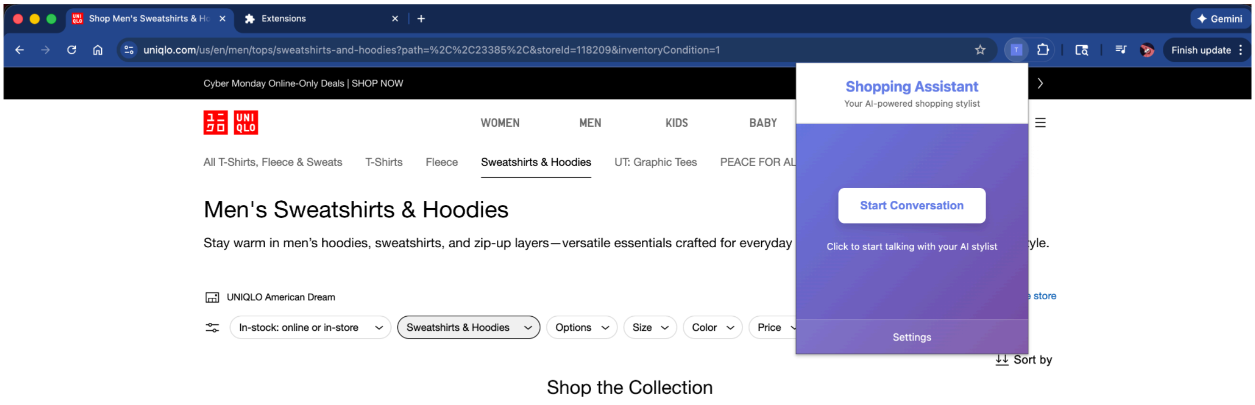
Tavus Personal Shopping Assistant – Project Debrief

Author: Vishal Patel

Loom

Github



uniqlo.com/us/en/men/tops/sweatshirts-and-hoodies?path=%2C%2C23385%2C&storeId=118209&inventoryCondition=1

Cyber Monday Online-Only Deals | SHOP NOW

Men's Sweatshirts & Hoodies

Stay warm in men's hoodies, sweatshirts, and zip-up layers—versatile essentials crafted for everyday yle.

Shopping Assistant

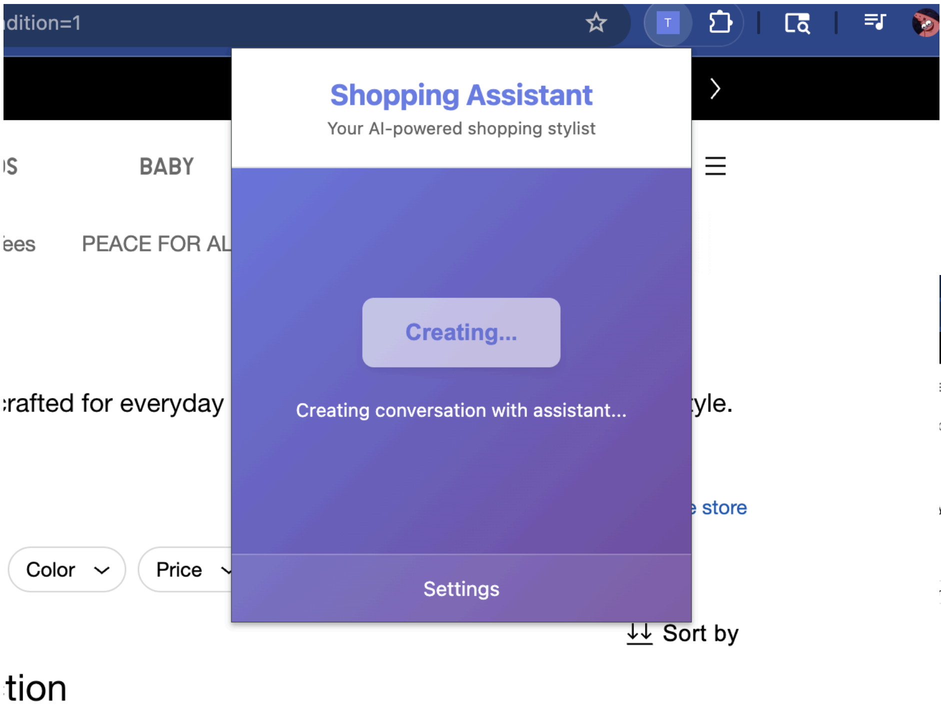
Your AI-powered shopping stylist

Start Conversation

Click to start talking with your AI stylist

Settings

Shop the Collection



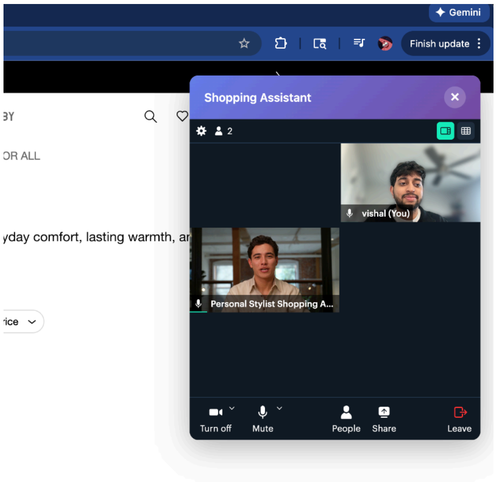
Shopping Assistant

Your AI-powered shopping stylist

Creating...

Creating conversation with assistant...

Settings



Shopping Assistant

Run it Yourself:

Everything is documented in the README on how to get started locally.

Main takeaways of my candidacy

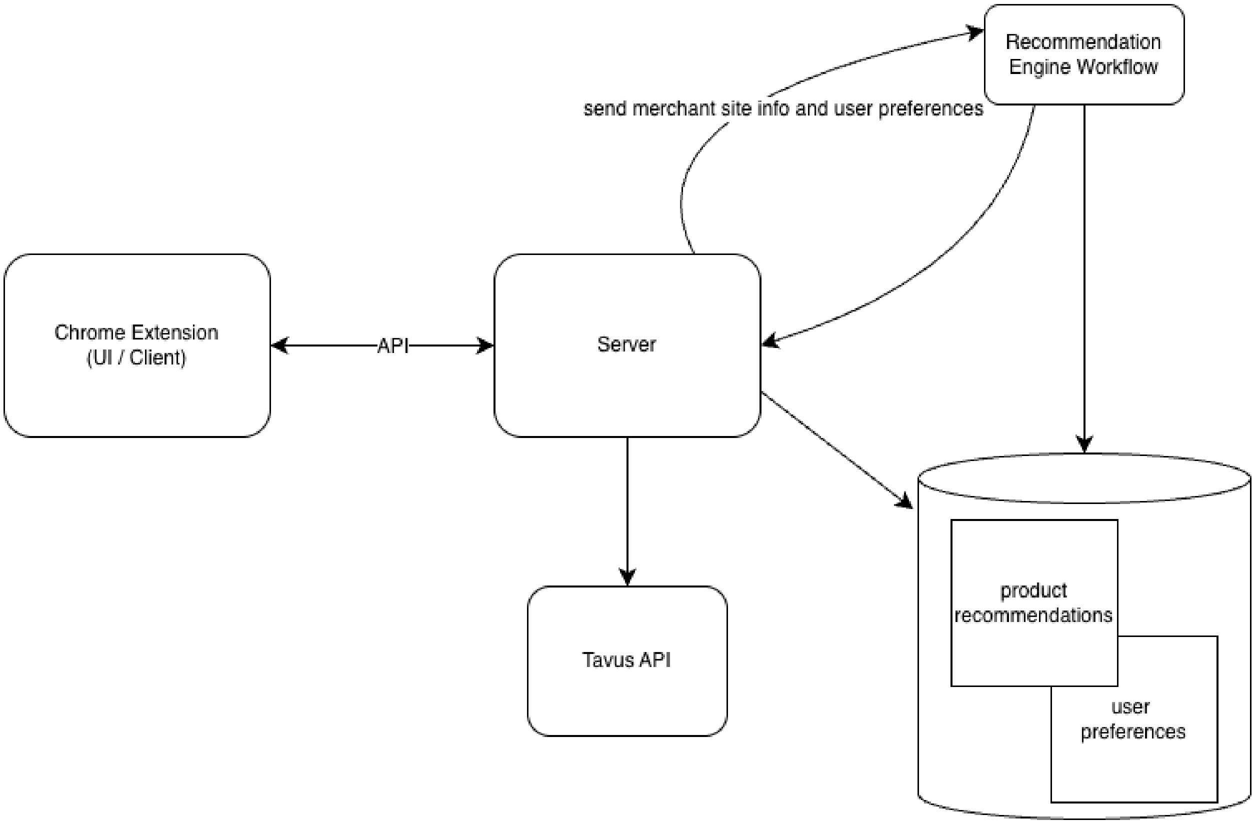
I wrote this debrief to highlight a few core things about myself as an engineer, builder, and product thinker.

- My speed: the entire project was built rapidly but thoughtfully. In a short window, I implemented a functioning Chrome extension with on-page widget injection, a backend FastAPI server, a Tavus persona with guardrails, greeting logic, persona engineering, and a recommendation search, all summing up to an end-to-end conversation flow
- My product vision and creativity: I didn't want to just build "a Tavus integration." I wanted to explore a direction where Tavus can exist simultaneously as B2B infrastructure and a B2C personal assistant. By creating a stylist that can live inside the browser (and eventually merchant pages) I aimed to demonstrate how the CVI can unlock entirely new shopping experiences. This projects shows how I think about new user behaviors, where Tavus can slot into real consumer experiences, and how to turn an API into a compelling, futuristic product
- My willingness to adapt and improve: throughout the project, there were several moments where things could have slowed me down, perception setup complexity, recommendation engineer latency issues, LLM formatting challenges, persona tuning, etc. Instead of stalling, I replaced slow components with fast stubs, iterated on person behavior, tuned guardrails, and refocused on what mattress most for the demo.

Product Vision:

The long term idea is to create a humanlike AI shopping assistant that can support both consumers and businesses. On the consumer side, the stylist would follow the user across the entire web and learn their taste, their budget, the types of outfits they like, and even signals from Pinterest boards, photos, social media profiles, or past purchases. Instead of shopping alone across many different sites, the user would have a consistent stylist who understands their style and guides them in a personal, helpful way.

For merchants, the same Tavus agent could live directly on product pages as a digital sales associate. It could help shoppers choose between items, explain fit, recommend alternatives, and support the buying journey in a way that feels more human than a traditional chatbot. This project is one demonstration of how a flexible, conversational stylist could work across both sides of the shopping experience and bring more intelligence, personality, and guidance to online retail.


Recommendation Engine Workflow
send merchant site info and user preferences
Chrome Extension (UI / Client)
API
Server
Tavus API
product recommendations
user preferences

High-Level Flow:

1. User browses a retailer
2. User clicks extension
3. Extension sends URL + metadata to backend server
4. Backend server starts a product recommendations engine (stubbed) based on user preferences & website
5. Backend converts recs → natural-language signals → JSON context
    - Can be stored in the db to cache recommendations and preferences
6. Backend creates a Tavus conversation with the following context:
    - Website
    - Product category
    - User style preferences
    - Recommendation insights
7. Extension displays Tavus widget
8. User chats with the stylist
9. Stylist provides tailored recommendations in natural, spoken language

Technical Specs:

More information in the README, and of course in code itself

Backend

Endpoints

POST /api/start_conversation

- Receives page URL + merchant
- Loads recommendation engine output
- Creates Tavus conversation
- Returns conversation_url

Extension

- Manifest V3
- Popup extracts current page
- Background script sends info to backend
  Content script injects widget
- Renders Tavus iframe

AI Layer (Tavus)

Persona

- Pre-configured stylist identity
- Natural conversational tone
- Knowledge of style principles
- Grounded with documents from Pinterest (fits, silhouettes, aesthetics)

Guardrails

- Prevent JSON reading
- Eliminate lists/tables in speech
- Enforce spoken language only
- Ensure natural stylist-like tone

Future improvements

- Perception for facial cues
- Perception for page understanding
- On-screen outfit evaluation

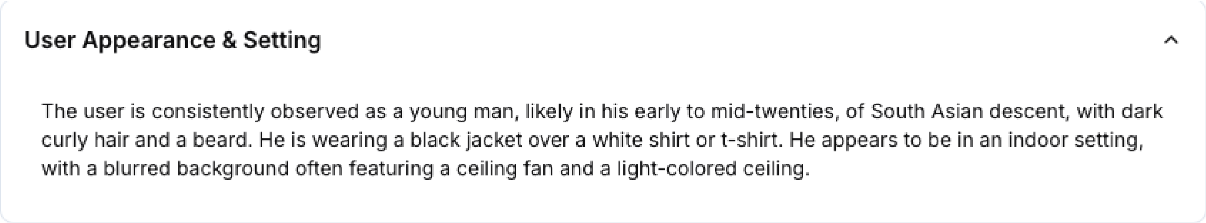
Improvement Areas

Product:

- Integrate perception model via React client to gain visual context from the physical appearance of the user and from screen sharing and actively seeing what is on the user's page. Can feed this information into the recommendation engine and pass the output into the conversation context before the call. Can also patch the info during live time
- With the benefit of the perception mode, can have the agent control the browser page to take you to product pages and show you outfits
- B2B mindset: embed Tavus agents within merchant sites as a personal shopping assistant
- B2C mindset: users can set up a Tavus agent as a stylist / shopping assistant across any merchant site to get personalized recommendations
    - Users can allow Tavus agent to connect to various forms of user data to get better context and style recommendations. ex) connect to socials, pinterest boards, selfies, etc.

Technical:

- Replace CSV stub with a real recommendation engine/workflow that takes in the user context and finds recommendations from the merchant site. Wrote the API call to Parallel AI (web search/scrape APIs) to replace the stub. Latency is high on the lower tier models for the Parallel API so a stub was made for a time sake
- Perception integration (mentioned above). We already get a description of what the model sees, just need to feed it into the data loop to get customized insights.



User Appearance & Setting

The user is consistently observed as a young man, likely in his early to mid-twenties, of South Asian descent, with dark curly hair and a beard. He is wearing a black jacket over a white shirt or t-shirt. He appears to be in an indoor setting, with a blurred background often featuring a ceiling fan and a light-colored ceiling.