Take-Home Challenge: BigSpring "Knowledge-to-Action" Search Agent

1. Project Context

BigSpring is an enterprise platform where companies upskill their workforce through **Plays**. A Play is a logical sequence of **Reps**, categorized into:

- **Watch Reps (Knowledge):** Multi-modal content (PDFs, Videos, Flashcards) meant to be consumed.
- **Practice Reps (Action):** Tasks where users submit responses (Video/Audio/Text) and receive **AI Coach Feedback**.

2. The Task

Build a secure, multi-tenant **Generative Search Engine** that allows Sales Representatives (**users**) to retrieve specific data from their **assigned knowledge** and their **own performance history**.

Note: *This is a Search Engine, not a general-purpose conversational assistant. Queries unrelated to the user's professional scope or assigned materials must be handled with strict guardrails.*

A. Core Workflow

1. **Context Selection:** A simple UI to select a **Company** and then a **User** within that company. Once the user is in, they can search for anything in natural language which then will be retrieved from the database.
2. **Scoped Search:** Search and results must be strictly limited to that user's assigned Plays and personal Practice submissions.
3. **Intent-Driven Retrieval:**
    - If the query is a valid search for assigned knowledge or history, provide a grounded answer with citations.
    - If the query is a general professional request (e.g., sales techniques), follow the fallback protocol described in section 4.
    - If the query is out-of-scope (non-professional), trigger the **Search Boundary Guardrail** described in section 4.
    - Make sure to restrict the search across different companies.

3. Data Requirements & Identity Layer

- **Companies:** 5 organizations (Veldra Therapeutics, Aetheris Pharma, Kyberon Cloud, Sentivue AI, Hexaloom Nanoworks).
- **Users:** Sales Representatives assigned to specific **Segments** or **Product Verticals**. Some may have different job titles.
- **Assignments:** Mapping between `user_id` and `play_id`. Users cannot search content from other users, segments (unless assigned), or companies.
- **Assets**: Simplified JSON files representing the content available for search.
- **History:** You are given submissions in the form of user transcripts (text) and AI Coach qualitative feedback.

4. Intelligent Fallback & Guardrails

Implement a three-tier logic for handling user queries:

1. **The Search Boundary (Out-of-Scope):**
    - If a user asks a query unrelated to their job or assigned materials (e.g., *"What does Earth look like from space?"* or *"Tell me a joke"*), the engine must recognize this as an out-of-scope request.
    - **Required Response:** *"I am a specialized search engine for your assigned BigSpring materials. I cannot assist with queries outside of your professional scope."*
2. **General Professional Knowledge (Fallback):**
    - If a user asks for a general sales technique (e.g., *"How do I handle price objections?"*) and it is not in their assigned Plays, the system should provide a helpful response based on **LLM internal knowledge or Web Search**, with a clear disclaimer: *"This response is based on general sales knowledge and is not found in your assigned company materials."*
3. **Proprietary Data Guardrail (No Hallucination):**
    - For specific company data (e.g., *"What is the mortality rate for Vax-Alpha?"*) where no grounding exists in the user's assigned Plays, the engine **must not** guess or use internal knowledge.
    - **Required Response:** *"I cannot find any specific information in your assigned materials regarding this query."*
4. **Conflict Prevention:** Ensure brand-specific data remains isolated (e.g., Zaloric vs. Nuvia) even if they share chemical ingredients.

5. Technical Guidelines & Architecture

A. Backend Orchestration & Intent Dispatching

You are free to design the backend architecture as you see fit. However, you must be prepared to justify your choice during the live review. Possible patterns include:

- **Single Chain:** One prompt managing multiple tools.
- **Router Pattern:** A classifier prompt that routes the query to specialized sub-prompts/tools (e.g., Knowledge Search vs. Practice History).
- **Agentic/Planner Pattern:** A "Planner" agent that identifies if a query needs Knowledge (Watch Reps), History (Practice Reps), or both, and orchestrates specialized retrievers accordingly.

At BigSpring, we use Python, Flask and/or FastAPI for generative AI development and PostgreSQL and Qdrant as DBs while Node.js, typescript, GraphQL for the remaining backend. For LLMs, GPT and Gemini are preferred models, however, you can use any other open-source alternatives. The main aim is not performance optimization, but to get the basic system working

B. Key Functionalities

- **Deterministic Partitioning:** Enforce Company/User filters at the DB level *before* vector search.
- **Recommendation Engine:** After every answer, recommend 2-3 **Follow-up Contents** (Reps/Plays) relevant to the query and assigned to the user.

C. Frontend Integration (**Nice to have,** *however the backend should provide all the fields required for the UI*)

- **Thought Trace and Response skeleton:** If needed, make use of skeleton elements in the UI. Also, if any reasoning needs to be shown, make sure to use the collapsable trace.
- **Streaming UI:** Answers must stream token-by-token. Use custom UI components (cards/side panels) for Citations and Recommendations.
- **Rich Citations:** Deep-linked citations like `[PDF: Page 12]` or `[Video: 04:20]`.
- **Component Hydration:** Use custom UI components to render "Rep Cards" or "Recommendation Carousels" in a side panel.

6. Bonus & Advanced Challenges

The following tasks are optional but highly encouraged. You can choose to implement or at least describe the approach of how you'd like to implement this if given enough resources:

A. Continual Learning & Feedback Loops

- **User Feedback Gathering:** Implement a UI mechanism for users to upvote/downvote or "Correct" an answer.

- **Self-Correction Pipeline:** When a user provides a correction, the system should store this in a "Verified Truth" store. Subsequent queries should prioritize this verified data over the original content.
- **Conflict Resolution:** Briefly describe or implement how you would handle conflicting feedback from two different expert users.

B. Prompt Refinement Pipelines

- **Automated Optimization:** Propose or implement a way to use user feedback to automatically refine system prompts or few-shot examples over time.

C. Evaluation Frameworks

- **Quality Metrics:** Implement or describe a simple framework to evaluate:
    - **Retrieval Accuracy:** Were the right chunks found?
    - **Answer Faithfulness:** Did the answer hallucinate beyond the provided grounding?

7. Submission Instructions

- **Timebox:** 2 weeks.
- **Submission format:** Please provide a GitHub repository along with the README detailing the instructions to run the code. You can also host the app on free platforms to demo a working prototype or include a walkthrough video in the repo.
- **AI Usage:** You are free to use (Cursor, ChatGPT, etc.). Include a brief summary of tools and the key prompts that helped develop the functionalities in your README.
- **Live Review:** We will perform a **Live Mod** (e.g., "Adjust the guardrail so that general sales advice is disabled for the Manufacturing company").

Directory Structure

`/database` (System Tables)

Contains the relational backbone in CSV/JSON format. Use these to build your permission filters.

- `companies.json`: Defines the companies present.
- `play.csv` / `rep.csv`: Defines the hierarchy.
- `play_assignment.csv`: **Crucial.** Maps `user_id` to `play_id`. Search queries must be filtered by the plays found here.
- `submission.csv` / `feedback.csv`: Maps user performance data to assets.
- `asset.csv`: The master manifest linking `asset_id` to file paths.
- `users.csv`: Defines the user details.

`/assets` (RAG-Ready Metadata)

Contains structured JSON files optimized for granular search and deep-linking. This is done for all the assets including PDF, video, audio, text and image. Make sure to verify the structure of different JSONs. For example:

- **PDF-to-JSON:** (e.g., `amproxin_guide.json`) Diarized by `page`. Includes `sections` for context and `tables` for mathematical reasoning.
- **Video/Audio-to-JSON:** (e.g., `sentilink_ai_speed.json`) Contains `full_transcript` and a `segments` array with `start`/`end` timestamps for temporal citations.

`/raw_assets` (Source & UI Files)

- Original `.pdf` documents for user download/viewing.
- `thumbnails/`: PNG/JPG files for the video/image training modules (e.g., `kstream_v_vorex_thumb.png`). You can use these to display in the search results.

Schema

1. Identity & Security Layer

- **Company**: `id`, `name`, `description`
- **User**: `id`, `username`, `display_name`, `role`, `segment`, `created_at`, `is_active`, `company_id`
    - `username` is unique

2. The Play Hierarchy

- **Play**: `id`, `company_id`, `title`, `description`, `created_at`, `is_active`
- **Play_Assignment**: `id`, `user_id`, `play_id`, `assigned_date`, `status` (assigned, in progress, completed), `completed_at`
- **Rep**: `id`, `prompt_text`, `rep_title`, `rep_type`, `play_id`, `company_id`, `asset_id`, `created_at`
    - **Logic**: Watch Reps (watch) = assigned knowledge. Practice Reps (practice) = actionable tasks.

3. The Multi-Modal Asset Layer

- **Asset**: `id`, `type`, `file_name`, `created_at`, `company_id`
    - **Logic**: `file_name` references a structured JSON file containing the converted content.
- **Asset Content (JSON Formats)**:
    - **PDF**: `[{ "page": integer, "text": string, "tables": array }]`
    - **Video**: `[{ "start": string, "end": string, "text": string }]`
    - **Flashcard**: `[string, string, ...]`
    - **Text**: Normal text string

4. Performance & Coaching Layer

- **Submission**: `id`, `user_id`, `rep_id`, `submitted_at`, `submission_type`, `asset_id`, `company_id`
    - **Logic**: Records user response (e.g. text transcript) to a Practice Rep.
- **Feedback**: `id`, `submission_id`, `company_id`, `score`, `text`, `created_at`
    - **Logic**: Used for "Coaching Insight" and "Performance" queries.

Test Case Suite for BigSpring Search Engine

Note: These are for reference and understanding only. The actual data may be different.

Happy Scenario 1: Valid Search (Authorized Access)

- **Case 1.1: Knowledge Base PDF Search**
  - **Username:** `aaron-veldra` (Aaron Montgomery)
  - **Assigned Play:** `play-vel-001` (Amproxin: Antibiotic Excellence)
  - **Search Query:** "What is the eradication rate for Streptococcus pneumoniae?"
  - **Expected Answer:** "The eradication rate for Streptococcus pneumoniae using Amproxin is 94.2%."
  - **Citations:** `[amproxin_guide.pdf: Page 1]`
- **Case 1.2: Personal Video Submission Search (Deep Link)**
  - **Username:** `daphne-kyberon` (Daphne Blake)
  - **Own Submission:** `sub-kyb-001` (GridMaster Pitch)
  - **Search Query:** "When did I mention cooling energy costs?"
  - **Expected Answer:** "You mentioned that GridMaster reduces cooling energy costs by up to 32 percent."
  - **Citations:** `[Video: kyb_sub_001.json at 00:26 - 00:38]`

Happy Scenario 2: Invalid Search (Unauthorized/Unassigned Content)

- **Case 2.1: Cross-Company Leakage**
  - **Username:** `sophie-aetheris` (Sophie Martin)
  - **Search Query:** "Show me the GridMaster PUE efficiency table." (This is Kyberon content).
  - **Expected Answer:** "No results found." (The engine must filter out `comp-kyberon-003` assets for an `aetheris` user).
- **Case 2.2: Assigned Play vs. Unassigned Play (Same Company)**
  - **Username:** `leo-aetheris` (Leo Fritz)
  - **Assigned Play:** `play-aet-001` (Somnirel)
  - **Unassigned Play:** `play-aet-002` (Nuvia)
  - **Search Query:** "How does Lydrenex protect the amygdala?"
  - **Expected Answer:** "No results found." (User is not assigned to the Nuvia play, even though it's the same company).
- **Case 2.3: Searching Peer Submissions**
  - **Username:** `aaron-veldra` (Aaron Montgomery)
  - **Search Query:** "Show me Aaron's pitch about antibiotics." (Searching for a submission by `quinn-veldra`).

- **Expected Answer:** "No results found." (Users cannot see transcripts of other users' practice reps).

Edge Cases (Context & Shared Ingredients)

- **Case 3.1: Shared Ingredient Disambiguation**
  - **Username:** `aaron-veldra` (Veldra user)
  - **Search Query:** "What is the dosage for Lydrenex?"
  - **Expected Answer:** "The high-strength dosage for Zaloric (Lydrenex) is 50mg - 100mg for muscle spasms."
  - **Citations:** `[nuvia_lowdose.pdf: Page 1 - Table 1]` (Note: Veldra users see the Zaloric row in the matrix).
  - **Developer Check:** Ensure the AI doesn't return the Aetheris (5mg-10mg) result unless that specific PDF is assigned to the Veldra user as a "Competitor Matrix."
- **Case 3.2: Multi-Page Reasoning**
  - **Username:** `daphne-kyberon`
  - **Search Query:** "Compare my rack temperature baseline vs my GridMaster results."
  - **Expected Answer:** "Your baseline rack temperature was 24.5°C, and with GridMaster it dropped to 21.8°C, an 11.02% gain."
  - **Citations:** `[gridmaster_blueprint.pdf: Page 2 - Table 2]`

Bad Data & Error Handling

- **Case 4.1: Fuzzy Matching / Typo**
  - **Username:** `clark-sentivue`
  - **Search Query:** "Sentalink acceleration speed" (Correct spelling: `Sentilink`)
  - **Expected Answer:** "The Sentilink integration provides a 15x increase in Time to First Insight."
  - **Citations:** `[sentilink_ai_speed.json: 04:11]`
- **Case 4.2: Out of Scope "General Knowledge"**
  - **Username:** `quinn-aetheris`
  - **Search Query:** "How do I make a chocolate cake?"
  - **Expected Answer:** "I'm sorry, I couldn't find any information about that in your assigned learning materials or submissions." (Ensures the LLM doesn't use its base training data to bypass the knowledge base).