Introducing Kura

Making sense of Data at scale

Learn More →

Yesterday, 1,000 Users Couldn't Find Their Contracts

Today, another 1,000 will fail the same way

Your logs show **"success"** but users are **quietly leaving**

LLM Apps Fail Silently

Performance Drift

Data Paralysis

Missing the Forest

Performance Drift

Silent degradation kills apps

- Model updates change behavior
- Prompt tweaks shift outputs
- Edge cases accumulate over time

Data Paralysis

Too much data, no direction

- Terabytes of logs to analyze
- User priorities buried deep
- Generic insights don't help

Missing the Forest

Hyper-focus on specifics blinds us

- Build XX for YY specific use case
- Miss broader capability patterns
- Reactive, not strategic

Every User Complaint Falls Into Two Buckets

Understanding this pattern changes everything

When Users Complain, They’re Really Saying

"I can't send emails through the AI" → Missing capability

"Why can't it find my signed contracts?" → Missing inventory

"Why doesn't it know when this was modified?" → Missing inventory

Every user request falls into one of two buckets

The Two Types of AI Failures

🔧 "I Can’t Do That"
Capability Gaps

- Send emails, book meetings
- Access external systems
- Execute workflows

📦 "I Can’t Find That"
Inventory Gaps

- Documents not indexed
- Metadata incomplete
- Search context missing

Traditional Methods Have Limitations

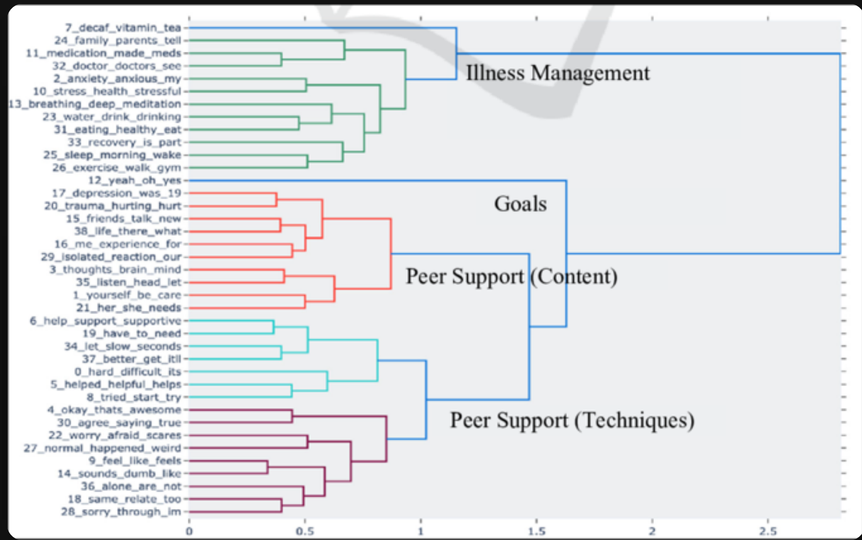
BERTopic is excellent for many use cases

But LLM conversations need a different approach

BERTopic: Great Tool, Different Goals

Why single-level clustering isn't enough

- Single granularity misses broader patterns
- Can't navigate from high-level to specific needs
- Explanation comes after, not during clustering



Can we do better?

Real-World Case Study

How Anthropic built Claude Education from user data

From 1M conversations to product innovation

LLM-Based Conversation Analysis

Step 1: CLIO’s Summarization → Clustering Process

- LLM summarizes each conversation
- Clusters similar business processes
- Reveals cognitive skill patterns

Raw → Summary → Cluster

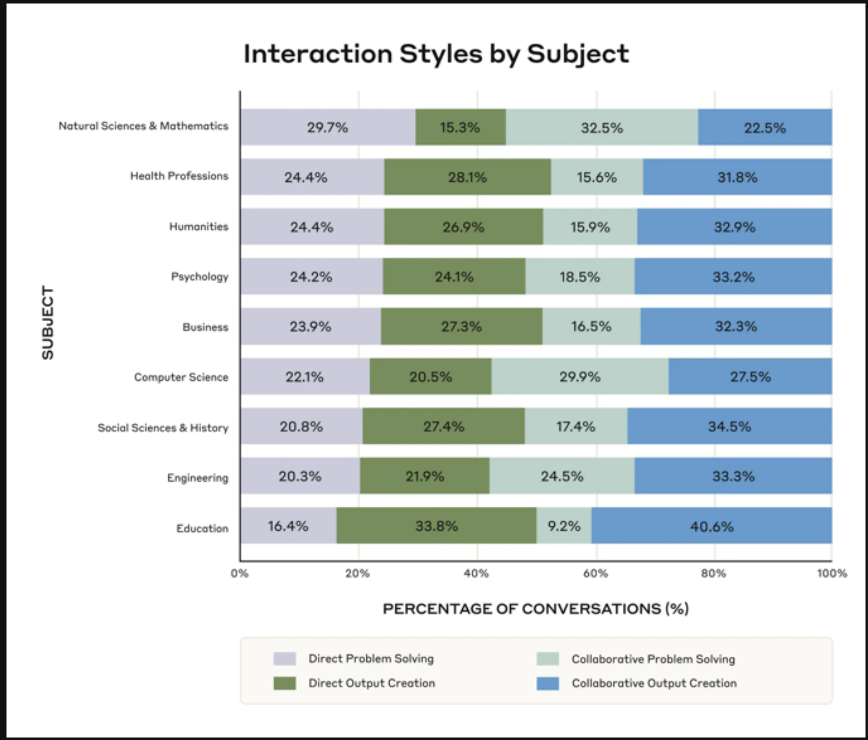
Business process discovery

1M conversations → actionable insights

Business Process Clusters Emerge

Step 2: From Summaries to Process Categories

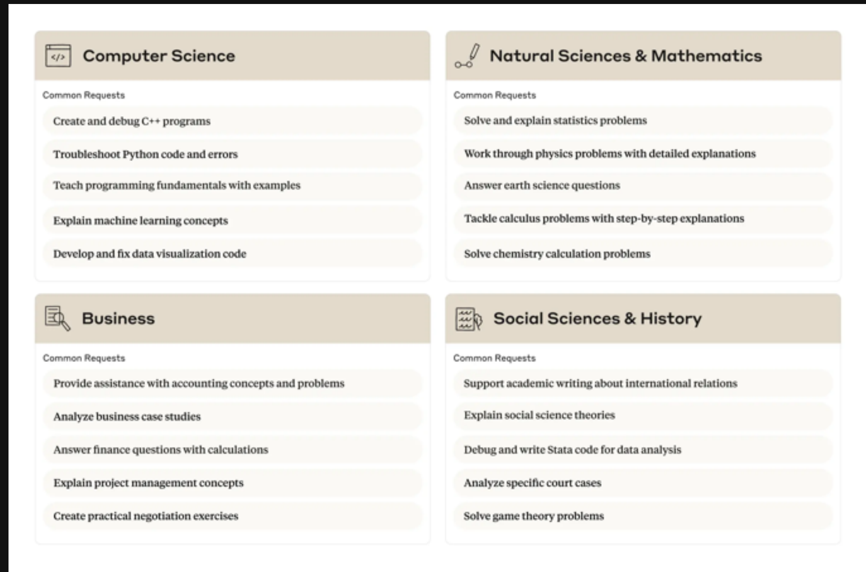
- Problem Solving processes (46-58%)
- Content Creation processes (42-54%)
- Collaboration style varies by process



What Students Actually Request

Step 3: Specific Use Cases by Field

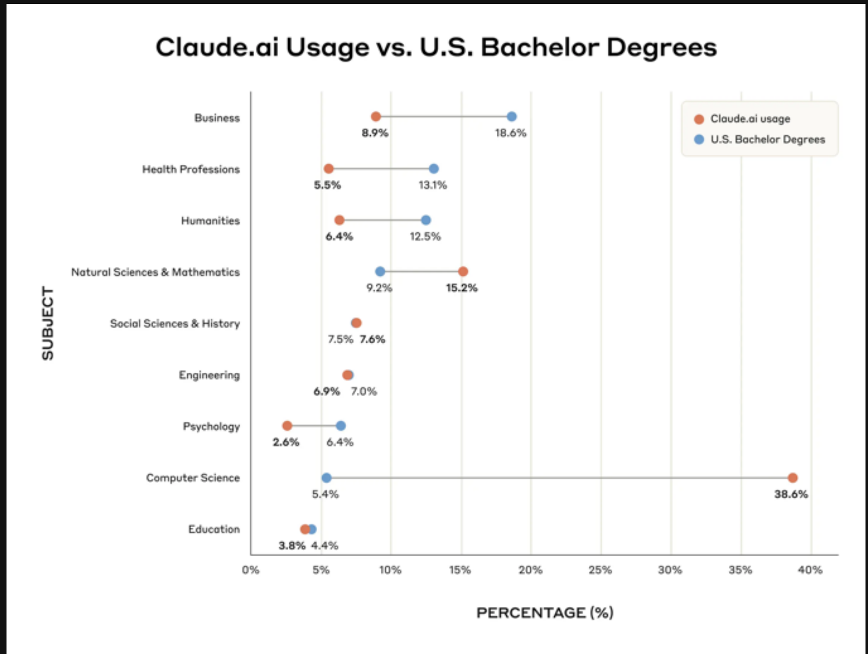
- CS: Debug code, implement algorithms
- Math: Step-by-step problem solving
- Business: Create presentations, case analysis



STEM Early Adoption Pattern

Step 4: Usage vs Enrollment Data

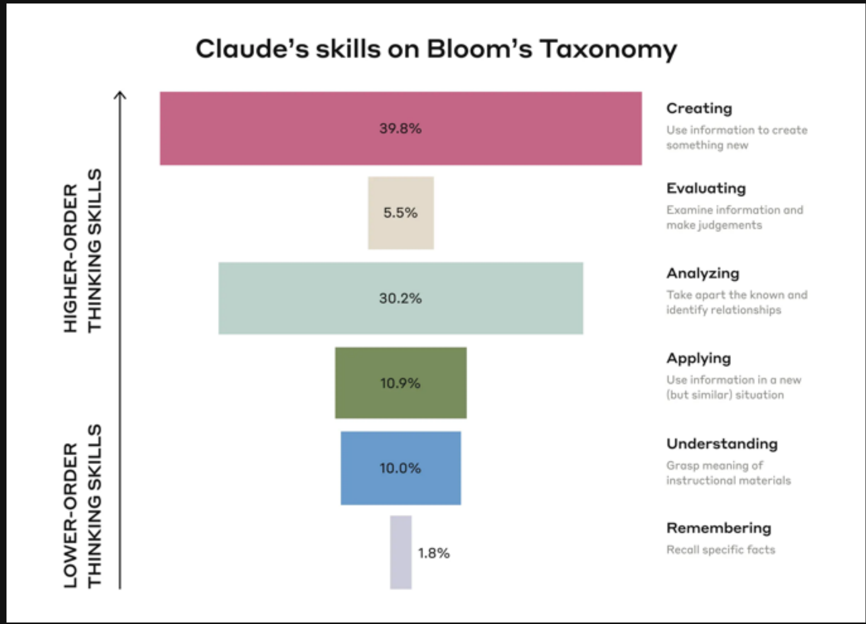
- CS: 38.6% usage vs 5.4% of degrees
- Natural Sciences: 15.2% vs 9.2%
- Business: 8.9% vs 18.6% (underrepresented)



The Critical Discovery

Step 5: Clustering Reveals Cognitive Skills

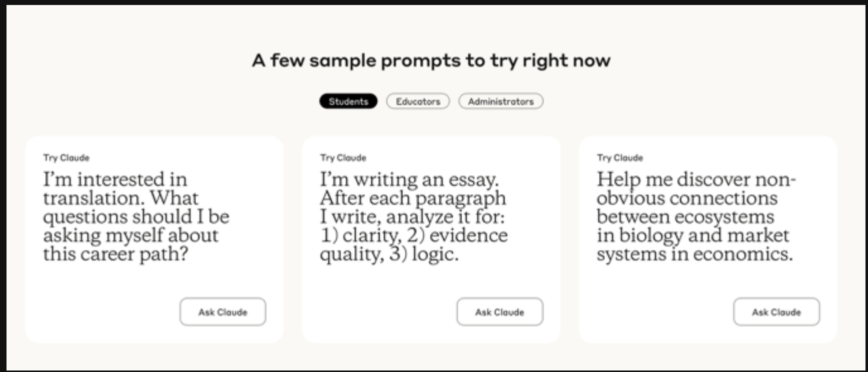
- Creating: 39.8% (highest cognitive skill)
- Analyzing: 30.2% (pattern recognition)
- Inverted pyramid → learning concern



Claude Education Launch

Step 6: Product Targets Discovered Use Cases

- Prompts target collaborative problem solving
- Messaging encourages guided analysis
- Product design supports learning goals



How Anthropic Solved This Problem

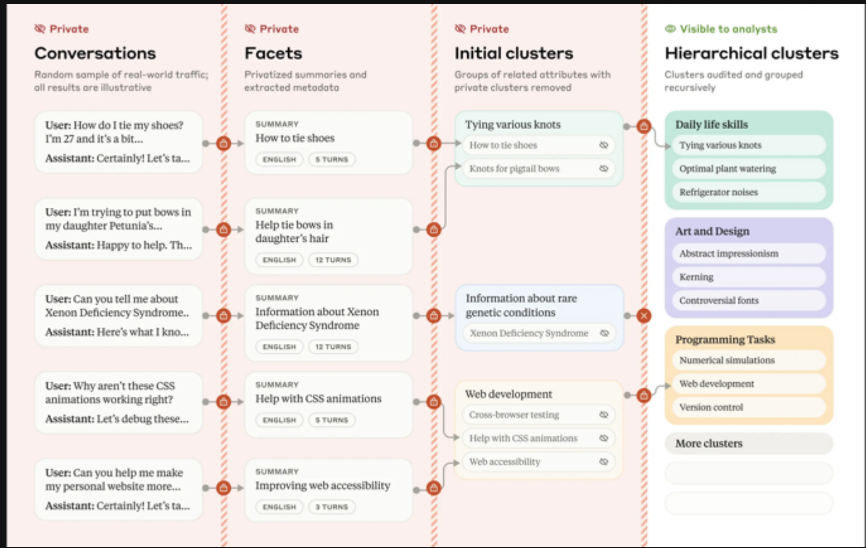
Clio: The inspiration behind Kura

Production-scale conversation analysis that led to Claude Education

Privacy-First Design

Multi-layered approach to user protection

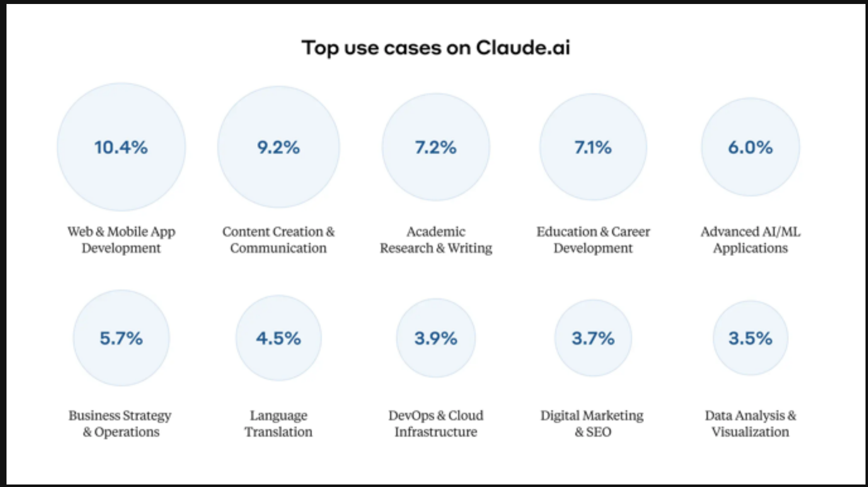
- Conversation summaries strip private data
- Minimum cluster size thresholds
- Automated privacy auditing



AI-Powered Pipeline

Models analyzing models at scale

- Extract conversation facets automatically
- Semantic clustering via embeddings
- Generate hierarchical insights



Real-World Impact

Concrete safety improvements delivered

- Detected coordinated SEO spam networks
- Monitored election period risks
- Improved safety classifier accuracy

Scalable Architecture

Cost-effective analysis at massive scale

- $0.0005 per conversation processed
- Interactive 2D visualization interface
- Hierarchical exploration from broad to specific

⚡

Production Ready

Millions of conversations daily

Clio Works, But It’s Not Available

That’s why we built Kura

Open source implementation of the same core principles

Your AI Has Thousands of Daily Conversations

But do you know what users actually need?

Kura reveals hidden patterns in chat data through a 5-step pipeline

The Kura Pipeline

5 Steps from Chaos to Clarity

- Load & summarize conversations
- Cluster by semantic similarity
- Visualize in 2D space

```python
# Complete Pipeline Overview
async def main():
    # 1. Load conversations
    conversations = Conversation.from_hf_dataset(
        "ivanleomk/synthetic-gemini-conversations"
    )

    # 2. Summarize conversations
    summaries = await summarise_conversations(conversat

    # 3. Generate base clusters
    clusters = await generate_base_clusters_from_conver

    # 4. Create meta clusters
    reduced_clusters = await reduce_clusters_from_base_

    # 5. Project to 2D and visualize
    projected = await reduce_dimensionality_from_cluste
    visualise_pipeline_results(projected)
```

Step 1: Load & Summarize

Transform Raw Conversations

- Load from Hugging Face datasets
- Custom prompts for summarization
- Disk caching for efficiency

```
# Setup models with caching
summary_model = SummaryModel(
    console=console,
    cache=DiskCacheStrategy(cache_dir="./.summary")
)

checkpoint_manager = JSONLCheckpointManager(
    "./checkpoints", enabled=True
)

# Load conversations
conversations = Conversation.from_hf_dataset(
    "ivanleomk/synthetic-gemini-conversations",
    split="train"
)

# Generate summaries with custom prompts
summaries = await summarise_conversations(
    conversations,
    model=summary_model,
    checkpoint_manager=checkpoint_manager
)
```

Step 2: Base Clustering

Group by Semantic Similarity

- Embeddings capture meaning
- K-means clustering by default
- Auto-generated cluster descriptions

```
# Setup clustering model
cluster_model = ClusterDescriptionModel(
    console=console  # Uses K-means by default
)

# Generate base clusters with titles and descriptions
clusters = await generate_base_clusters_from_conversati
    summaries,
    model=cluster_model,
    checkpoint_manager=checkpoint_manager
)

# Each cluster gets:
# - Semantic grouping based on embeddings
# - Descriptive title (e.g., "API Integration Issues")
# - Detailed description of common patterns
# - List of conversation summaries in cluster
```

Step 3: Meta Clustering

Cluster the Clusters

- Reduce cluster count hierarchically
- Find higher-level patterns
- Maintain meaningful granularity

```
# Setup meta clustering
meta_cluster_model = MetaClusterModel(console=console)

# Reduce base clusters into meta clusters
reduced_clusters = await reduce_clusters_from_base_clus
    clusters,
    model=meta_cluster_model,
    checkpoint_manager=checkpoint_manager
)

# Example transformation:
# Base clusters (20):
# - "React Component Errors", "Vue.js Issues", "Angular
# - "API Rate Limiting", "Authentication Failures", "CC
#
# Meta clusters (5):
# - "Frontend Framework Issues"
# - "API Integration Problems"
```

Step 4: Dimensionality Reduction

Project to 2D Space

- HDBSCAN + UMAP algorithms
- Preserve cluster relationships
- Enable interactive visualization

```
# Setup dimensionality reduction
dimensionality_model = HDBUMAP()

# Project clusters into 2D space
projected_clusters = await reduce_dimensionality_from_c
    reduced_clusters,
    model=dimensionality_model,
    checkpoint_manager=checkpoint_manager,
)

# High-dimensional embeddings → 2D coordinates
# Similar clusters appear close together
# Cluster density reflects conversation volume
# Interactive exploration of patterns
```

Step 5: Visualization

Interactive Pattern Discovery

- 2D scatter plot of clusters
- Click to explore conversations
- Identify patterns and gaps

```
# Generate final visualization
visualise_pipeline_results(
    projected_clusters,
    style="basic"
)

# Creates interactive plot showing:
# - Each cluster as a point in 2D space
# - Cluster size proportional to conversation count
# - Hover for cluster descriptions
# - Click to drill down into conversations
# - Identify user pain points and opportunities
```

Expected Output

Hierarchical Conversation Insights

- Clear hierarchical structure
- 10x performance with caching
- Actionable conversation patterns

```
Programming Assistance (190 conversations)
├── Data Analysis (38 conversations)
│   ├── "R plots for statistics"
│   ├── "Tableau performance"
│   └── "Excel to pandas"
├── Web Development (45 conversations)
│   ├── "React re-rendering"
│   ├── "Stripe API integration"
│   └── "CSS grid responsive"
└── ... (more clusters)

Performance: 21.9s → 2.1s (10x faster!)
```

Enhanced Pattern Discovery

Clusters + Feedback + Metadata = Actionable Insights

- Chat clusters reveal conversation themes
- User feedback shows satisfaction levels
- Metadata enables smart prioritization

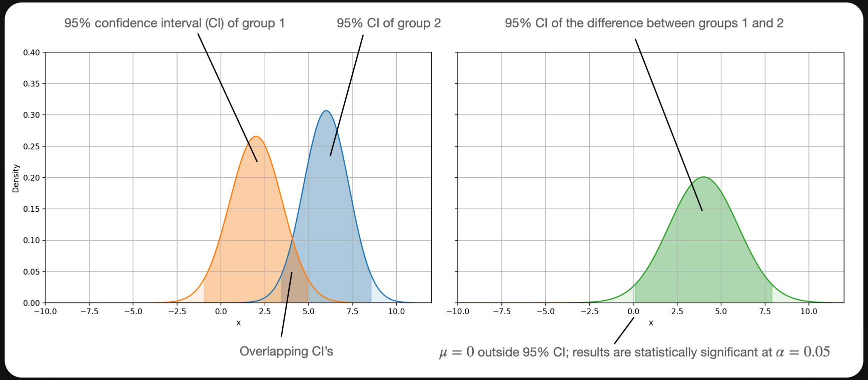
```sql
-- Cluster insights with user feedback
SELECT
    cluster_id,
    cluster_name,
    COUNT(*) as conversations,
    AVG(thumbs_up) as avg_satisfaction,
    SUM(revenue_impact) as total_revenue_impact
FROM chat_clusters c
JOIN user_feedback f ON c.conversation_id = f.conversat
JOIN user_metadata u ON c.user_id = u.user_id
GROUP BY cluster_id, cluster_name
ORDER BY total_revenue_impact DESC;

-- Results:
-- API Issues     | 156 | 0.27 | $2.3M
-- Bulk Export    | 89  | 0.85 | $500K
-- UI Feedback    | 203 | 0.64 | $200K
```

Why Explicit Classifiers?

Topic modeling is inherently lossy

- Stochastic algorithms produce different results
- Production needs consistent categorization
- Explicit classifiers provide reliability



Bootstrapping Training Data

Clusters become labeled examples

- Clustered conversations are pre-labeled
- LLMs use few-shot examples for classification
- Efficiently identify data for labeling

Discovery → Production

Topic modeling for exploration
Classifiers for monitoring

Getting Started with Kura

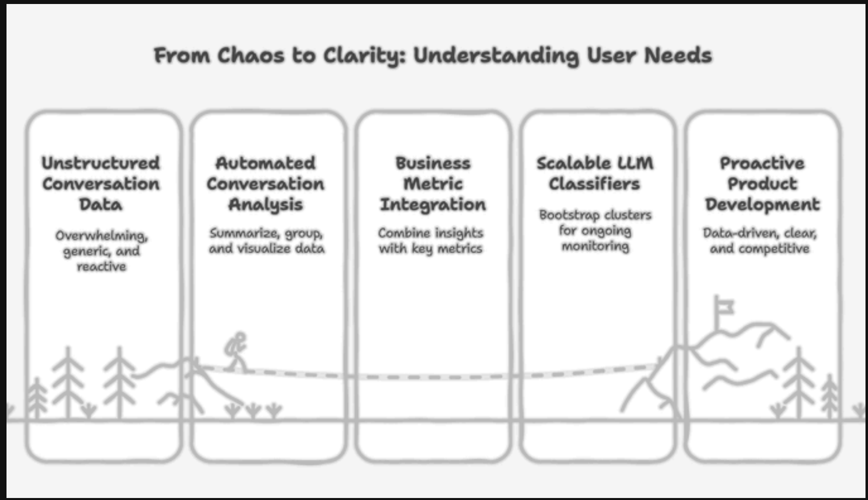
From Raw Data to Production Insights

A practical workflow for analyzing your LLM conversations

The Kura Workflow

Simple process from data to insights

- Run the 5-step pipeline on your chat data
- Combine clusters with business metrics
- Deploy classifiers for ongoing monitoring



Stop Flying Blind

Your users are already telling you what they need

Kura helps you listen at scale

Try Kura Today

```
uv pip install kura
```

📚 Documentation: **usekura.xyz**