82 일차 · 단위 프로젝트 최종 발표

# 발표 · 시연 · 질의응답 가이드

팀당 20 분 ( 설명 + 시연 ) + 질의응답 5 분 — 만든 것을 ‘증명’ 하는 25 분



## 발표

짧게 · 숫자로 · 함께



## 시연

이야기로 · 대본대로



## 질의응답

결론 먼저 · 맡은 사람이

# 25 분은 이렇게 나눕니다

설명이 길어지면 시연 시간이 줄어듭니다 — 시연과 결과가 평가의 핵심입니다




| | |
|---|---|
| **팀장** | 문제 · 시나리오 · 범위 · 역할 |
| **워크플로우** | 흐름도 · 에이전트 · 분기 · 병렬 · 루프 |
| **RAG** | 청킹 비교 · 임베딩 · 권한 필터 |
| **통합** | 연동 · 오류 처리 · VM 구성 |
| **시험 · 품질** | 시연 4 분 + 결과 숫자 · 실패 사례 2 분 |
| **질의응답** | 맡은 사람이 답함 |

### 시간 지키는 법

- 타이머 담당 1 명 ( 발표 안 하는 순간에 )
- 15 분 · 18 분에 손 신호
- 넘치면 설명을 줄이고 시연은 지킴
- 장표는 10 ~ 14 장이면 충분

배분은 예시 — 팀 사정에 맞게 바꾸되 , 시연 + 결과는 6 분 이상 확보

# 듣는 사람 ( 평가자 ) 은 이 네 가지를 봅니다

멋진 말보다 ‘보여 줄 수 있는 증거’가 점수가 됩니다

## 문서 = 구현인가 ?

설계서의 흐름도 · 노드 이름이 시연 화면과 같은가

→ 설계서 화면 ↔ n8n 화면을 나란히

## 숫자가 실제인가 ?

통과 수 · 처리시간 · 청킹 비교가 runs/ 보고서에서 왔나

→ 보고서 파일 이름을 장표에

## 예외를 보여 주나 ?

정상만이 아니라 보완 질문 · 장애 · 이관이 동작하나

→ 시연에 장애 장면 1 개 이상

## 모두가 설명할 수 있나 ?

질문을 받았을 때 맡은 사람이 바로 답하나

→ 역할별로 나눠 발표 · 답변

# 첫 1 분 : 무엇을 만들었고 , 무엇을 보여 줄지 먼저

듣는 사람이 ‘끝까지 무엇을 보면 되는지’ 알게 해 줍니다

**오프닝 예시 ( 팀장 , 약 40 초 )**

“ 저희는 사내 IT 헬프데스크 문의를 자동으로 처리하는 워크플로우를 만들었습니다 .

직원이 텔레그램으로 물으면 , 규정 문서에서 근거를 찾아 답하고 , 근거가 없거나 긴급하면 담당자에게 티켓으로 넘깁니다 .

공통 시나리오 14 개 중 13 개 , 팀 시나리오 6 개 중 5 개를 통과했습니다 .

오늘 시연에서는 정상 문의 , 정보가 부족한 문의 , 외부 장애 상황 세 가지를 보여 드리겠습니다 .”

숫자는 예시 — 우리 팀의 실제 값으로

1. **누가 · 어디서 · 무엇을**
   사용자와 문제를 한 문장으로
2. **어떻게 처리하나**
   자동 답변 / 이관의 기준 한 문장
3. **얼마나 되나**
   통과 수 — 결론부터 숫자로
4. **오늘 볼 것**
   시연 장면을 미리 예고

**피할 시작 : “ 저희 팀은 4 명이고요 , 역할은요…” 로 2 분 쓰기 · 과정 소개 · 기술 이름 나열**

# 장표는 읽는 문서가 아니라 보여 주는 그림

설계서를 그대로 붙이지 말고 , 설계서에서 '한 장면' 씩 꺼내 옵니다

### 1 장 = 1 메시지

제목을 문장으로 : " 근거가 없으면 반드시 이관합니다"

### 글 대신 흐름도

설계서의 Mermaid 흐름도 · n8n 화면 캡처를 크게

### 숫자는 크게

통과 13/14 · p95 4.2 초 — 보고서 파일 이름을 작게 함께

### 뒤에서도 읽히게

본문 18pt 이상 · 코드 · JSON 은 핵심 줄만 확대

### 캡처 전에 가리기

키 · 토큰 · 전화번호 · 메일이 화면에 없게

### 10 ~ 14 장

자세한 표는 부록으로 — 질문이 오면 그때 펼치기

# 말하기 : 이렇게 · 이렇게 말고

듣는 사람 절반은 비전공자라고 생각하고 말합니다

**이렇게**

- 화면을 가리키며 “여기 보시면” — 시선을 화면으로
- 용어는 한 번 풀어서 : “RAG, 즉 문서를 먼저 찾아서 답하는 방식”
- 주장 뒤에 숫자 : “ 빨라졌습니다 → p95 6.1 초에서 4.2 초로”
- 한계를 먼저 말하기 : “WF-10 은 아직 흔들립니다 , 원인은 …”
- 다음 사람에게 넘기며 한 문장 연결

**이렇게 말고**

- 장표 글을 그대로 읽기
- “ 아마 될 거예요” , “ 원래는 됐는데요”
- 기술 이름만 나열 (n8n, PGVector, LiteLLM …)
- 화면 앞을 가로막고 서기 · 마우스 빙빙 돌리기
- 다른 팀원 파트에 끼어들어 보충 설명하기

**바톤 넘기기 예** “이렇게 분류된 질문이 어떤 문서에서 근거를 찾는지는 RAG 를 맡은 OOO 이 설명하겠습니다 .”

# 발표에서 자주 나오는 실수 여섯 가지

리허설 (3 · 4 교시 ) 때 이 표로 서로 점검해 주세요

| 실수 | 모습 | 대신 이렇게 |
|---|---|---|
| **시간 초과** | 앞 설명이 길어 시연을 못 함 | **시연 시작 시각을 정해 두고 타이머로 지킴** |
| **안 보이는 화면** | 작은 글씨 · 어두운 터미널 · 긴 JSON | **확대 (Ctrl + +) · 핵심 줄만 · 미리 띄워 두기** |
| **숫자 없는 주장** | “ 잘 됩니다” , “ 빠릅니다” | **통과 수 · 처리시간 · 보고서 이름** |
| **예시 값 그대로** | 템플릿의 500 토큰 · 80 오버랩을 우리 값처럼 | **우리가 잰 값 · 비교한 근거** |
| **화면에 키 · 개인정보** | Credential 화면 · .env · 전화번호 노출 | **발표 전 캡처 · 화면 점검 , 필요 없는 창 닫기** |
| **실패 숨기기** | 안 되는 시나리오를 건너뛰고 말 안 함 | **실패 사례로 당당히 : 원인 · 조치 · 재시험** |

# 시연은 기능 목록이 아니라 ‘한 사람의 이야기’ 로

같은 직원이 차례로 문의하는 이야기 — 장면마다 ‘넣는 것 → 나와야 할 것 → 실제’

장면 1

## 정상 문의

넣는 것
박서연 : “VPN 이 계속 끊겨요”

나와야 할 것
**자동 답변 + 근거 문서**

장면 2

## 정보 부족

넣는 것
이준호 : “ 접속이 안 돼요”

나와야 할 것
**보완 질문 → 다시 물으면 답**

장면 3

## 장애 · 이관

넣는 것
티켓 API 장애 상태에서 근거 없는 문의

나와야 할 것
**담당자 이관 · DLQ · 알림**

장면 4

## 기록 확인

넣는 것
n8n 실행 화면 · ops

나와야 할 것
**어느 노드를 지났는지 · 티켓**

**말하는 순서** “이걸 넣으면 → 이렇게 나와야 하고 → 실제로 이렇게 나왔습니다”

**보여 줄 곳** 응답의 route · sources · ticket 칸 + n8n 실행 화면에서 지나간 노드

**장면 수** 3 ~ 4 장면이면 충분 — 14 개를 다 보여 주지 않음 ( 나머지는 결과표로 )

# 시연  30 분 전 무대 세팅 체크리스트

시연 실패의 대부분은 ‘준비 안 된 화면’에서 나옵니다

- [ ] **팀  VM  접속 · doctor ready**
  발표 노트북에서 팀  VM 에 접속되는지
- [ ] **fault all ok · llm up**
  앞 팀 · 리허설에서 켜 둔 고장이 없게
- [ ] **요청 파일 준비**
  requests/demo-*.json —  손으로 타이핑하지 않기
- [ ] **창 미리 띄우기**
  n8n  워크플로우 · Executions  탭 · 터미널 · 발표 자료
- [ ] **글씨 크게**
  터미널 · 브라우저 확대 , 밝은 테마 권장
- [ ] **방해 요소 끄기**
  메신저 알림 · 자동 업데이트 · 화면 보호기
- [ ] **키 · 개인정보 창 닫기**
  .env · Credential  화면 · 개인 메일
- [ ] **백업 준비**
  장면별 캡처 · 녹화 영상 · runs/  보고서

**시연 담당 ( 시험 · 품질 ) 이 체크하고 , 팀장이 한 번 더 확인합니다**

# 시연 대본 : 장면마다 '명령 · 보여 줄 곳 · 한 줄'

대본을 손에 들고 하면 4 분 안에 끝납니다 — 리허설 때 시간을 재 보세요

| 장면 | 명령 / 요청 파일 | 화면에서 보여 줄 곳 | 말할 한 줄 | 시간 |
|---|---|---|---|---|
| **1 정상** | `call … demo-1-normal.json` | route · sources | *"근거 문서와 절이 함께 나옵니다"* | **50 초** |
| **2 정보 부족** | `call … demo-2-missing.json`<br>`→ demo-2b-context.json` | route = 보완 질문 → 자동 답변 | *"빠진 정보를 되묻고, 답하면 이어 갑니다"* | **60 초** |
| **3 장애 · 이관** | `fault tickets error`<br>`call … demo-3-handoff.json`<br>`fault all ok` | route = 담당자 이관 · ticket null | *"외부가 멈춰도 정해진 방식으로 실패합니다"* | **70 초** |
| **4 기록** | `ops · n8n Executions` | DLQ 1 · 알림 1 · 지나간 노드 | *"모든 처리는 개인정보 없이 기록됩니다"* | **40 초** |

**명령은 짧게** 긴 명령은 터미널에 미리 써 두고 Enter 만 — 또는 짧은 스크립트 (demo.sh) 로 장면마다 한 줄

# 시연이 실패하면 : 당황하지 않는 3 단계

라이브 시연은 실패할 수 있습니다 — 평가자가 보는 것은 '실패를 다루는 방식'

### 1 한 번만 다시

같은 요청을 한 번만 다시 보냄 (OpenAI 지연은 흔함 )

10 초

### 2 한 줄로 설명

" 지금 외부 응답이 늦습니다 . 이럴 때를 위해 시간 제한을 두었는데요…"

15 초

### 3 백업으로 전환

미리 만든 캡처 · 녹화 · runs/ 보고서로 같은 장면을 보여 줌

바로

**가점이 되는 태도**

- 실패한 것도 예외 처리로 설명 — LLM 이 멈추면 이관되는 것이 설계
- " 리허설 때는 3 회 모두 통과했습니다 (runs/...)" 로 근거 제시

**감점이 되는 태도**

- 화면 앞에서 1 분 넘게 디버깅
- " 원래는 됐는데…" 만 반복 · 사과만 하기

# 질문에 답하는 4 단계 : 결론 먼저 , 30 ~ 60 초

5 분이면 질문 3 ~ 4 개 — 짧게 답할수록 더 많은 질문을 받을 수 있습니다

| | | | |
|---|---|---|---|
| 1 | **다시 말하기** | *"~ 를 어떻게 정했는지 질문하신 거죠?"* | 질문을 정확히 이해했는지 확인 · 생각할 시간 |
| 2 | **맡은 사람이** | *팀장이 "RAG 담당 OOO 이 답하겠습니다"* | 역할별로 답하면 '모두가 이해했다' 의 증거 |
| 3 | **결론 한 문장** | *"점수 0.55 를 기준으로 정했습니다."* | 이유보다 결론을 먼저 |
| 4 | **근거 · 한계** | *"정상 · 결과 없음 질문 20 개 분포를 보고… 다만 질문이 적어서…"* | 숫자 · 부록 장표 · 인정할 한계 |

**팀장의 역할 : 질문을 받아 맡은 사람에게 연결하고 , 시간을 보며 마무리**

# 자주 나올 질문 10 개와 답할 사람

리허설 때 서로 이 질문을 던져 보세요 — 답이 30 초 안에 안 나오면 부록 장표를 준비

| | | |
|---|---|---|
| Q1 | 청킹 크기 · 오버랩은 왜 그 값인가요 ? | RAG |
| Q2 | 점수 기준 ( 신뢰도 ) 은 어떻게 정했나요 ? | RAG |
| Q3 | 왜 이 판단은 LLM 이 아니라 Code 로 했나요 ? | 워크플로우 |
| Q4 | 에이전트가 JSON 을 잘못 주면 어떻게 되나요 ? | 워크플로우 |
| Q5 | 외부 API 가 멈추면 사용자는 무엇을 보나요 ? | 통합 |
| Q6 | 프롬프트 주입 · 개인정보는 어떻게 막았나요 ? | 통합 |
| Q7 | 3 번 반복에서 흔들린 시나리오가 있었나요 ? | 시험 · 품질 |
| Q8 | 처리시간 · 비용은 얼마나 드나요 ? | 시험 · 품질 |
| Q9 | 실제 회사에서 쓰려면 무엇을 바꿔야 하나요 ? | 팀장 · 통합 |
| Q10 | 다시 한다면 무엇을 다르게 하겠어요 ? | 팀장 |

# 모르는 질문 · 날카로운 질문에는

‘모릅니다’도 좋은 답이 될 수 있습니다 — 확인할 방법까지 말하면

**이렇게**

- “지금 측정한 값은 없습니다. 확인하려면 bench로 p95를 재면 됩니다”
- “좋은 지적입니다. 현재는 ~까지만 했고, 다음 단계는 ~입니다”
- 질문이 길면 핵심을 되물어 확인
- 팀원이 막히면 팀장이 짧게 보충

**이렇게 말고**

- 모르는 것을 그럴듯하게 지어내기
- “그건 시간이 없어서…”로만 끝내기
- 질문자와 논쟁하거나 방어적으로 길게 설명
- 여러 명이 동시에 답하기

## 다른 팀에게 질문할 때 (동료 평가)

### 좋았던 점 하나

“장애 장면에서 DLQ까지 보여 준 게 인상적이었습니다”

### 질문 하나

“그 점수 기준은 질문 몇 개로 정하셨나요?” — 구체적으로

### 가져갈 것 하나

동료 평가지에: 우리 팀이 따라 하고 싶은 것

정리

# 발표 직전 최종 체크 10

6 교시 시작 전 , 팀이 함께 소리 내어 확인하세요

- [ ] 첫 1 분에 결과 숫자 ( 통과 수 ) 를 말한다
- [ ] 역할별 발표 순서와 바톤 문장을 정했다
- [ ] 실패 사례 1 건 이상을 발표에 넣었다
- [ ] 요청 파일과 대본으로 리허설을 끝까지 했다
- [ ] 화면에 키 · 개인정보가 없다
- [ ] 시연 시작 시각을 정했고 타이머 담당이 있다
- [ ] 모든 숫자에 보고서 (runs/) 근거가 있다
- [ ] 팀 VM · doctor ready · fault all ok · llm up
- [ ] 백업 ( 캡처 · 녹화 ) 이 바로 열린다
- [ ] 예상 질문 10 개에 답할 사람이 정해져 있다

**만든 것을 ‘보여 주고 · 숫자로 말하고 · 함께 답하는’ 25 분**