CS-421 Project Assignment 1 Report

Kaan Atakan Aray

21703187

Section 2

November 11, 2021

1 Introduction

This is a report for describing the work done in PA1 of CS-421. How is the program sends a http GET request and how it is handling the response from the server.

2 HTTP Requests

There are 2 types of requests that are sent by the program to the server these are HEAD and GET requests. Also GET has two forms not ranged and ranged. If user enters a range information to the argument a ranged GET request will be used and if user does not specify a range, not ranged GET request will be used.

HTTP requests are in the string form and there are functions which describe their form whose headers are as follows, 'formatted_http_get(cls, file_name, host_addr)', 'formatted_http_partial_get (cls, file_name, host_addr, range)', 'formatted_http_head(cls, file_name, host_addr)' and here cls argument is here because they are class functions, range argument is for the range info provided by the user. file_name and host_addr arguments are self explanatory. HTTP request strings returned from these functions are of the form:

- `"GET /{file_name} HTTP/1.1\r\\nHost:{host_addr}\r\n\r\n"`

- `"HEAD /{file_name} HTTP/1.1\r\nHost:{host_addr}\r\n\r\n"`

There is also another function with the header 'send_http_req(cls, host_addr, http_req, port)' which is used for sending a request to the server and receiving a data from server as a form of http response. To do this, the function creates a socket, then creates a connection, sends the http request to the server, closes the connection and returns the response as a dictionary object by using a helper function 'dictify_response(cls,response)' . An example output (dictionary object) looks like:

- " {'date': 'Wed, 10 Nov 2021 12:12:08 GMT',
  'server': 'Apache/2.4.25 (FreeBSD) OpenSSL/1.0.2u-freebsd PHP/7.4.15',
  'last-modified': 'Mon, 25 Oct 2021 17:48:47 GMT',
  'etag': '"b-5cf30f914cb18"',
  'accept-ranges': 'bytes',
  'content-length': '11',
  'content-type': 'text/plain',
  'http': 'HTTP/1.1 200 OK',
  'body': 'Cras nunc.\n'}"

General flow of the program is a follows, firstly the program starts by asking user the arguments. After that, if the arguments correctly entered, it first downloads the index file and turns it into a list, then downloads the entries of the list and creates files if the download is successful.