

# Assignment 1 Design Document

Alan Caro  
CruzID: alcaro

CSE130, Fall 2019

## 1 Goal

The goal of this program is to implement a single-threaded HTTP server. The server will be capable of handling GET and PUT request sent by a client.

## 2 Assumptions

I am assuming the user will only send one request at a time and that the user will only use the curl command. Also, I am assuming the user will send request with this format:

### PUT:

*curl -T localfile http://localhost:8080 --request-target filename\_27\_character*

### GET:

*curl http://localhost:8080 --request-target filename\_27\_character*

## 3 Design

The general approach I am taking is to check what request and file name the user passed. Then, if the request and file name is valid, I open a socket then process the GET or PUT command accordingly. If there is any errors like an invalid file or request I send the appropriate error code.

## 4 Pseudocode

This is the core pseudocode for the program. Note that it's pseudocode, not C (or Java or Python) code.

```
procedure main
```

```

Declare string hostname
Declare string port

if argc > 1
    hostname = argv[1]
    if argv[2] != NULL
        strcpy(port, argv[2])
    else
        strcpy(port, "80")

    Declare struct addrinfo addrs, hints
    hints.ai_family = AF_INET
    hints.ai_socktype = SOCK_STREAM
    getaddrinfo(hostname, port, &hints, &addrs)
    main_socket = socket(addrs.ai_family, addrs.ai_socktype, addrs.ai_protocol)
    enable ← 1
    setsockopt(main_socket, SOL_SOCKET, SO_REUSEADDR, &enable,
    sizeof(enable))
    bind(main_socket, addr.ai_addr, addr.ai_addrlen)
    listen(main_socket, 16)

    Declare a client_socket

loop
    Declare buffer
    memset(buffer, 0, 1024);
    Declare request
    Declare fileName

    client_socket = accept(main_socket, NULL, NULL);
    recv(client_socket, buffer, 1024, 0);

    sscanf(buffer, "%s %s", request, fileName);
    processOneRequest(fileName, request, client_socket, buffer);

else
    fprintf(stderr, "usage: ./httpserver localhost port")
    return -1
return 0

procedure processOnerequest
if fileName[0] == '/'
    memmove(fileName, fileName + 1, strlen(fileName))

```

```

if isValidName(fileName) == -1
    send(socket, "HTTP/1.1 400 Bad Request", strlen("HTTP/1.1 400 Bad
Request"),0)
    return
if strcmp(request, "GET") == 0
    processGet(fileName, socket)
else if strcmp(request, "PUT") == 0
    line ← strtok(buffer, "\r\n")
    array[7]
    word[20]
    i ← 0
    j ← 0

loop
    array[i++] = line
    line = strtok(NULL, "\r\n")
loop
    if strstr(array[j], "Content-Length: ") != NULL
        break

    if array[j] != NULL
        sscanf(array[j], "%*s, %s", word)
        i = atoi(word)
    else
        i = -1
    processPut(fileName, socket, i)
else
    send(socket, "HTTP/1.1 400 Bad Request", strlen("HTTP/1.1 400 Bad

procedure isValidName
    Declare variable j
    Declare struct path_stat
    loop
        j+= 1
    if j != 27
        return -1
    loop
        c ← fileName[i]
        if isalpha(c)
            continue
        if isdigit(c)
            continue
        if c == '_'

```

```

        continue
if c == '_'
    continue
return -1

return 0

procedure processGet
fd ← open(fileName, O_RDONLY)
buffer[32]

if fd == -1
    if access(fileName, F_OK) == -1
        send(socket, "404 Not Found" strlen("404 Not Found"),0)
    else
        send(socket, "403 Forbidden" strlen("403 Forbidden"),0)
return

```

Declare struct of type stat st

```

if stat(fileName, &st) == 1
    send(socket, "500 Internal Server Error" strlen("500 Internal Server Error"),0)

size ← st.st_size
Declare char array str[1024]
sprintf(str, "HTTP/1.1 200 OK \r\nContent-Length: %d\r\n\r\n", size);
send(socket, str, strlen(str), 0);

loop read(fd,buffer,1)
    send(socket, buffer, 1, 0)

close(fd)

```

```

procedure processPut
fd ← open(fileName, O_CREAT | O_RDWR | O_TRUNC, 0644)

if fd == -1
    send(socket, "403 Forbidden" strlen("403 Forbidden"),0)
    return
i ← 0

loop
    read (socket, buffer, 1)

```

```
    write(fd, buffer, 1)
    i+=1
    send(socket, "HTTP/1.1 201 Created \r\nContent-Length: 0\r\n\r\n",
strlen("HTTP/1.1 201 Created \r\nContent-Length: 0\r\n\r\n"), 0);

close(fd);
```