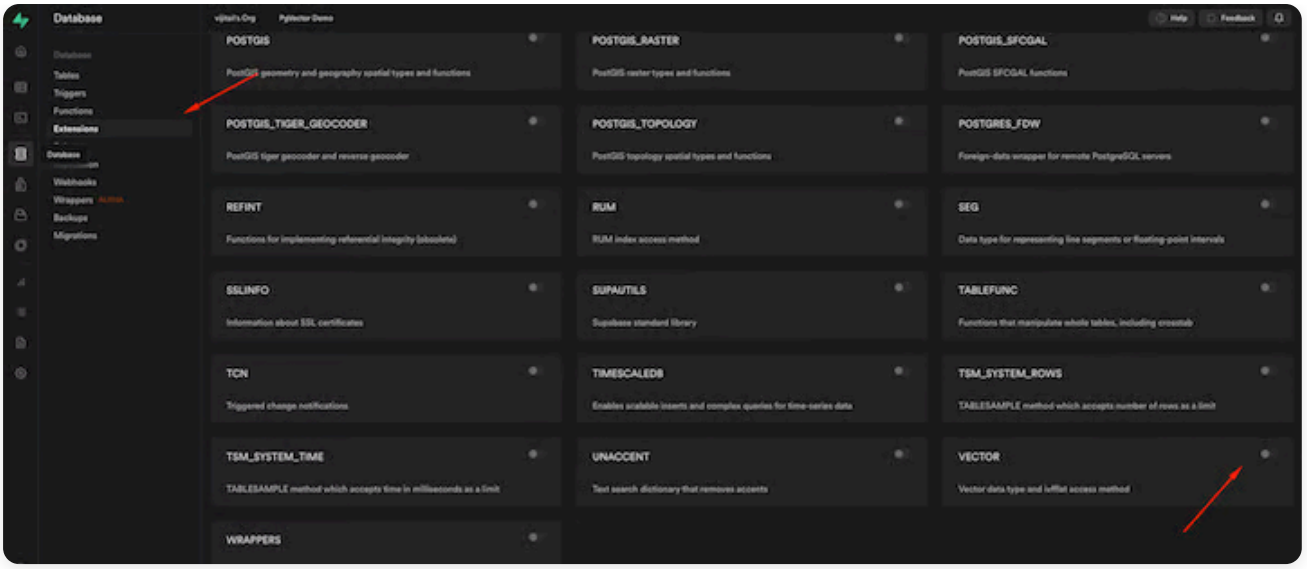
먼저 Supabase에 가입하고 새 프로젝트를 생성해야 합니다. 과정은 매우 간단합니다.

1. Supabase 공식 홈페이지에 접속하여 로그인합니다.
2. 대시보드에서 'New Project' 버튼을 클릭합니다.
3. 프로젝트 이름(예: `rag-agent`), 데이터베이스 비밀번호를 설정합니다. 이 비밀번호는 나중에 n8n과 연동할 때 필요하므로 반드시 안전한 곳에 기록해두어야 합니다. (영상 3:45)
4. 리전(Region)은 물리적으로 가까운 'Asia Pacific (Seoul)'을 선택하는 것이 좋습니다.
5. 'Create new project'를 클릭하면 몇 분 내로 프로젝트 생성이 완료됩니다.

벡터 스토어 테이블 생성

프로젝트가 생성되었다고 해서 바로 문서를 저장할 수 있는 것은 아닙니다. 문서 벡터를 저장할 '책장', 즉 테이블을 만들어주어야 합니다. 이 과정은 SQL 코드를 복사하여 실행하는 것만으로 간단히 완료할 수 있습니다.

1. n8n 공식 문서 템플릿에 제공된 SQL 쿼리를 복사합니다. 이 쿼리는 Supabase와 LangChain에서 벡터 스토어를 설정하기 위해 표준적으로 사용되는 스크립트입니다. (영상 7:31)
2. 생성한 Supabase 프로젝트 대시보드에서 왼쪽 메뉴의 'SQL Editor'로 이동합니다.
3. 'New query'를 클릭하고, 복사한 SQL 코드를 붙여넣습니다. (영상 7:49)
4. 'RUN' 버튼을 눌러 쿼리를 실행합니다. 'Success. No rows returned'라는 메시지가 뜨면 성공적으로 테이블이 생성된 것입니다.



Supabase 대시보드에서 pgvector와 같은 데이터베이스 확장 기능을 활성화하는 화면

이 SQL 코드는 다음의 세 가지 중요한 작업을 수행합니다:

- `create extension if not exists vector;` : Supabase의 PostgreSQL 데이터베이스에서 벡터 연산을 가능하게 하는 **pgvector** 확장 기능을 활성화합니다. (영상 7:56)
- `create table documents (...);` : 'documents'라는 이름의 테이블을 생성합니다. 이 테이블에는 문서의 원본 내용(`content`), 메타데이터(`metadata`), 그리고 가장 중요한 벡터 데이터(`embedding`)를 저장할 열이 정의되어 있습니다.
- `vector(1536)` : 임베딩 벡터를 저장할 열의 차원을 1536으로 지정합니다. 이 숫자는 우리가 사용할 OpenAI의 `text-embedding-3-small` 모델이 생성하는 벡터의 차원 수와 일치해야 합니다. 모델마다 차원 수가 다르므로, 사용할 임베딩 모델에 맞춰 이 값을 설정하는 것이 중요합니다. (영상 8:14)
- `create function match_documents (...);` : 유사도 검색을 수행할 함수를 정의합니다. 이 함수는 사용자의 질문 벡터와 가장 유사한 문서들을 데이터베이스에서 찾아주는 역할을 합니다.

이제 왼쪽 메뉴의 'Table Editor'로 이동하면 'documents' 테이블이 성공적으로 생성된 것을 확인할 수 있습니다. (영상 9:01) 이것으로 우리의 AI를 위한 지식 창고가 준비되었습니다.

1단계: 문서 업로드 및 임베딩 자동화 워크플로우

이제 데이터베이스가 준비되었으니, 문서를 자동으로 데이터베이스에 저장하는 n8n 워크플로우를 구축해 보겠습니다. 목표는 Google Drive의 특정 폴더에 문서를 올리기만 하면, n8n이 이

를 감지하여 자동으로 내용을 읽고, 청킹과 임베딩을 거쳐 Supabase에 저장하도록 하는 것입니다.

n8n 노드 구성

n8n에서 새로운 워크플로우를 생성하고 다음 노드들을 순서대로 연결합니다.

1. **Google Drive Trigger (파일 생성 감지):**
   - 새 노드를 추가하고 'Google Drive'를 검색하여 'Google Drive Trigger'를 선택합니다.
   - 'Event' 옵션에서 'File Created'를 선택합니다. 이는 폴더에 새 파일이 생성될 때마다 워크플로우가 실행되도록 합니다. (영상 4:23)
   - Google 계정을 연동하고, 감시할 폴더(예: `n8n-projects`)를 지정합니다. 주기적으로 폴더를 체크하도록 'Poll Time'을 설정할 수 있습니다.

2. **Google Drive (파일 다운로드):**
   - 앞선 트리거 노드에 '+'를 눌러 'Google Drive' 노드를 추가합니다.
   - 'Resource'는 'File', 'Operation'은 'Download'를 선택합니다.
   - 'File ID' 필드에 이전 노드(Trigger)에서 넘어온 파일의 ID 값을 드래그 앤 드롭으로 연결해줍니다. 이렇게 하면 트리거에서 감지된 바로 그 파일을 다운로드하게 됩니다. (영상 6:09)

3. **Supabase Vector Store (문서 추가 및 임베딩):**
   - 가장 핵심적인 노드입니다. '+'를 눌러 'Supabase Vector Store' 노드를 추가합니다.
   - **계정 연동:** 'Credential'에서 'Create New'를 선택합니다. Supabase 프로젝트의 'Settings' > 'API' 메뉴에서 'Project URL'(Host)과 'API Keys' 섹션의 'service_role' secret 키를 복사하여 붙여넣습니다. (영상 6:50)
   - **작업 설정:** 'Operation'은 'Add/Insert Documents'를 선택하고, 'Table Name'은 우리가 생성한 'documents'를 선택합니다.
   - **텍스트 분할 (Text Splitting):** 'Text Splitting' 옵션을 활성화합니다. 'Splitter'에서 'RecursiveCharacterTextSplitter'를 선택하는 것을 강력히 추천합니다. (영상 10:30) 이 스플리터는 단순히 글자 수로 자르는 것이 아니라, 문단의 흐름(줄바꿈), 문장(마침표), 단어(공백) 순으로 의미를 최대한 보존하며 지능적으로 텍스트를 분할해줍니다.
   - **메타데이터 추가:** 'Metadata' 옵션을 통해 검색 성능을 높일 수 있습니다. 예를 들어, 'Name'에 `document_title`을 입력하고, 'Value'에 이전 노드에서 가져온 파일 이름을 매

핑해주면, 나중에 특정 문서 제목으로 검색을 제한하는 등의 고급 기능을 구현할 수 있습니다. (영상 11:02)

- **임베딩 모델 설정:** 'Embeddings' 섹션에서 'Use OpenAI'를 선택하고, OpenAI 계정을 연동합니다. 'Model Name'은 `text-embedding-3-small`을 선택합니다. 이 모델이 바로 1536차원의 벡터를 생성하는 모델입니다. (영상 11:48)

실행 및 검증

이제 모든 설정이 끝났습니다. Google Drive의 지정된 폴더에 샘플 문서(예: '구시컴퍼니_취업규칙.docx')를 업로드한 후, n8n 워크플로우를 실행합니다. 실행이 완료되면 Supabase의 'documents' 테이블을 새로고침하여 데이터가 잘 들어왔는지 확인합니다. (영상 12:26) 테이블에는 문서 내용이 여러 개의 행으로 나뉘어 저장되어 있고, 각 행에는 `content`(분할된 텍스트), `metadata`(문서 제목 등), 그리고 `embedding`(1536개의 숫자로 이루어진 벡터 값)이 채워져 있는 것을 볼 수 있습니다. 이것으로 문서 라이브러리 구축이 자동화되었습니다.

2단계: AI Agent 구축 및 질문/답변 테스트

이제 구축된 지식 베이스를 활용하여 사용자의 질문에 실시간으로 답변하는 AI Agent 챗봇을 만들 차례입니다. 이 워크플로우는 사용자와의 대화창 역할을 합니다.

n8n 노드 구성

별도의 워크플로우를 새로 만들거나, 기존 워크플로우에 이어서 구성할 수 있습니다.

1. **Chat Trigger (채팅 시작):**
   - 'Chat Trigger' 노드를 추가하여 사용자가 메시지를 보낼 때 워크플로우가 시작되도록 합니다. 이 노드는 n8n 내에서 테스트용 채팅 인터페이스를 제공합니다.

2. **AI Agent (핵심 로직):**
   - 'AI Agent' 노드를 추가합니다. 이 노드가 RAG의 두뇌 역할을 합니다.
   - **모델 설정:** 'Model' 섹션에서 답변 생성에 사용할 LLM을 선택합니다. 예를 들어, 'OpenAI Chat Model'을 선택하고 모델로 `gpt-4-turbo` 또는 `gpt-4o`를 지정합니다. (영상 13:36)
   - **시스템 프롬프트 설정:** Agent의 정체성과 행동 규칙을 정의하는 'System Prompt'를 작성합니다. 이는 매우 중요합니다. 예를 들어, 다음과 같이 명확하게 지시할 수 있습니다: "당신은 회사의 내부 문서 전문가입니다. 반드시 제공된 문서 내용을 기반으로만 답변해야 합니다. 정보가 문서에 없으면 '해당 문서에는 관련 정보가 없습니다'라고 솔직하게 답변하세요. 모든 답변은 한국어로 작성해야 합니다." (영상 13:06)

- **도구(Tools) 설정:** Agent가 사용할 수 있는 '연장'을 등록해줍니다.
  - 'Add Tool'을 클릭하고 'Supabase Vector Store'를 선택합니다.
  - 'Operation'은 'Retrieve from Documents'로 설정합니다.
  - 'Description'에 Agent가 언제 이 도구를 사용해야 하는지 명확하게 설명해줍니다. (예: "회사 취업 규칙, 업무 매뉴얼, CS 가이드 등 내부 문서와 관련된 질문에 답변할 때 사용하세요.") (영상 14:01)
  - 'Table Name'은 'documents'를 선택합니다.
  - 'Embeddings' 모델은 **1단계에서 문서를 저장할 때 사용했던 것과 반드시 동일한 모델(`text-embedding-3-small`)**을 선택해야 합니다. 이는 질문과 문서를 같은 기준으로 비교하기 위해 필수적입니다. (영상 14:38)

- **메모리(Memory) 설정:**
  - Agent가 이전 대화 내용을 기억하게 하여 더 자연스러운 대화를 이어가도록 'Memory'를 설정합니다.
  - 'Postgres Chat Memory'를 선택합니다. 이 메모리는 대화 기록을 데이터베이스에 저장하여 영구적으로 보관할 수 있습니다. (영상 15:11)
  - 'Credential'에서 Supabase의 데이터베이스 연결 정보를 사용하여 새 계정을 생성합니다. 이 정보는 Supabase 프로젝트의 'Settings' > 'Database' > 'Connection string'에서 찾을 수 있습니다. (영상 15:28)
  - 'Table Name'을 지정하면(예: `chat_history`), n8n이 자동으로 해당 테이블을 생성하고 대화 기록을 저장합니다.

테스트 및 결과 확인

모든 설정이 완료되었습니다. n8n 우측 상단의 'Open Chat' 버튼을 눌러 채팅 인터페이스를 열고, 실제 질문을 던져봅니다.

> *"구시 컴퍼니의 겸업 관련 조항이 어떻게 되나요?" (영상 16:37)*

질문을 입력하면, AI Agent는 Supabase 도구를 사용하여 지식 베이스에서 '겸업'과 관련된 문서 조각들을 검색하고, 이를 바탕으로 "임직원은 회사의 사전 승인 없이 다른 직업에 종사할 수 없습니다..."와 같이 정확한 답변을 생성하는 것을 확인할 수 있습니다. (영상 16:49)

만약 Agent가 예상과 다르게 동작한다면, n8n의 'Executions' 탭에서 실행 기록을 확인할 수 있습니다. AI Agent 노드의 로그를 살펴보면, Agent가 어떤 검색어로 문서를 검색했는지, 그리고 어떤 문서 청크들을 참고하여 답변을 생성했는지 상세히 볼 수 있어 디버깅에 매우 유용합니다. (영상 17:26) 이 과정을 통해 우리는 성공적으로 나만의 RAG Agent를 구축하고 그 작동을 검증했습니다.

3. 기업에서의 RAG Agent 활용 사례와 확장 아이디어

단순히 기술을 구현하는 것을 넘어, 이 RAG Agent를 실제 비즈니스 환경에서 어떻게 활용하여 가치를 창출할 수 있는지 구체적인 사례를 통해 살펴보겠습니다. 우리가 만든 기본 모델은 다양한 기업 시나리오에 적용될 수 있는 강력한 잠재력을 가지고 있습니다.

핵심 활용 사례 (영상 시연 기반)

영상에서 시연된 두 가지 사례는 RAG Agent가 기업의 반복적인 업무를 얼마나 효과적으로 자동화할 수 있는지 명확하게 보여줍니다.

- **사내 HR 문의 자동화:** 모든 기업에는 취업 규칙, 복지 정책, 휴가 규정, 경비 처리 지침 등 수많은 내부 규정이 존재합니다. 직원들은 궁금한 점이 생길 때마다 HR팀에 문의하거나 사내 문서를 검색해야 합니다. RAG Agent에 이러한 규정 문서를 학습시키면, 직원들은 24시간 언제든지 AI 챗봇을 통해 "입사 6개월 미만인데 연차 사용이 가능한가요?" (영상 17:51) 와 같은 질문에 대한 즉각적이고 정확한 답변을 얻을 수 있습니다. 이는 HR팀의 반복적인 문의 응대 업무 부담을 줄여주고, 직원들의 만족도를 높이는 효과를 가져옵니다.
- **고객 지원(CS) 업무 효율화:** 고객 지원 부서는 신입 사원이 업무에 적응하는 데 많은 시간이 소요되는 대표적인 분야입니다. 복잡한 제품 정보, 다양한 상황별 응대 매뉴얼, 환불 규정 등을 모두 숙지하기 어렵기 때문입니다. RAG Agent에 고객 응대 매뉴얼을 학습시키면, 신입 사원도 베테랑 직원처럼 일관성 있고 정확한 답변을 고객에게 제공할 수 있습니다. (영상 18:48) 예를 들어, "고객이 배송 지연으로 환불을 요구할 때 어떻게 응대해야 하나요?" (영상 19:45) 라는 질문에 Agent가 단계별 응대 절차를 즉시 알려주어, 실수를 줄이고 고객 만족도를 유지하는 데 크게 기여할 수 있습니다.

추가 비즈니스 활용 아이디어

위 사례 외에도 RAG Agent는 기업의 다양한 부서에서 활용될 수 있습니다.

- **영업 및 마케팅:** 방대한 제품 기술 자료, 제안서 템플릿, 경쟁사 분석 보고서, 과거 성공 사례 등을 RAG Agent에 학습시킬 수 있습니다. 영업 사원은 고객과의 미팅 중에도 "A 제품과 경

쟁사 B 제품의 주요 차이점은 무엇인가?" 또는 "C 산업 분야의 성공 사례를 요약해줘"와 같은 질문을 통해 실시간으로 필요한 정보를 얻어 고객 문의에 기민하게 대응할 수 있습니다.

- **법무 및 컴플라이언스:** 수만 페이지에 달하는 법률 문서, 과거 계약서, 개인정보보호 규정, 내부 통제 지침 등을 검색하는 것은 법무팀의 주요 업무 중 하나입니다. RAG Agent를 활용하면 "최신 GDPR 규정에서 데이터 처리 동의와 관련된 조항은?"과 같은 복잡한 질의에 대해 관련 조항을 신속하게 찾아내어 법적 리스크 검토 시간을 획기적으로 단축할 수 있습니다.
- **R&D 및 기술 기획:** 연구 개발 부서에서는 새로운 기술을 개발하기 위해 수많은 기술 논문, 특허, 연구 보고서를 검토해야 합니다. RAG Agent에 관련 학술 자료 데이터베이스를 연결하면, 연구원들은 "나노 기술을 활용한 배터리 효율 개선에 대한 최신 연구 동향은?"과 같은 질문을 통해 자료 조사 및 분석 업무의 효율을 높이고, 혁신적인 아이디어를 얻는 데 더 많은 시간을 할애할 수 있습니다.
- **제조 및 생산:** 복잡한 장비의 유지보수 매뉴얼, 안전 수칙, 공정 관리 문서를 학습시켜 현장 작업자가 문제 발생 시 신속하게 해결책을 찾도록 도울 수 있습니다. "X-200 장비에서 에러 코드 503이 발생했을 때 조치 절차는?"과 같은 질문에 즉시 답변하여 생산 라인의 가동 중단 시간을 최소화할 수 있습니다.

시스템 확장 방안

우리가 만든 RAG Agent는 시작에 불과합니다. 몇 가지 기능을 추가하여 훨씬 더 강력하고 사용자 친화적인 시스템으로 발전시킬 수 있습니다.

- **협업 도구 연동:** 현재는 n8n의 테스트용 채팅창을 사용했지만, 이를 직원들이 매일 사용하는 Slack, Microsoft Teams, Telegram과 같은 사내 메신저와 연동할 수 있습니다. (영상 20:34) 이렇게 하면 직원들은 별도의 시스템에 접속할 필요 없이, 익숙한 환경에서 자연스럽게 AI 챗봇과 대화하며 필요한 정보를 얻을 수 있습니다.
- **다중 문서 및 데이터 소스 처리:** 현재는 하나의 테이블에 모든 문서 조각을 저장했지만, 시스템이 커지면 데이터 소스별로 테이블을 분리하거나 메타데이터를 활용하여 검색 범위를 동적으로 제어할 수 있습니다. 예를 들어, 사용자가 "인사 규정에서 찾아줘"라고 명시하면 'hr-documents' 메타데이터 태그가 붙은 문서들 내에서만 검색하도록 하여 검색의 정확도와 속도를 높일 수 있습니다.
- **Agentic RAG로의 진화:** 현재의 RAG는 주어진 질문에 대해 한 번의 검색과 답변 생성을 수행합니다. 이를 넘어, 복잡한 질문을 스스로 여러 하위 질문으로 분해하고, 여러 도구를 순차적으로 사용하여 정보를 수집하고 종합하여 최종 답변을 내놓는 '에이전트(Agent)'적인 접근

법으로 고도화할 수 있습니다. 이를 'Agentic RAG'라고 하며, 더 복잡한 문제 해결이 가능해집니다.

4. 한계점 및 향후 과제

n8n과 Supabase를 활용해 강력한 RAG Agent를 손쉽게 만들 수 있었지만, 이 기본 모델이 모든 문제를 해결해주는 만능 열쇠는 아닙니다. 프로덕션 환경에서 안정적으로 운영하기 위해서는 현재 구현된 기본 RAG의 기술적 한계를 명확히 인지하고, 이를 극복하기 위한 향후 과제를 고민해야 합니다.

기본 RAG의 명확한 한계

영상 마지막 부분에서 언급된 것처럼, 우리가 만든 RAG Agent는 몇 가지 명확한 한계를 가지고 있습니다. (영상 20:58)

- **키워드 기반 검색의 한계:** RAG는 문장의 '의미'를 기반으로 유사성을 검색하는 데 강점을 보입니다. 하지만 'A-모델-SN12345'와 같은 특정 제품 코드나 모델명, 고유명사 등 정확한 키워드 일치가 중요한 검색에는 상대적으로 취약할 수 있습니다. 의미적으로는 관련이 없다고 판단하여 중요한 정보를 놓칠 수 있습니다. 이를 보완하기 위해 전통적인 키워드 검색(Lexical Search)과 의미 검색(Semantic Search)을 결합하는 '하이브리드 검색(Hybrid Search)' 전략이 필요합니다.
- **전체 문서 요약 불가:** RAG의 작동 원리는 질문과 가장 관련성 높은 '일부' 청크만을 검색하여 답변을 생성하는 것입니다. 따라서 "이 100페이지짜리 보고서 전체를 요약해줘"와 같은 요청은 제대로 수행할 수 없습니다. (영상 21:29) 전체 요약은 문서 전체를 컨텍스트로 제공해야 가능하며, 이는 RAG의 기본 설계 목적과 다릅니다.
- **계산 및 정형 데이터 분석 불가:** RAG는 기본적으로 비정형 텍스트 데이터를 다루는 데 특화되어 있습니다. 스프레드시트나 데이터베이스 테이블에 있는 숫자 데이터를 가져와 합계를 내거나, 평균을 계산하거나, 복잡한 분석을 수행하는 작업은 수행하지 못합니다. (영상 21:36) 이러한 작업은 SQL 데이터베이스에 직접 질문을 던지는 Text-to-SQL 기술이나 코드 실행 기능을 갖춘 AI Agent가 더 적합합니다.

고려해야 할 추가 사항

실제 기업 환경에서 RAG 시스템을 성공적으로 도입하고 운영하기 위해서는 기술적 구현 외에도 다음과 같은 사항들을 반드시 고려해야 합니다.

엔터프라이즈 RAG 시스템의 7가지 실패 지점

최근 연구에서는 기업 환경에서 RAG 시스템을 설계할 때 흔히 발생하는 7가지 실패 지점을 지적했습니다. 이 중 몇 가지 핵심적인 실패 요인은 다음과 같습니다.

1. **콘텐츠 부재 (Missing Content):** 사용자가 질문한 내용에 대한 정보가 애초에 지식 베이스(문서)에 존재하지 않는 경우입니다. 이 경우 RAG는 답변을 생성할 수 없거나 환각을 일으킬 수 있습니다.
2. **상위 순위 검색 실패 (Missed Top Ranked):** 관련 정보가 문서에 존재하지만, 검색 알고리즘이 해당 문서를 상위 순위로 가져오지 못하는 경우입니다. 이는 부적절한 임베딩 모델 선택이나 비효율적인 청킹 전략 때문에 발생할 수 있습니다.
3. **컨텍스트에 없음 (Not in Context):** 문서는 제대로 검색했지만, LLM에 전달된 청크 내에 질문에 대한 답변이 포함되어 있지 않은 경우입니다. 예를 들어, 질문에 대한 답이 청크의 경계에 걸쳐 나뉘어 저장된 경우 발생할 수 있습니다.
4. **부정확한 구체성 (Incorrect Specificity):** 검색된 정보가 너무 일반적이거나, 반대로 너무 세부적이어서 사용자의 질문 의도에 맞지 않는 답변을 생성하는 경우입니다.

이러한 실패를 줄이기 위해서는 데이터 품질 관리, 청킹 전략 고도화, 그리고 지속적인 시스템 평가가 필수적입니다.

- **데이터 품질의 중요성 (Garbage In, Garbage Out):** RAG 시스템의 성능은 전적으로 학습하는 원본 문서의 품질에 달려 있습니다. 내용이 부정확하거나, 오래되었거나, 구조가 엉망인 문서를 제공하면 AI Agent 역시 저품질의 답변을 내놓을 수밖에 없습니다. 따라서 RAG 시스템을 구축하기 전에 데이터 정제 및 관리 프로세스를 수립하는 것이 매우 중요합니다.
- **청킹 전략의 고도화:** 단순 텍스트 문서는 `RecursiveCharacterTextSplitter`로 충분할 수 있지만, 복잡한 표, 차트, 이미지가 포함된 문서는 다른 접근이 필요합니다. 표는 행이나 열 단위로 의미를 유지하며 분할해야 하고, 이미지는 별도의 이미지 캡셔닝 모델을 통해 텍스트로 변환하는 등의 고급 청킹 전략이 요구됩니다.
- **지속적인 평가 및 튜닝 (Evaluation & Monitoring):** RAG 시스템은 한 번 구축하고 끝나는 것이 아닙니다. 검색 정확도(Retrieval Accuracy), 답변의 사실성(Faithfulness), 사용자 만족도 등 핵심 성능 지표(KPI)를 지속적으로 모니터링해야 합니다. 사용자 피드백을 수집하고, 실패 사례를 분석하여 프롬프트를 개선하거나, 청킹 전략을 수정하거나, 임베딩 모델을 파인튜닝하는 등 끊임없이 시스템을 개선해 나가는 과정이 필수적입니다.

- **보안 및 접근 제어:** 기업 환경에서는 모든 직원이 모든 문서에 접근할 수 없습니다. RAG 시스템 역시 사용자의 역할과 권한에 따라 접근 가능한 문서 범위를 제어하는 기능이 반드시 필요합니다. Supabase의 RLS(Row-Level Security)와 같은 기능을 활용하면, 사용자의 인증 정보에 따라 특정 문서에 대한 검색 결과를 필터링하여 민감한 정보 유출을 방지할 수 있습니다.

결론: 지금 바로, 당신의 첫 AI 업무 비서를 만들어보세요

지금까지 우리는 RAG라는 기술의 개념부터 시작하여, n8n과 Supabase라는 강력하고 접근성 높은 도구를 활용해 직접 AI Agent를 구축하는 전 과정을 함께했습니다. RAG라는 용어가 처음에는 복잡하고 어렵게 느껴질 수 있지만, 이 가이드에서 보았듯이 로우코드 플랫폼을 활용하면 누구나 자신의 필요에 맞는 강력한 AI 업무 비서를 만들 수 있다는 것을 확인했습니다.

우리가 만든 RAG Agent는 단순한 기술 시연을 넘어, 조직의 업무 방식을 근본적으로 변화시킬 수 있는 잠재력을 지니고 있습니다. 사내 규정에 대한 끝없는 질문에 답변하는 일, 신입 사원을 위해 매번 같은 내용을 설명하는 일, 방대한 자료 속에서 특정 정보를 찾아 헤매는 일 등. 이러한 단순 반복적인 정보 검색 업무를 자동화함으로써, 우리는 더 창의적이고 전략적인 가치를 창출하는 핵심 업무에 집중할 수 있습니다.

> *"RAG는 조직의 숨겨진 지식 자산을 가장 효과적으로 활용하는 방법 중 하나입니다. 이는 단순히 효율성을 높이는 것을 넘어, 데이터 기반의 의사결정을 가속화하고 집단 지성을 강화하는 촉매제가 될 수 있습니다."*

이 가이드는 여러분의 여정의 시작점입니다. 오늘 배운 내용을 바탕으로 여러분의 업무 환경에 맞는 문서를 학습시켜 보세요. 고객 응대 매뉴얼, 제품 개발 문서, 마케팅 자료 등 무엇이든 좋습니다. 여러분만의 RAG Agent를 직접 만들어보고, 그 과정에서 얻은 경험과 새로운 활용 아이디어를 동료들과 공유해보는 것은 어떨까요? 여러분의 작은 시도가 팀과 조직 전체의 생산성을 한 단계 끌어올리는 강력한 첫걸음이 될 것입니다.

참고 자료

[1] A guide on how retrieval-augmented generation (RAG) works
https://www.merge.dev/blog/how-rag-works

[2] n8n으로 RAG+AI 에이전트 챗봇 만들기 (노코드 자동화 가이드)
https://aiheroes.ai/community/317

[3] Build a Retrieval Augmented Generation (RAG) App: Part 1
https://python.langchain.com/docs/tutorials/rag/

[4] SK하이닉스의 RAG 플랫폼 구축 및 성능 평가/분석 연구 사례 - AWS
https://aws.amazon.com/ko/blogs/tech/sk-hynix-rag-platfrom-analysis-evaluation/

[5] 여러분, 회사에
https://static-us-img.skywork.ai/prod/analysis/2025-07-27/5363768134482727537/1949337399479050240_3f3849b46a6e4140ce3791baa59eb3b7.txt

[6] Interactive Knowledge Base Chat with Supabase RAG using AI - N8N
https://n8n.io/workflows/3799-interactive-knowledge-base-chat-with-supabase-rag-using-ai/

[7] 거북이처럼 n8n 노드 구축 끙끙대며 한땀한땀 시작하기 (feat. AI agent ...
https://www.gpters.org/nocode/post/getting-started-n8n-implementing-lEAItMp3FZcZXMz

[8] What is RAG? - Retrieval-Augmented Generation AI Explained - AWS
https://aws.amazon.com/what-is/retrieval-augmented-generation/

[9] Building a RAG with Supabase Vector & OpenAI - Rachit Khurana
https://blog.rachitkhurana.tech/building-a-rag-with-supabase-vector-openai

[10] Seven Failure Points When Engineering a Retrieval Augmented ...
https://arxiv.org/abs/2401.05856

[11] Supabase for Beginners
https://supabase.com/solutions/beginners

[12] Top 10 RAG Use Cases and 17 Essential Tools for Implementation
https://www.chatbees.ai/blog/rag-use-cases

[13] The Road Ahead for Governance, Risk, and Compliance (GRC) and ...
https://www.cioapplications.com/cxoinsights/the-road-ahead-for-governance-risk-and-compliance-grc-and-cybersecurity-with-retrieval-augmented-generation-nid-11004.html

[14] 검색증강생성(RAG)으로 다양한 제조 문제를 신속하게 해결하세요
https://ahha.ai/2024/07/26/rag_application/

[15] Automating AI workflows with n8n — from Chatbot to RAG in minutes
https://medium.com/@sabarishds03/automating-ai-workflows-with-n8n-from-chatbot-to-rag-in-minutes-952ffeb80d5e

[16] Best Practices for Production-Scale RAG Systems - Orkes
https://orkes.io/blog/rag-best-practices/

[17] 7 Common Failure Points in RAG Systems - LinkedIn
https://www.linkedin.com/pulse/7-common-failure-points-rag-systems-tanmay-patra-bnyfc

[18] Deploying RAGs in Production: A Guide to Best Practices - Medium
https://medium.com/@himanshu_72022/deploying-rags-in-production-a-guide-to-best-practices-98391b44df40

[19] RAG: Fundamentals, Challenges, and Advanced Techniques
https://labelstud.io/blog/rag-fundamentals-challenges-and-advanced-techniques/

[20] Optimizing RAG Performance: Key Metrics to Track - Galileo AI

https://galileo.ai/blog/top-metrics-to-monitor-and-improve-rag-performance

[21] RAG with Permissions | Supabase Docs
https://supabase.com/docs/guides/ai/rag-with-permissions