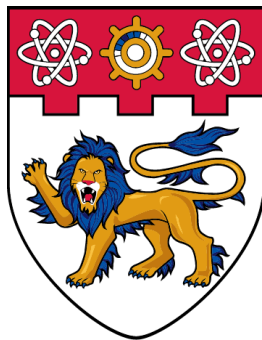


NANYANG TECHNOLOGICAL UNIVERSITY
College of Computing and Data Science (CCDS)
SC4052: Cloud Computing



**NANYANG
TECHNOLOGICAL
UNIVERSITY**

SINGAPORE

Academic Year: 2024-2025, Semester 2

Project Report: Interesting X-as-a-Service

M Hisham B Khairul A (U2121992E)

Contents

Introduction.....	3
Background.....	3
Problem.....	3
Solution.....	4
Outline.....	4
Backend.....	5
Frontend.....	7
Roadmap.....	10
Outline.....	10
Item classification.....	10
“Irrelevant” item storage.....	10
Deployment.....	10

Introduction

Background

This report is written for the Cloud Computing module project, under the topic *Interesting X-as-a-Service*. It describes a proof-of-concept project for tracking and analyzing online marketplaces using an LLM.

The code can be found at <https://github.com/mhbka/itemtracker>

Problem

A personal hobby of mine is to collect rare T-shirts, in particular shirts that are vintage, only printed in limited runs, and containing niche media. Such shirts cannot be found in any traditional stores. Instead, they have to be scoured for online, in places such as user marketplaces, specialty stores, and even social media.

Depending on the rarity of the shirt, they can be incredibly hard to find; in some cases, these shirts may have only appeared on the internet less than 5 times. On the other hand, highly sought-after shirts may not be too hard to find, but are sold for very high prices (hundreds to even thousands of dollars). In this case, it is preferable to hunt for bargain deals, posted by a seller who may not know the shirt's value, or who wants to make a quick sale. However, such item listings are always quickly sold.

Either way, such a hobby involves some time investment, as I need to look online regularly in case a shirt I want, or a good deal, has appeared online. To reduce time spent, I use a narrow search criteria, but this also runs the risk of losing shirts I want that didn't fit into the search criteria (i.e a "false negative"). In an ideal world, I would be able to:

1. Look online as regularly as possible
2. Use a wide search criteria to reduce "false negatives"
3. Quickly filter out what I'm not interested in

In recent history, point 3 would've been impossible due to its subjective nature. However, the recent advent of LLMs, and associated tooling such as image recognition, means that the problem can now be tackled. Thus, this project was built as a proof-of-concept for solving a generalization of this problem.

Solution

Outline

This project uses Vue for the frontend, Rust for the backend, and Supabase for auth + Postgres storage.

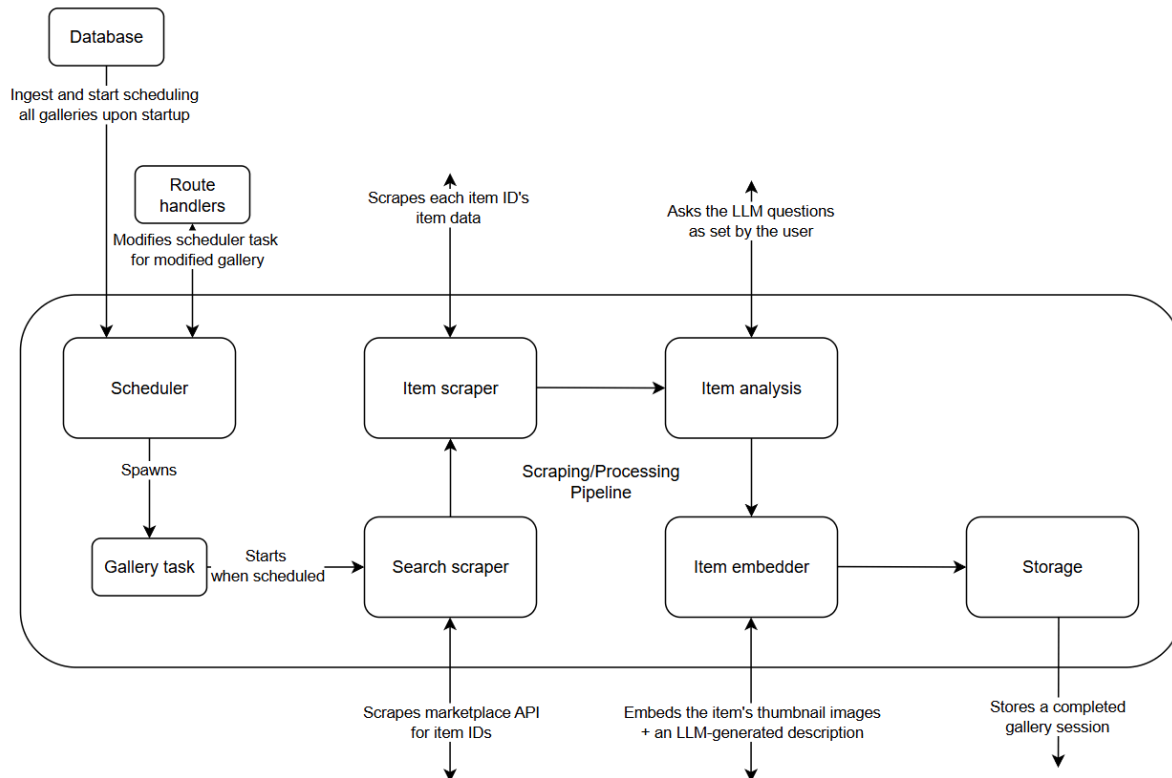
The basic flow of this service is:

- Search criteria
The user sets some search criteria to scrape an online marketplace with (for simplicity, only 1 marketplace, Mercari, is supported).
- Periodicity
The user can set how often the marketplace is to be scraped and processed (for example, every 2 hours).
- Evaluation criteria
The user can set a structured list of questions to ask the LLM about each scraped listing. For example, in my case, it could be *"Is this a vintage shirt?"*. Several types of questions can be asked, where the corresponding answer can be parsed appropriately:
 - Yes/No: A strict yes/no question
 - Yes/No/Uncertain: A yes/no question, but skippable if not determinable
 - Int: An integer question, such as "Which year is the media of this shirt from?"
 - Float: A float question, such as "Rate the shirt from 0.0 to 10.0"
 - Open-ended: An open-ended question, where any answer is allowed

Certain questions can then be chosen as a "hard filter", to quickly filter out objectively unwanted shirts. For example, for *"Is this a vintage shirt?"*, I could choose "Yes" as a hard filter, meaning all items to which the LLM responded "No" would not be shown to me. Thus, a user can use the LLM to accurately apply their subjective reasoning to a wide search criteria.

- Running the "gallery"
The above details are used to create a "gallery", which represents all item listings scraped under it. Scraping and processing of item listings occurs according to the chosen periodicity, after which the curated items can be viewed as a "gallery session".

The structure of the backend is fairly simple. Database migrations and queries are written using the Diesel ORM. Data stores abstract over the database and provide interfaces for accessing/storing data across different domains, like galleries/gallery sessions/items. The Axum framework is used for composing routes and handlers. Gallery-modifying operations, such as add/update/delete, are also sent to the scraping/processing pipeline.



The pipeline handles the scraping and processing for each gallery.

It begins at the scheduler, which is populated with all galleries at startup. A task is spawned for each gallery, which repeatedly triggers the pipeline according to the user-set periodicity.

The search scraper scrapes the marketplace's search using the user-set search criteria, getting the IDs of all items created after the gallery's last-ran time. This is passed to the item scraper, which scrapes the actual item data for each ID.


The item analyzer handles querying of the LLM, using the user-set evaluation criteria. In addition, the item analyzer also asks for a "brief description" of the item, to be used in the item embedder later.

The item embedder handles embedding of the item listing's thumbnail + its brief description. The embedding is done by a separate Python service running a CLIP model. The intent is that item listings advertising the same item can be classified together using these 2 embeddings, allowing users to see availability and pricing of items as the gallery grows. However, due to time constraints, this feature hasn't been built yet, so the embeddings aren't being used yet.

Finally, the storage module handles creation of a new gallery session and storing of the items, which can then be viewed by the user. The gallery's last-ran time is updated in the scheduler, allowing it to only scrape new items on its next run.

Frontend

Once logged in, the first page a user sees is the gallery dashboard. It contains summary information for each gallery the user has created. The user can also create a new gallery from here:


track what you're looking for

Galleries Dashboard

Create New Gallery

NAME	GALLERY ID	TOTAL SESSIONS	TOTAL ITEMS	LAST SCRAPED
test	2deb88ac-f289-4f7b-84ae-5dffa11a07302	5	26	4/17/2025, 9:51:07 PM

The “Create New Gallery” button leads to a form for creating a new gallery, where important information such as the periodicity, search criteria, and evaluation criteria must be filled in:

Create New Gallery

Details
Name

Scraping Periodicity (as Cron string)

Search Criteria
Keywords

Excluded Keywords

Minimum Price

Maximum Price

Evaluation Criteria
Question

Criterion Type

Hard Criterion (optional)

Add Criterion

Create GalleryCancel

The evaluation criteria section allows the user to enter as many as they require. The type of criterion can only be chosen. Optionally, they can also set the criterion to be “hard”, meaning it must be fulfilled or the item listing will be filtered out. For example:

Evaluation Criteria

Question

Is this a vintage shirt?

Criterion Type

YesNo

Hard Criterion (optional)

Yes

Hitting the “*Create Gallery*” button registers the gallery, which can now be accessed by clicking on it from the dashboard. The gallery page shows basic statistics about the gallery, and lists each scraping session for it:

← Back to Dashboard

test

Delete Gallery

Basic Information

ID: 2deb88ac-f289-4f7b-84ae-5dff11a07302
Status: Active
Cron: */3 * * * *
Schedule:
Last Scraped: 4/17/2025, 1:51:00 PM

Search Criteria

Keywords: 00s cospa shirt
Excluded: -
Keywords:
Price Range: - - -

Evaluation Criteria

Question	Type	Hard Criterion
is this a vintage shirt?	YesNo	No

Gallery Sessions

Session ID	Created	Total Items
63	4/17/2025, 1:27:56 AM	17
64	4/17/2025, 1:39:01 AM	0
65	4/17/2025, 2:24:05 AM	1
66	4/17/2025, 2:39:06 AM	1
67	4/17/2025, 9:51:07 PM	7

Clicking on a gallery session brings the user to its page, where all its items can be seen.

Session Details

[← Back to Gallery](#)

Gallery ID: 2deb88ac-f289-4f7b-84ae-5dff11a07302

Created: 4/17/2025, 1:27:56 AM

Total Items: 17

Marketplace Items

Image	ID	Name	Description	Price
	m17997193867	【レア古着】寺田てら ジャンクモール ニコ 漫画キャラシヤツ 黒L Ado	Black graphic t-shirt featuring anime-style character illustration by Terada Tera, JUNK MALL/Nico character design	¥8,880
	m75993970606	00s COSPA ヴィンテージアニメTシャツ 黒T エヴァンゲリオン	COSPA Evangelion anime black t-shirt with red geometric warning design	¥5,550
	m93353407545	ああっ女神様っ 90s 00s アニメtシャツ vintage cospa	Ah! My Goddess anime t-shirt featuring globe design with character prints	¥5,999
	m37513257524	【良品】00s COSPA ひぐらしのなく頃に プリントTシャツ Lサイズ	Higurashi no Naku Koro ni anime t-shirt by COSPA, black with orange text design	¥11,000
				

Roadmap

Outline

Due to time constraints, only the minimal features for a proof-of-concept were built. Below are a few features which are planned for the future.

Item classification

Embeddings of the item thumbnail and an LLM-generated description can be used to compare similarity with other items in the user's gallery. A formula can be used to determine if the embeddings are sufficiently close to be the same item, and a new "item group" can be generated. Thus, as the gallery grows over time, the user can see pricing and availability trends of different items.

"Irrelevant" item storage

Despite the accuracy of LLMs today, it is still possible that items are filtered out during item analysis, which the user may actually want. Thus, these filtered-out items should also be stored in a separate section for each gallery session, for users to occasionally vet through.

Deployment

The project has not been prepared for deployment on the cloud. A CI/CD build + Docker pipeline is currently being worked on.