SŠPU
STŘEDNÍ ŠKOLA PRŮMYSLOVÁ
A UMĚLECKÁ, OPAVA

ZÁVĚREČNÁ STUDIJNÍ PRÁCE

dokumentace

Teploměr pro obchod

Štěpán Wysoglad

Obor:	18-20-M/01 INFORMAČNÍ TECHNOLOGIE se zaměřením na počítačové sítě a programování
Třída:	IT4
Školní rok:	2021/2022

Poděkování

Jako první bych chtěl podělkovat panu Ing. Petru Grusmannovi za pomoc s komunikační částí projektu, panu Ing. Marceli Godovskému za pomoc s hardwarovou částí projektu, a nakonec bych chtěl poděkovat také panu Mgr. Marku Lučnému, za rady v oblasti webu a webových stránek.

Prohlašuji, že jsem závěrečnou práci vypracoval samostatně a uvedl veškeré použité informační zdroje.

Souhlasím, aby tato studijní práce byla použita k výukovým účelům na Střední průmyslové a umělecké škole v Opavě, Praskova 399/8.

V Opavě 31. 12. 2021

podpis autora práce

ANOTACE

Cílem projektu bylo vytvoření teploměru pro lednice v obchodech, který by zajistil rychlý zápis naměřených hodnot. Pro zápis byl použit čip ESP8266 naprogramovaný v jazyce C a technologie AJAX. Čip prostřednictvím wifi komunikuje se zařízeními a stanovené IP adrese servíruje webové rozhraní, které pomocí JavaScript a HTML5 zobrazuje čas a teplotu a umožňuje export do CSV souboru.

OBSAH

ÚVOD

Jako v každém obchodě se i u nás musí zapisovat teploty v mrazácích a lednicích z důvodu hygienických požadavků.

Proto mě jednoho dne napadlo, že bych vytvořil teploměr s automatickým zápisem, aby se to nemuselo každý den vypisovat ručně. Takto stačí jednou začas vyexportovat hodnoty do souboru a nikdo se nemusí o nic starat.

V této dokumentaci se zabývám především problematikou programování čipu ESP8266 v jazyce C. Naznačuji rovněž princip fungování webové aplikace, která využívá JavaScript a HTML5 Canvas pro zobrazení grafu.

1 TEORETICKÁ A METODICKÁ VÝCHODISKA

1.1 AJAX

AJAX (Asynchronous JavaScript And XML) je technologie pro vývoj interaktivních webových aplikací, které mění svůj obsah bez nutnosti opětovného načtení za pomoci asynchronního zpracování webových stránek pomocí knihovny v JavaScriptu.

```
function loadDoc() {
  const xhttp = new XMLHttpRequest();
  xhttp.onload = function() {
    document.getElementById("demo").innerHTML = this.responseText;
    }
  xhttp.open("GET", "ajax_info.txt", true);
  xhttp.send();
}
```

Ukázka načítání souboru pomocí AJAX

1.2 REST API

Je možnost, jak jednoduše tvořit, číst, nebo mazat informace ze serveru pomocí jednoduchých HTTP volání.

REST je, na rozdíl od XML-RPC či SOAP, orientován datově, nikoli procedurálně. Webové služby definují vzdálené procedury a protokol pro jejich volání, REST určuje, jak se přistupuje k datům.

Rozhraní REST je použitelné pro jednotný a snadný přístup ke zdrojům (resources). Zdrojem mohou být data, stejně jako stavy aplikace (pokud je lze popsat konkrétními daty). Všechny zdroje mají vlastní identifikátor URI a REST definuje čtyři základní metody pro přístup k nim.

```
void handleRoot() {
 server.send(200, "text/html", “Hello World!“); //Send web page
}
 //reading temperature
void handleTemp() {
  float t = dht.readTemperature();
  if (isnan(t)) {
    Serial.println(F("Failed to read from DHT sensor!"));
    return;
  }
 String value = String(t);
 server.send(200, "text/plane", value);
}

void setup(void){
server.on("/", handleRoot);      //This is display page
  server.on("/readADC", handleTemp);
}
void loop(void){
  server.handleClient();
  delay(1);
 }
```

Příklad REST API

2 VYUŽITÉ TECHNOLOGIE

2.1 ESP8266 – D1 mini lite

ESP8266 je levný wifi modul, který se dá použít jak ve spolupráci s Arduinem nebo jiným čipem, tak i samostatně. Tento model obsahuje 1 MB flash paměti. V poměru cena výkon je skvělou volbou.

2.2 C++

C++ je programovací jazyk, který vyvinul Bjarne Stroustrup a je rozšířením jazyka C. C++ podporuje několik programovacích stylů jako je procedurální programování, objektově orientované programování a generické programování, není tedy jazykem čistě objektovým. V současné době patří C++ mezi nejrozšířenější programovací jazyky.

Jazyk C++ jsem zvolil, protože se jej učíme ve škole, dobře se mi v něm programuje a je vhodný pro programování mikrokontrolerů i ESP.

2.3 PlatformIO

Rozšíření ve Visual Studio Code, funguje jako spouštěč arduino kódů, je to Open Source uživatelské rozhraní a debugger pro Arduino a ESP zařízení.

Je mnohem přehlednější než obyčejné Arduino IDE, rozděluje program do jednotlivých souborů už od začátku. Lze jednoduše přidat knihovny jen do určeného projektu nebo do všech, knihovny přitom zůstávají aktualizované. Má na výběr z mnoha různých čipů, které se zde mohou programovat, knihovna je rovněž velice obsáhlá.

3 ZPŮSOBY ŘEŠENÍ A POUŽITÉ POSTUPY

3.1 Řešení webového serveru

K fungování celého zařízení je potřeba aby fungoval webový server. Zprvu jsem myslel, že bude stačit jednoduchý AccessPoint, tedy že si ESP vytvoří svou vlastní síť, ke které by se nakonec stačilo jen připojit a do vyhledávače následně zadal jen pevně nastavenou IP adresu. Nakonec jsem se ale rozhodl pro druhý způsob a to, že se ESP připojí již na vytvořenou wifi síť, která následně přiřadí IP adresu. V tomto případě není zapotřebí internetové připojení, ale jde o bezpečnost. Takto se nikdo nemůže napojit na ESP nezná-li kredence od wifi sítě a IP adresu ESP čipu.

```
const char* ssid = "YOUR-SSID-HERE";
const char* password = "YOUR-PASSWORD-HERE";

void setup(void){
  Serial.begin(9600);
  Serial.println();
  Serial.println("Booting Sketch...");

//ESP32 connects to your wifi ----------------------------------
  WiFi.mode(WIFI_STA); //Connect to your wifi
  WiFi.begin(ssid, password);

  Serial.println("Connecting to ");
  Serial.print(ssid);

  //Wait for WiFi to connect
  while(WiFi.waitForConnectResult() != WL_CONNECTED){
      Serial.print(".");
    }

  //If connection successful show IP address in serial monitor
  Serial.println("");
  Serial.print("Connected to ");
  Serial.println(ssid);
  Serial.print("IP address: ");
  Serial.println(WiFi.localIP());  //IP address assigned to your ESP

  server.begin();                  //Start server
  Serial.println("HTTP server started");
}
```

Ukázka nastavení webového serveru

3.2 Webové rozhraní

Nejtěžší bylo asi vytvořit webovou stránku a připojit ji k web serveru. Nad tímto jsem strávil měsíce, ale nakonec za pomocí technologie AJAX se mi povedlo ji zprovoznit a posílat zde i data k zobrazení. Webová stránka kromě HTML využívá i JavaScriptu a metodu pro dynamické zobrazení posílaných dat. Data jsou přijímána na stránku, v tomto na endpoint readADC, která si následně vezme skript, který nakonec data rozdělí a zobrazí je na hlavní stránce.

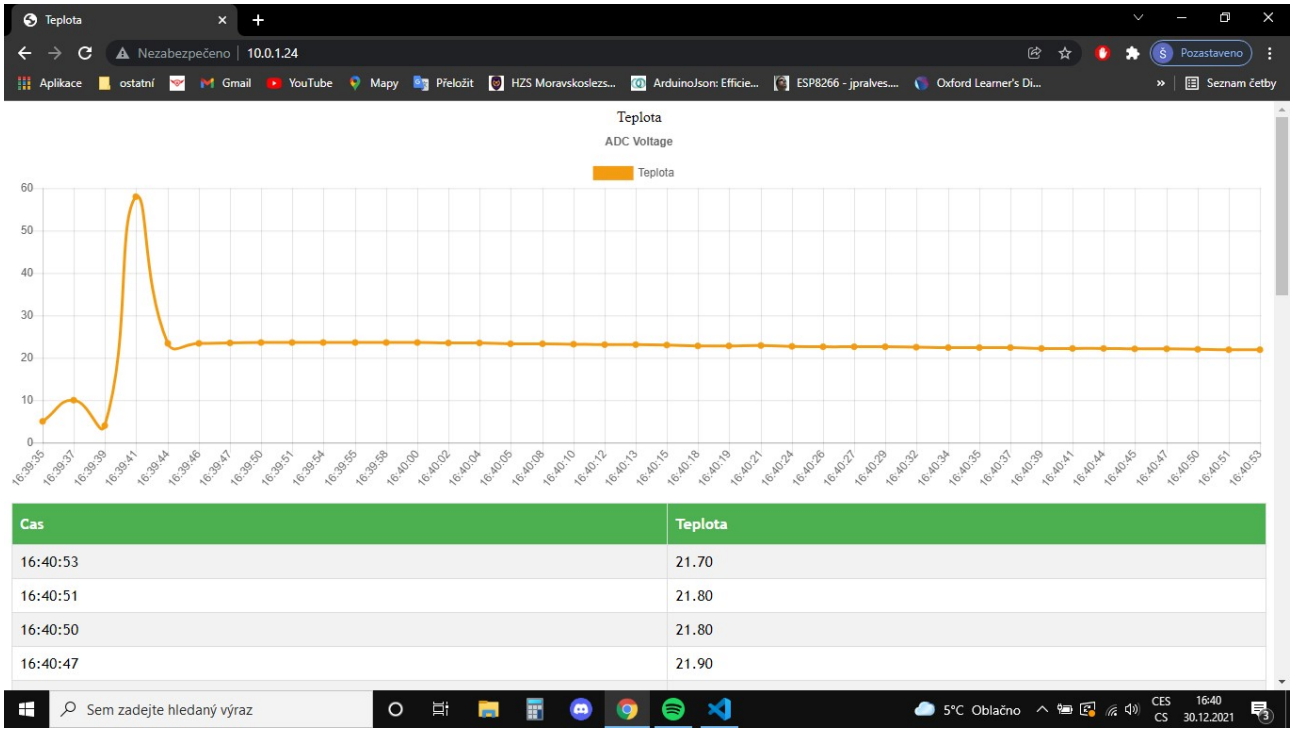
```
<!doctype html>
<html>
<head>
    <title>Teplomer</title>
    <script src =
"https://cdnjs.cloudflare.com/ajax/libs/Chart.js/2.7.3/Chart.min.js"></script>
    <link rel="stylesheet" type="text/css" href="style.css">

</head>

<body>
    <div style="text-align:center;">Teplota</div>
    <div class="chart-container" position: relative; height:350px; width:100%>
        <canvas id="Chart" width="400" height="400"></canvas>
    </div>
    <button><a href="readADC" download="file.csv">Export</a></button>
<div>
    <table id="dataTable">
      <tr><th>Cas</th><th>Teplota</th></tr>
    </table>
</div>
<br>
<br>
<script src="script.js"></script>
</body>

</html>
```

Ukázka hlavní stránky



Graf a zapsané hodnoty

3.3 Zápis a výpis souboru

Pro ukládání naměřených hodnot byl použit FileSystem LittleFS, ve kterém lze zapisovat a vytvářet soubory, zároveň mi umožnil separovat webovou stránku od kódu. LittleFS je do jisté míry stejný jako SPIFFS, který jsem používal ze začátku, ale při dokončování projektu se vyskytly malé nesrovnalosti při zapisování do souboru, kdy již naměřená data spojil dohromady dvěma až třemi nečitelnými znaky. Tento problém byl vyřešen náhradou za již zmíněný LittleFS.

Nejdřív bylo nutné vytvořit soubor do kterého se data budou zapisovat. To bvlo vyřešeno funkcí write()(viz. obr. níže). Následně se data musela zapsat bez jakéhokoliv přepisování, to řeší funkce append()(viz. obr. níže). Pro kontrolu jsem daný soubor vypisoval do terminálu za pomocí funkce read()(viz. obr. níže).

Následně se celý soubor posílal na vedlejší stránku s názvem readADC.

```
void write(){
    File file = LittleFS.open("/data.csv", "w");
    file.close();
}

void append(){
    File file = LittleFS.open("/data.csv", "a");
    if(!file){
        Serial.println("Chyba otevření souboru");
        return;
    }
    float t = dht.readTemperature();
    String value = "";
    Serial.println(t);
  if (isnan(t)) {
    Serial.println(F("Failed to read from DHT sensor!"));
    return;
  }
    value = String(timeClient.getHours()) + ":" + String(timeClient.getMinutes()) + ":" +
String(timeClient.getSeconds()) + "," + String(t) + "\n";
    file.print(value);
    file.close();
}

void read(){
    File file = LittleFS.open("/data.csv", "r");
    while(file.available()){
        Serial.write(file.read());
    }
    Serial.println();
    file.close();
}
```

Ukázka zápisu a výpisu souboru

3.4 Zjišťování času

Zjišťovat čas bylo celkem zajímavé, nejdřív jsem si myslel, že pro to budu potřebovat RTC(Real Time Clock) čip, následně jsem zjistil, že lze dostat data taky z GPS, problém byl že ani jeden čip nemám. Později jsem našel třetí řešení, a to za pomocí NTP serveru, kde přímo vybírám čas z tzv. timeserveru, v mém případě pool.ntp.org.

```
const long utcOffsetInSeconds = 3600;
WiFiUDP ntpUDP;
NTPClient timeClient(ntpUDP, "pool.ntp.org", utcOffsetInSeconds);
String value = String(timeClient.getHours()) + ":" + String(timeClient.getMinutes()) + ":"
+ String(timeClient.getSeconds()) + "," + String(t) + "\n";
```

Ukázka výpisu aktuálního času

3.5 Měření teploty

Tato část byla asi nejjednodušší. K měření teploty byl použit teploměr typu DHT22, který dokáže měřit i záporné hodnoty. Měří s přesností <+-0.5 °C od -40~80 °C. Tento teploměr dokáže měřit i vlhkost vzduchu a pocitovou teplotu.

Teploměr DHT22

Naprogramovat jej bylo snadné, vlastně k tomu stačilo jen pár řádků. Jednoduše se definuje typ sensoru

```
#include <DHT.h>
#define DHTPIN 4
#define DHTTYPE 22
DHT dht(DHTPIN, DHTTYPE);
void handleTemp() {
  float t = dht.readTemperature();
  if (isnan(t)) {
    Serial.println(F("Failed to read from DHT sensor!"));
    return;
  }
 String value = String(t);
 Serial.println(value);

}
void setup(void){
 handleTemp;

  dht.begin();
}
```

Ukázka výpisu teploty

4 SHRNUTÍ A POROVNÁNÍ S KONKURENCÍ

4.1 Shrnutí

Aplikace dokáže zapisovat reálnou hodnotu právě naměřenou v reálném čase, kterou vyobrazí v grafu. Graf funguje bez problémů, dokáže zapsat velké množství dat, ale při určitém počtu už není tak čitelný a změna hodnot není tak dobře vidět jako ze začátku. Tuto nečitelnost způsobuje neustále oddalování v důsledku velkého množství zápisů. Také stránka je postupně delší a delší tudíž dostat se až na konec je třeba po 100 zápisech velice zdlouhavé.

Měření teploty v lednici ještě nebylo provedeno z důvodu chybějící krabičky, která by byla voděodolná. Krabička se však už vyrábí a za chvíli můžou být provedeny první testy v chladu. Momentálně bude muset stačit pokojová a venkovní teplota.

Celkově bych svůj projekt ohodnotil asi za splněný i přes to, že má některé nedostatky, které určitě šly udělat lépe. Doufám, že projekt bude použitelný, alespoň pro vlastní účely, a tudíž že se do našeho obchodu nebudou muset kupovat teploměry od konkurence.

Nejdůležitější na celém projektu jsou ale vědomosti, které jsem za celou tu dobu nabral, hlavně o fungování webových serverů. V budoucnu snad budu moct tyto vědomosti použít při sestavování jiných, podobných projektů.

4.2 Porovnání s konkurencí

Konkurence má jistě lepší výsledky, než mám já. Mé zařízení stojí okolo 300 korun bez práce, načež konkurence je 3x a dražší, našel jsem jeden model s cenou 1500 korun za podobnou práci, kterou mám já. Uznávám, že konkurence bude mít asi lepší technologie než, které jsem využil, a že bude mít i lepší naprogramování.

Uplatnění na trhu asi není v tomto případě možné, chtělo by to ještě hodně práce, než by se projekt vyrovnal konkurenci, ale pokusím se o co největší přiblížení.

ZÁVĚR

Cílem projektu bylo naprogramovat takový teploměr, který by zapisoval teploty v lednicích s možností následného vyexportování do CSV souboru. Byl použit čip 8266 s wifi modulem a naprogramovaný v jazyce C++. Čip má nastavenou statickou IP, na které servíruje webové rozhraní.

Myslím, že projekt svůj úkol splňuje. Zařízení dokáže zobrazovat webovou stránku s naměřenými hodnotami, reálným časem a dokáže exportovat data do CSV souboru.

Co mě ještě napadá pro zlepšení projektu je vylepšení webové stránky, jelikož se stránka nedokáže rozdělovat na více, tudíž se po chvíli stává nepřehlednou při vyhledávání starších údajů.

SEZNAM POUŽITÝCH INFORMAČNÍCH ZDROJŮ

[1] ESP8266 Datasheets [online].

[cit. 2021-12-30].

< https://jpralves.net/post/2016/11/15/esp8266.html#esp-201>

[2] Arduino libraries [online].

[cit. 2021-12-30].

< http://arduino.esp8266.com/Arduino/versions/2.1.0-rc1/doc/libraries.html>

[3] AJAX – kde jsou hranice? [online].

[cit. 2021-12-30].

< https://www.snizekweb.cz/c/ajax-kde-jsou-hranice>

[4] AJAX Tutorial [online].

[cit. 2021-12-30].

< https://www.w3schools.com/js/js_ajax_intro.asp>

[5] DHT22 Datasheets [online].

[cit. 2021-12-30].

< http://www.datasheetcafe.com/dht22-datasheet-pdf>

[6] Grafy [online].

[cit. 2021-12-30].

< https://cdnjs.com/libraries/Chart.js>