2026 Global Public HR Conference | Session 3 | Oct 30, 2026

Generative AI Training for Busan Civil Servants

From using AI,

To owning its results.

3 Levels

Groups · Contest

AI internalization

Lee Chang-sub · Busan City



From using AI as a tool to owning its results

Here is how today's talk is organized

01

Background

Budget to people
Why we changed

02

Design

From tool use to accountability in 4 steps

03

Levels

Beginner · Intermediate · Advanced

04

Scaling

9 services · pitfalls
contest · live services

05

Next Steps

All-staff
AI internalization

Key Message

When those who know the work **build and own it**, AI becomes our strength.

Use → Build → Verify → Own

AI Training : What Should We Teach?

The tools change every month

ChatGPT	Gemini	Claude	Copilot	Perplexity	NotebookLM
Cursor	**Officer Bugi**	EXAONE	Llama	DeepSeek	Grok

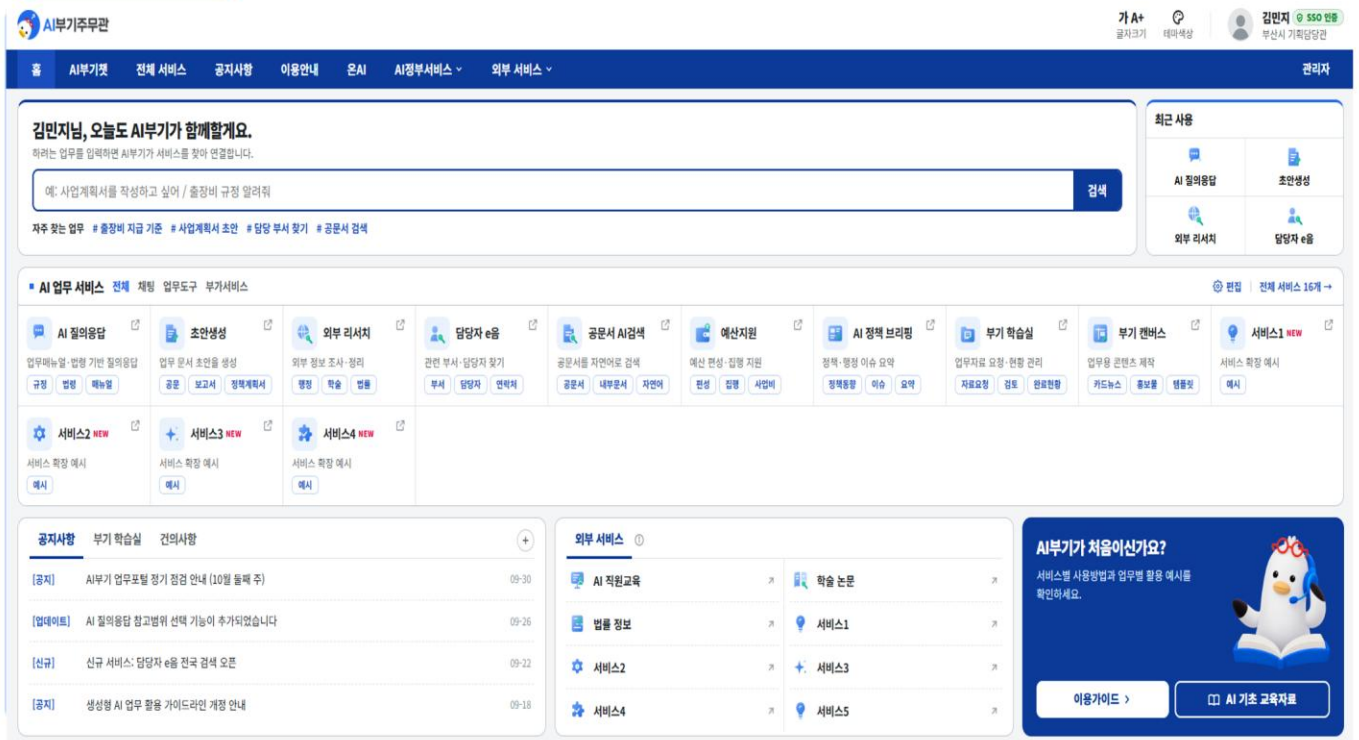
What we taught last year has already changed.

So what should civil-servant AI training leave behind?

We built our own AI inside the government network

Screenshot

First, we invested budget to build Busan's AI service, "AI Officer Bugi"



Internal Gov. Network

Built and run by the city,
not dependent on vendors

For All Staff

Reports, replies to complaints,
document summaries

Secure Environment

Closed network — administrative
data never leaves

Budget built a **safe foundation**. But civil servant usage remained low.

From buying systems, to growing people

From spending budget to build systems, to developing the staff who build them

Old way

Build with budget

- Budget spent to build AI Officer Bugi
- New features need new budget and time
- Field needs get lost in the handoff

New way

Grow people who build

- Staff who know the work build it
- Needed features made and fixed fast
- Builders run it and own the answers

Build the platform once with budget; the services on top are built **by staff who know the work**

From training to live services: one pipeline

Learn → Build → Verify → Compete → Deploy

1	2	3	4	5
Skills Training	**Study Groups**	**AI Best Practice Contest**	**Live Services**	**AI Champion Certified Institution**
Beginner · Inter. · Advanced	From planning to building 9 services developed	Verify real-world use	Deployed to real work	Gov. certified from 2027 Upgraded curriculum

The goal is not course completion. It is for what trainees build to be deployed in real work.

AI Capability Training

Beginner → Intermediate → Advanced: what changes?

STEP 1 · Beginner

Use

Use AI

Gemini · NotebookLM
Prompt-based drafts

Output: press release draft

STEP 2 · Intermediate

Build

Make with AI

Vibe coding · Apps Script
No-code app building

Output: work automation app

STEP 3 · Advanced

Engineer

Engineer AI

LangChain · RAG · sLLM
Design pipelines in code

Output: department chatbot

STEP 4 · Study Group

Scale

Scale AI

10-session deep-dive
Apply & refine on the job

Output: live platform

AI as a collaboration tool

AI I control and am accountable for

Instead of working within rules set by others, we set the rules ourselves. The more control, the more we can own the results.

Beginner: AI as a Collaboration Tool

Textbook "Public AX Insight" — from concepts to text, visuals, images and video

1 AI Basics — First understand how AI produces answers, then move on to the tools

Text Session 2 **Prompt writing**	Text Session 3 **Writing official reports**	Text Session 4 **Press releases (Gems)**
Visual Session 5 **NotebookLM**	Image Session 6 **Google Mixboard**	Video Session 7 **Google Flow**

Turning one task into text, visuals, images and video shows that the core is not the tool but the ability to handle the work.

Textbook: "Public AX Insight: AI Capability Training"

01 Beginner

Making with Gemini & Google tools

STEP 1 · Use



Bugi introducing the Gwangalli drone show



Bugi promoting the wholesale produce market

Intro Design drafts and prompts were made in Gemini, then turned into short videos with Flow.

Intermediate: Building what your department needs

A structured course, from understanding generative AI to building services that actually run

1 **Generative AI Basics**
Generative AI for gov. innovation

2 **What is Vibe Coding?**
A new way to build with AI, by talking

3 **Using Gemini Wisely**
Using it safely matters as much as using it well

4 **Prompt Design**
Good results start with structured prompts

5 **Gemini + Canvas**
Start light, but finish with real output

6 **Canvas Practice**
Build a Busan festival guide web app

7 **Apps Script Apps**
Turn repetitive tasks into automation apps

8 **AI Studio Monitor**
A decision-support dashboard linking data and AI

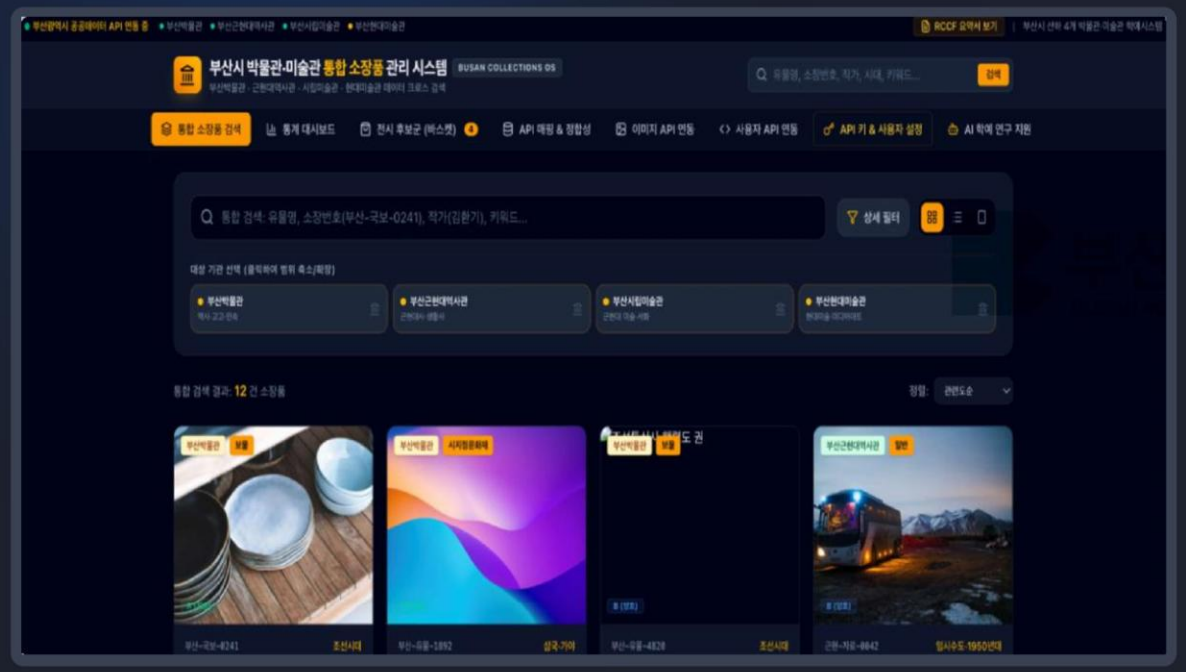
9 **Antigravity Adv.**
Beyond screens, to live services

The last module goes 'beyond screens to running services.' This is where training leads to live services.

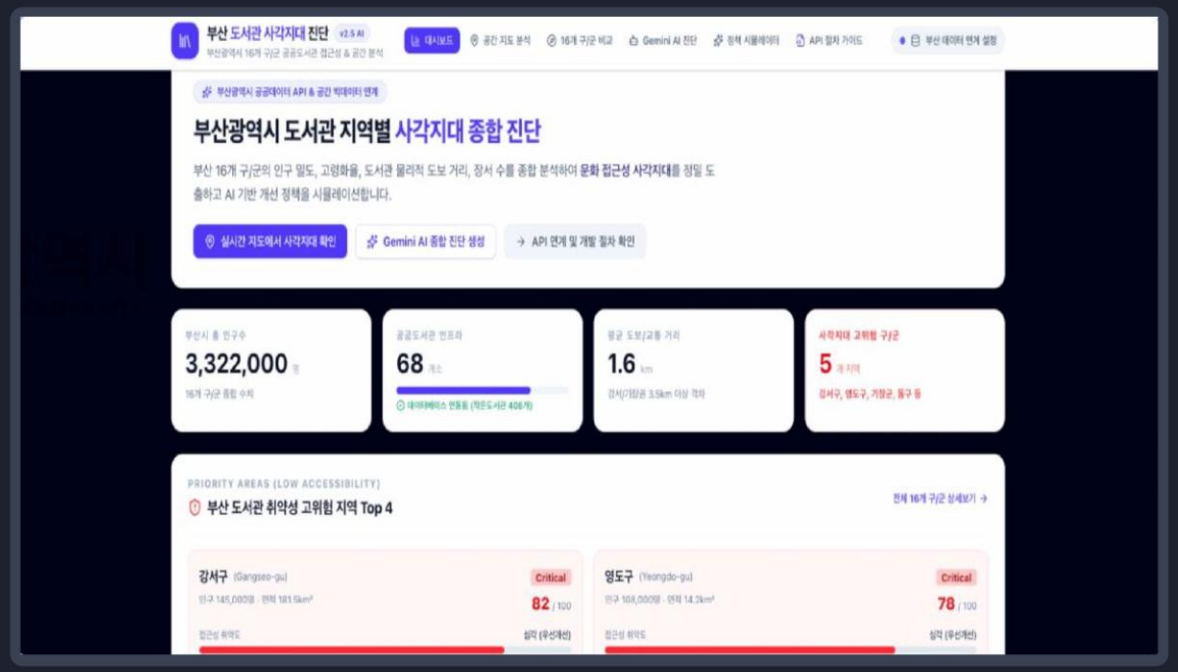
02 Intermediate

Programs built by staff themselves

STEP 2 · Build



Integrated museum & gallery collection management



Diagnosing public library service gaps

Intro

In class, staff refined ideas with Gemini and, without writing code themselves, built working programs with Antigravity.

Advanced: Leave in 3 days with a deployed service

Hands-on building, not concept lectures — 8 hours × 4 modules a day, 12 modules in total

Day 1

LangChain basics · work use

- M1 Onboarding · dev setup
- M2 LangChain basics
- M3 Work use · agents
- M4 Preprocessing · de-ID

Outputs
3 work automation chains
Tool-calling agent
Cleaned, de-identified dataset

Day 2

RAG · LangGraph build

- M5 RAG basics
- M6 Advanced RAG
- M7 LangGraph basics
- M8 Modular RAG

Outputs
Dept. rules Q&A chatbot
Retrieval quality report
Self-correcting graph

Day 3

Local sLLM · deployment

- M9 Running sLLMs
- M10 Moving on-premises
- M11 Streamlit development
- M12 Deploy · operate

Outputs
Closed-network sLLM RAG
Streamlit web app
A URL you can open now

Quote "What you take home tonight is not code but a single URL — a chatbot you can open and use right away." (Textbook, Day 3)

'Accountability' built into every module

Where verification, security and trust sit across the 12 modules

M1 How LLMs work

We start with why AI gets things wrong.
You must understand to doubt.

M4 PII masking · de-ID

What goes in and out is decided
at the data stage.

M6 Retrieval quality

Not “it seems better” —
we prove improvement with numbers.

M8 Self-correction

We build systems that find
and fix their own errors.

M10 Zero external calls

We confirm, not assume, that data
never left the network.

M11 Source-showing UI

Screens are designed so users can
check the evidence themselves.

Not a separate class — verification and security are built into every stage.

Learned separately, they become rules; learned while building, they become habits.

03 Advanced

Vibe coding: how far can it go?

STEP 3 · Engineer

10_onpremise_rag.ipynb

```python
from langchain_ollama import ChatOllama

# 어제:  llm = ChatOpenAI(model="gpt-4o-mini", temperature=0)
# 오늘:
llm_local = ChatOllama(model="gemma3:4b", temperature=0)

# 리트리버도 로컬 인덱스로
retriever_local = vs_local.as_retriever(search_kwargs={"k": 4})

# 어제의 rag() 함수를 그대로 재사용
def rag_local(question: str) -> dict:
    docs = retriever_local.invoke(question)
    ctx = "\n\n".join(
        f"[{d.metadata.get('source')} {d.metadata.get('article')}]\n"
        f"{d.page_content}" for d in docs)
    msg = [("system", SYSTEM),
           ("human", f"참고자료:\n{ctx}\n\n질문: {question}")]
    return {"answer": llm_local.invoke(msg).content, "docs": docs}

print(rag_local("출장여비 정산 기한은?")["answer"])
```

On-premises RAG on a local sLLM

app.py (배포본)

```python
import os, streamlit as st

# 배포 환경에서는 st.secrets 로 키를 읽는다
os.environ["OPENAI_API_KEY"] = st.secrets["OPENAI_API_KEY"]

@st.cache_resource
def load_engine():
    from langchain_openai import (OpenAIEmbeddings,
                                  ChatOpenAI)
    emb = OpenAIEmbeddings(model="text-embedding-3-small")
    vs = Chroma(persist_directory="./chroma_public",
                embedding_function=emb)
    llm = ChatOpenAI(model="gpt-4o-mini", temperature=0)
    return vs.as_retriever(search_kwargs={"k": 4}), llm
```

Streamlit deployment (app.py)

Intro Trainees ran their own RAG system locally with no security risk, and even refined the tone of its answers.

Study Groups: Turning training into services

To keep and grow what they learned, graduates plan and build services themselves

1 Plan
Define real dept. problems as service ideas

2 Design
Structure data and features

3 Build
Make it with AI tools and test it works

4 Verify
Peer demos, trust & security checks

9 services built
From planning to real build, all the way to completion

The real start came after training ended

One-off training is forgotten within weeks. In the study groups, graduates stayed on their own, took on real department problems, and carried them through planning, design, building and verification.
No one told them to. They defined problems, verified results and presented to peers, showing real ownership.

Study groups built 9 services

All started from real problems in everyday department work

1 **Aerini**
Real-time dashboard of work status across departments

2 **AI Jump**
Dashboard showing youth facilities on a map and list

3 **Kaltoe Fairy**
Platform supporting Agricultural Center staff and trainees

4 **SafeDOC**
Detects personal data in documents to reduce leak risk

5 **Urban Flood Zero**
Uses weather radar to auto-show CCTV of rainy areas

6 **AI Jaemmini**
Structures types of workplace harassment to help judge cases

7 **Ildangbaek**
Gathers scattered EV info to answer citizen inquiries

8 **Ichangyong**
AI platform managing complaint-response quality

9 **BHUB**
Integrated support system for Busan's veterans

01

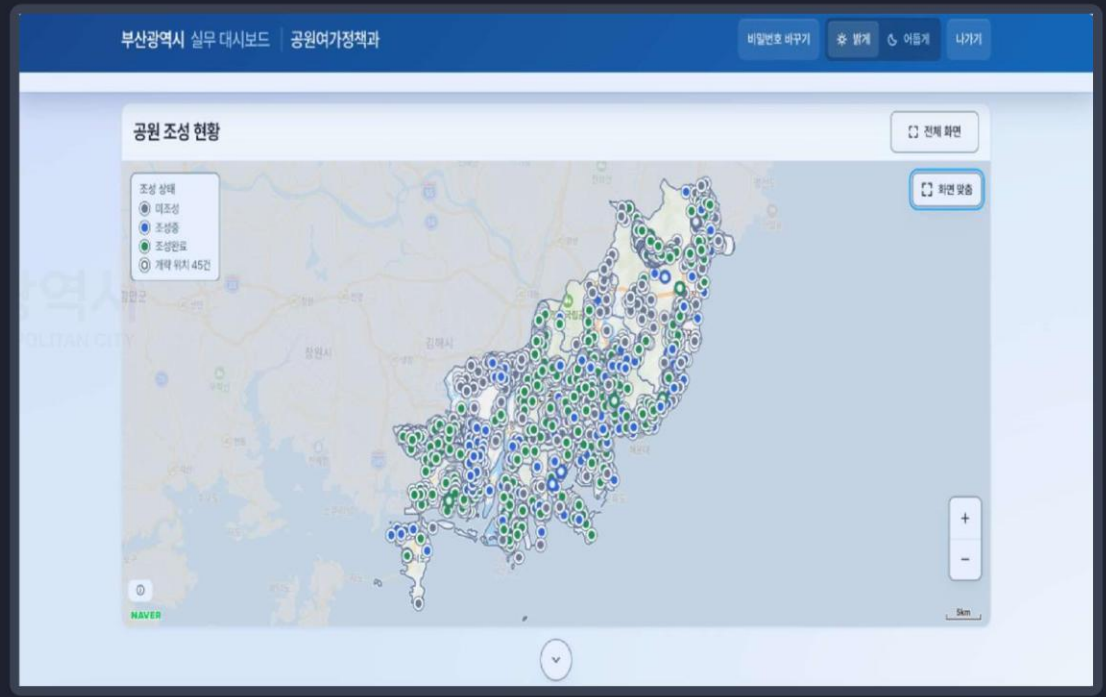
Aerini

Integrated dept. work-status dashboard

Study Group

9 teams · 15 members





Intro

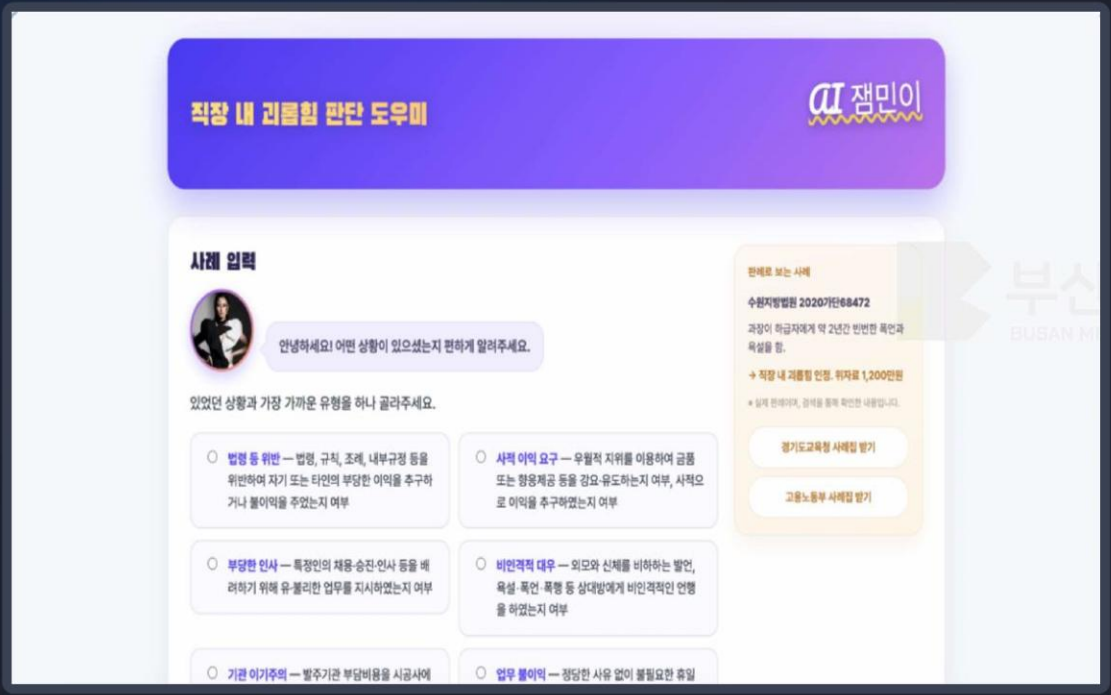
Staff from 3 departments merged repetitive, inefficient tasks into a single dashboard. It combines Naver Maps API, the Public Data Portal API, V-World, Eum land data and internal Excel files.

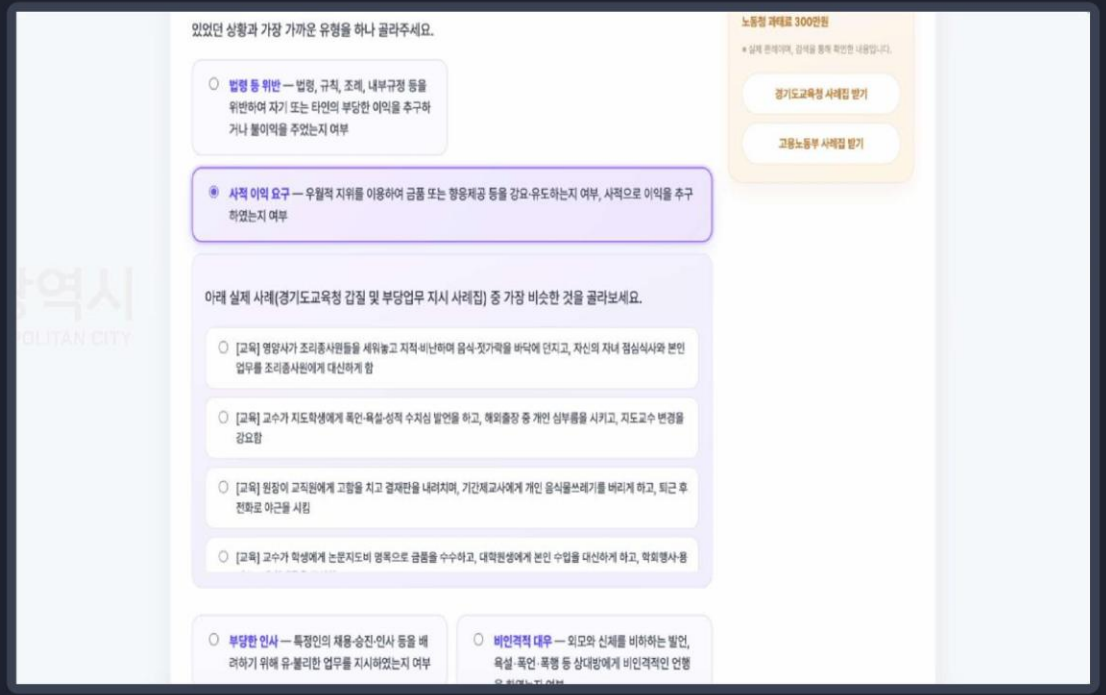
02 AI Jaemmini

Harassment assessment helper

Study Group

9 teams · 15 members





Intro

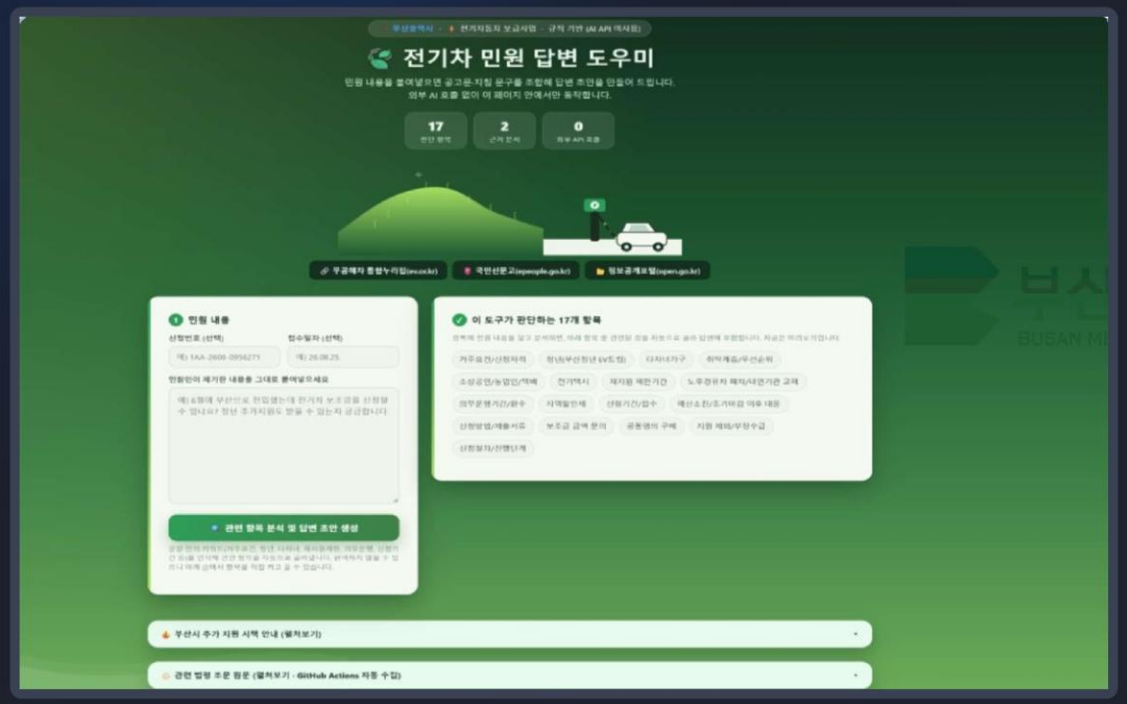
Using RAG to structure types of workplace harassment, it helps staff judge whether their situation qualifies and helps the organization prevent similar cases.

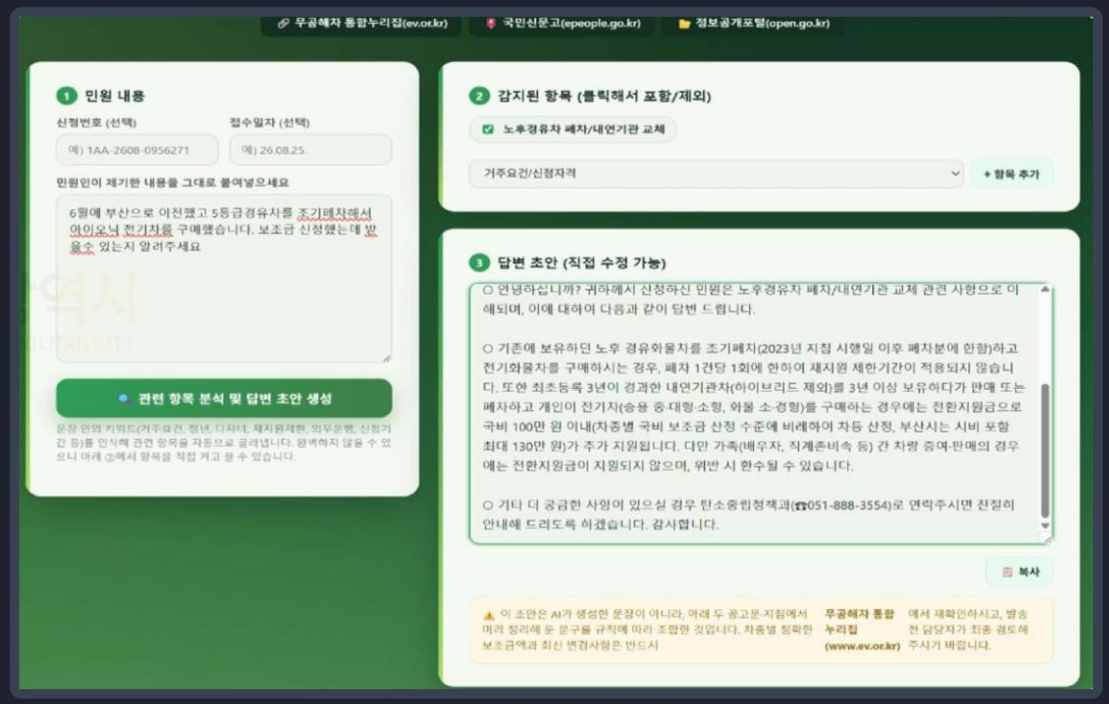
03 Ildangbaek

EV inquiry response assistant

Study Group

9 teams · 15 members





Intro

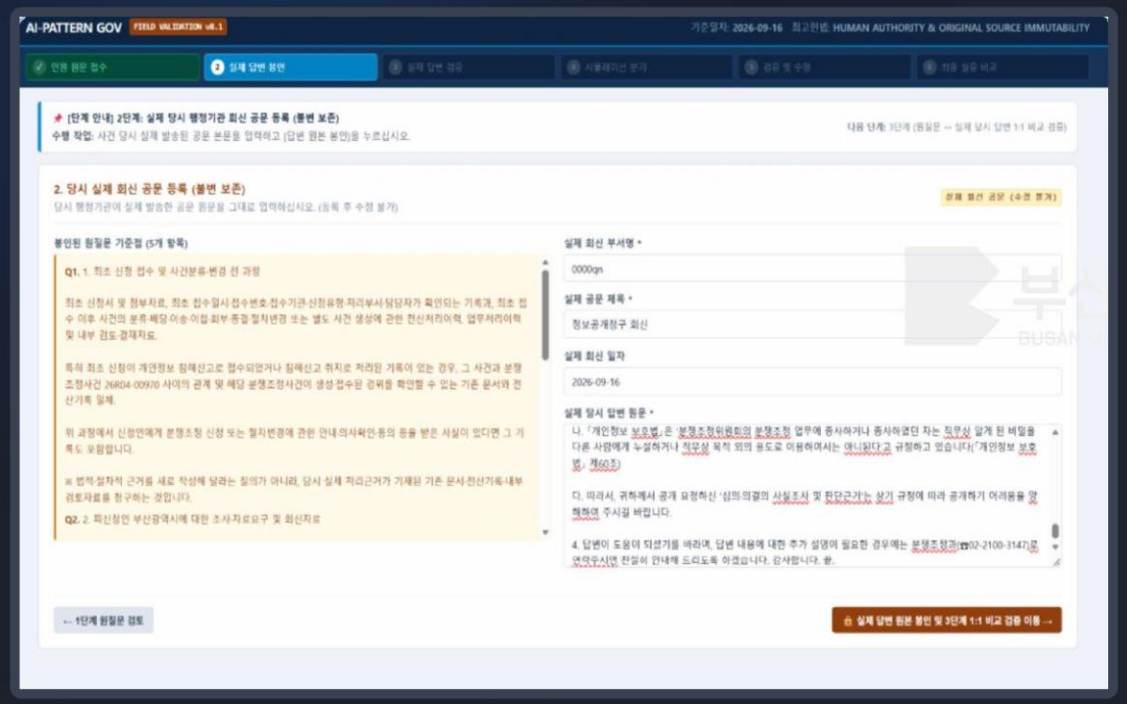
It gathers EV information scattered across many websites, analyzes situations citizens describe in plain language, and tells them which subsidies they can receive.

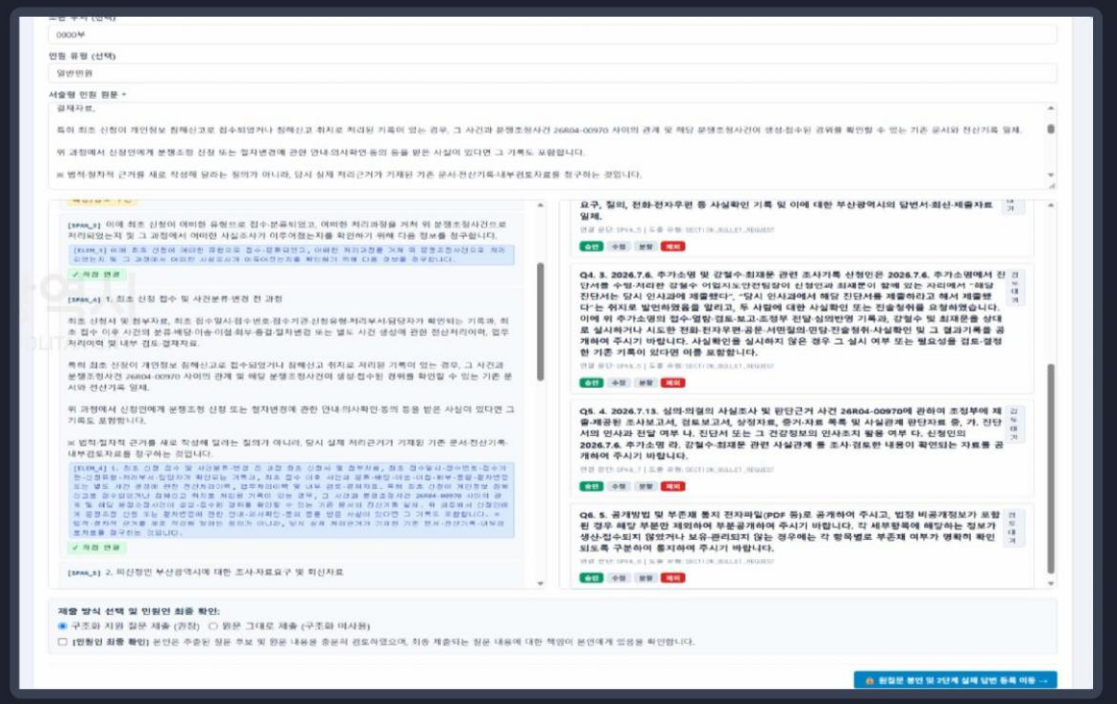
04 Ichangyong

Complaint history & quality

Study Group

9 teams · 15 members





Intro

Complaints are hard to track and easily lost when staff change. It organizes similar complaints and past replies in time order to keep answers consistent and suggest next steps.

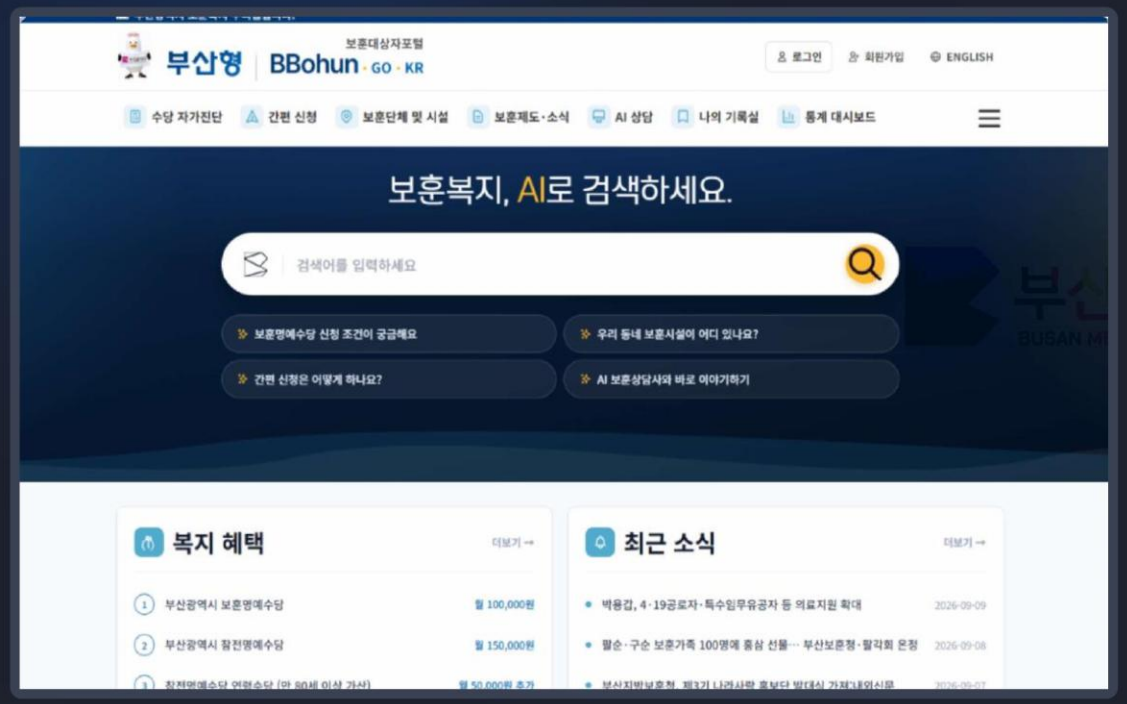
05

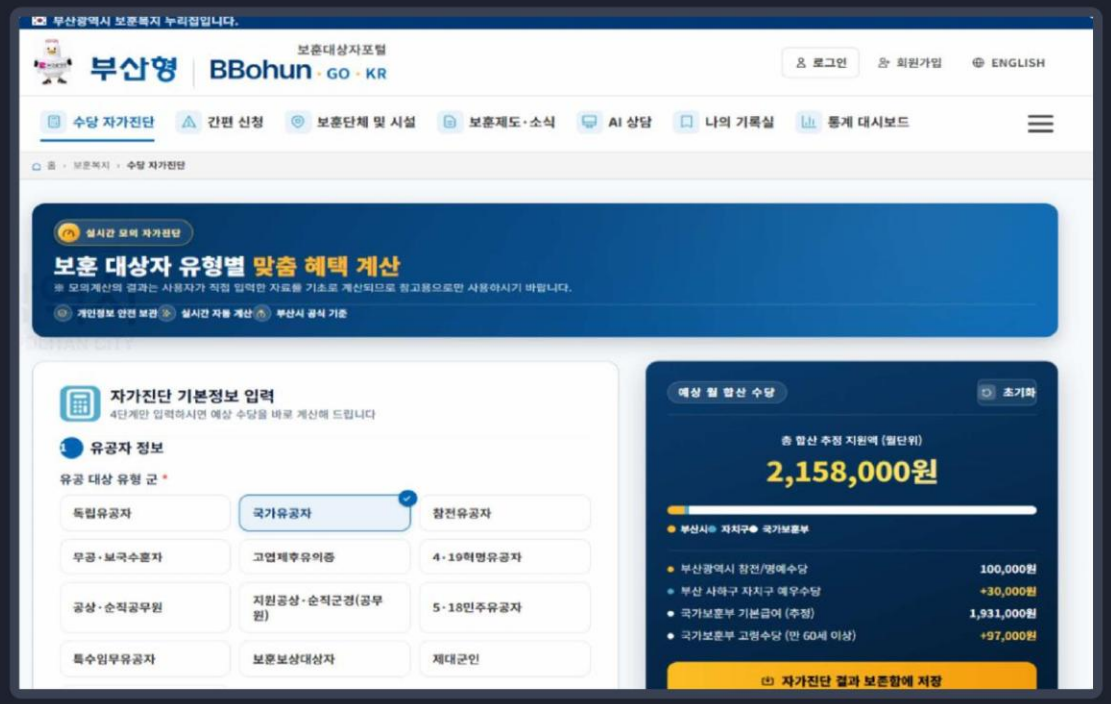
BHUB

Integrated veterans' support

Study Group

9 teams · 15 members





Intro

Many veterans missed benefits they didn't know they qualified for, and staff had to search many sites to check eligibility.
With BHUB, anyone can check eligibility and benefits in one place.

What we learned by building

Once vibe coding let anyone build, the same mistakes kept repeating

45% AI code that failed security tests

Veracode, 2025

~20% AI-suggested packages that don't exist

USENIX Security, 2025

170+ Vibe-coded apps exposing personal data

CVE-2025-48757

Data · Privacy

① Personal data into external AI

Code · Dev Env.

② API keys written in code
③ Login but no auth check
④ Unvalidated input
⑤ Unverified packages

AI-Specific Risks

⑥ Ignoring prompt injection
⑦ Unchecked AI answers to citizens
⑧ Agents with prod DB access

Operations

⑨ Deploying without approval
⑩ Services with no handover

"Working code" ≠ "safe code." Better models don't automatically mean better security.

※ Training material: "Build AI Services Like This and You'll Have an Incident" (10 mistakes and how to prevent them)

Block or allow? Manage by level of use

Block everything and innovation stops; allow everything and incidents happen

Level 1 Personal Use

Used alone on my PC,
no personal data

**Use freely —
just follow the basic rules**

Level 2 Team Sharing

Shared with colleagues,
handles internal data

**Fill out self-checklist,
register with the dept.**

Level 3 Public · PII

Used by citizens or
handles personal data

**Security review +
secure-coding check**

10-Item Self-Check Before Sharing or Deploying

Tested with fake data?	No API keys in the code?	Server-side auth checks?	Input validated?	Package actually exists?
Ignores instructions in docs?	Sources shown in answers?	Minimal agent permissions?	Registered with the dept.?	Operations doc written?

Fake data | **Least privilege** | **Review before deploy**

The Contest: Verifying it really works

We check whether what was built can really be used — and turn winners into live services

1 2 3 4

Discover

Services & ideas from study groups and depts.

Review

Verify practical use:
fit · reliability · security

Select

Bonus points for teams that built it themselves

Go Live

Deployed to real work, used by all staff

Why a contest?

What's built in training easily stops at 'looks plausible'

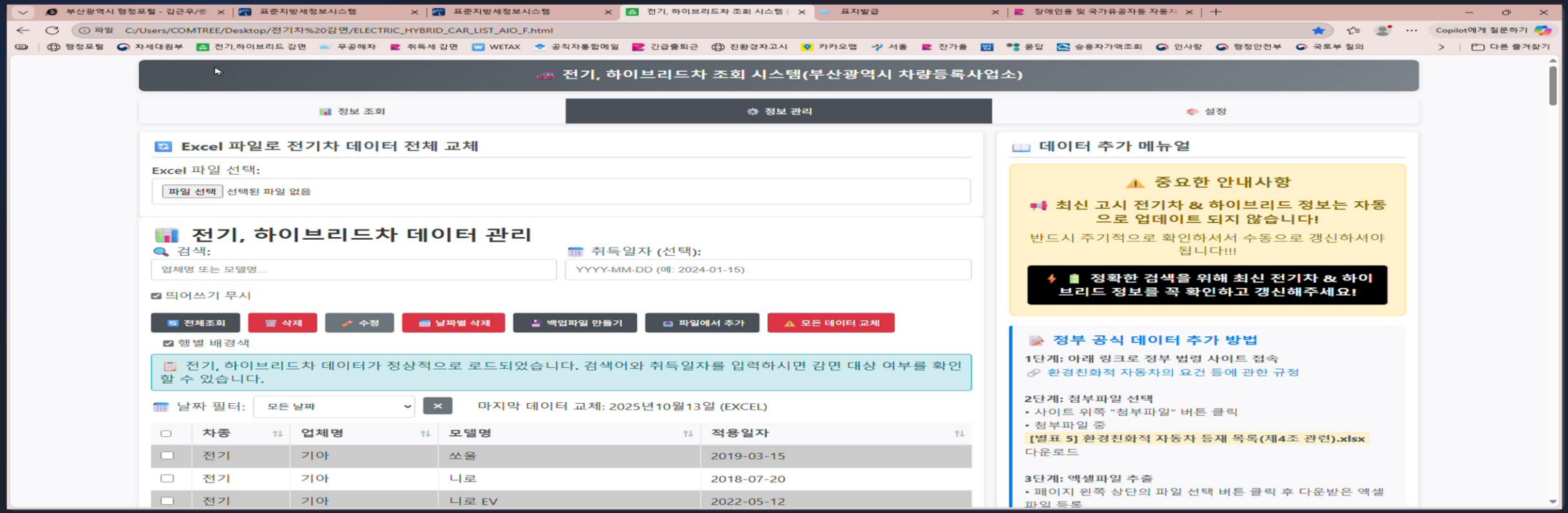
Only with a gate asking “Can we really use this?” do makers take responsibility for results.
Winning services are developed and deployed for real.



01 EV Info System

EV & hybrid vehicle lookup system

Contest Winner



Intro

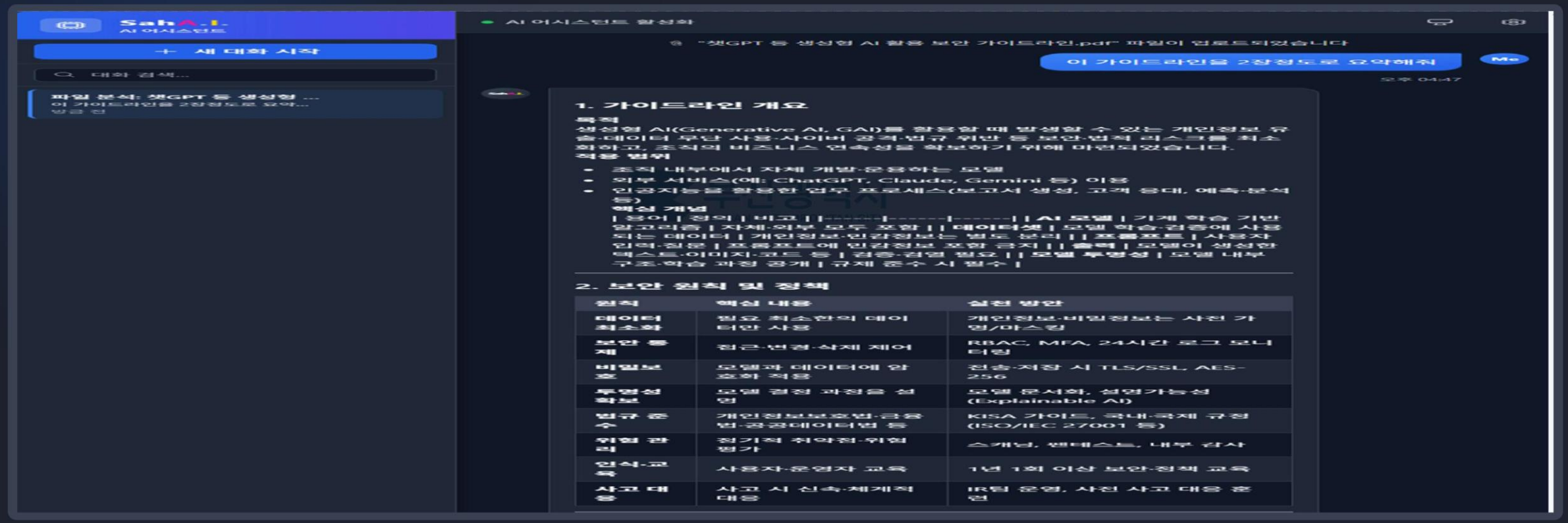
Vehicle registration office staff instantly check tax-exempt EVs and hybrids by keyword and acquisition date.
After passing the contest review, it became a live service in production.

02

SahA.I.

AI document summarizer

Contest Winner



Intro

It analyzes uploaded documents (e.g., Generative AI Security Guidelines) and summarizes them to any length.
It passed the contest review and is now a live service.

All-staff AI internalization: build, run, own

An organization where those who know the work own its AI services

"I know my work better than anyone."

1 Build

Staff who know the domain build the services they need.

"Whoever builds it looks after it."

2 Run

From registration to handover, the building team runs it.

"AI's answers carry my sign-off too."

3 Own

After verifying the sources, the officer owns the answer.

AI isn't just IT's job. It takes root when it becomes **every employee's work skill**.

2027: Certified AI Champion Institution

Making a proven system the standard — wider and deeper

So Far

Busan's own training

- 3 levels: Beginner to Advanced
- Scaled via study groups & contest
- For Busan city staff

From 2027

AI Champion Institution

- Run as a gov.-certified course
- Standardize proven curriculum
- Expand audience and scale

We now share a curriculum refined through 2 years of trial and error with more agencies via a national certification system.

If you design training, remember these five

For central and local governments preparing AI training

01	Write goals in two layers	Include both tool use (beginner/intermediate) and responsible use (advanced)
02	Always teach how AI works	Without knowing RAG or fine-tuning, no one can verify results
03	Teach verification & security	Accountability needs both how to check and how to protect
04	Build a path to deployment	Without study groups and contests, outputs stay in the classroom
05	One plan: training, scaling, leaders	When plans move separately, so does execution

What isn't in your goals **won't be in your curriculum**.

미래 대전환의 중심
해양수도 부산

If using AI well is the starting point,

owning AI results is the destination.

Use → Build → Verify → Own

Start — Destination

Tools change. But a civil servant's ability to question and own the results does not.

Thank you! Lee Chang-sub, Busan Metropolitan City

