Networking Fundamentals

Assignment 2

Lucky M. Kispotta

luckymk.mcs2024@cmi.ac.in

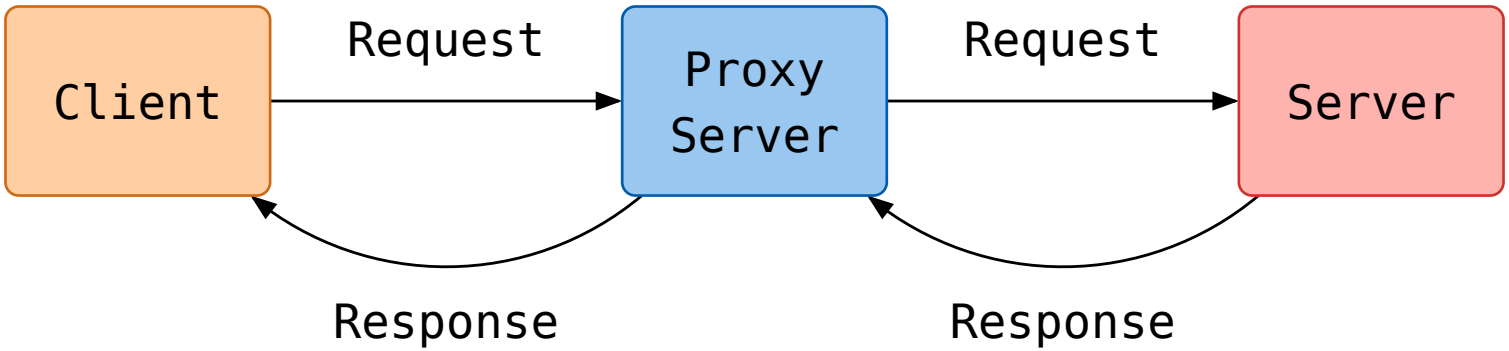
Chennai Mathematical Institute

2025-04-23

Table of Contents

Content

Objective


Client
Request
Proxy Server
Request
Server
Response
Response

Content

Client

A system or node in the network which is able to retrieve information from the network.

Proxy Server

An intermediate server which acts as a intermediary between the client and the server.

Server

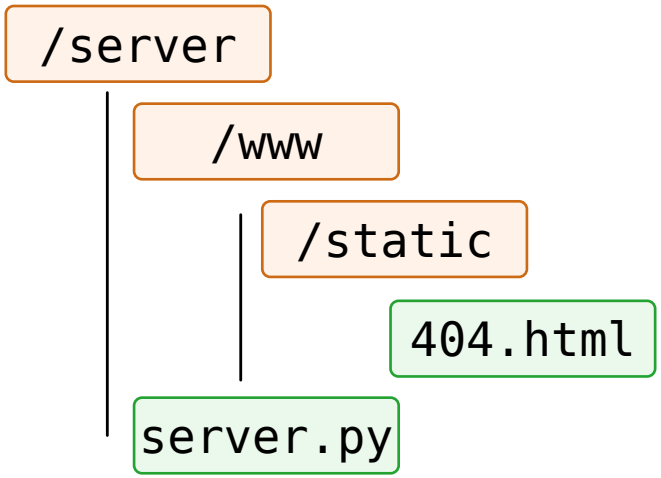
A perpetually running system which serves the client.

Content

Features :

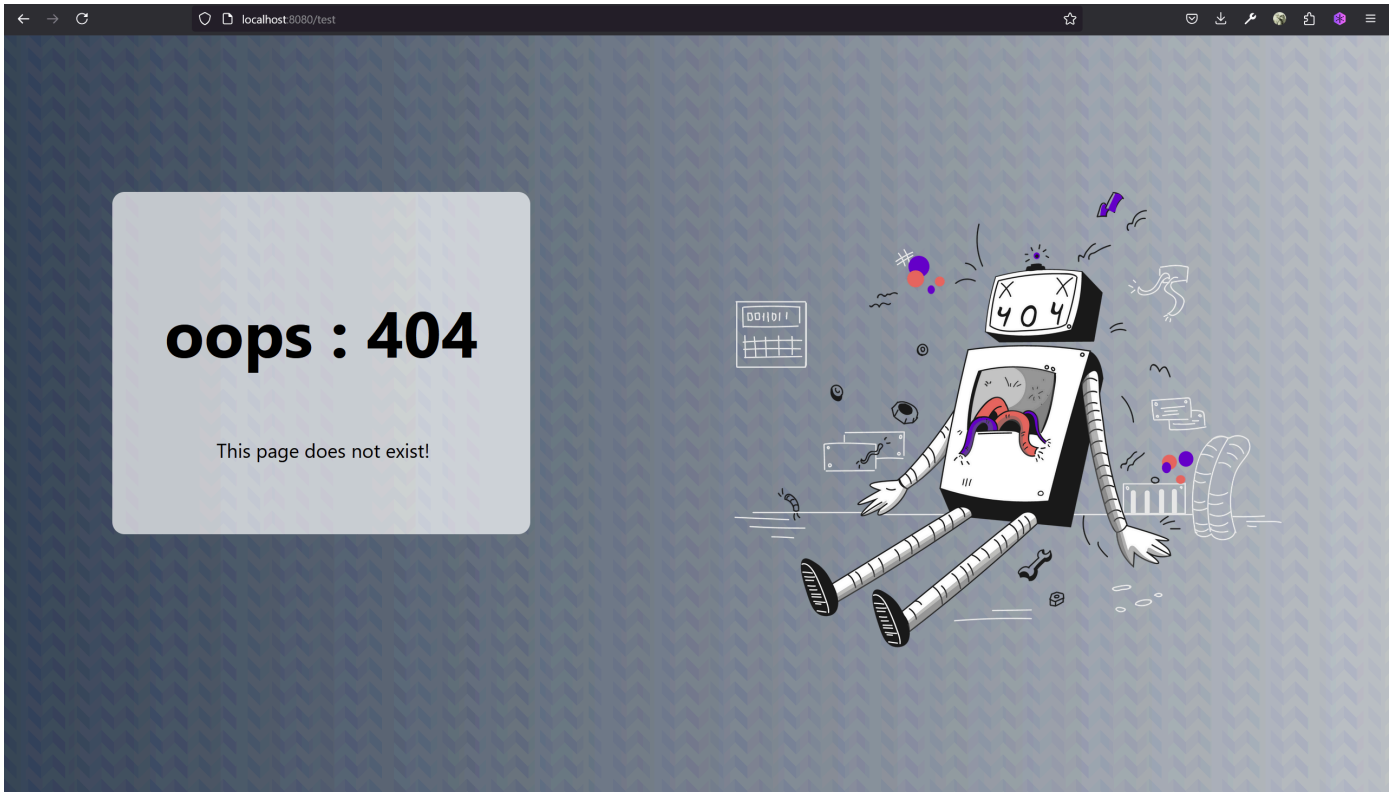
- Written in Python.
- Serve pages to the client.
- Return a **404** page when the requested page in missing.

File structure of the server :



404 page

The server returns this page if the requested page is not present.



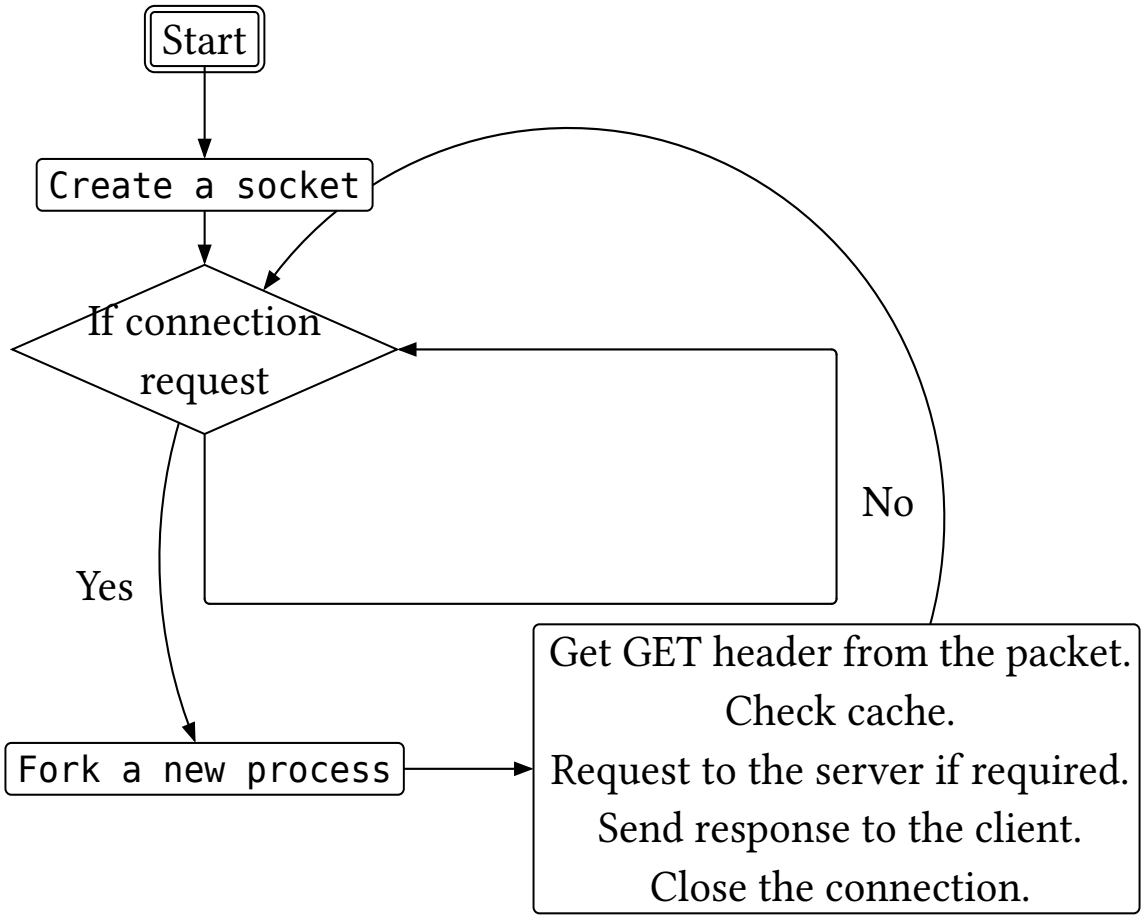
Content

Features :

- Written in C++.
- Handle client requests.
- Cache every page requested.
- `proxy <PORT NUMBER>` needs to be executed.
- For every connection request checks cache.
- Uses functions declared in **socketp.h**.

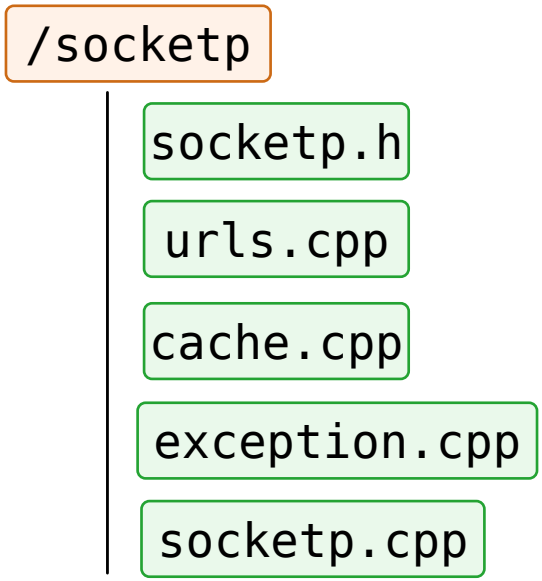
proxy.cpp

This contains the functionalities of the proxy server.



Socketp.h

`Socketp.h` contains declarations of various helper functions defined in the below files.



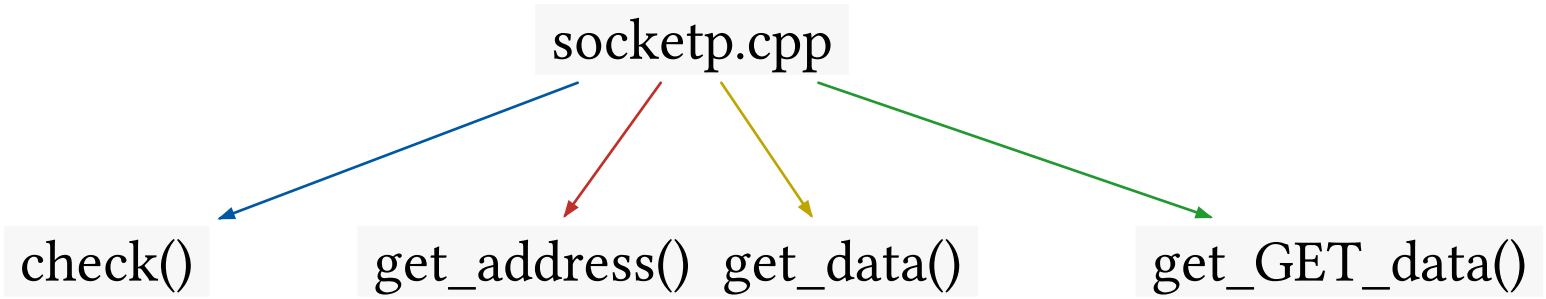
Some helper functions

1. **exception.cpp** - This defines two types of *custom* errors `SocketCreationError` and `CacheError`.

2. **urls.cpp** - Helper class for *URLS*. Takes input as a string containing the url. Parses hostname, port, path from it. [uses `Regex`]

Other two files are defined in the following slides.

socketp.cpp



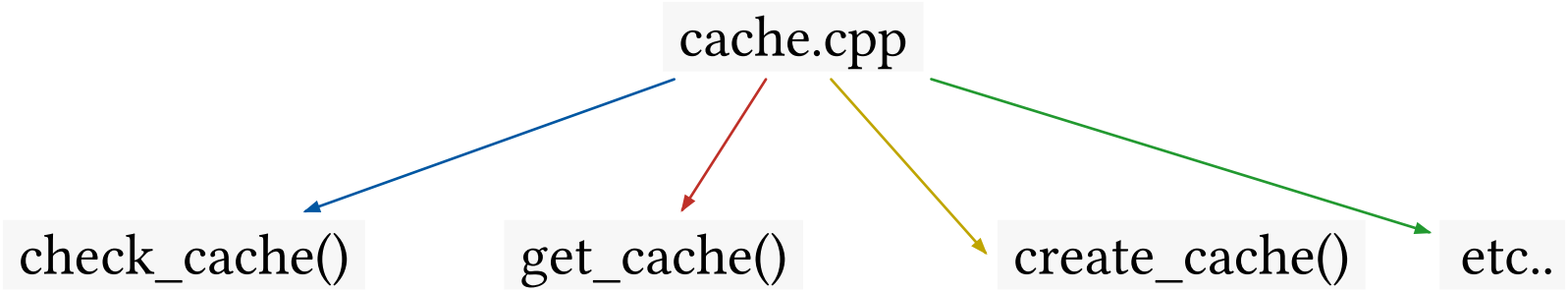
1. **check()** - Takes in return value of a function, `good` message and `bad` message.

2. **get_address()** - Returns an object of `sockaddr_in struct`.

3. **get_data()** - Takes in the contents of the entire packet as char array. Returns the body of the packet.

4. **get_GET_data()** - Takes in the contents of the entire packet as char array. Returns the GET entry of the packet.

cache.cpp

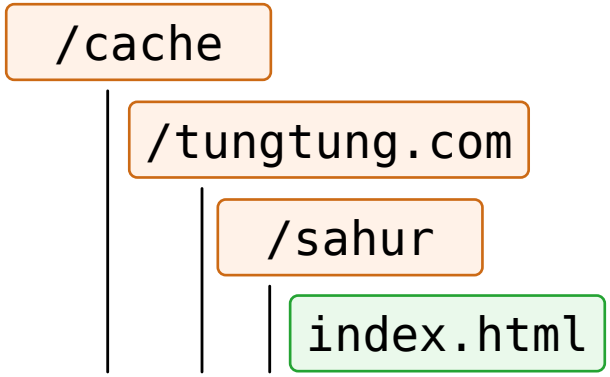


1. **check_cache()** - Checks if a particular page is cached or not.
2. **get_cache()** - Returns the cached page as string.
3. **create_cache()** - Takes in the *url* and *contents* of a page and caches it.
4. **etc..** - Other misc helper functions include `adv_tokenizer` and `recur_dir`.

Content

Caching Scheme

Caching is handled by `cache.cpp`. The caching scheme is very `trivial`.
For every uncached page create a new subfolder in `cache` folder and save the page in that.
For example, if the requested page is `tungtung.com/sahur/index.html`.
Folders, `tungtung.com`, `sahur` will be created and `index.html` will be inside that.



This is a very *dumb* caching scheme.

Thanks for Listening.

Bye Bye