**Coding Challenge**

Description

Most of us are familiar with seeing URLs like bit.ly or t.co on our Twitter or Facebook feeds. These are examples of shortened URLs, which are a short alias or pointer to a longer page link. For example, I can send you the shortened URL http://ic.click/hnN58bl that will forward you to a very long CBC URL Young Canadian Olympians poised to lead the way at Milan-Cortina in 2026 | CBC Sports.

Mandatory Requirements

* Design and implement an API for short URL creation.
* Implement forwarding of short URLs to the original ones. It means, if I open the short URL I should be redirected to the original one.
* There should be some form of persistent storage.

Bonus Requirements

* The application should be distributed as one or more Docker images
* Design and implement an API for gathering different statistics

My thought process

For simplicity sake, I will implement all APIs into one single instance where functionality:

* Create, Read, Delete of shortened URL
* Administration for URL CRD
* Public viewer able of sample short URL and redirect
* Use of simple file storage for persistent storage instead of database
* Implement a simple click count for different URL for statistics gathering, although there could be more.
* No authentication or security tokens will be implemented for simplicity sake. Only SSL is implemented.

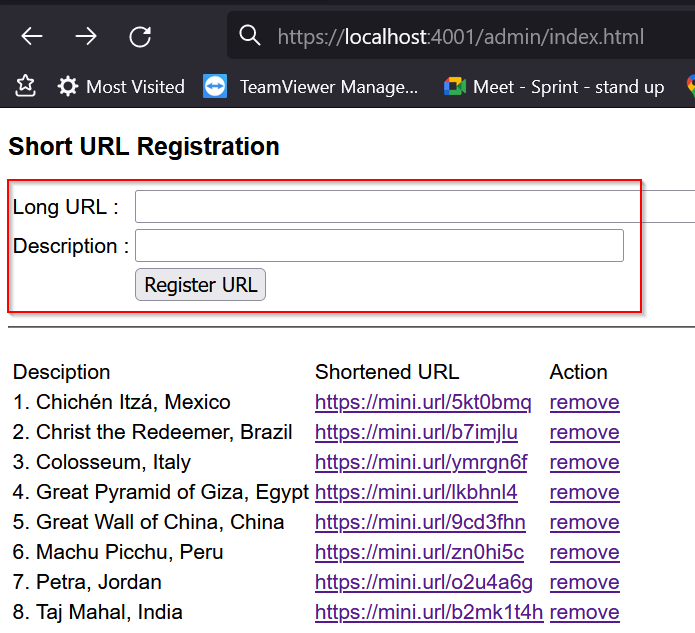
In actual production, this will be broken up into different API instances for security. This project will be implemented as a Restful API with no state management.

Since this is a code dev project, I will use localhost as the API endpoint. In actual production, the endpoint will be hosted in the cloud.

Design concept

A paired key of short and original URL will be store on disk. A description added to help identify them. The short URL is generate by random string of predetermine length (here is 7 characters and can be customized). The URL prefix is also predetermine in a config file settings. Here we use "https://mini.url/".

The first step is to generate and persist the URL by using the admin page.



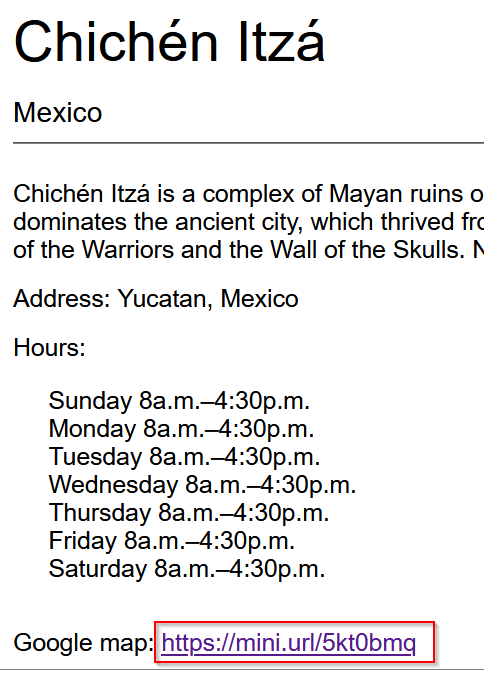
URL can be removed by clicking on the 'remove' link next to the shortened URL.

Once the shortened URL is generated, the URL can be placed on a HTML page or send in an email as a hyper link. The link will be directed to the API passing in the short URL as key. If the key is registered with the API and valid, the API will retrieve the original URL and issues a 301 redirect to the client.

On the email, the hyper link will look like this:

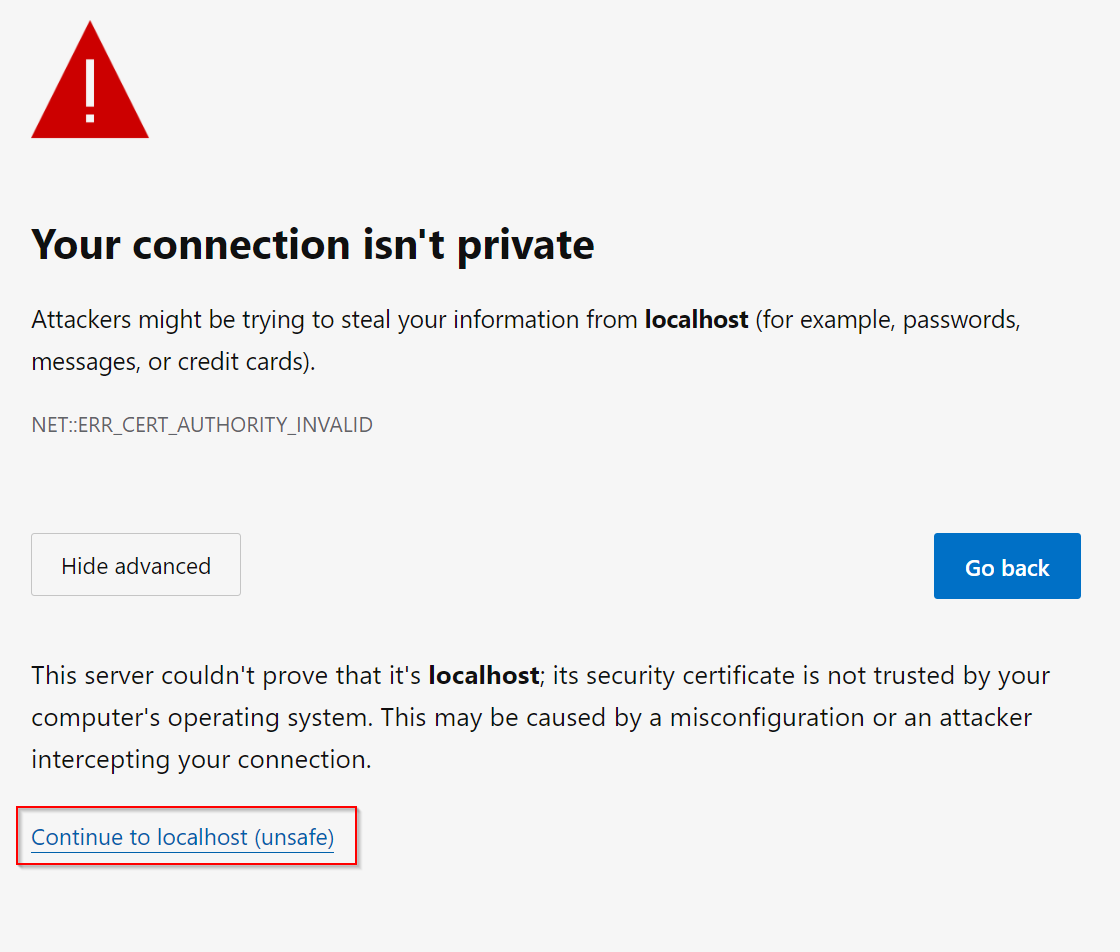
<a href="https://localhost:4001/getURL?id=https://mini.url/b2mk1t4h">https://mini.url/b2mk1t4h</a>

On the web page, it will look like this:



Implementation

* Using self signed SSL, API will be SSL enabled. However, running it locally will need to bypass browser security prompt when running for the first time.



* Using port 4001 on localhost.
* Will not be using non-secure protocol (http).
* Data stored will be simple non-encrypted JSON file. (In actual production, this could be encrypted)

Assumption

As this is a dev environment, I'm using localhost and a single API endpoint for serving admin, public API calls. In actual production system, these will most likely be different endpoint.

Running locally

2 ways of running this locally:

* Run from local source

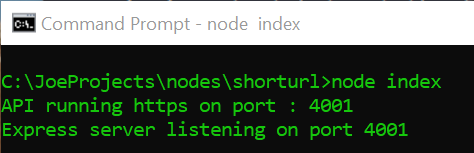
1. Clone the project from my github repo : <https://github.com/joewkm/shorturl>

2. Install all dependencies packages using npm install <package name>

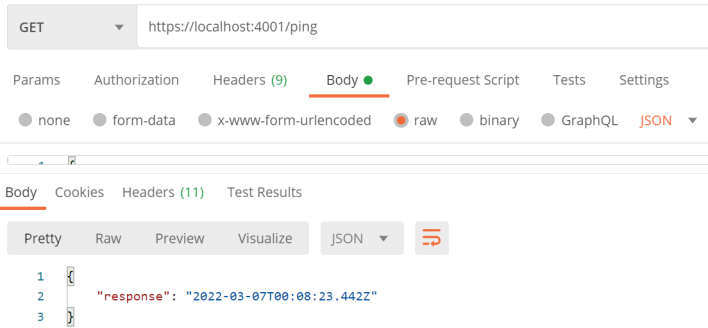
3. Navigate to the root of the project in a command prompt/terminal and start the server with the following command:

*node index.js*

You will see the following screen :

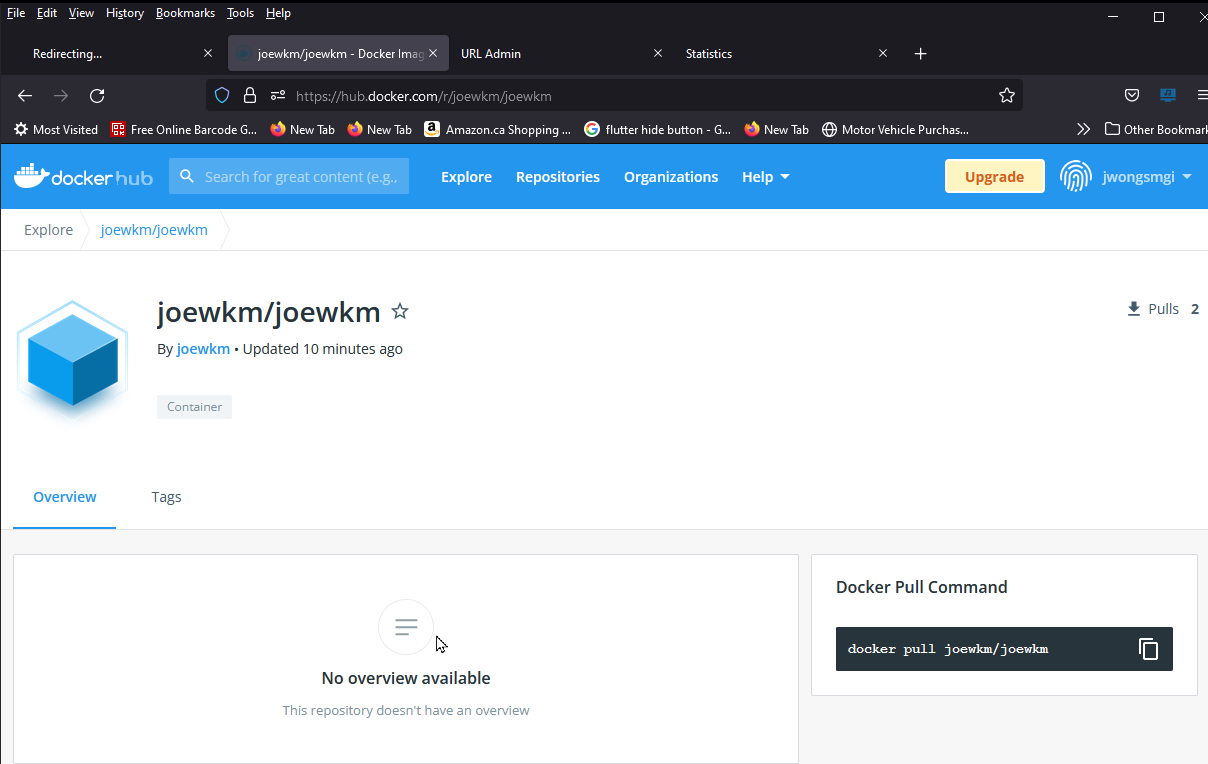


4. When you see the above, API server is up and running. Using postman you can check the health of the API by pinging it and getting the following response.



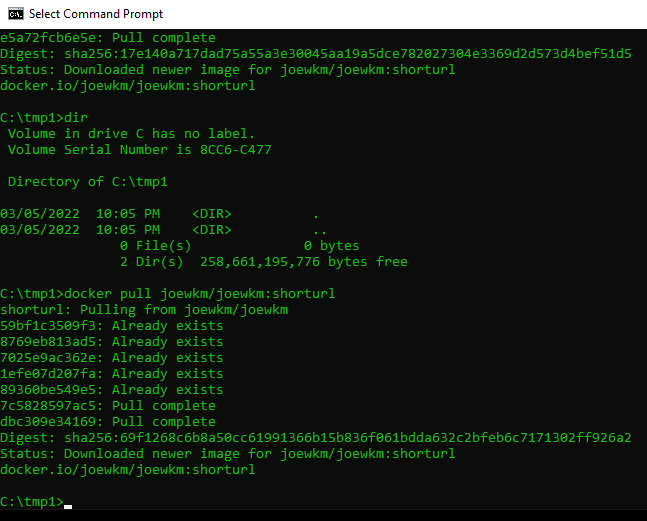
* Run from local docker image

Pull Docker image from Docker hub (note : docker desktop must be installed)

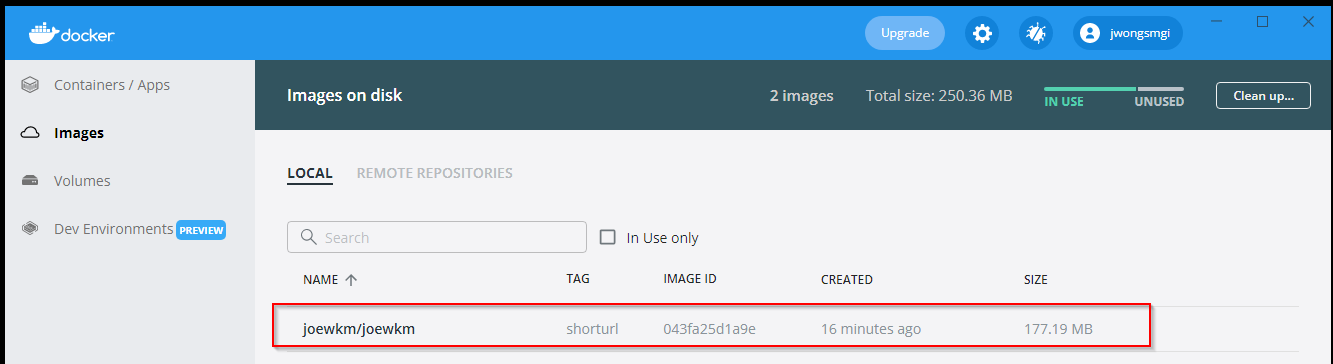


1. From command line/terminal:

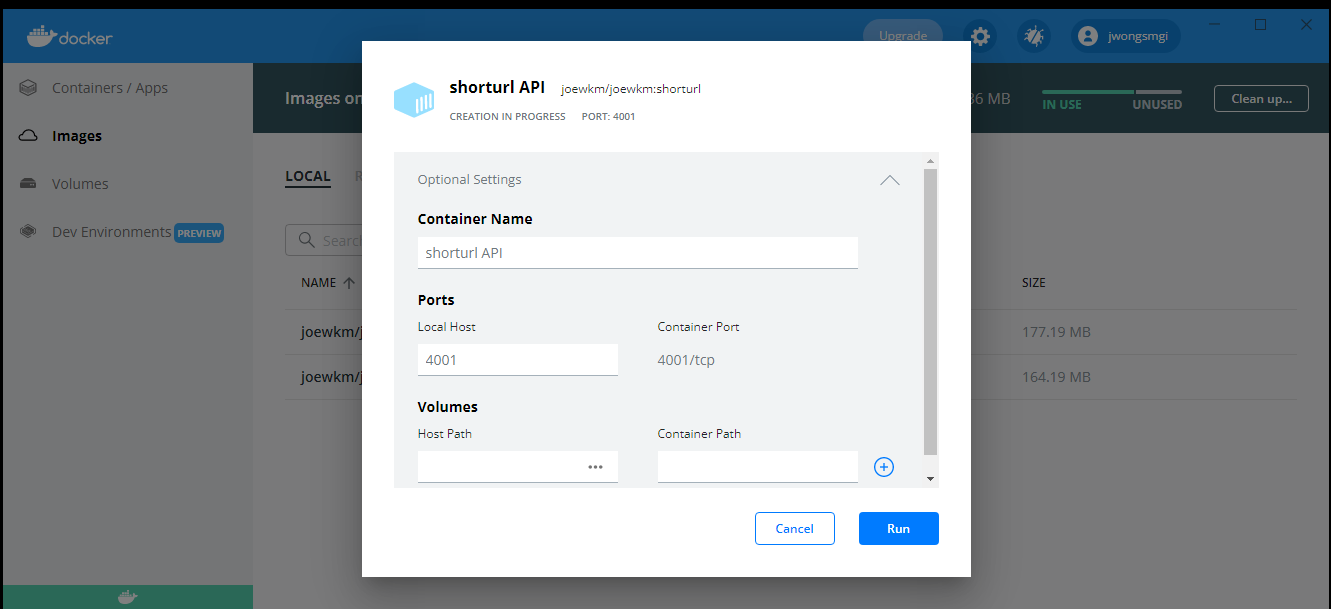
*docker pull joewkm/joewkm/shorturl*



2. Once the image is pulled, you will see a new docker image in the desktop docker container.

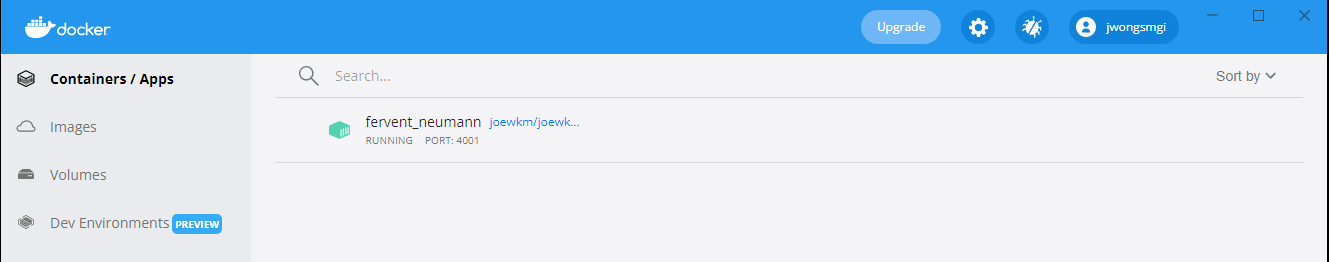


3. Start the docker image by clicking on the 'run' button. Enter the port 4001 in the optional settings.

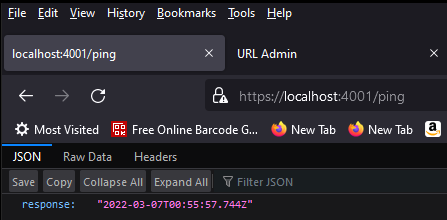


*Note: Very important to enter the port 4001 in the above dialogue box or you wouldn't be able to access the API services.*

4. Once the docker is started, you should see the following:



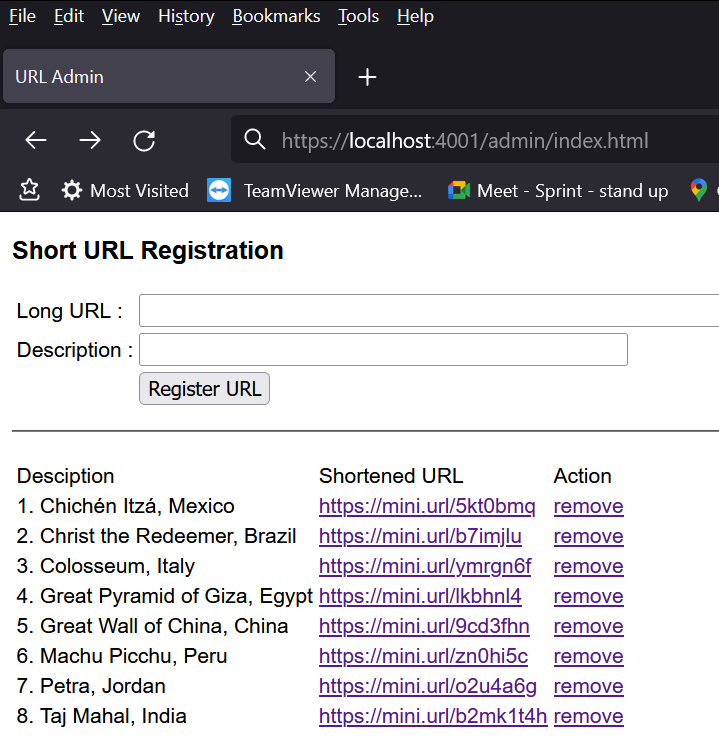
5. Now you can test if the API is up and running by typing the following URL in a browser on a browser and will get the following response:



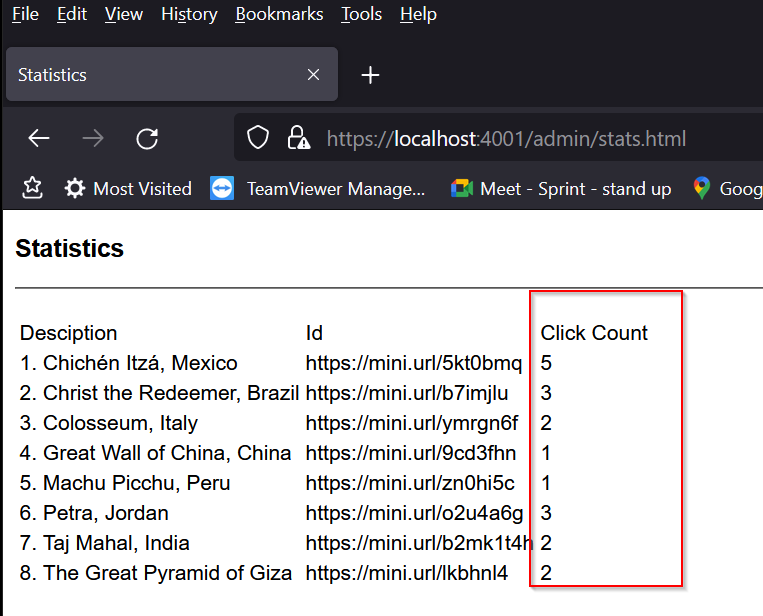
With either one of the above 2 steps completed (but not both), You should be able to navigate to the following on the browser.

For administrator:

1. Admin page => https://localhost:4001/admin/index.html



2. Statistic page => https://localhost:4001/admin/stats.html



For public viewer

1. Sample page => https:local.host:4001/public/index.html



The clickable short URL is highlighted above will call the API endpoint and get the original URL before redirecting the page to the URL.

**Welcome to the Wonders of the Ancient World!**

Local version:

https://localhost:4001/public

https://localhost:4001/admin

https://localhost:4001/admin/states.html

Cloud version :

A cloud version of the above can be found on

https://shorturl-c3bb9.web.app/public

https://shorturl-c3bb9.web.app/admin

https://shorturl-c3bb9.web.app/admin/stats.html

Which is hosted on firebase hosting.

Note in order for the cloud version to be running, the local API must be running either on Docker image or local source.

shortenedurl

Designed and coded by :

Joe Wong

Full Stack Developer

https://github.com/joewkm

https://hub.docker.com/u/joewkm