See our prioritized requirements below.

(1) Natural language search to replace the current tag-based search

Goal:

Replace the current tag-based search with an on-device natural language search capability that works offline. The new search should go beyond exact keyword matches by retrieving relevant results from meaning-based queries while remaining lightweight enough for local device operation.

Requirements:

- natural language query input (see our current search bar) that accepts free-text
- Retrieval should support both keyword and semantic retrieval
- The search backend is local and offline, meaning the system shall perform indexing and search locally on the device.
- Search results are the same as the current result presentation
- The system shall no longer require manual tagging to make content searchable, though existing tags may be retained as optional filters.
- The system shall support local index creation and refresh as part of the device update process.

(2) PWA proof of concept for mobile app (viability first)

Goal:

Build a Proof of Concept (POC) for a Progressive Web App (PWA) to validate whether a web-based, installable mobile experience can support the current functionalities before committing to a full mobile app rebuild.

Requirements:

- PWA prototype that supports a core workflow end-to-end (example: search and open a result, or view content).
- Confirm feasibility and constraints

(3) Packaging and deployment for device software (Docker vs Ansible)

Goal:

Establish a repeatable packaging and deployment approach for the device software to reduce manual setup effort and improve consistency across devices. The solution should evaluate and support at least two candidate approaches (for example, Ansible and Docker) and may include other viable options.

Requirements:

- Produce a repeatable packaging pipeline for one version (for example, the current mini-computer software version).
- The solution shares produce a released artifact that can be stored and distributed.
- A documented, repeatable process that a non-expert can run to install and test.

(4) Mobile update experience

Goal:

Improve the “Connect to Update” experience so users understand what is happening and do not lose confidence mid-update. (see the video demo by Manoj below)

Share link:
https://cgu.zoom.us/rec/share/lzNYA7BxFgYFLfysekmUjn_hKVofnWvDrYXfonsOzKMUVF9nNI-2hVpGY0Am9H8-.hkprwSNHgnZ6u7qf

Requirements:

- UX workflow cleanup to make the update steps clear and error-free.
- Real-time status updates during the update process (not feasible now, see video).

(5) Content usage tracking

Goal:

Improve CME content analytics by tracking meaningful usage for PDF and audio materials beyond the current “content opened” event.

Requirements:

- The system shall capture, at minimum, events that distinguish engagement from opening, including content opened; progress update (PDF page progress and/or audio playback progress); content closed / session ended, and completion.
- Create a completion definition for different content.

(6) In-app continuity for CME Content Access

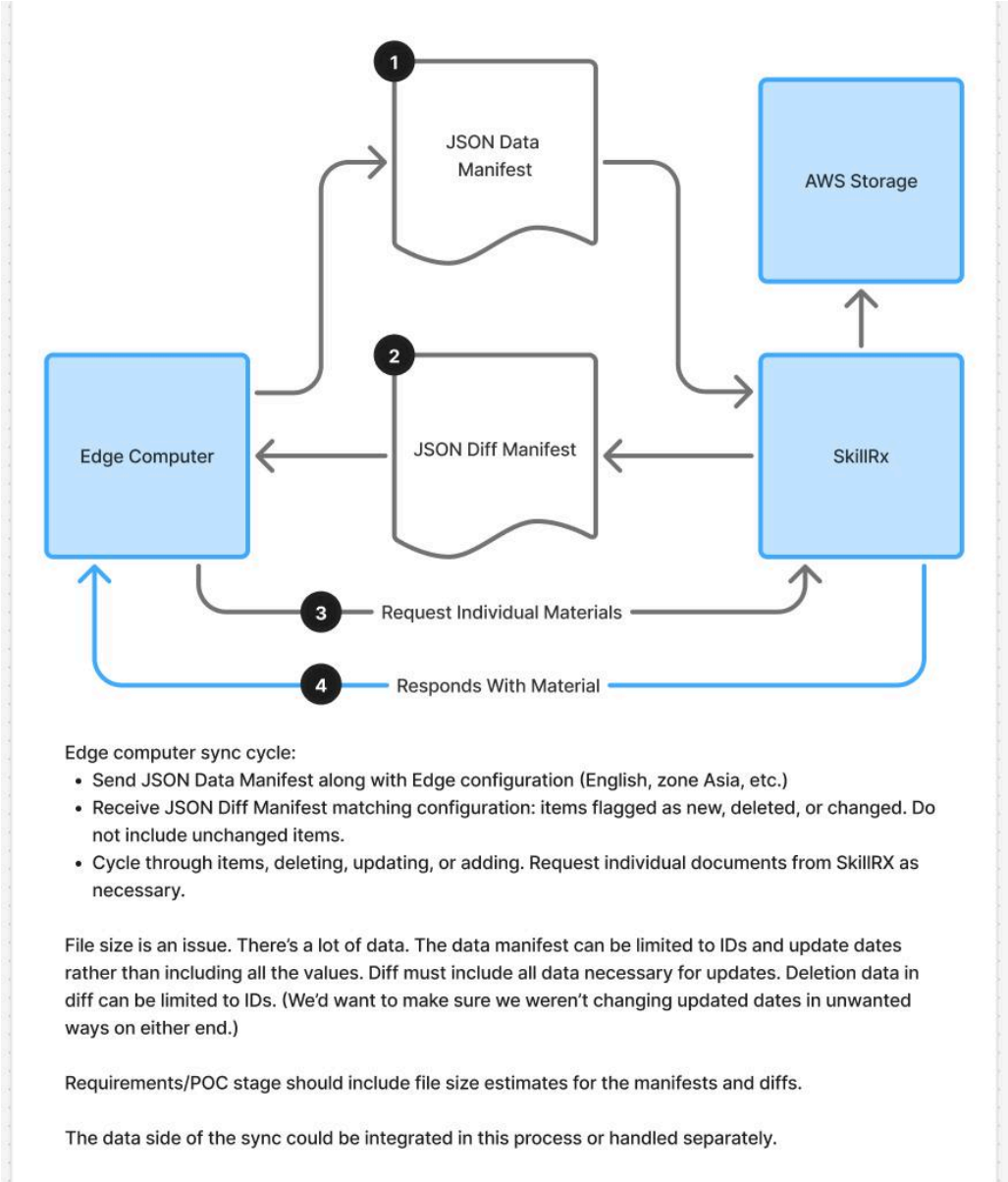
Goal:

Ensure users can open CME content without being redirected to separate webpages.

Requirements:

Any CME content (under CMES extras) currently opens a separate page. They should be opened within the app.

(7) Updated Sync Functionality



Edge computer sync cycle:

- Send JSON Data Manifest along with Edge configuration (English, zone Asia, etc.)
- Receive JSON Diff Manifest matching configuration: items flagged as new, deleted, or changed. Do not include unchanged items.
- Cycle through items, deleting, updating, or adding. Request individual documents from SkillRX as necessary.

File size is an issue. There's a lot of data. The data manifest can be limited to IDs and update dates rather than including all the values. Diff must include all data necessary for updates. Deletion data in diff can be limited to IDs. (We'd want to make sure we weren't changing updated dates in unwanted ways on either end.)

Requirements/POC stage should include file size estimates for the manifests and diffs.

The data side of the sync could be integrated in this process or handled separately.

(8) Integration of Clinical Decision Support Tools in CMES

Goal:
Provide offline access to Clinical Decision Support Tools

Requirements:

- Clinical Decision Support Tools like CorePendium, Core UltraSound, RCH Pediatric care, Life in The Fast Lane (LIFTL) should be listed and accessible under “CMES Extra”.
- For CorePendium, create a pipeline to load content from CorePendium site to CMES cloud storage for sync with edge device
- For Core UltraSound

    - Create repo similar to “alfred” under ‘CMES Extra” for the “Fundamental Course”
    - Replicate the functioning question bank for users to do knowledge assessment in offline mode
    - Required documentation and content can be provided
- For LIFTL, create a pipeline that scrapes and loads marked content from LIFTL website to SkillRx for future sync with edge device.

https://www.dataplicity.com/