DOCUMENTATION

Pipecat Cloud – Conversation Source:

Pipecat Cloud is where all voice conversations happen. Voice agents run on Pipecat Cloud, and customers talk to these agents in real time.

Each conversation produces raw logs that include:

- User questions
- Agent responses
- Start and end timestamps
- Language information
- System and debug-level noise

Why We Don't Store Raw Logs

Raw Pipecat logs are:

- Very large in size
- Difficult to understand or analyze directly
- Not suitable for dashboards or insights
- Expensive to store at scale

Pipecat Cloud remains completely unchanged. We simply consume its output, clean and structure the data, and store only what is needed for analytics and dashboards.

Stateless Ingestion & Filtering (Smart Gatekeeper Layer):

This layer receives raw conversation logs from Pipecat Cloud and keeps only what matters for the business, such as:

- User questions
- Bot responses
- Session start & end time
- Call duration

All unnecessary data—debug logs, system noise, and heartbeats—is discarded.

Why This Is Scalable

This service is stateless, so:

- Multiple instances can run in parallel
- Traffic spikes are handled by simply adding more instances
- Databases stay lean with no unnecessary data load

Like airport security—only relevant passengers are allowed through, not everyone.

Message Queue / Buffer (Scalability Safety Net):

This layer temporarily stores filtered events and smooths traffic spikes, ensuring data is delivered downstream at a controlled pace.

Why This Is Critical

- Protects PostgreSQL from sudden load spikes
- Prevents data loss
- Allows ingestion and storage to scale independently
- Keeps the system reliable during peak traffic

Like a traffic signal—it prevents jams during rush hour. Even if traffic grows suddenly, nothing breaks and nothing is lost.

Structured Session Records (Clean, Minimal Output):

After filtering, thousands of raw logs are condensed into one clean record per conversation session, containing:

- Session ID
- Agent ID
- User question
- Bot response
- Start & end time
- Duration
- Status
- Language

Why This Is Powerful

- Very small data size
- Easy to store and query
- Fast analytics
- Scales effortlessly

Less data. More value.

PostgreSQL (Analytics-Ready Enterprise Database):

What is stored here

- Clean session records only
- Question–answer pairs
- Session duration and metadata

Why PostgreSQL

- Enterprise-grade reliability
- Fast analytics queries
- Strong data consistency
- Works well with reporting tools

How It Scales

- Data is partitioned by date
- New data never slows old data
- Indexes keep queries fast
- Read replicas can be added as usage grows

Because we store only clean and minimal data, the database stays fast, efficient, and scalable—even at very large volumes.

Backend APIs (Controlled & Optimized Access Layer):

These APIs provide secure and efficient access to the data. They:

- Fetch analytics summaries
- Fetch detailed session records
- Apply filters like date range, agent, and language
- Enforce admin-only access

Why This Scales Well

- APIs are stateless and horizontally scalable
- Frequently requested data is cached
- Database is protected from heavy or repeated queries

Commonly requested answers are kept ready instead of being recalculated every time.

React Admin Dashboard (What Admins See):

The dashboard gives admins clear, actionable insights at a glance:

- Agent-wise usage
- Total conversation sessions
- Average call duration
- Success vs failure rates

Admins can click any session to instantly view:

- User message
- Bot response
- Session duration

Why the UI remains fast

- Uses pre-processed, clean data
- No heavy computation on the frontend
- Lazy loading and pagination for large datasets
- No raw or noisy logs exposed

Clear insights. No noise. Instant visibility.