SIGNUP.C

---------

#include<common.h>

int process\_signup(int csfd, const char \*username, const char \*password) {

FILE \*file = fopen("userfile.txt", "a+");

if (file == NULL) {

log\_action("Error opening users file for signup.");

send(csfd, "Error opening users file.\n", 25, 0);

return 0;

}

char existing\_username[MAX\_USERNAME\_LEN];

char existing\_password[MAX\_PASSWORD\_LEN];

while (fscanf(file, "%s %s", existing\_username, existing\_password) != EOF) {

if (strcmp(existing\_username, username) == 0) {

send(csfd, "Username already exists. Please choose a different one.\n", 58, 0);

fclose(file);

return 0;

}

}

fprintf(file, "%s %s\n", username, password);

fclose(file);

log\_action("New user signed up successfully.");

send(csfd, "Signup successful.\nYou can now log in.\n", 39, 0);

return 1;

}

LOGIN.C

---------

#include<common.h>

int process\_login(int csfd, const char \*username, const char \*password) {

FILE \*file = fopen("userfile.txt", "r");

if (file == NULL) {

log\_action("Error opening users file for login.");

send(csfd, "Error opening users file.\n", 25, 0);

return 0;

}

char existing\_username[MAX\_USERNAME\_LEN];

char existing\_password[MAX\_PASSWORD\_LEN];

while (fscanf(file, "%s %s", existing\_username, existing\_password) != EOF) {

if (strcmp(existing\_username, username) == 0 && strcmp(existing\_password, password) == 0) {

fclose(file);

log\_action("User logged in successfully.");

send(client\_socket, "Login successful.\n", 18, 0);

return 1;

}

}

send(csfd, "Invalid username or password.\n", 30, 0);

log\_action("Invalid login attempt.");

fclose(file);

return 0;

}

PROCESSCDR.C

-------------

#include<common.h>

int process\_cdr\_file() {

FILE \*file = fopen("../data.txt", "r");

if (file == NULL) {

perror("fopen() ");

return;

}

char line[MAXBUFF];

while (fgets(line, MAXBUFF, file)) {

CDRRecord record;

sscanf(line, "%s %s %d %d %d %d", record.msisdn, record.operator, &record.call\_duration,

&record.sms\_count, &record.data\_downloaded, &record.data\_uploaded);

pthread\_mutex\_lock(&mutex);

cdr\_records[cdr\_count++] = record;

pthread\_mutex\_unlock(&mutex);

}

return 1;

fclose(file);

printf("[+]Server: CDR File Processed\n");

}

CUSTOMERBILLING.C

------------------

#include<common.h>

int process\_customer\_billing(int csfd,const char \*msisdn) {

FILE \*file = fopen("../data.txt", "r");

FILE \*cb\_file = fopen("CB.txt", "a");

if (file == NULL || cb\_file == NULL) {

send(csfd, "Error opening data or CB file.\n", 31, 0);

log\_action("Error opening data or CB file.");

return;

}

char line[MAX\_LINE\_LEN];

int found = 0;

while (fgets(line, sizeof(line), file) != NULL) {

if (strstr(line, msisdn)) {

fprintf(cb\_file, "%s", line);

found = 1;

send(csfd, line, strlen(line), 0);

}

}

if (!found) {

// send(csfd, "MSISDN not found.\n", 18, 0);

return 0;

}

fclose(file);

fclose(cb\_file);

}

INTEROPERATORBILLING.C

-----------------------

#include<common.h>

int search\_brand(int csfd, const char \*brand\_name) {

FILE \*file = fopen("../data.txt", "r");

FILE \*iosb\_file = fopen("IOSB.txt", "a");

if (file == NULL || iosb\_file == NULL) {

send(csfd, "Error opening data or IOSB file.\n", 32, 0);

log\_action("Error opening data or IOSB file.");

return;

}

char line[MAX\_LINE\_LEN];

int found = 0;

while (fgets(line, sizeof(line), file) != NULL) {

if (strstr(line, brand\_name)) {

fprintf(iosb\_file, "%s", line);

found = 1;

send(csfd, line, strlen(line), 0);

}

}

if (!found) {

// send(csfd, "Brand not found.\n", 17, 0);

return 0;

}

fclose(file);

fclose(iosb\_file);

}