#define TEMAPROJEKAT "ugradbeni/projekat/youtube"

#include "mbed.h"

#define MQTTCLIENT\_QOS2 0

#include "easy-connect.h"

#include "MQTTNetwork.h"

#include "MQTTmbed.h"

#include "MQTTClient.h"

#include <string.h>

#include <vector>

#include <string>

#include "stm32f413h\_discovery\_ts.h"

#include "stm32f413h\_discovery\_lcd.h"

TS\_StateTypeDef TS\_State = { 0 };

int channelCount = 0;

int usernameLength = 0;

char \*channelLink;

char \*option;

char \*channelUsername;

std::vector<std::string> channelsNames;

void getDefaultValues(){

BSP\_LCD\_Clear(LCD\_COLOR\_WHITE);

BSP\_LCD\_SetTextColor(LCD\_COLOR\_RED);

BSP\_LCD\_FillRect(0, 0, BSP\_LCD\_GetXSize(), 50);

BSP\_LCD\_SetTextColor(LCD\_COLOR\_BLACK);

BSP\_LCD\_SetBackColor(LCD\_COLOR\_RED);

}

void drawChannelNameBanner(){

/\*channelsNames.push\_back("Dnevnjak");

channelsNames.push\_back("MrBeast");

int n = channelsNames[1].length();

char pom[n];

strcpy(pom, channelsNames[1].c\_str());\*/

BSP\_LCD\_FillRect(0, 50, BSP\_LCD\_GetXSize(), 40);

BSP\_LCD\_SetTextColor(LCD\_COLOR\_WHITE);

BSP\_LCD\_SetBackColor(LCD\_COLOR\_BLACK);

BSP\_LCD\_SetFont(&Font12);

BSP\_LCD\_DisplayStringAt(0, 65, (uint8\_t \*)"Channel: ", LEFT\_MODE);

BSP\_LCD\_SetFont(&Font16);

BSP\_LCD\_DisplayStringAt(0, 63, (uint8\_t \*)/\*pom\*/"Drzavni posao", RIGHT\_MODE);

}

void drawYouTubeTriangleLogo(){

Point points[3];

points[0].X = 30; points[0].Y = 15;

points[1].X = 50; points[1].Y = 25;

points[2].X = 30; points[2].Y = 35;

BSP\_LCD\_SetTextColor(LCD\_COLOR\_WHITE);

BSP\_LCD\_FillPolygon(points, 3);

}

void drawAppTitleBanner(){

getDefaultValues();

drawYouTubeTriangleLogo();

BSP\_LCD\_SetTextColor(LCD\_COLOR\_BLACK);

BSP\_LCD\_SetFont(&Font16);

BSP\_LCD\_DisplayStringAt(0, 15, (uint8\_t \*)"YouTube", RIGHT\_MODE);

BSP\_LCD\_SetFont(&Font12);

BSP\_LCD\_DisplayStringAt(0, 30, (uint8\_t \*)"Subscriber Counter", RIGHT\_MODE);

BSP\_LCD\_SetTextColor(LCD\_COLOR\_BLACK);

}

void drawInfoFrame(){

BSP\_LCD\_SetTextColor(LCD\_COLOR\_RED);

BSP\_LCD\_DrawRect(0, 90, BSP\_LCD\_GetXSize()-1, 109);

}

void setChannelInfo(){

drawInfoFrame();

BSP\_LCD\_SetBackColor(LCD\_COLOR\_WHITE);

BSP\_LCD\_SetTextColor(LCD\_COLOR\_BLACK);

BSP\_LCD\_SetFont(&Font16);

BSP\_LCD\_DisplayStringAt(0, 95, (uint8\_t \*)"Channel Number: 1", LEFT\_MODE);

BSP\_LCD\_DisplayStringAt(0, 115, (uint8\_t \*)"Subscribers: 318000", LEFT\_MODE);

BSP\_LCD\_DisplayStringAt(0, 135, (uint8\_t \*)"Views: 572214256", LEFT\_MODE);

BSP\_LCD\_DisplayStringAt(0, 155, (uint8\_t \*)"Uploads: 1615", LEFT\_MODE);

BSP\_LCD\_DisplayStringAt(0, 175, (uint8\_t \*)"Country: Serbia", LEFT\_MODE);

}

void drawInstructionBanner(){

BSP\_LCD\_SetTextColor(LCD\_COLOR\_BLACK);

BSP\_LCD\_FillRect(0, BSP\_LCD\_GetYSize()-40, BSP\_LCD\_GetXSize(), 40);

BSP\_LCD\_SetTextColor(LCD\_COLOR\_GREEN);

BSP\_LCD\_SetBackColor(LCD\_COLOR\_BLACK);

BSP\_LCD\_SetFont(&Font12);

BSP\_LCD\_DisplayStringAt(0, 205, (uint8\_t \*)"Button 1: Next Channel", LEFT\_MODE);

BSP\_LCD\_DisplayStringAt(0, 220, (uint8\_t \*)"Button 2: Previous Channel", LEFT\_MODE);

}

void setLCD(){

BSP\_LCD\_Init();

drawAppTitleBanner();

drawChannelNameBanner();

setChannelInfo();

drawInstructionBanner();

}

void findChannelUsername(char \*linkYT){

char \*username = linkYT + 24;

usernameLength -= 28;

if(username[0] == 'c'){

username = username + 8;

usernameLength -= 8;

printf("ChannelId: %.\*s\n", usernameLength, username);

//ovdje pozvati funkciju za JSON; bez drugog parametra

}

else if(username[0] == 'u'){

username = username + 5;

usernameLength -= 5;

printf("Username: %.\*s\n", usernameLength, username);

//ovdje pozvati funkciju za JSON; drugi parametar = true

}

}

void messageArrived(MQTT::MessageData& md)

{

MQTT::Message &message = md.message;

++channelCount;

option = (char\*)message.payload;

option[3] = '\n';

channelLink=(char\*)message.payload + 4;

usernameLength = message.payloadlen;

if(option[0] == 'a'){

findChannelUsername(channelLink);

}

}

int main() {

BSP\_LCD\_Init();

setLCD();

printf("Ugradbeni sistemi\r\n");

printf("Test projekta\r\n\r\n");

NetworkInterface \*network;

network = NetworkInterface::get\_default\_instance();

if (!network) {

return -1;

}

MQTTNetwork mqttNetwork(network);

MQTT::Client<MQTTNetwork, Countdown> client(mqttNetwork);

const char\* hostname = "broker.hivemq.com";

int port = 1883;

printf("Connecting to %s:%d\r\n", hostname, port);

int rc = mqttNetwork.connect(hostname, port);

if (rc != 0)

printf("rc from TCP connect is %d\r\n", rc);

MQTTPacket\_connectData data = MQTTPacket\_connectData\_initializer;

data.MQTTVersion = 3;

data.clientID.cstring = "ugradbeni";

data.username.cstring = "";

data.password.cstring = "";

if ((rc = client.connect(data)) != 0)

printf("rc from MQTT connect is %d\r\n", rc);

if ((rc = client.subscribe(TEMAPROJEKAT, MQTT::QOS2, messageArrived)) != 0)

printf("rc from MQTT subscribe is %d\r\n", rc);

MQTT::Message message;

char buf[100];

while (1) {

BSP\_TS\_GetState(&TS\_State);

rc = client.subscribe(TEMAPROJEKAT, MQTT::QOS0, messageArrived);

wait\_ms(10);

}

}