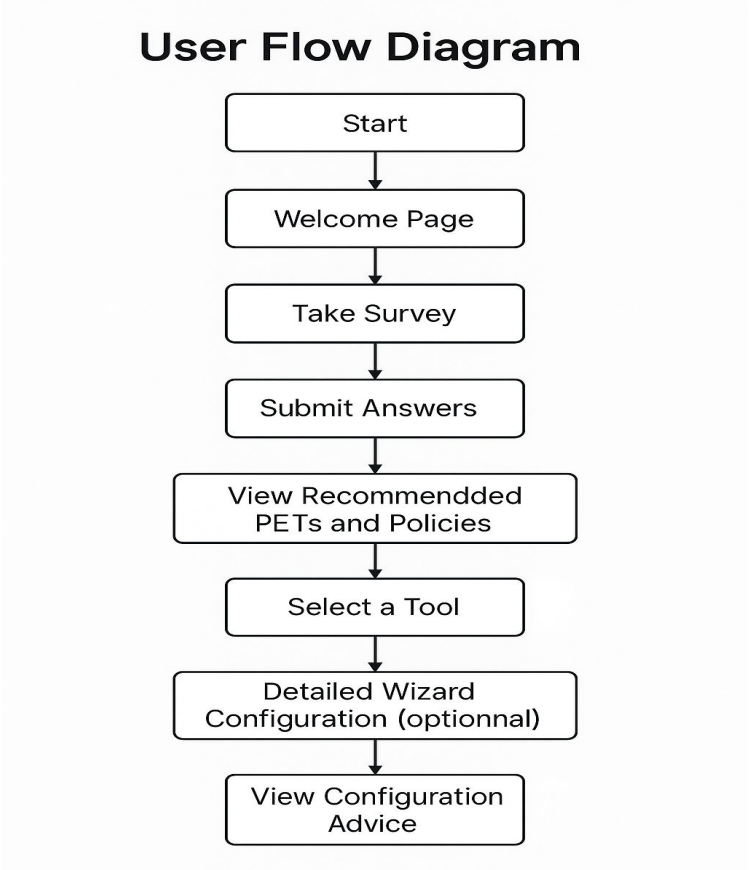
PET Advisor: Privacy Enhancing Technologies Recommendation Tool

- Webform Interactive survey and guidance tool

Tool Overview: Dynamic survey, automatic recommendations



PET Advisor

Your interactive guide to privacy-enhancing technologies

Welcome to PET Advisor

Answer a few quick questions about your data, risk profile, and compliance needs, and we'll generate a customized set of privacy-enhancing techniques—plus the key legal and regulatory measures you should follow—for your project!

Start Survey →

Questions Structure

Question 2 of 27

How accurate do results need to be?

- ○ Exact counts (0% error)
- ○ ±1–2% error
- ○ ±5–10% error

Question 3 of 27

How fast must your data analytics results appear?

- ○ Real-time/interactive
- ○ Same-day batch
- ○ Weekly/monthly

Question 12 of 27

Do you collect or process information about people in the European Economic Area (EU/EEA)?

- ◉ Yes
- ○ No

‹ Previous

Next ›

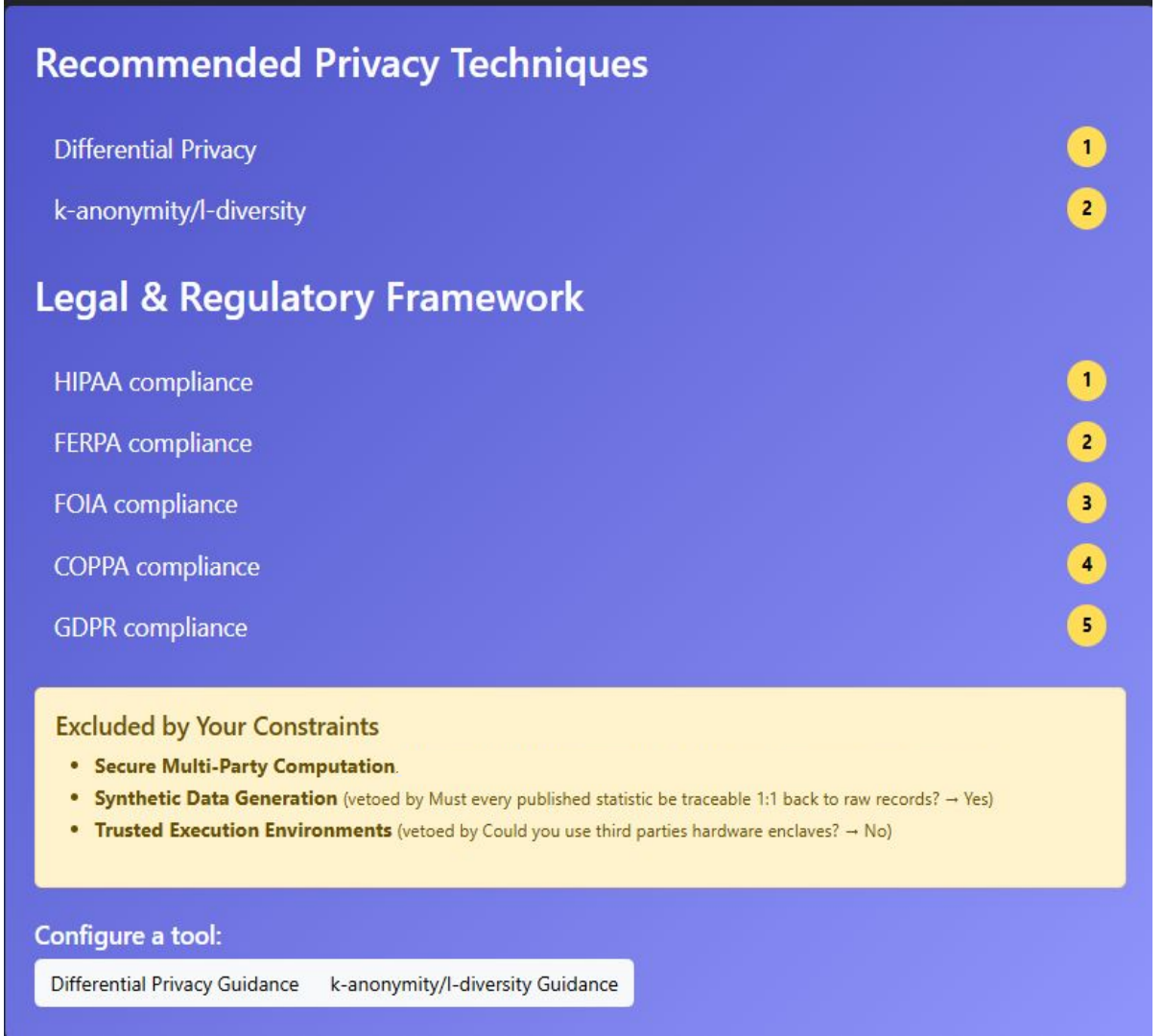
Final Output

Recommended Privacy Techniques: technology that most fit the customer needs

Legal & Regulatory Framework: legal protection can be utilized to protect users privacy, but also is the legal responsibility for the user

Excluded by Your Constraints: Explanation for the ruling out privacy tools

Configure a tool : Further explanation for the chosen tool



Database:

	Question	Answer Option
2	Could you use third parties hardware enclaves?	Yes
3	Could you use third parties hardware enclaves?	No
4	How accurate do results need to be?	Exact counts (0% error)
5	How accurate do results need to be?	±1-2% error
6	How accurate do results need to be?	±5-10% error
7	How fast must your data analytics results appear?	Real-time/interactive
8	How fast must your data analytics results appear?	Same-day batch
9	How fast must your data analytics results appear?	Weekly/monthly
10	If real-time or interactive results are needed, how do you plan to protect the data during processing and transmissio	I want hardware-based secure environments to protect data in real-time.
11	If real-time or interactive results are needed, how do you plan to protect the data during processing and transmissio	I want lightweight mathematical protection with fast outputs, even if it's a
12	If real-time or interactive results are needed, how do you plan to protect the data during processing and transmissio	I mainly need to encrypt the data in transit and storage, without adding sign

Recommended Techniques	Dealbreakers
k-anonymity/l-diversity; Differential Privacy;Trusted Execution Environments;Secure Multi-Party Computation;private-key cryptography;public-key cryptography	
Differential Privacy;Secure Multi-Party Computation;private-key cryptography;public-key cryptographyl;k-anonymity/l-diversity	Trusted Execution Environments
Secure Multi-Party Computation; Trusted Execution Environments;	Differential Privacy; Synthetic Data Generation
Differential Privacy; k-anonymity/l-diversity;	
Differential Privacy; Synthetic Data Generation	
Trusted Execution Environments; Differential Privacy; private-key cryptography; private-key cryptography; private-key cryptography; private-key cryptogra	public-key cryptography
Differential Privacy; k-anonymity/l-diversity; public-key cryptography; public-key cryptography; public-key cryptography; Secure Multi-Party Computation	
Secure Multi-Party Computation; Trusted Execution Environments	
Trusted Execution Environments; Trusted Execution Environments; Trusted Execution Environments; private-key crytography	
Differential Privacy; Differential Privacy; Differential Privacy; private-key crytography	
private-key cryptography; private-key cryptography; private-key cryptography; private-key cryptography; private-key cryptography; private-key cryptography	
Secure Multi-Party Computation; Trusted Execution Environments; private-key cryptography	
Differential Privacy	
k-anonymity/l-diversity; Differential Privacy	

Look into the database:

Question:

How accurate do results need to be?	A
How accurate do results need to be?	B
How accurate do results need to be?	C

Answer options:

Exact counts (0% error)	A
±1-2% error	B
±5-10% error	C

Recommend technique:

Secure Multi-Party Computation; Trusted Execution Environments;	A
Differential Privacy; k-anonymity/l-diversity;	B
Differential Privacy; Synthetic Data Generation	C

Dealbreaker:

Differential Privacy; Synthetic Data Generation	A
	B
	C

Decision making logic

-- Each survey answer maps to one or more privacy techniques(PT).

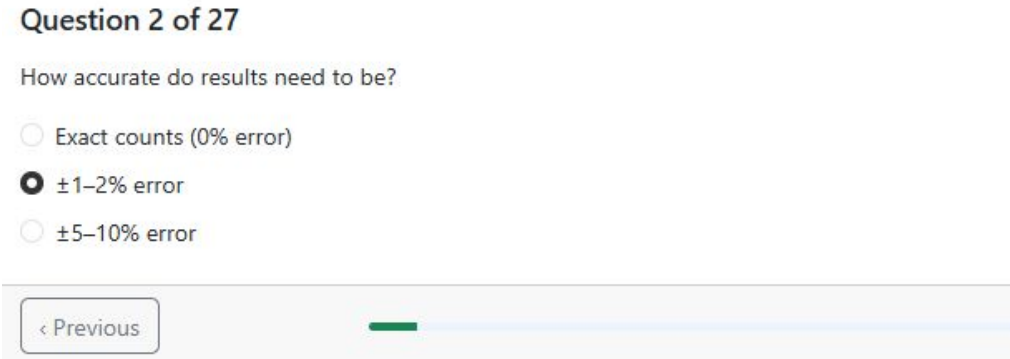
-- When selected, each mapped PTs gets +1 point.

-- Scores accumulate as user answers.

-- At the end, PTs are ranked by total score.

-- Top PTs are recommended.

Question on the web app:



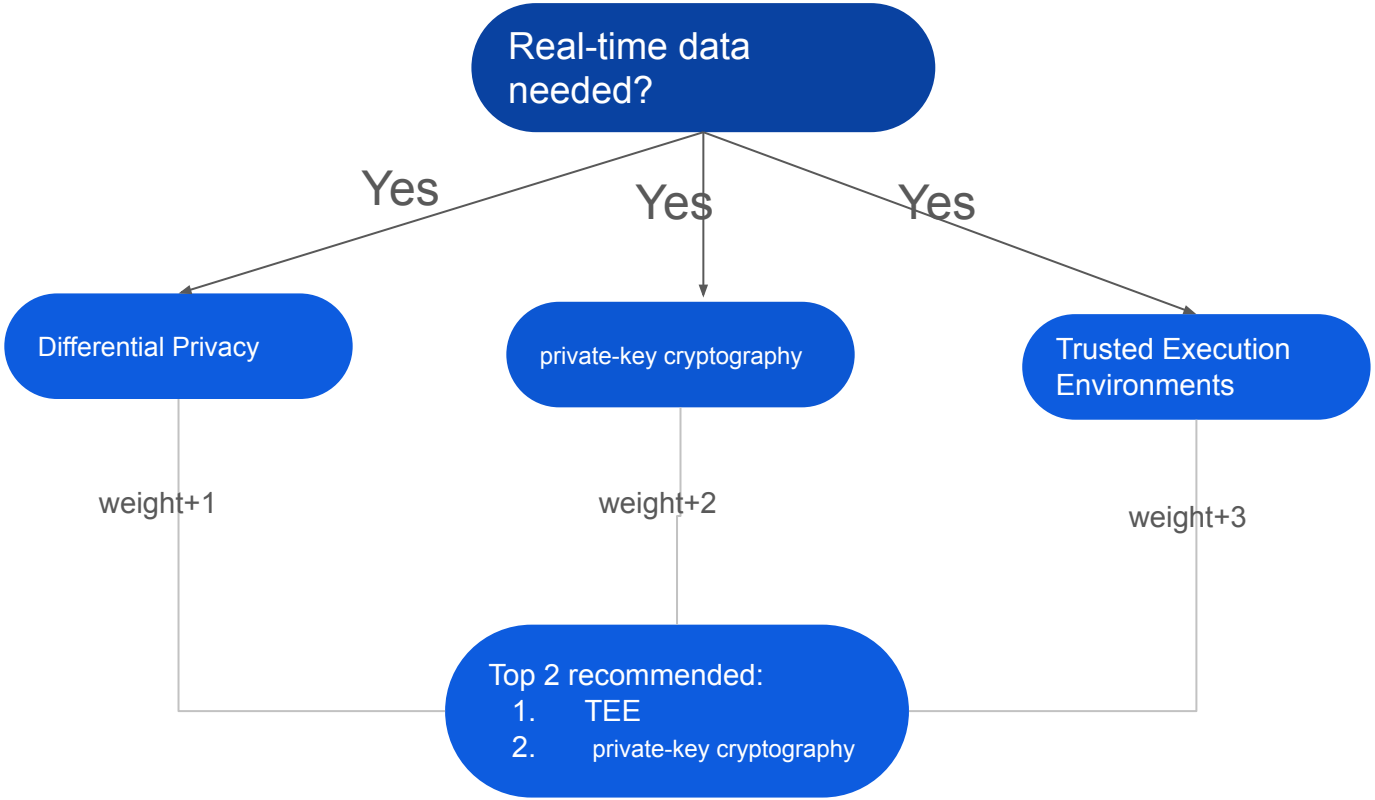
Question in the database looks like:

How accurate do results need to be?
How accurate do results need to be?
How accurate do results need to be?

add the following elements weight to the model

Secure Multi-Party Computation; Trusted Execution Environments;
Differential Privacy; k-anonymity/l-diversity;
Differential Privacy; Synthetic Data Generation

Example


Real-time data needed?
Yes
Yes
Yes
Differential Privacy
private-key cryptography
Trusted Execution Environments
weight+1
weight+2
weight+3
Top 2 recommended:
1. TEE
2. private-key cryptography

Question and answers- specific, meaningful and user-friendly

For example,

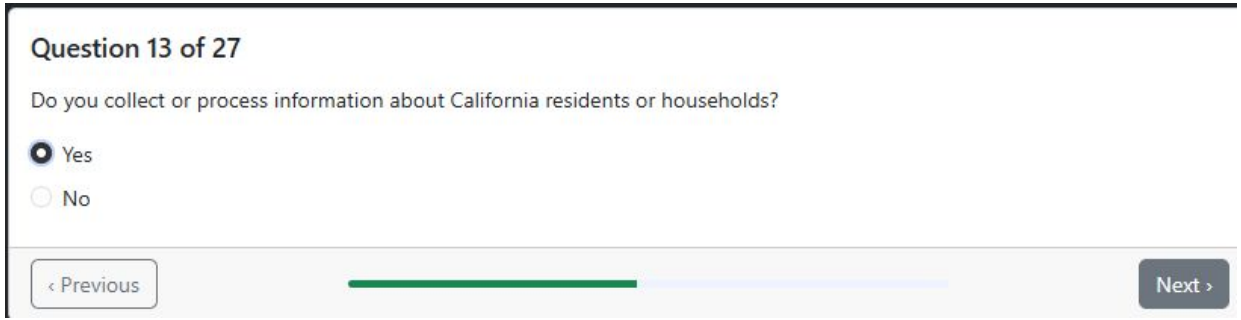
"If you share data externally, how much do you trust these partners?",

"No trust": Prefer locally process the data, (DP,TEE)

"Low trust": map strongly to recommend Secure Multiparty Computation (MPC) with a higher weight (e.g., weight = 3).

"High trust": All techniques can be recommend

Law implements



Question in the database looks like:

Do you collect or process information about California residents or households?	Yes
Do you collect or process information about California residents or households?	No

We process Legal protection differently. Instead of counting the weight, we suggest the Legal tool if it appears.

Model Details

- *Backend:* Flask (Python) manages data flow, question processing, and results generation.
- *Data Source*: privacy.xlsx contains questions, answer options, and recommendation mappings.
- *Frontend*: Vue.js (JavaScript) handles dynamic survey pages, progress tracking, and user inputs.
- *Styling*: Bootstrap 5 and custom CSS ensure responsive, clean design.

Code(partially): Loading data, extracting each column connect to the user interference

```python

# ------------------------------------------------------------------------------
# Configuration & data load
# ------------------------------------------------------------------------------
DATA_FILE = Path(__file__).parent / "privacy.xlsx"
df = pd.read_excel(DATA_FILE)

deal_col = next(
    (c for c in df.columns if "deal" in c.lower()),
    None
)

deal_map: dict[str, dict[str, list[str]]] = {}
if deal_col:
    for _, row in df.iterrows():
        q = row["Question"].strip()
        a = str(row["Answer Option"]).strip()
        raw = row.get(deal_col, "")            # e.g. "MPC; Differential Privacy"
        if pd.isna(raw) or not raw:
            continue
        pets = [p.strip() for p in str(raw).split(";") if p.strip()]
        if pets:
            deal_map.setdefault(q, {})[a] = pets

```

```python
# 1) Normalize question text (strip whitespace) so duplicates unify
df["Question"] = df["Question"].astype(str).str.strip()

# 2) Detect if a Parameter Suggestions column exists
param_col = next(
    (c for c in df.columns if c.lower().startswith("parameter suggestion")),
    None
)

# 3) Build fast lookup { question_text -> { answer_option -> {techs, params} } }
lookup: dict[str, dict[str, dict]] = {}
for _, row in df.iterrows():
    q = row["Question"]
    a = str(row["Answer Option"]).strip()

    # Recommended Techniques → list of strings (safe split)
    raw_techs = row.get("Recommended Techniques", "")
    raw_techs = "" if pd.isna(raw_techs) else str(raw_techs)
    techs = [t.strip() for t in raw_techs.split(";") if t.strip()]

    # Parameter Suggestions → string or empty
    params = ""
    if param_col:
        raw_params = row.get(param_col, "")
        params = "" if pd.isna(raw_params) else str(raw_params)

    lookup.setdefault(q, {})[a] = {
        "techs": techs,
        "params": params
    }
```

Code for counting the score(weight)

```
# ------------------------------------------------------------------------------
# Score screening answers
# ------------------------------------------------------------------------------
def evaluate(answers: dict):
    votes = {}
    params_out = []
    vetoed = set()

    # 1) Tally votes & collect params as before
    for q_text, sel in answers.items():
        sels = sel if isinstance(sel, list) else [sel]
        for ans in sels:
            entry = lookup.get(q_text, {}).get(ans)
            if not entry:
                continue
            for tech in entry["techs"]:
                votes[tech] = votes.get(tech, 0) + 1
            if entry["params"]:
                params_out.append(entry["params"])

    # 2) Apply any deal-breakers
    #    Any PET listed under deal_map[q_text][ans] is vetoed
    for q_text, sel in answers.items():
        if not deal_col:
            break
        sels = sel if isinstance(sel, list) else [sel]
        for ans in sels:
            pets_to_veto = deal_map.get(q_text, {}).get(ans, [])
            for pet in pets_to_veto:
                vetoed.add(pet)

    # 3) Filter out vetoed PETs entirely
    for pet in vetoed:
        votes.pop(pet, None)

    # 4) Build the final ranked list
    ranked = sorted(votes.items(), key=lambda kv: kv[1], reverse=True)
    ranked_pets = [
        {"name": tech, "score": cnt, "rationale": ("VETOED" if tech in vetoed else "Matches survey")}
        for tech, cnt in ranked
    ]

    # 5) Deduplicate parameter suggestions
    param_suggestions = sorted(set(params_out))
```

Show results: Privacy technologies; Laws

```
@app.get("/results")
def show_results():
    ranked = session.get("ranked", [])
    params = session.get("params", [])
    vetoed = session.get("vetoed", [])
    all_tools = request.args.get("tools", "").split(",")
    top3_tools = all_tools
    session["wizard_tools"] = top3_tools[:3]

    # Split ranked into privacy techniques vs policy recommendations
    recommended_privacy_techniques = []
    recommended_policies = []

    for item in ranked:
        name = item.get("name", "").lower()
        if any(keyword in name for keyword in ["compliance", "policy", "regulation", "ferpa", "hipaa
            recommended_policies.append(item)
        else:
            recommended_privacy_techniques.append(item)

    # Only take top 2 from each
    top_privacy_techniques = recommended_privacy_techniques[:2]
    top_policies = recommended_policies

    return render_template(
        "results.html",
        privacy_techniques=top_privacy_techniques,
        policies=top_policies,
        parameters=params,
        vetoed=vetoed,
        wizard_tools=top3_tools
    )
```

Code for follow up configure tool :

```
# ------------------------------------------------------------------------------
# Implementation-wizard sub-questions
# ------------------------------------------------------------------------------
def dp_steps():
    return [
        {
            "id": "D1",
            "text": "1) Maximum absolute error you can tolerate (Δ=1):",
            "input_type": "number",
            "placeholder": "e.g. 2.0"
        },
        {
            "id": "D2",
            "text": "2) Expected number of queries per day:",
            "input_type": "number",
            "placeholder": "e.g. 50"
```

```
def ka_steps():
    return [
        {
            "id": "K1",
            "text": "1) Approximately how many unique records does your dataset contain?",
            "options": ["<10k", "10k–100k", "100k–1M", ">1M"]
        },
        {
            "id": "K2",
            "text": "2) What maximum re-identification risk do you accept?",
            "options": ["Very low (<1%)", "Low (1–5%)", "Moderate (5–10%)"]
        }
    ]
```

We hard coded the follow up tool suggestion with legal compliance.

For example:

DP calculation:

```python
def wizard_submit():
    data = request.get_json(force=True)
    tool = data.pop("tool", None) or ""
    session["last_tool"] = tool
    config = []

    if tool == "Differential Privacy":
        # DP logic
        try:
            err = float(data.get("D1", 0))
            qpd = int(data.get("D2", 0))
            eps_q = 1.0/err if err>0 else 0.0
            eps_tot = eps_q * qpd
            config.append(f"ε per query ≈ {eps_q:.3f}")
            config.append(f"Total ε/day ≈ {eps_tot:.3f}")
        except Exception:
            config.append("⛔ Invalid DP inputs—could not compute ε.")
```

K&L logic by if and else

```python
elif tool == "k-anonymity & ℓ-diversity":
    # k-anonymity logic
    config.append("Your anonymization settings:")
    k1 = data.get("K1", "")
    k2 = data.get("K2", "")
    config.append(f"  Dataset size: {k1}")
    config.append(f"  Risk tolerance: {k2}")

    # map to k-value
    size_map = {
        "<10k":      {"Very low (<1%)": 5,   "Low (1–5%)": 10,  "Moderate (5–10%)": 20},
        "10k–100k":  {"Very low (<1%)": 10,  "Low (1–5%)": 20,  "Moderate (5–10%)": 50},
        "100k–1M":   {"Very low (<1%)": 20,  "Low (1–5%)": 50,  "Moderate (5–10%)": 100},
        ">1M":       {"Very low (<1%)": 50,  "Low (1–5%)": 100, "Moderate (5–10%)": 200}
    }
    k_val = size_map.get(k1, {}).get(k2)
    l_map = {"Very low (<1%)": 2, "Low (1–5%)": 3, "Moderate (5–10%)": 5}
    l_val = l_map.get(k2, 2)

    config.append("")
    if k_val:
        config.append(f"• Generalize/suppress to achieve k={k_val} and ℓ={l_val}.")
        config.append("  – k (anonymity): each record is indistinguishable from at least k–1 others sha
        config.append("  – ℓ (diversity): each such group must contain at least ℓ distinct sensitive-at
        config.append("  Use a library like ARX (Java) or sdcMicro (R/Python).")
    else:
        config.append("• Unable to derive k/ℓ for those choices—please adjust settings.")


else:
```

Thank you!