About AutoAce

AutoAce builds **AI agents for car dealerships** to handle inbound/after-hours/overflow calls—answer questions, capture customer intent, and schedule service appointments. We integrate into dealership workflows and systems, and we care deeply about **reliability, call quality, and operational visibility**.

In production we use **Retell**, which is a **telephony + AI engine** (not just telephony). Retell emits webhook events about AI-handled calls (transcripts, outcomes, metadata, etc.).

This project is **internal tooling** to help our team monitor and improve call quality per dealership.

Goal

Build an internal system that:

1. Ingests **Retell-style AI call events** via a webhook
2. Uses an LLM to assess whether calls are going well or not
3. Presents results in a **per-dealership monitoring dashboard**
4. Supports **multi-role login** for internal users (CSMs + Managers)

Reference (Important)

Use Retell's webhook docs to understand real event semantics and shapes:

🔗 https://docs.retellai.com/features/webhook-overview#webhook-overview

Clarifications

- You do **not** need Retell credentials or live telephony
- Your system should accept a **normalized event payload inspired by Retell**
- We are evaluating system design, robustness, and clarity

Core Requirements

1) Webhook Ingestion (Required)

You must provide a **publicly reachable** webhook endpoint that we can POST into.

Endpoint

`POST /webhooks/retell`

Hard requirements

- **Authentication required** (choose one):
    - `Authorization: Bearer <shared_secret>` (OK)
    - HMAC signature header (better)
- Must return **HTTP 200 in < 1 second** (ACK only)
- Must support **idempotency** using `event_id` (ignore duplicates)
- Must persist raw payload + normalized fields
- Any expensive work must happen **asynchronously** (not in the request)

You must provide

- Webhook URL
- Auth method + secret
- A working `curl` command we can use to test ingestion

---

2) Input Event Contract (Minimum)

Your webhook must accept at least:

```
{
  "event_id": "evt_123",
  "dealership_id": "toyota-braintree",
  "call_id": "call_456",
  "timestamp": "2025-12-17T14:32:00Z",
  "transcript": "...",
  "customer": {
    "phone": "+15551234567",
    "name": "Optional"
  },
  "metadata": {
    "duration_sec": 420,
    "channel": "phone",
    "ai_engine": "retell"
  }
}
```

You may extend this schema as needed, but keep it clean and documented.

---

3) Backend System & Data Model (Required)

You own backend architecture and persistence.

Must include

- Postgres-backed storage (Supabase acceptable)
- Multi-tenant isolation by `dealership_id`
- Clear separation between:
    - Ingested events
    - LLM analysis results
- Defensive error handling (LLM failure ≠ ingestion failure)

Suggested tables

- `events`
- `call_analyses`
- `processing_errors` (optional)
- `users`, `roles`, `user_dealerships` (for access control)
- `call_reviews` or `resolutions` (CSM workflow)

---

4) LLM-Assisted Call Analysis (Required)

You will be provided a **trial-only OpenAI API key**.

Key handling rule

- **Do not expose the key to the frontend.** (backend only)

Objective
Classify whether the AI-handled call:

- Went well
- Needs human review
- Created customer or business risk

Required Output Format (JSON only)

```
{
  "verdict": "good | needs_review | bad",
  "score": 0-100,
  "reasons": ["short, concrete explanations"],
  "flags": [
    "missed_appointment_confirmation",
    "customer_upset",
    "handoff_failed",
    "hallucinated_information"
```

```
  ],
  "recommended_action": "none | human_follow_up",
  "confidence": 0.0-1.0
}
```

Failure behavior
If LLM analysis fails or times out:

- Store the event
- Set `verdict = needs_review`
- Record reason `analysis_failed`

---

5) Internal Dashboard (Required)

Build a simple internal monitoring dashboard.

Must include

- Dealership selector (or scoped access by dealership)
- Call list (last 24h / 7d)
- Verdict, score, top reasons
- Filters by verdict / score / flags
- Drill-down view:
    - Transcript
    - Stored JSON analysis
    - Timestamps + metadata

---

6) Authentication + Multi-Role Access Control (Required)

The dashboard must support **login** and **role-based access**.

Roles

Minimum roles:

1. **CSM**
    - Can view calls for assigned dealerships
    - Can mark calls as “reviewed”
    - Can record a resolution status + notes (e.g., “follow-up needed”, “resolved”, “escalated”)
2. **Manager / Admin**
    - Can view across dealerships (or across all assigned CSMs)
    - Can see what CSMs are resolving or not resolving
    - Can filter by: unresolved items, by CSM, by dealership, by time

Required UI views for roles

- **CSM view:** “My queue” (calls flagged `needs_review`/`bad`) + resolution workflow
- **Manager view:** “Team overview” (CSM throughput + unresolved backlog)

Chat Consideration (Bonus — Not Required)

We want the system designed with **future chat capability** in mind.

Expectation (bonus)

- Add a “Chat” section in the call detail view with a disabled button or placeholder:

    - “Send message to customer (future)”

- Add brief notes in your architecture doc on how you’d implement:
    - If a call is flagged as `bad` / `needs_review`, a CSM could send a chat/SMS to the customer
    - What provider you’d use (e.g., Twilio Conversations/SMS) and what data you’d store
    - Safety/consent considerations and audit logs

You do not need to implement chat. If you do, it’s extra credit—keep it minimal and safe.

Explicitly Out of Scope (Do NOT Build)

- Live telephony integration
- Retell configuration / phone numbers
- Real-time streaming updates
- Alerting/paging
- Automatic actions (no booking changes; no autonomous outbound messaging)

Timeline

1 week total.
We will test by POSTing sample payloads into your webhook and validating the dashboard, roles, and workflow.

Evaluation Criteria

You pass if:

- We can POST into your webhook reliably (auth + idempotency)
- Calls are stored and visible per dealership
- LLM analysis runs async and produces structured outputs
- Dashboard answers: **“Are RetelI AI calls going well for this dealership?”**
- Role-based access works (CSM queue + Manager oversight)
- Resolution tracking is clear and usable

Red flags:

- No webhook auth
- No multi-tenant isolation by dealership
- LLM blocks ingestion
- No role separation / no workflow for resolutions

Final Handoff

At the end, provide:

- Webhook URL + auth details + working `curl`
- Dashboard URL (or local run instructions)
- Repo link + setup steps
- Short architecture/tradeoffs note (1–2 pages max), including chat approach