|  |  |  |
| --- | --- | --- |
|  | | |
| **Závěrečná studijní práce**  **dokumentace** | | |
| **Teploměr pro obchod** | | |
| Štěpán Wysoglad | | |
|  | | |
|  | |  |
| **Obor:** | 18-20-M/01 INFORMAČNÍ TECHNOLOGIE  se zaměřením na počítačové sítě a programování | |
| **Třída:**  **Školní rok:** | IT4  2021/2022 | |

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 5](#_Toc91795815)

[1 Teoretická a metodická východiska 6](#_Toc91795816)

[1.1 AJAX 6](#_Toc91795817)

[2 Využité technologie 7](#_Toc91795818)

[2.1 ESP8266 – 201 7](#_Toc91795819)

[2.2 C++ 7](#_Toc91795820)

[2.3 PlatformIO 8](#_Toc91795821)

[3 Způsoby řešení a použité postupy 9](#_Toc91795822)

[3.1 Řešení webového serveru 9](#_Toc91795823)

[3.2 Webové rozhraní 10](#_Toc91795824)

[3.3 Měření teploty 11](#_Toc91795825)

[4 Výsledky řešení, výstupy, uživatelský manuál 12](#_Toc91795826)

[4.1 Shrnutí 12](#_Toc91795827)

[4.2 Porovnání s konkurencí 12](#_Toc91795828)

[Závěr 13](#_Toc91795829)

[Seznam použitýCH INFORMAČNÍCH ZDROJů 14](#_Toc91795830)

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.

# Teoretická a metodická východiska

## 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*

## 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.

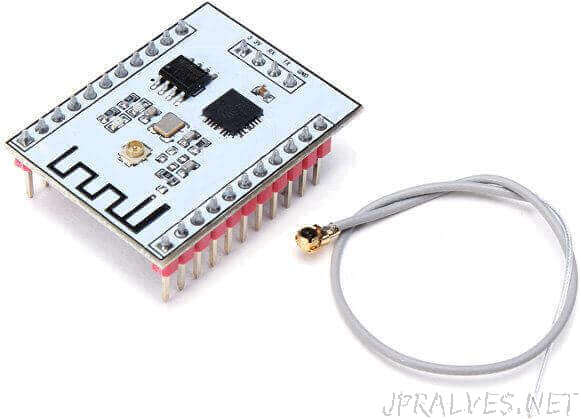
|  |
| --- |
| void handleRoot() {  server.send(200, "text/html", “Hello World!“); //Send web page  }  //reading temperature  void handleTemp() {  float t = dht.readTemperature();  if (isnan(t)) {  Serial.println(F("Failed to read from DHT sensor!"));  return;  }  String value = String(t);  server.send(200,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

# Využité technologie

## ESP8266 – 201

ESP8266 je levný wifi modul, který se dá použít jak ve spolupráci s Arduinem nebo jiným čipem, tak i samostatně. Tento konkrétní model 201 obsahuje 512 MB flash paměti. Je možné ho programovat v jazyce C, C++, Lua, Python, JavaScript.



*ESP8266 - 201*

## 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.

## 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.



*Logo PlatformIO*

# Způsoby řešení a použité postupy

## 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");  } |

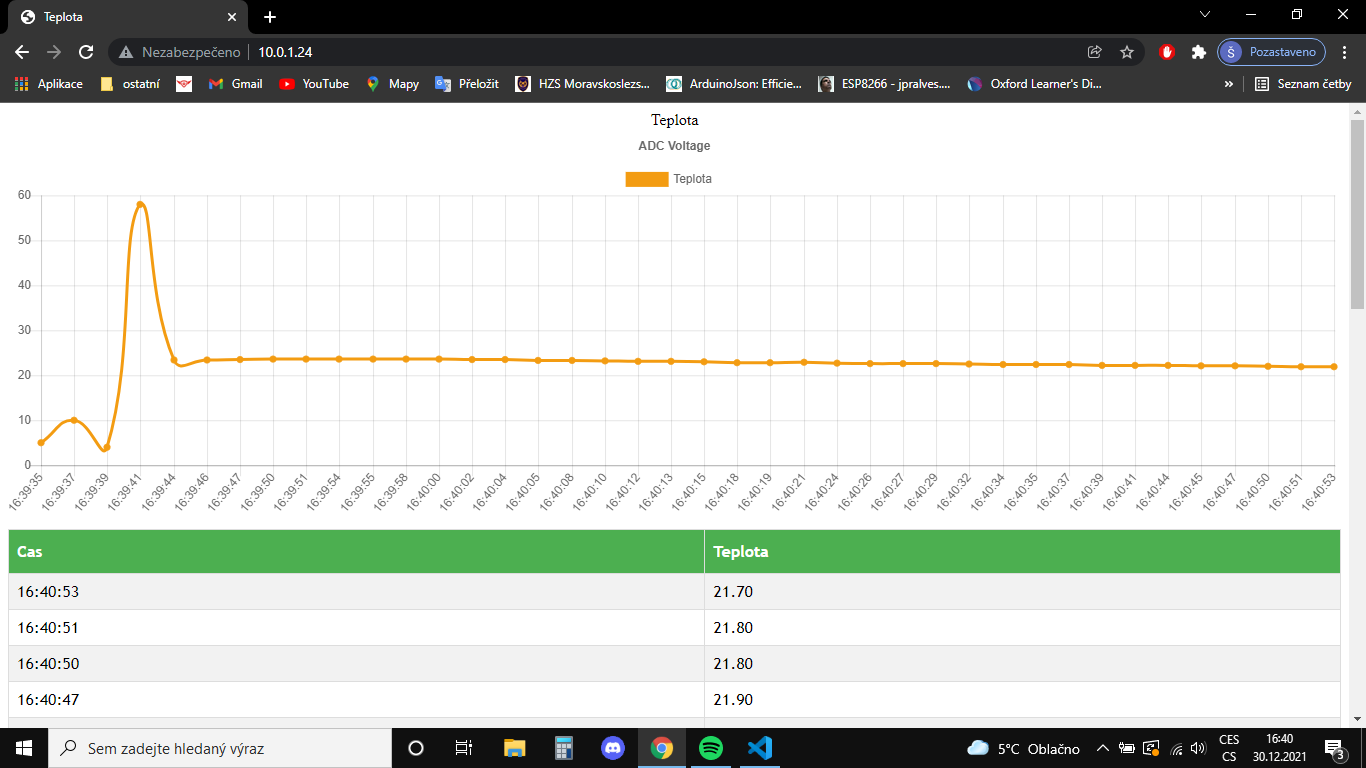
*Ukázka nastavení webového serveru*

## Webové rozhraní

Nejtěžší bylo asi vytvořit webovou stránku a připojit ji k webserveru. Nad tímto jsem strávil měsíce, ale nakonec za pomocí technologie AJAX se mi povedlo ji zprovoznit a posílat zde i data zobrazení. Celá webová stránka má vlastní soubor, který je následně implementován do hlavního kódu. Webová stránka kromě HTML využívá i JavaScriptu a XMLHTTP pro dynamické zobrazení posílaných dat. V tomto případě se data aktualizují každé dvě sekundy.

|  |
| --- |
| <!doctype html>  <html>  <head>    <title>Teplomer</title>  </head>  <body>  <h1>Hello world!</h1>  </body>  </html> |

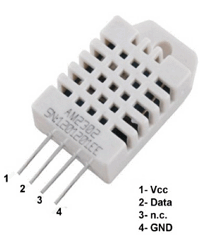
*Příklad jednoduché stránky*



*Graf a zapsané hodnoty*

## 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*

# Výsledky řešení, výstupy, uživatelský manuál

## Shrnutí

Aplikace dokáže zapisovat Pravdivou hodnotu právě naměřenou v reálném čase, kterou vyobrazí v grafu. Graf potřebuje pár vstupů pro nastartování, prvních asi šest hodnot je nepravdivých i když v tabulce správně zapsaných. Následně pak funguje bez problémů, dokáže zapsat velké množství dat, ale při určitém počtu už graf není čitelný a změna hodnot není tak 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 50 zápisech velice zdlouhavé.

## 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

Závěr

Cílem projektu bylo naprogramovat takový teploměr, který by zapisoval teploty v lednicích s možností následného vyexportování do CSV souboru.

Myslím, že projekt svůj úkol splňuje. Zařízení dokáže zobrazovat webovou stránku s naměřenými hodnotami a reálným časem.

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.

Jediné, co se mi nepovedlo je právě exportování dat do CSV souboru, což mě trošku mrzí. A doufám, že to brzy napravím společně s úpravou stránky pro lepší čitelnost záznamů. Také graf budu muset ještě trošku poladit, protože se mi nelíbí, jak zapisuje prvních pár hodnot. Graf je však jen přídavkem a má tedy nejnižší prioritu, jelikož data nijak neovlivňuje.

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.

Celkově bych svůj projekt ohodnotil asi za splněný i přes to, že má spoustu chyb a spoustu nevyladěných věcí, 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.

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.

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>