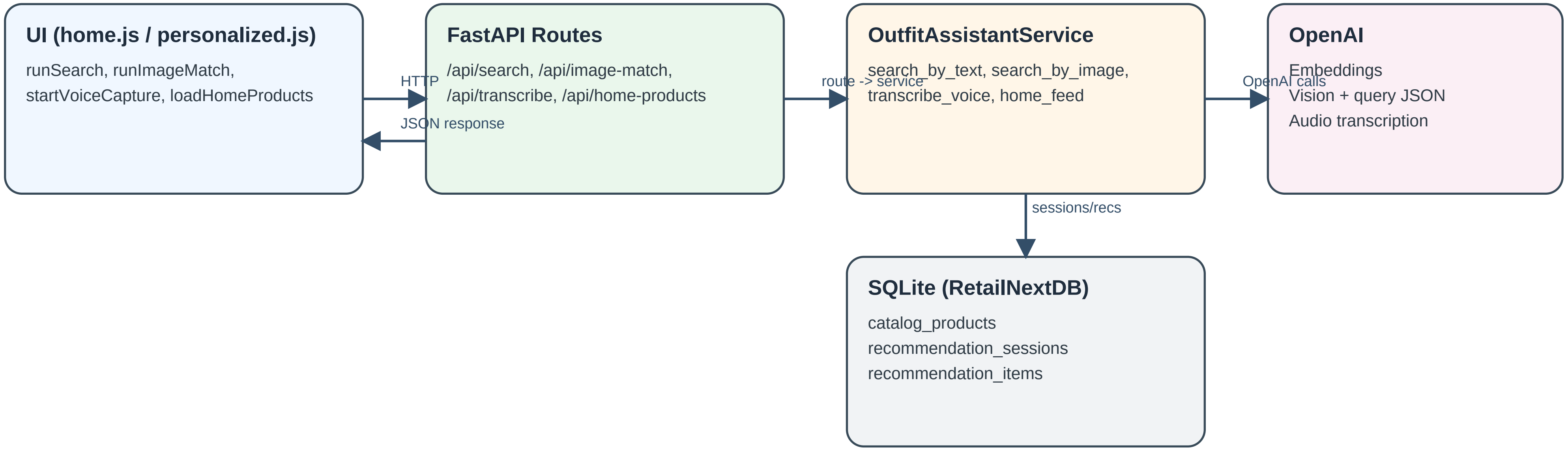
RetailNext Outfit Assistant - Request Trace Sheet

Generated from current codebase: 2026-02-08 14:17 UTC

End-to-End Request Flow Diagram



Trace Matrix (A-D)

Flow	UI Trigger	Backend Route	Service Method	OpenAI Calls (exact models)	Retrieval / DB Logic	Output to UI
A) Text search	Form submit handler **runSearch** in **home.js** and **personalized.js** posts to **/api/search**.	**search(request: SearchRequest)** in **app/api_server.py**.	**search_by_text** -> **_rank_from_queries** -> **_embed** in **src/retailnext_outfit_assistant/service.py**.	Embeddings: **text-embedding-3-large** (default **RN_EMBEDDING_MODEL**). GPT chat model configured: **gpt-4o-mini** (default **RN_CHAT_MODEL**) but not invoked directly by **/api/search**.	Cosine retrieval via **top_k_cosine**; ranked IDs saved using **create_session** + **store_recommendations**.	**session** (session_id, shopper_name, source, query_text, image_summary, created_at), **recommendations[]** (id, name, gender, master_category, sub_category, article_type, base_colour, season, year, usage, image_url, rank, score, optional match), **assistant_note**, **ai_powered**.

Flow	UI Trigger	Backend Route	Service Method	OpenAI Calls (exact models)	Retrieval / DB Logic	Output to UI
B) Image match	Image upload handler **runImageMatch** in **home.js** and **personalized.js** posts multipart form to **/api/image-match**.	**image_match(...)** in **app/api_server.py**.	**search_by_image** in **src/retailnext_outfit_assistant/service.py**.	Vision parsing + query generation in **analyze_outfit_image** using **gpt-4o-mini** (default **RN_VISION_MODEL**) via **chat.completions.create**. Returns JSON with gender, occasion, colors, article_types, search_queries.	Multi-query fusion: for up to 6 generated queries, embed each query, retrieve **top_k*8** candidates, keep max similarity per product ID, sort globally, truncate to top_k.	Same shape as text search plus **image_analysis** (parsed vision JSON), **assistant_note**, **ai_powered**.
C) Voice	**MediaRecorder** flow in **startVoiceCapture** -> **transcribeAudioBlob** posts to **/api/transcribe**; transcript fills search box; user confirms with Find Items (**runSearch** -> **/api/search**).	**transcribe(audio: UploadFile)** in **app/api_server.py** returns service payload.	**transcribe_voice** in **src/retailnext_outfit_assistant/service.py**.	Audio transcription via **client.audio.transcriptions.create** with **gpt-4o-mini-transcribe** (default **RN_AUDIO_MODEL**).	No retrieval in transcribe step; transcribe output is then fed into the same text-search flow.	**/api/transcribe** returns **{text, model, ai_powered}**; then **/api/search** returns the normal recommendation payload.
D) Home feed	Tab click / refresh triggers **loadHomeProducts** in **home.js** to request **/api/home-products?limit=24&gender;=Women|Men**.	**home_products(limit, gender)** in **app/api_server.py**.	**home_feed** in **src/retailnext_outfit_assistant/service.py**.	**None** (no OpenAI call in home feed route).	Reads SQLite table **catalog_products** via **list_random_products**. If gender is provided: **WHERE lower(gender)=lower(?) ORDER BY RANDOM() LIMIT ?**; else random global sample.	**{shopper_name, gender_filter, products[]}** where each product includes id, name, gender, master_category, sub_category, article_type, base_colour, season, year, usage, image_url.