Task completion:

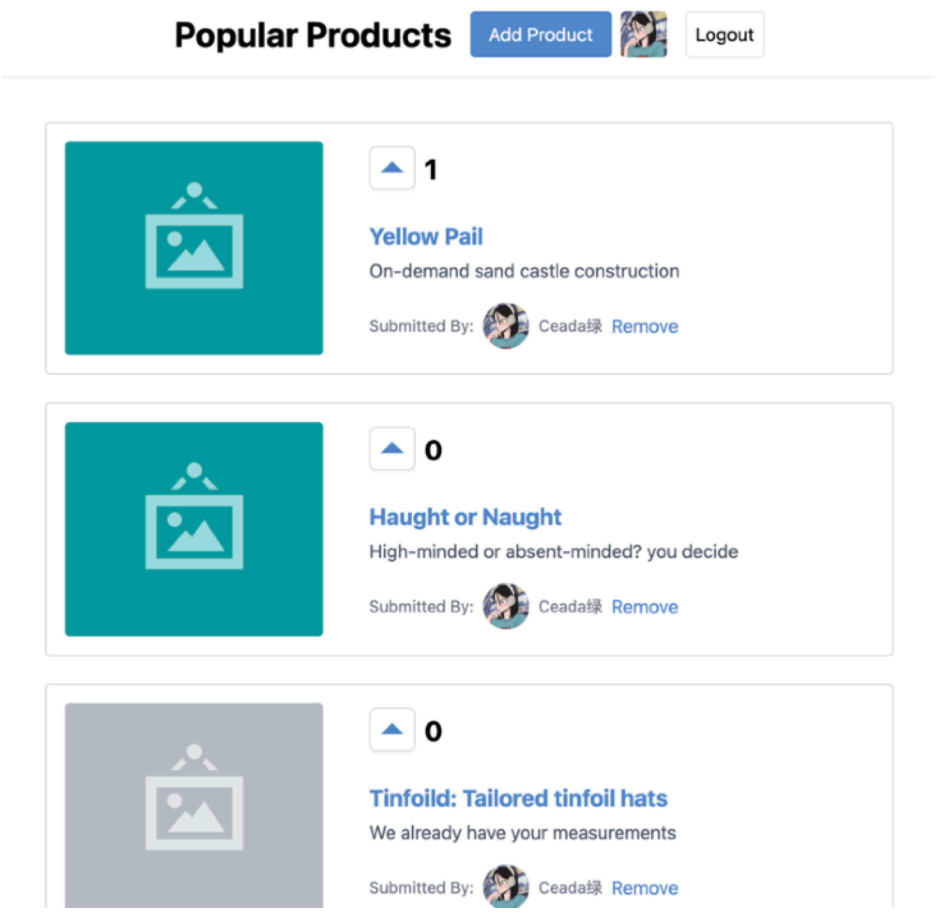
I completed the implementation of both front-end and back-end, established the database, deployed it online, and integrated lint and unit tests. The following is a detailed document introduction, including product introduction, how to install, technical selection considerations, and how many hours I spent on this exercise.

Product Introduction:

Access URL:https://yu-tian-woven.vercel.app/

This is a popular product voting list that allows you to log in using your GitHub account.
Administrators have the ability to add and remove products.
Both administrators and regular users are able to cast votes.
The product list will be sorted by the number of votes from high to low.

Administrator's Perspective: (with add product and remove)

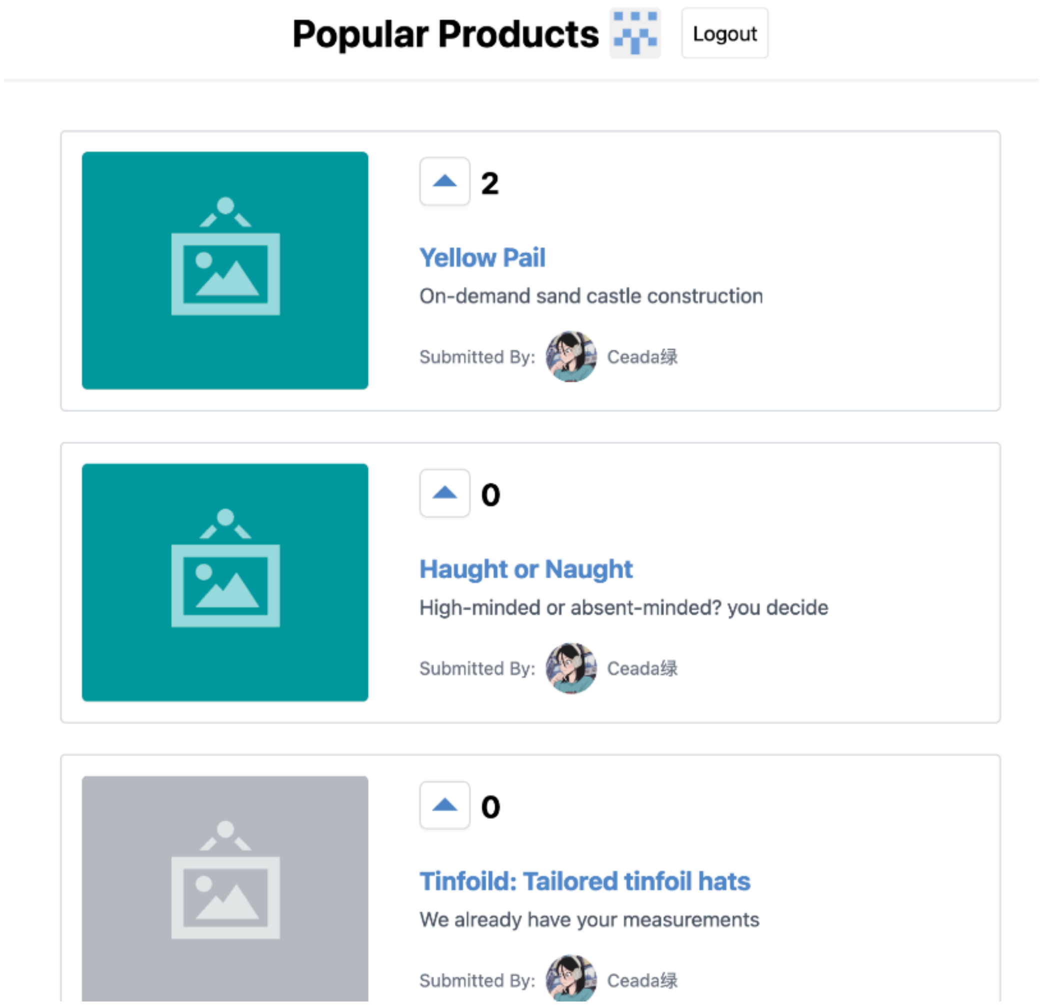


1
Yellow Pail
On-demand sand castle construction
Submitted By: Ceada绿 Remove

0
Haught or Naught
High-minded or absent-minded? you decide
Submitted By: Ceada绿 Remove

0
Tinfoild: Tailored tinfoil hats
We already have your measurements
Submitted By: Ceada绿 Remove

Common user's Perspective:

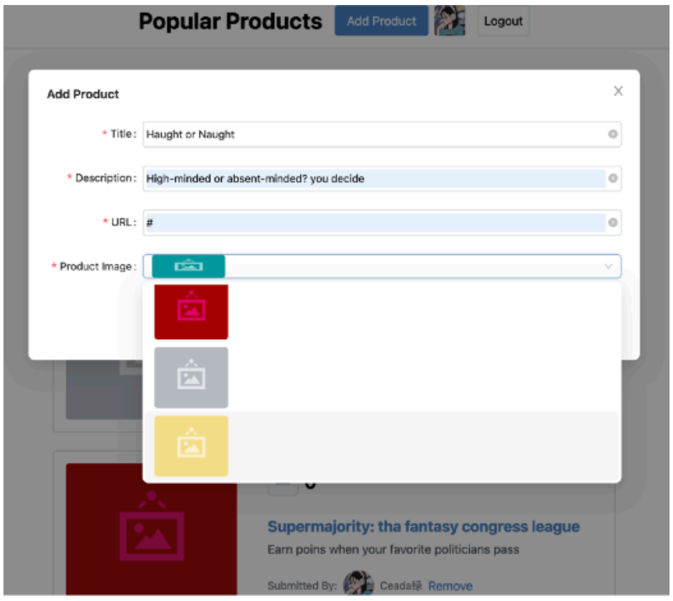

Popular Products
Logout

2
Yellow Pail
On-demand sand castle construction
Submitted By: Ceada绿

0
Haught or Naught
High-minded or absent-minded? you decide
Submitted By: Ceada绿

0
Tinfoild: Tailored tinfoil hats
We already have your measurements
Submitted By: Ceada绿

Administrators have the capability to publish products and edits to the product title, description, image, and product URL

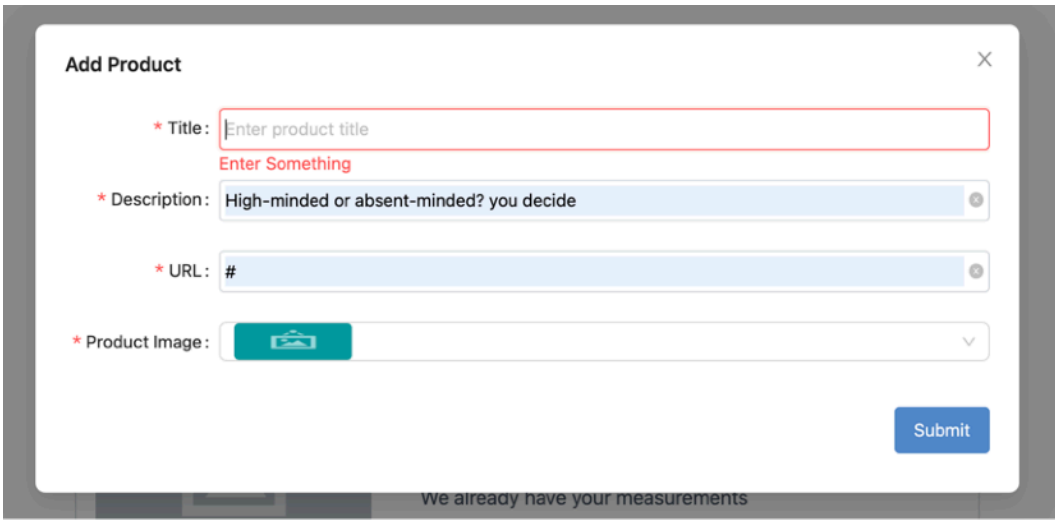

Popular Products
Add Product
Logout

Add Product

* Title:	Haught or Naught
* Description:	High-minded or absent-minded? you decide
* URL:	#
* Product Image:	

Supermajority: tha fantasy congress league
Earn poins when your favorite politicians pass
Submitted By: Ceada绿 Remove

During the editing process, the validity of the parameters for the administrator's input will be verified.



Add Product

* Title: Enter product title

Enter Something

* Description: High-minded or absent-minded? you decide

* URL: #

* Product Image:

Technique stack

Language	Typescript
framework	Next.js
component library	ant design
styling	Tailwind
Database	Upstash
Auth	next-auth
data fetching	swr
Unit test	Jest
Lint	Eslint

How to Install

1. Set up project

Unzip the Yu.Tian file, open your terminal, and enter the Yu.Tian folder, install all the dependencies.

You can use “npm install” to install all required dependencies, but it is recommended to use "yarn" instead for faster installation and consistency.

2. Set your environment variables

To set up your environment variables, follow these steps:

1. Copy the `.env.local.example` file to `.env.local`

   enter this in the terminal： `cp .env.local.example .env.local`

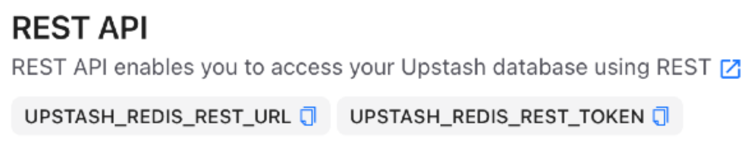
2. The `.env.local.example` file contains a description of all the environment variables. Here is a brief overview of each variable:

```
# A URL for the Redis REST API
UPSTASH_REDIS_REST_URL=

# Redis REST API token that verifies that the sender of the request has access to the Redis
REST API.
UPSTASH_REDIS_REST_TOKEN=

# Authentication ID of GitHub
GITHUB_ID=
# Authentication secret of GitHub
GITHUB_SECRET=

# A list of administrator github email, separated by ','
NEXT_PUBLIC_ADMIN_EMAILS=ceadatian@gmail.com

# Secret key of NextAuth
# generate one here: https://generate-secret.vercel.app/32
NEXTAUTH_SECRET=

# Mandatory next-auth URL for localhost
NEXTAUTH_URL=http://localhost:3000
```

3. Register database

To access Upstash, log in to the Upstash Console using your GitHub account.

Create a new database and retrieve the UPSTASH_REDIS_REST_URL and UPSTASH_REDIS_REST_TOKEN. Here is it:

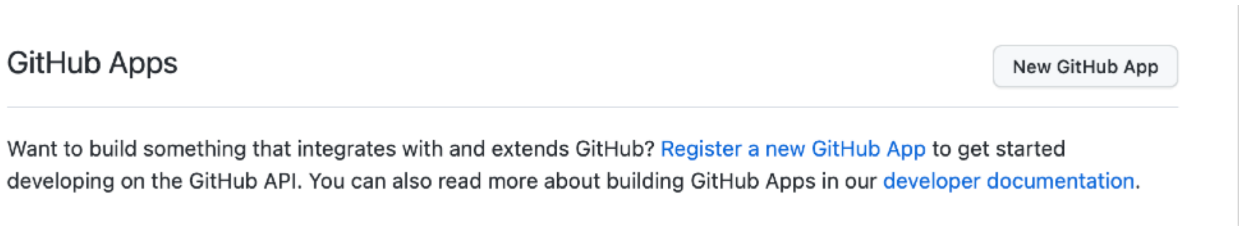

REST API

REST API enables you to access your Upstash database using REST

4. Set up user authentication by next-auth

To set up GitHub OAuth using the next-auth library, follow these steps:

1. Navigate to the GitHub Developer Settings on GitHub.
2. Click on "New GitHub App". See the image below:

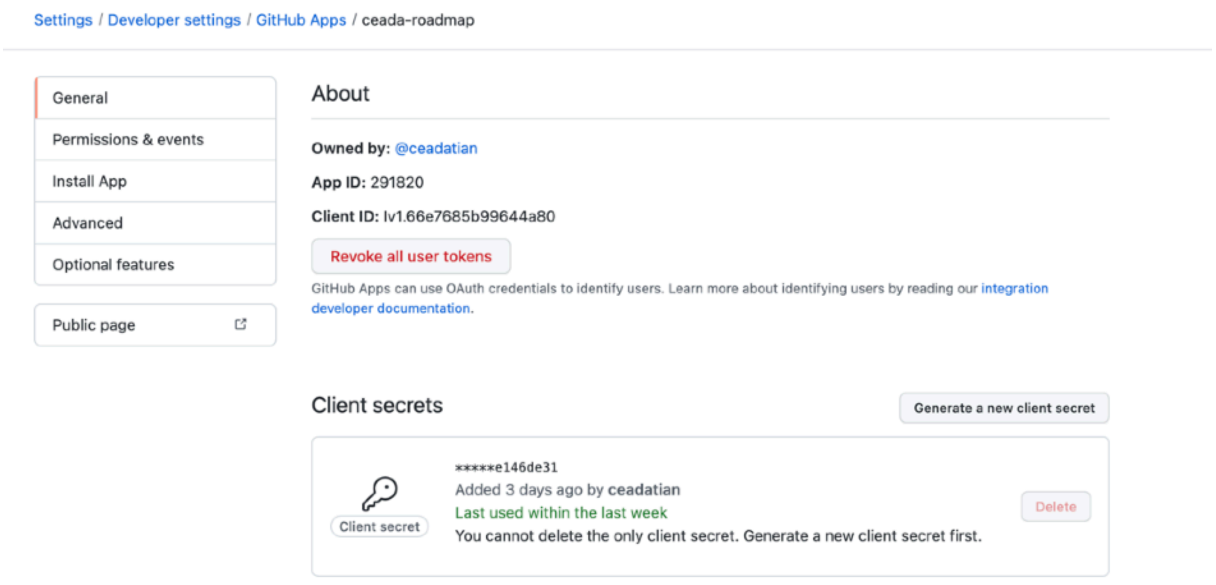


GitHub Apps

New GitHub App

Want to build something that integrates with and extends GitHub? Register a new GitHub App to get started developing on the GitHub API. You can also read more about building GitHub Apps in our developer documentation.

3. Name your "GitHub App name"
4. Set the "Homepage URL" (or just placeholder, if you don't have a website yet).
5. Add the "Callback URL", and put **http://localhost:3000**. The "Callback URL" is crucial as it is where the user will be redirected after logging in with their GitHub account. If deploying the website online, don't forget to update the "Callback URL" to the online domain. In my case, I changed it to https://yu-tian-woven.vercel.app/
6. Generate a new client secret by clicking on the button after successfully creating your app. Show in the following image:



Settings / Developer settings / GitHub Apps / ceada-roadmap

General
Permissions & events
Install App
Advanced
Optional features
Public page

About

Owned by: @ceadatian

App ID: 291820

Client ID: Iv1.66e7685b99644a80

Revoke all user tokens

GitHub Apps can use OAuth credentials to identify users. Learn more about identifying users by reading our integration developer documentation.

Client secrets

Generate a new client secret

Client secret

*****e146de31
Added 3 days ago by ceadatian
Last used within the last week
You cannot delete the only client secret. Generate a new client secret first.

Delete

7. Copy the client secret you generated and paste it under the `GITHUB_SECRET`value in your .env file
8. Copy the Client ID and paste it under the `GITHUB_ID` value in your .env file

5. Run Your Project

1. To run the project locally, open the project folder and run either `` `npm run dev` `` or `` `yarn dev` ``.
2. To run the lint and automatically fix lint errors, use either `` `npm run lint` `` or `` `yarn lint` ``. To fix lint errors, use either `` `npm run lint:fix ` `` or `` `yarn lint:fix. ` ``
3. To build the project, run either `` `npm run build` `` or `` `yarn build` ``

6. Add Administrators

Only administrators can add and remove products. To specify the administrator's Github email, set the environment variable `NEXT_PUBLIC_ADMIN_EMAILS` with a comma-separated list of emails.

7. Deploy to Vercel

To deploy the project to Vercel, first create a new Github repository and push your local changes. Then, follow the instructions in Vercel's documentation for Deploying a Git Repository. Make sure to add all the necessary Environment Variables.

Technical selection considerations

1. Tailwind CSS (https://tailwindcss.com/)

Tailwind CSS is an advanced CSS framework that utilizes a set of CSS classes to define styles. I chose it for the following reasons:

- Atomic CSS: Tailwind adopts a highly granular and atomic approach to styling, enabling maximum customization and versatility.
- Design System: Tailwind offers a consistent design system, ensuring a uniform look and reducing the need to spend time thinking about styling and design.

In conclusion, Tailwind CSS is a robust and flexible CSS framework that enables fast and effortless application development.

2. Database: Upstash

Upstash provides a serverless architecture for managing and maintaining all your database services. With Upstash, you can easily integrate its SDK into your project, configure the request URL in your environment variables, and you're ready to go.

All server management and maintenance work is completed by Upstash.
If I use a traditional way like express + MySQL, I need to write Github Action deployment, write my own container orchestration solution and finally ship it to a container service. So upstash is much more convenient and let me focus on the main logic of product itself.

Serverless databases like Upstash boast strong performance as evidenced by articles like "One Million Queries Per Second with MySQL – PlanetScale (Serverless MySQL)" at https://planetscale.com/blog/one-million-queries-per-second-with-mysql.

3. Fetching data: swr （https://swr.vercel.app/ja）

In my project, I centralized the management of data requests by abstracting the API and data layers in the store/index file. I chose swr as my data-fetching library for many advantages.

swr provides faster updates by serving stale data while fetching fresh data in the background. It also ensures that users have a seamless experience, even when offline, by automatically caching the data. Additionally, SWR revalidates the data whenever it becomes stale, keeping the user's information up-to-date.

swr is also lightweight and efficient, with a small size and a simple, intuitive API that is easy to use, even for developers new to data-fetching libraries.

4. full-stack framework： next.js

Next.js is a full-stack JavaScript framework. I chose Next.js as it provides a better way of organizing front-end and back-end code compared to traditional front-end and back-end separation development methods.

It combines the benefits of both front-end UI frameworks and back-end Node.js server frameworks and provides a streamlined solution for building full-stack applications.

With Next.js, you can enjoy server-side rendering, built-in optimizations, and a simple and intuitive API. This allows developers to focus on building their applications in a very fast and easy way.

In Next.js, the pages directory acts as the application's router, automatically creating a route for each .tsx file. The file name is the route path, for instance, pages/index.tsx is the root path of the application, that is "/". The _app.tsx file in Next.js is a special file that is used to customize the overall behavior of the application, serving as a wrapper for all pages and providing common functionality.

When reviewing the code, be sure to pay attention to both pages/index.tsx and pages/_app.tsx.

Costing Time:

Technique stack selection: 6 hours

Setting up the database and implementing both front-end and back-end: 12 hours

Integrating lint and unit tests, deploying online: 6 hours