2026 AI 활용 사례 공모전 · 지역사회 문제해결을 위한 AI 활용

동네 가게 AI가 손님 번호를 가리고 있는지 59개 문장으로 재봤습니다 — 3분의 1이 새고 있었습니다

운영본 실측 탐지율 66.7% · 오탐률 21.7% / 개선안 v9 로컬 측정 83.3% · 4.3%(운영 반영 예정) · 활용 도구: Claude (Cowork · Claude Code) · 측정 2026-09

① 어떤 상황에서 AI를 활용했나요?

종로의 한 미용실에 이런 문자가 온다고 해 봅시다. "저 010-1234-5678인데 오늘 6시 예약되나요?" 요즘은 사장님이 시술 중이어도 AI가 매장 FAQ를 찾아 바로 답을 보냅니다. 사장님은 편해졌습니다. 그런데 그 문장은 사장님 휴대폰을 떠나 해외 AI 서버로 나갑니다. 사장님은 그 사실을 모르고, 번호를 보낸 손님은 더더욱 모릅니다.

「개인정보 보호법」상 그 번호의 책임자는 AI 회사도 앱 개발자도 아닌 매장 사장님(개인정보처리자)이고, 앱은 그 처리를 위탁받은 쪽(수탁자)입니다. 동네 가게가 AI를 들이는 순간 사장님은 자기도 모르게 개인정보를 국외로 내보내는 당사자가 됩니다. 국내 소상공인 기업체는 613만 4천 개, 종사자는 961만 명입니다(중소벤처기업부·소상공인시장진흥공단, 2026년 3월 발표). 중소기업중앙회 조사(소상공인 500개사, 2026년 6월)에서는 디지털·AI를 쓴다는 응답이 80.0%였지만(키오스크·배달앱·SNS 등 일상 도구 포함) 그중 83.3%가 자기 수준을 '기초·입문 단계'라고 답했습니다. 물어볼 사람이 없다는 뜻입니다.

저는 종로에서 1인 사업자로 일하며 소상공인용 예약·고객관리 앱을 만들었고, 거기에 손님 문자를 매장 FAQ로 답해 주는 AI 응대 기능을 넣었습니다. 그래서 두 가지를 했습니다. 하나는 제 앱이 정말로 손님 번호를 가리고 있는지 직접 재본 것이고, 다른 하나는 개발자 없는 사장님이 자기가 쓰는 AI 도구에 그대로 물어볼 수 있는 확인 질문과 손님 고지 문구를 만든 것입니다. 두 번째가 이 글에서 바로 가져가실 부분입니다.

② 어떤 AI를 어떻게 활용했나요?

먼저 구조를 바꿨습니다. 손님 질문이 오면 곧장 AI에게 답을 만들게 하지 않고, 사장님이 등록·확인한 FAQ 안에서만 찾습니다. 못 찾으면 지어내지 않고 사장님에게 넘깁니다. 기본 설정에서는 FAQ에 없는 질문이 와도 생성 AI(LLM)를 부르지 않습니다. 다만 FAQ에서 뜻이 비슷한 항목을 찾기 위한 호출(임베딩)은 질문마다 OpenAI로 나가므로, 그 경로에도 아래 마스킹을 먼저 적용했습니다.

그다음 Claude(Cowork·Claude Code)와 함께 마스킹 함수를 짜서 국외로 나가기 직전 지점, 즉 임베딩 호출과 생성 호출 양쪽에 끼웠습니다. 생성 호출에는 저장을 막는 옵션(store:false)을 함께 보냅니다. 손님 문장에서 전화번호·주민번호·이메일·카드번호처럼 보이는 것을 지우는 함수이고, 아래가 서버에 들어간 코드입니다(원본은 TypeScript이며 줄 끝 주석만 뺐습니다).

```
function scrubPII(s: string): string {
  return String(s || "")
    .replace(/\d{6}[-\s]?[1-4]\d{6}/g, "[주민번호]")
    .replace(/01[016789][-\s]?\d{3,4}[-\s]?\d{4}/g, "[전화]")
    .replace(/[\w.+-]+@[\w-]+\.[\w.-]+/g, "[이메일]")
    .replace(/\b0\d{1,2}-\d{3,4}-\d{4}\b/g, "[전화]")
    .replace(/\b\d{4}[-\s]?\d{4}[-\s]?\d{4}[-\s]?\d{4}\b/g, "[카드번호]");
}
```

이 함수는 앱 안이 아니라 서버 함수에서 동작하고, 2026년 7월 14일 운영에 반영해 앱 1.1 시점의 운영본이 되었습니다. 보통은 여기서 "개인정보를 보호합니다"라고 쓰고 끝냅니다. 저는 그 문장을 쓰기 전에 재보기로 했습니다.

③ 활용 결과 어떤 변화가 있었나요?

손님이 보낼 법한 문장 59개를 만들고, 서버에 들어간 정규식을 그대로 옮겨 돌렸습니다. 개인정보가 든 문장 36개와 개인정보가 없는 평범한 문의 23개입니다.

■ 측정 결과 (시험 문장 59개 기준)

· 운영본 v7 : 탐지율 66.7% / 오탐률 21.7%

· 개선안 v8 : 탐지율 83.3% / 오탐률 26.1%

· 개선안 v9 : 탐지율 83.3% / 오탐률 4.3%

첫째, 제가 이미 운영에 올린 코드가 개인정보 3분의 1을 놓치고 있었습니다. 놓친 12건은 점을 쓴 번호(010.1234.5678, 02.123.4567) 2건, 국제표기(+82 10-1234-5678 등) 3건, 구분자(하이픈·점·띄어쓰기) 없는 유선번호(0212345678) 1건, 은행 계좌번호 2건, 한글로 쓴 번호(공일공 일이삼사) 1건, 차량번호·여권번호·주소 각 1건입니다.

둘째, 구분자와 국제표기를 넓혀 탐지율을 83.3%로 올렸더니 오탐이 같이 늘었습니다(21.7% → 26.1%). 탐지만 보고 고치면 반대쪽이 나빠집니다.

셋째, 오탐의 진짜 원인은 카드번호 규칙이 길이만 봤다는 것이었습니다. 예약·고객관리를 하는 매장에는 "주문번호 2026 0915 0001 0042" 같은 문장이 일상적으로 오는데, 전부 카드번호로 가려지고 있었습니다. 그래서 실제 카드번호만 통과하는 자릿수 검사식(룬(Luhn) 검증)을 돌리고, 앞에 주문·예약·쿠폰·기프티콘·영수증 같은 말이 오면 건드리지 않게 했습니다. 두 장치를 함께 붙여야 합니다 — 오탐 5건 중 2건은 룬 검증을 통과해 버려서, 문맥어 규칙이 없으면 그대로 남습니다. 탐지율은 그대로 두고 오탐률만 26.1%에서 4.3%로 내렸습니다. 남은 오탐 1건은 손님이 가게 유선번호를 확인차 적은 경우인데, 이는 가려도 손해가 작다고 보고 남겼습니다.

④ 나만의 방식 또는 개선 포인트는 무엇인가요?

첫째, 제 코드를 제가 깨는 문장부터 만들었습니다. 잘 되는 예시가 아니라 "이렇게 쓰면 새겠는데" 싶은 표기법을 먼저 적었고, 그래서 첫 측정에서 66.7%라는 낮은 숫자가 나왔습니다. 그게 이 작업에서 가장 쓸모 있는 결과였습니다. 오탐 쪽 시험 문장도 같은 방식으로, 일부러 걸리기 쉽게 만들어야 합니다. 처음에 저는 평범한 문의만 넣어 오탐 0%가 나왔는데, 긴 숫자가 들어간 실제 문의를 넣자 21.7%가 나왔습니다. 물러터진 시험지는 통과해도 의미가 없습니다.

둘째, 이 일을 AI에게 맡기지 않기로 한 판단이 오히려 중요했습니다. 정규식이 못 잡는 주소·계좌번호는 원래 AI 분류기(문장을 읽고 종류를 판정하는 모델)가 잘 잡는 영역입니다. 그런데 그러려면 개인정보를 가려낼 목적으로 그 개인정보를 통째로 외부 AI에 한 번 더 보내야 합니다. 국외 이전을 줄이려고 만든 장치가 국외 이전을 늘리는 역설입니다. 그래서 정규식으로 확실하게 가려낼 수 있는 것만 정규식에 맡기고, 나머지는 구조로 막았습니다.

셋째, 그 구조가 실제 방어선입니다. 계좌번호·한글로 쓴 번호·차량번호·여권번호·주소 5개 유형(시험 문장 기준 6건)은 v9에서도 못 막습니다. 대신 애초에 손님 이름·전화·방문 이력·고객 프로필은 AI로 보내지 않고, 나가는 것은 손님이 방금 보낸 질문 문장 하나뿐입니다. 마스킹은 마지막 방어선이지 첫 번째 방어선이 아닙니다.

⑤ 다른 사람도 따라 할 수 있나요?

사장님이 오늘 바로 하실 수 있는 것부터 적겠습니다. 쓰시는 AI 응대 도구에 이 세 가지를 그대로 물어보십시오.

1. 손님이 보낸 문장이 어디로 갑니까? 국내입니까, 국외입니까?
2. 그 문장이 그쪽 AI 학습에 쓰입니까?
3. 전화번호는 보내기 전에 가려집니까?

손님께 드리는 고지 문구도 한 줄이면 됩니다. 매장 안내나 자동 응답 첫 줄에 쓰시면 됩니다. "문의 내용은 예약 확인과 답변을 위해 국외 AI 응대 서비스에 전달됩니다. 전화번호 등 개인정보는 전달 전 자동으로 가리도록 하고 있으나, 표기 방식에 따라 일부는 걸러지지 않을 수 있습니다. 민감한 정보는 문자로 보내지 마시고 전화로 문의해 주세요." 제 측정에서 3분의 1이 안 걸러졌기 때문에, "가려집니다"라고 단정하지 않고 이렇게 적는 것이 정확합니다.

직접 도구를 만드는 분이라면 ②의 함수를 출발점으로 삼으셔도 됩니다(제가 작성한 코드이고, 자유롭게 가져다 쓰시라고 공개합니다). 다만 ③에서 보듯 이 버전은 점 표기와 국제표기를 놓치므로 구분자를 넓혀 쓰시고, 카드번호에는 룬 검증과 문맥어 규칙을 반드시 함께 붙이십시오. 그다음 본인 손님 말투로 문장 50개 정도만 만들어 돌려보시면 됩니다. 요령은 셋입니다.

1. 개인정보가 든 문장과 들지 않은 문장을 함께 넣어 탐지율과 오탐률을 같이 잽니다.
2. 표기 변형(점·띄어쓰기·국제표기)을 반드시 넣습니다.
3. 못 막은 유형을 지우지 말고 그대로 적어 둡니다.

> **측정 방법**
>
> 시험 문장 59건(개인정보 포함 36건 / 정상 23건)을 유형 표시와 함께 사전에 고정하고, 서버 함수의 정규식 5개를 측정 직전 현재 소스와 한 줄씩 대조해 동일함을 확인한 뒤 파이썬으로 옮겨 실행했습니다(대조일 2026년 9월 16일). '성공'은 원문에 있던 개인정보 글자가 결과에 한 글자도 남아 있지 않은 경우로만 인정했습니다. 스크립트 전문, 문장 59건 전문, 실행 출력 원문을 첨부 PDF에 그대로 실었습니다. 스크립트를 실행하면 같은 표가 다시 나옵니다.

【출처 · 윤리 · 한계】

· 통계는 중소벤처기업부·소상공인시장진흥공단 「2024년 기준 소상공인실태조사」(2026.3.13. 발표)와 중소기업중앙회 「소상공인 DX·AX 현황 및 정책 수요 조사」(소상공인 500개사, 2026.6.9. 발표)에서 인용했습니다. 80.0%에는 키오스크·배달앱 등 일상 도구가 포함되므로 AI 응대 보급률로 읽으시면 안 됩니다.

· 탐지율 66.7%·83.3%와 오탐률 21.7%·26.1%·4.3%는 제가 직접 측정한 값입니다. 시험 문장은 제가 만든 것이고 실제 손님이 보낸 문장이 아닙니다. 따라서 이 숫자는 실사용 환경의 유출률이 아니라 제가 설계한 59개 문장에 대한 값이며, 표본을 늘리면 달라집니다. ①의 도입 장면도 실제 관찰이 아니라 설명을 위해 구성한 것입니다.

· 시험 문장에 쓴 전화번호·주민등록번호·사업자등록번호·계좌번호·여권번호·차량번호·이메일 주소·이름·주소는 모두 형식만 맞춰 지어낸 값이며 실존하는 개인·사업자·계좌의 정보가 아닙니다. 주민등록번호는 공식 검증식을 통과하지 않는 값으로 만들었고, 이메일은 배달되지 않는 예약 도메인(example.com)을 썼습니다. 카드번호는 결제사가 공개 문서에 게시한 시험용 번호입니다(실제 발급 카드 아님).

· 개선안 v8·v9는 로컬 측정까지 마친 상태이고 운영 반영은 다음 버전 예정입니다. 아직 반영되지 않은 것을 반영된 것처럼 쓰지 않았습니다. 현재 운영에 들어간 것은 탐지율 66.7%·오탐률 21.7%인 v7입니다.

· 측정 대상이 제 제품이므로 유리한 결과를 고를 유인이 있었습니다. 그래서 실패 목록을 먼저, 그리고 전부 공개했습니다.

· 도입 매장 수·사용자 수·매출은 심사위원이 검증할 수 있게 공개된 수치가 아니어서 쓰지 않았고, 이 글은 제품 홍보가 아니라 측정 기록입니다.

· 법적 근거: 「개인정보 보호법」 제28조의8(개인정보의 국외 이전), 제29조(안전조치의무). 과학기술정보통신부 「인공지능(AI) 윤리기준」(2020.12.23.) 10대 핵심요건 중 '프라이버시 보호'(전송 전 마스킹·최소 전송), '데이터 관리'(국외 이전 경로와 수탁 구조 공개), '투명성'(못 막은 유형과 오탐을 본문에 그대로 공개)에 해당합니다. 최종 법률 검토는 개인정보 전문 변호사 확인을 권합니다.

· 첨부한 표·도표는 실행 출력과 스크립트의 텍스트를 그대로 옮겨 가독성을 위해 조판한 것이며 원본 화면 캡처가 아닙니다. 내용의 추가·삭제·수정은 없습니다. 조판에도 Claude를 사용했습니다.

· 같은 제품을 소재로 업무 생산성 분야에도 응모했으나, 그 편은 출시 전 검수 구조에 대한 기록이고 이 편은 개인정보 마스킹 측정 기록입니다.

· 이 글의 초안 정리에 Claude를 사용했고, 표의 모든 값은 실행 출력과 한 줄씩 대조했습니다.
