Community Hub Requirements

Product Development

Define:

- Website - Collection of multiple, interconnected web pages under a single domain name
- Webpage - A single, distinct document on the internet that can be displayed in a web browser.

Identifying and Categorizing Information Sources

A method for inputting, storing, viewing, updating, and categorizing websites (in order to crawl them). Example here.

Collecting Information

The ability to crawl the stored sites automatically with different rules for different pages.

A. To start, each website is crawled 1x per week (n). This is done at different times to avoid every site being crawled at the same time.
B. If there is a change on a specific page within that site during the next crawl (n+1), the specific page is crawled every day (m).
C. If there is a change in the next 2 days (m+1 or m+2), then that specific page is to be crawled every hour from 7 a.m. to 12 a.m.
D. If, after 30 days, there is no change to the sites in either B or C, then it reverts to the previous crawl schedule (i.e. C reverts to B, B reverts to A).

The contents of the websites are to be stored in a database in order to be compared to future versions of the site. After each crawl, the contents of the page/site are to be compared to the previous version. If there was no change, do nothing.

Generating Content

If there was a change to the content of the site, then apply rules in order to determine if the new content is to be sent to an LLM.

- Rules:
    - Any text has to be at least 50 characters in length.
    - Any image has to have text associated with it.

For new content that survives the rules, it is to be sent to an LLM with a prompt.

The output from the LLM is to be stored. It is important to be able to apply the categorization from the originating website to the output from the LLM.

The stored content is to be sent to the publishing mechanism along with the categorization.

Organizing Content

The basic organizing concept is the community. Each community will have their own integrated news feed.

In the publishing mechanism, each Geographic Location will have their own queue. And, each will have their own human editor assigned to it.

The LLM content will be transferred to the appropriate queue based on the 1st order classification.

When new content enters the queue, a notification will be sent to the editor of the queue.

The editor will be able to access the content and edit the content (HUMAN CONTROL)

Once the content is ready for publication, the editor is able to publish the content (HUMAN CONTROL).

There are two choices: Publish Now or Scheduled Post. The default is Scheduled Post. The publishing mechanism will automatically schedule posts in different time slots to ensure there is a steady stream of content to the community news feeds.

Publishing and Sharing Content

The content will be automatically sent to the appropriate community news feed.

If 25% of users select the content, it is deemed to be “Hot” and it is automatically placed in the news feeds of neighbouring communities:

- Local to Local +
- Local+ to Regional
- Regional to the neighbouring regions
- Provincial to neighbouring provinces
- Federal to neighbouring countries.

For this to continue, the number of users that have to select the content will decrease by ⅕, or 25% (local) to 20% (Local+) to 15% (Regional) to 10% (Provincial) to 5% (federal). This will need to be carefully tested to ensure effectiveness and accuracy.

Users will be assigned geographic location(s). They will have a primary geographic location based on where they live. This will be obtained when they first sign up to the application. They will be able to add additional geographic locations at their discretion.

Users will be assigned interests based on the content in which they engage. The user will be shown content from other communities that they can reasonably be expected to want to see.

Mobile App

This has to be mobile first. It's the way most people consume content.

The app is to be a very simple, blog style app.

To access content, the user has to sign up. The sign up process involves three things: (1) username/password, (2) Credit card, and (3) Location.

The user can browse content (Home), view saved content (Saved), submit a news story (Submit), search for content (Search), update their feed settings and view their profile (Settings).

In a story, there is the option to save the story. There is the option to provide a news tip. There is the option to submit feedback on the story.

Software Development

Core Agent Concept

- Accepts a **user's location** (e.g. Vancouver, Toronto, Seattle, etc.).
- Automatically **searches the internet** (and later, other data sources) for **relevant local information** — news, updates, community events, blog posts, etc.
- Stores and classifies the results in a **structured way** (with metadata like source, category, topic, date).
- Detects **content changes over time** — so it knows when something new or significant happens.

Essentially, it's a **crawler + summarizer + classifier** pipeline with adaptive frequency.

Research Agent (Phase 1)

Goal: "Find and summarize what's happening around a user's location."

Responsibilities:

1. Take location input.

2. Crawl or query relevant online sources (news, blogs, organizations, social feeds, etc.).
3. Filter noise (apply text/image length or relevance rules).
4. Generate clean summaries of discovered content via an LLM.
5. Store both the summaries and metadata (location, source, timestamp, category).

In the future, this agent could:

- Leverage a **vector database** (like Azure AI Search or Pinecone) for similarity search.
- Support **feedback loops** (if a user marks a topic as interesting, the agent learns their preferences).
- Use **Azure AI Foundry Actions** to call REST APIs for crawling, storage, and retrieval.

3. Profile Agent (Phase 2)

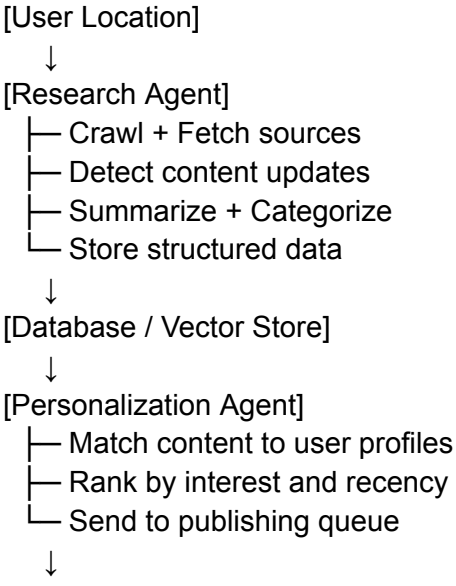
The second agent (let's call it the *Personalization Agent*) would:

- Take a **user's profile** (interests, browsing habits, engagement history, demographic data).
- Filter or rank all content from the Research Agent based on relevance to that user.
- Continuously learn from what the user clicks, saves, or ignores.
- Generate a **personalized feed** (like a dynamic "For You" page).

Later, this could integrate with your mobile app or a React front end, where:

- Users can **follow communities**, **save stories**, or **submit new tips**.
- The app's backend invokes the agent endpoints (via Azure OpenAI endpoint or Functions) to fetch curated content.

4. System Overview (Conceptual Flow)



[Human Editor] → [Community Feed]

5. Azure Foundry

Component	Azure Foundry Role	Description
Research Agent	Primary Agent	Handles crawling, summarization, categorization.
Profile Agent	Connected Agent	Handles personalization, filtering, user modeling.
Storage Layer	Vector Store / Azure AI Search	Stores embeddings, summaries, categories.
Rules + Scheduling	Azure Functions / Logic Apps	Automates crawl frequency and rule transitions.
Publishing Mechanism	Web App + API	Pushes content to editors and feeds.