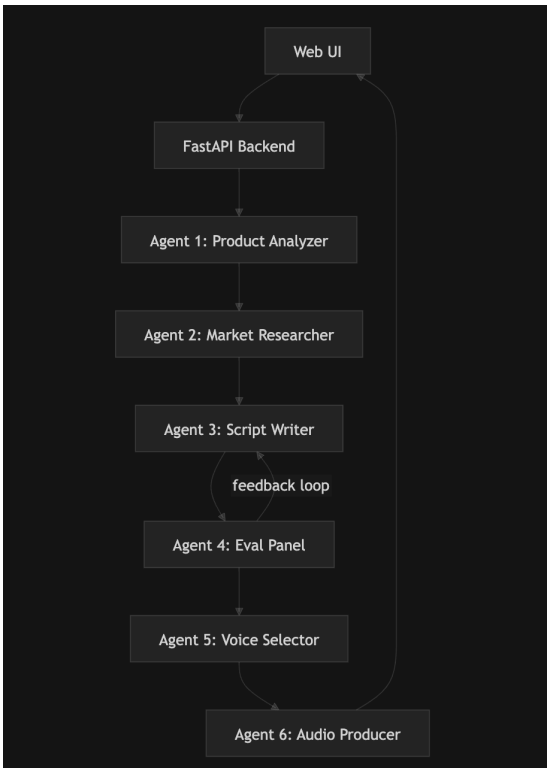
OBD SuperStar Agent

Architecture Overview

A Next.js (React) frontend with a Python FastAPI backend powering a multi-agent pipeline. The system takes three inputs -- **product documentation**, **country/telco**, and **target audience** -- and outputs fully produced OBD audio files ready for deployment.



Agent Pipeline (6 Agents)

Agent 1: Product Analyzer

- **Input**: Product documentation (PDF/PPT/text), pricing info
- **Role**: Reads and fully understands the product -- features, pricing, USPs, target use cases
- **Output**: Structured product brief (JSON) with key selling points, pricing tiers, and value propositions

Agent 2: Market Researcher

- **Input**: Country name, Telco name, product brief from Agent 1

- **Role**: Acts as a local market analyst. Researches the country's demographics, mobile usage patterns, cultural nuances, current affairs, local trends, feature phone vs smartphone split, language preferences
- **Output**: Market analysis report with target audience psyche, cultural hooks, current affairs angles, and language recommendations

Agent 3: Script Writer (Hook + Body + CTA)

- **Input**: Product brief + Market analysis
- **Role**: Acts as the best copywriter in that country who understands local culture, nuances, and psychology. Creates scripts under 30 seconds with:
    - **Hook** (first 5 seconds): Must prevent call disconnect. Culturally relevant, attention-grabbing
    - **Body**: Product pitch delivered compellingly so recipients want to subscribe
    - **CTA**: DTMF-based call to action
    - **Fallback 1**: If no DTMF pressed -- urgency-based follow-up CTA
    - **Fallback 2**: Third attempt using psychological techniques (scarcity, loss aversion, social proof)
    - **Polite closure**: If still no response
- **Output**: 5 complete script sets, each with main flow + 2 fallback variants. Scripts include ElevenLabs V3 audio tags (`[excited]`, `[whispers]`, `[pause]`, etc.) baked in
- **Key constraint**: Each script must be under 30 seconds when spoken

Agent 4: Evaluation Panel

- **Input**: All 5 script sets from Agent 3
- **Role**: A panel of 10 AI evaluators, each with a distinct persona:
    1. Award-winning radio jingle writer
    2. Consumer psychologist
    3. Behavioral economist
    4. Top copywriter #1 (direct response specialist)
    5. Top copywriter #2 (brand storytelling)
    6. Top copywriter #3 (local market expert)
    7. Dark psychology / persuasion expert
    8. Telco marketing director
    9. UX/voice interface designer
    10. Local cultural consultant
- **Process**: Each evaluator scores and critiques. Consensus feedback is sent back to Agent 3 for revision. One iteration loop.
- **Output**: Final approved script sets with scores and rationale

Agent 5: Voice Selector

- **Input**: Final scripts + country/language info
- **Role**: Queries ElevenLabs API (`GET /v2/voices`) to find voices matching the target country/language. Evaluates voice characteristics (gender, age, tone, accent) against the

script requirements. Recommends optimal voice with API parameters (stability, similarity_boost, style, speed).
- **Output**: Voice ID, voice settings JSON, and rationale for selection

Agent 6: Audio Producer

- **Input**: Final scripts (with V3 audio tags) + Voice selection
- **Role**: Calls ElevenLabs TTS API (`POST /v1/text-to-speech/:voice_id`) using the `eleven_v3` model. Generates audio for all script variants. Uses V3 audio tags for expressiveness: `[excited]`, `[whispers]`, `[pause]`, `[laughs]`, `[sigh]`, `[curious]`, capitalization for emphasis, ellipses for pacing.
- **Output**: MP3 audio files for each script variant, downloadable from the UI

Tech Stack

Frontend: Next.js + Tailwind CSS + shadcn/ui

- `frontend/` directory
- Modern, clean UI with a step-by-step wizard
- Real-time progress tracking as agents execute
- Audio player for previewing generated recordings
- Download functionality for final audio files

Backend: Python FastAPI

- `backend/` directory
- Agent orchestration engine
- LLM integration supporting both **Claude** (Anthropic) and **OpenAI GPT-4** (configurable)
- ElevenLabs API integration
- WebSocket for real-time progress updates to frontend

Key Dependencies

- **Python**: `fastapi`, `uvicorn`, `anthropic`, `openai`, `elevenlabs`, `python-multipart`, `websockets`, `pydantic`
- **Node**: `next`, `react`, `tailwindcss`, `shadcn/ui`, `lucide-react`

Project Structure

```
OBD_SuperStarAgent/
  backend/
    main.py                 # FastAPI app, routes, WebSocket
    agents/
      __init__.py
      base.py               # Base agent class with LLM abstraction
```

```
    product_analyzer.py    # Agent 1
    market_researcher.py   # Agent 2
    script_writer.py       # Agent 3
    eval_panel.py          # Agent 4
    voice_selector.py      # Agent 5
    audio_producer.py      # Agent 6
  orchestrator.py          # Pipeline orchestrator
  config.py                # API keys, model config
  requirements.txt
frontend/
  app/
    page.tsx               # Main wizard UI
    layout.tsx
    globals.css
  components/
    ProductUpload.tsx       # Step 1: Upload product docs
    CountryTelcoSelect.tsx  # Step 2: Select country/telco
    PipelineProgress.tsx    # Step 3: Real-time agent progress
    ScriptReview.tsx        # Step 4: Review generated scripts
    AudioPlayer.tsx         # Step 5: Listen and download audio
  package.json
  tailwind.config.ts
.env                       # API keys (ANTHROPIC_API_KEY, OPENAI_API_KEY,
ELEVENLABS_API_KEY)
README.md
```

Key Design Decisions

- **LLM Abstraction**: Base agent class supports both Claude and OpenAI. Each agent call specifies which provider to use via config. Default: Claude for creative tasks (script writing, evaluation), OpenAI as fallback.
- **Feedback Loop**: Agent 4 (Eval Panel) sends structured feedback to Agent 3 (Script Writer) for exactly one revision cycle to balance quality vs. speed.
- **V3 Audio Tags**: Agent 3 outputs scripts pre-formatted with ElevenLabs V3 audio tags (`[excited]`, `[whispers]`, `[pause]`, etc.) so Agent 6 can pass them directly to the API.
- **30-Second Constraint**: Agent 3 estimates word count (~150 words = 60 seconds, ~75 words = 30 seconds) and enforces the limit. Agent 6 validates actual audio duration.
- **Streaming Progress**: WebSocket connection from frontend to backend shows real-time agent status (which agent is running, intermediate outputs).