|  |  |  |  |
| --- | --- | --- | --- |
|  | | |  |
| **Závěrečná studijní práce**  **dokumentace** | | |  |
| **Distance Warning Device - DWD** | | |  |
| Andreas Piskoř | | |  |
|  | | |  |
|  | |  |  |
| **Obor:** | 18-20-M/01 INFORMAČNÍ TECHNOLOGIE  se zaměřením na počítačové sítě a programování | |  |
| **Třída:**  **Školní rok:** | IT4  2020/2021 | |  |

#### Poděkování

*Děkuji panu učiteli Ing. Petru Grussmannovi za rady a vylepšení projektu, panu učiteli Mgr. Marcelu Godovskému za rady, které se týkaly napájení, a především panu Ing. Jiřímu Miekischovi za pomoc se zapojením hardwaru a ukázku pájení.*

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. 2020

*podpis autora práce*

**ANOTACE**

Projekt se zabývá zpracováním přístroje k monitoringu držení bezpečných odstupů od ostatních lidí kvůli pandemii. Skládá se z hardwarové a softwarové části. Softwarová část je napsaná v Platformě IDE, což znamená, že kód je napsán v kombinaci jazyků C a C++. V této části si může uživatel měnit maximální vzdálenost od osob, jak jen ultrazvukový senzor dovolí. V hardwarové části je základem čip Arduino nano, který celý projekt řídí. Ultrazvukový senzor měří vzdálenost od objektů (lidí), hodnoty ze senzoru se vypisují na LCD display, a je-li maximální dovolená hodnota překročena, začne tonizovat bzučák a svítit červená LED, naopak, když je hodnota vzdálenosti v nastaveném rozmezí, svítí zelená LED. Vše je propojeno na napájecím poli.

**Klíčová slova**

Platformio IDE; C++; C; Arduino nano; ultrazvukový senzor; LCD display; LED

**OBSAH**

[Úvod 6](#_Toc28475)

[1 Výroba distančního systému 7](#_Toc15343)

[2 Využité technologie 8](#_Toc22620)

[2.1 Hardware 8](#_Toc15781)

[2.1.1 Seznam součástek 8](#_Toc22411)

[2.1.2 Arduino nano 9](#_Toc71)

[2.1.3 Ultrazvukový senzor HC-SR04 11](#_Toc26556)

[2.1.4 LCD Display 16x02 I2C 12](#_Toc32141)

[2.1.5 Bzučák 14](#_Toc17402)

[2.1.6 Tlačítko 14](#_Toc21447)

[2.1.7 LED 15](#_Toc19223)

[2.1.8 Rezistor 15](#_Toc16692)

[2.2 Napájení 9V baterie 16](#_Toc6693)

[2.3 Software 17](#_Toc12070)

[2.3.1 VS Code 17](#_Toc23441)

[2.3.2 Platformio IDE 17](#_Toc29494)

[3 Způsoby řešení a použité postupy 18](#_Toc30725)

[3.1 Funkčnost součástek 18](#_Toc24333)

[3.1.1 Senzorová část 18](#_Toc32648)

[3.1.2 Displayová část 19](#_Toc15251)

[3.1.3 Bzučáková část 20](#_Toc5234)

[3.2 Kompletní řešení projektu 21](#_Toc29316)

[3.2.1 Problém s napájením 21](#_Toc12079)

[3.2.2 Problémy s knihovnami 21](#_Toc442)

[4 Výsledky řešení 22](#_Toc2277)

[4.1 Finální podoba hardwaru 22](#_Toc28075)

[4.2 Finální podoba kódu 23](#_Toc11384)

[4.3 Schéma 24](#_Toc21735)

[Závěr 25](#_Toc26754)

[Seznam použitýCH INFORMAČNÍCH ZDROJů 26](#_Toc5290)

[Seznam příloh 27](#_Toc29240)

# Úvod

Rozhodl jsem se, zpracovat tento projekt z toho důvodu, že na jaře tohoto roku vypukla celosvětová pandemie COVIDU-19. Chtěl bych, aby lidé dbali na rozestupy, které si mají držet od ostatních lidí tak, aby nemoc nedostali či neroznášeli. U zařízení, které v práci popisuji, se může nastavit hodnota v kódu, a může se libovolně měnit. Zařízení nemusí nutně sloužit jen na tento případ, když si vezmete novější auto a začnete couvat, zjistíte, že zařízení v tomto autě funguje na obdobném principu, akorát trochu inteligentněji.

Na tento projekt jsem nahlížel tak, aby skutečně fungoval a měl i své vychytávky. Hardwarovou část tvoří tudíž čip Arduino nano, který má hodně výhod. Je zde propojeno mnoho součástek, o kterých se rozepíšu později, a to u Hardwarové části. U softwarové části jsem místo původního Arduina IDE vybral Platformio IDE. Cíle byly v tento moment jasné, udělat nejlépe funkční měření se senzorem, a aby se hodnota vypisovala na display, a když se hodnota překročí, začne bzučet bzučák.

V této dokumentaci se rozepíšu hlavně o problematice u součástek a napájení. Nebude chybět ani samotné schéma projektu. Na závěr práci zhodnotím, co se mi povedlo a co naopak ne.

# Výroba distančního systému

První část projektu se zabývala postavením a konstrukcí zařízení. Bylo to poprvé, co jsem musel něco pájet, součástky byly koupeny z mých prostředků, nejedná se o až tak drahé součástky, ale i přesto věřím, že projekt bude zcela funkční. Součástky jsem vybíral po náležité rozvaze, protože jak jeden moudrý člověk řekl: “Jak si usteleš, tak si také lehneš,” a proto jsem si radši připlatil za lepší. Tudíž jsem nekupoval nic z Číny, vše je vyrobeno v České republice, což se v tomto projektu doufám vyplatí.

Nejdříve jsem uvažoval o použití jádra projektu ESP8266, ale řekl jsem si, že je zbytečné tento modul kupovat. Projekt totiž nevyžaduje Wi-Fi a také už doma mám Arduino nano. Arduino nano je vytvořené na takové malé projekty, jako je ten můj. Je taky založen na mikrokontroleru ATmega328 od firmy Atmel.

Ostatní součástky jsem příliš neřešil, takže jsem se podíval na to, co mám doma k dispozici a co ne, to jsem objednal z Arduino shopu. Jednalo se především o senzor, display a rezistory.

Po zapojení všech součástek jsem se rozhodoval, zda použít programovací prostrědí Arduino IDE nebo Platformino IDE. Ale uposlechl jsem pana učitele Grussmanna a rozhodl jsem se pro Platformio. I když si do nynějška říkám, ne že by to byla chyba, ale chtěl jsem opravdu projekt udělat dle svého uvážení, tak snad někdy příště. Ale na druhou stranu jsem si procvičil jazyk C a C++.

# Využité technologie

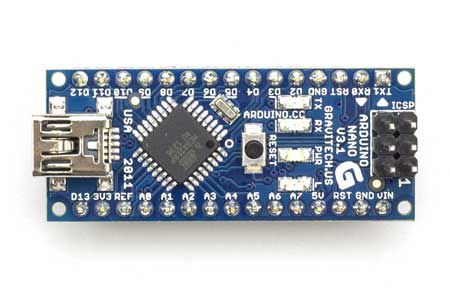
## 2.1 Hardware

### 2.1.1 Seznam součástek

* Arduino nano,
* ultrazvukový senzor HC-SR04,
* LCD display 16×02 I2C,
* bzučák,
* 2x tlačítko,
* červená LED,
* zelená LED,
* rezistor 100 Ω,
* 2× rezistor 220 Ω.

### 2.1.2 Arduino nano

Arduino nano je v [informatice](https://cs.wikipedia.org/wiki/Informatika" \o "Informatika) název malého [jednodeskového počítače](https://cs.wikipedia.org/wiki/Jednodeskov%C3%BD_po%C4%8D%C3%ADta%C4%8D" \o "Jednodeskový počítač) z [otevřené](https://cs.wikipedia.org/wiki/Otev%C5%99en%C3%BD_software" \o "Otevřený software) [platformy](https://cs.wikipedia.org/wiki/Po%C4%8D%C3%ADta%C4%8Dov%C3%A1_platforma" \o "Počítačová platforma) [Arduino](https://cs.wikipedia.org/wiki/Arduino" \o "Arduino), který je založen na [mikrokontrolerech](https://cs.wikipedia.org/wiki/Jedno%C4%8Dipov%C3%BD_po%C4%8D%C3%ADta%C4%8D" \o "Jednočipový počítač) [ATmega](https://cs.wikipedia.org/wiki/AVR" \o "AVR) od firmy [Atmel](https://cs.wikipedia.org/wiki/Atmel" \o "Atmel). Arduino nano je podobné Arduinu mini, s tím rozdílem, že má vlastní USB port a převodník. Je tedy trošku větší, ovšem není třeba mít extra převodník při vkládání programu.



*Obrázek č. 1 Arduino nano*

Vybral jsem si Arduino nano, protože jsem s ním pracoval i v minulosti a pracovalo se mi s ním velice dobře. Sice jedno Arduino nano během projektu shořelo, ale to nebyl podmět k tomu vzdávat se. Také jsem si ho vybral proto, že jsem jej už doma měl a nechtěl jsem pracovat s něčím, co neznám, např. s ESP8266.

Dělal jsem na něm hodně pokusů, ještě dříve, než jsem se dostal k závěrečnému projektu, a byly pozitivní, už jen z tohoto hlediska jsem usoudil, že Arduino nano je to správné řešení pro můj projekt.

**Technické informace o Arduinu**

|  |  |
| --- | --- |
| Mikroprocesor | ATmega328 |
| Architektura | AVR |
| Provozní napětí (logická úroveň) | 5V |
| Vstupní napětí (doporučeno) | 7–12V |
| Počet digitálních I/O pinů | 22 pinů, z toho 6 s PWM |
| Počet analogových vstupů | 8 pinů |
| Proudové zatížení na 1 pin | 40 mA |
| Flash paměť | 32 KB |
| SRAM | 2 KB |
| EEPROM | 1KB |
| Rychlost hodin | 16 MHz |
| Výška | 18 mm |
| Šířka | 45 mm |
| Váha | 7g |

Arduino vlastní 22 digitálních pinů, což umožňuje přidávat hodně vychytávek na projekty a má i 8 digitálních. Do Arduina se můžou i ukládat hodnoty pomocí EEPROM. Je malé a dobře se s ním manipuluje. Prostě pro potřeby mého projektu jak dělané.

### 2.1.3 Ultrazvukový senzor HC-SR04

Automatické měření vzdálenosti nemusí být zdaleka takový problém, jak by se mohlo na první pohled zdát. **Pomocí ultrazvukového principu můžeme vzdálenost měřit velmi pohodlně**, bezkontaktně a dokonce přesně. Ultrazvukový měřič vzdálenosti najde využití v mém projektu úplně přesně, kde se kontroluje vzdálenost od objektu. K dispozici je Arduino knihovna, a tak je získávání dat ze senzoru velmi jednoduché.



*Obrázek č. 2 HC-SR04*

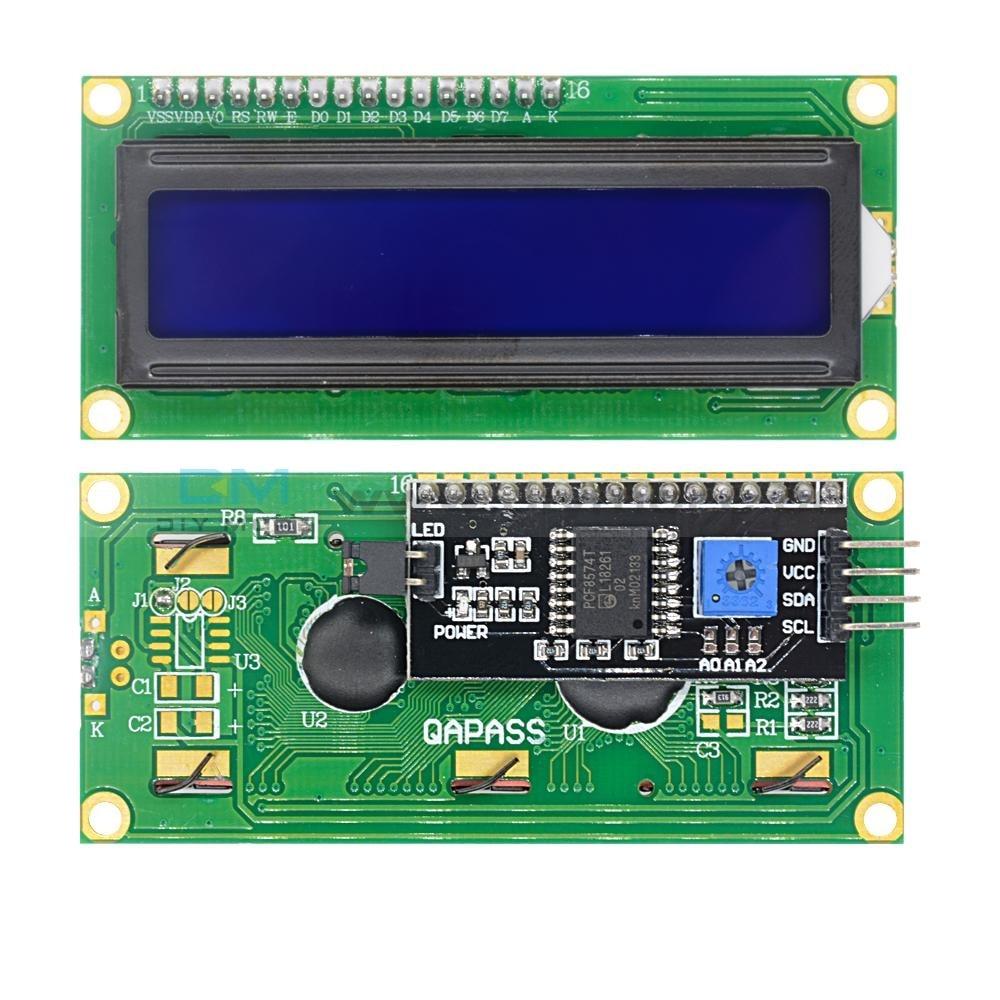
**Specifikace:**

* Pracovní rozsah: 2 – 400 cm
* Vysoká přesnost: až 3 mm
* Pracovní úhel: < 15 stupňů
* Klidový proud: < 2mA
* Napájecí napětí: 5V
* Velikost: 45×20×16 mm

Tento senzor jsem si vybral, protože mi ho pan učitel Grussmann doporučil, a byl k dispozici i ve škole.

### 2.1.4 **LCD** Display 16x02 I2C

I2C\_LCD je snadno použitelný zobrazovací modul, který může usnadnit zobrazení. Jeho použití může snížit obtížnost zapojení, abych se mohl soustředit na jádro práce. Knihovna Arduino pro I2C\_LCD usnadňuje uživatelům práci, potřebuje jen pár řádků kódu, aby dosáhl složitých funkcí grafiky a zobrazení textu. Může na nějakém místě nahradit sériový monitor Arduina, můžeme získat informace o běhu bez počítače, a to jsem zrovna potřeboval.



*Obrázek č. 3 LCD Display 16x02 I2C*

Tento display má strašně moc výhod oproti normálnímu LCD 16x02. Nejenže se lépe pracovalo s knihovnami, ale bylo i lehčí zapojení. Velkou výhodou je i podsvícení, takže i ve tmě se dá dobře přečíst, co zrovna vypisuje.

**Funkce displaye**

* Obsazeny jsou pouze 2 piny Arduino (použijte rozhraní I2C).
* Podporuje standardní režim I2C (100 Kbit / s) a rychlý režim I2C (400 Kbit / s).
* Kompatibilní s několika úrovněmi komunikační logiky: 2,8 ~ 5V.
* Knihovna Arduino podporována, k dokončení zobrazení použije řádek kódu.
* Integruje 7 velikostí písem ASCll, 5 grafických funkcí.
* Poskytuje vyhrazený software pro převod obrazových dat (Bitmap Converter).
* Většinu složitých operací zpracovává nezávislý řadič I2C\_LCD, což šetří prostředky uživatelského řadiče.
* Podporuje funkci kurzoru, může nastavit frekvenci 16 blikání kurzoru.
* Podporuje 128 úrovní nastavení podsvícení.
* Podpora 64 úrovňového nastavení kontrastu obrazovky.
* Podpora úpravy adresy zařízení.
* Podporuje paralelní práci 127 I2C\_LCD.
* Při ladění kódu může místo sériového monitoru sledovat stav běhu programu.
* K dispozici jsou dvě neobvyklé metody obnovení: reset a obnovení továrního nastavení.
* 4 symetrické pevné otvory pro snadnou instalaci.

**Specifikace:**

|  |  |  |  |
| --- | --- | --- | --- |
| Parametry | Hodnota |  |  |
| Typ obrazovky | Dvoubarevný LCD |  |  |
| Rozlišení obrazovky | 128 \* 64 pixelů |  |  |
| Velikost jednotlivých pixelů | 0,33 \* 0,33 mm |  |  |
| Režim komunikace | I2C (100 Kbit / s a 400 Kbit / s) |  |  |
| Ovladač | STM8S005KBT6 |  |  |
| Provozní frekvence | 16 MHz |  |  |
| Hmotnost | 20 g |  |  |

### 2.1.5 Bzučák

Bzučák je elektro-akustický měnič, který přeměňuje [elektrickou energii](https://cs.wikipedia.org/wiki/Elektrick%C3%A1_energie" \o "Elektrická energie) na [mechanickou energii](https://cs.wikipedia.org/wiki/Mechanick%C3%A1_energie" \o "Mechanická energie) ve formě [zvuku](https://cs.wikipedia.org/wiki/Zvuk" \o "Zvuk). Obvykle se skládá z [membrány](https://cs.wikipedia.org/w/index.php?title=Technick%C3%A9_membr%C3%A1ny&action=edit&redlink=1" \o "Technické membrány (stránka neexistuje)), z pohonné části, do které je přiváděn vstupní [signál](https://cs.wikipedia.org/wiki/Sign%C3%A1l" \o "Signál) a dalších dílů. V mém projektu bude sloužit k tomu, aby bzučel, když je porušena podmínka v kódu. To je tehdy, když je překročena maximální dovolená vzdálenost od objektu.



*Obrázek č. 4 Bzučák*

### 2.1.6 Tlačítko

Tlačítko je jednoduchý [spínač](https://cs.wikipedia.org/wiki/Sp%C3%ADna%C4%8D" \o "Spínač), který slouží k ručnímu ovládání elektrického zařízení. Hmatník tlačítka je obvykle vyroben z tvrdého materiálu (plastu nebo kovu) a tvar je přizpůsoben tak, aby bylo možné tlačítko obsluhovat tlakem prstu nebo ruky. Já tlačítko v tomto případě potřebuji, abych mohl vpustit napětí z baterie do oběhu, kdy chci.

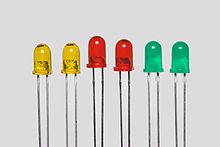


*Obrázek č. 5 Tlačitko*

### 2.1.7 LED

LED je označení v elektrotechnice pro diodu, která emituje světlo případně [infračervené](https://cs.wikipedia.org/wiki/Infra%C4%8Derven%C3%A9_z%C3%A1%C5%99en%C3%AD" \o "Infračervené záření) nebo [ultrafialové](https://cs.wikipedia.org/wiki/Ultrafialov%C3%A9_z%C3%A1%C5%99en%C3%AD" \o "Ultrafialové záření) záření, čímž se liší od standardní diody. LED vyzařuje z obnaženého [PN přechodu](https://cs.wikipedia.org/wiki/PN_p%C5%99echod" \o "PN přechod), a vede [stejnosměrný proud](https://cs.wikipedia.org/wiki/Stejnosm%C4%9Brn%C3%BD_proud" \o "Stejnosměrný proud) pouze jedním směrem. Na rozdíl od [žárovky](https://cs.wikipedia.org/wiki/%C5%BD%C3%A1rovka" \o "Žárovka) dosahuje vysoké účinnosti, je mechanicky odolná, levná na výrobu, a proto je čím dál více využívána.

Já v projektu využívám 2 LED a to červenou a zelenou. Červená svítí, když je překročena maximální dovolená vzdálenost od objektu. Zelená naopak, když je vše v pořádku a je vzdálenost dodržena.



*Obrázek č. 6 LED*

### 2.1.8 Rezistor

Rezistor je pasivní [elektrotechnická součástka](https://cs.wikipedia.org/wiki/Elektrotechnick%C3%A1_sou%C4%8D%C3%A1stka" \o "Elektrotechnická součástka) projevující se v [elektrickém obvodu](https://cs.wikipedia.org/wiki/Elektrick%C3%BD_obvod" \o "Elektrický obvod) v ideálním případě jedinou vlastností – [elektrickým odporem](https://cs.wikipedia.org/wiki/Elektrick%C3%BD_odpor" \o "Elektrický odpor) (jednotka [Ohm](https://cs.wikipedia.org/wiki/Ohm" \o "Ohm), značka Ω). Důvodem pro zařazení rezistoru do obvodu je obvykle snížení velikosti [elektrického proudu](https://cs.wikipedia.org/wiki/Elektrick%C3%BD_proud" \o "Elektrický proud) nebo získání určitého úbytku [napětí](https://cs.wikipedia.org/wiki/Elektrick%C3%A9_nap%C4%9Bt%C3%AD" \o "Elektrické napětí). Rezistory se také mohou používat jako topné články, testovací zátěže pro [generátory](https://cs.wikipedia.org/wiki/Elektrick%C3%BD_gener%C3%A1tor" \o "Elektrický generátor) apod. Rezistory rozdělujeme na pevné a proměnné. Pevné rezistory mají pevně danou hodnotu odporu, která se mírně mění pouze v závislosti na teplotě, procházejícím proudem a životnosti rezistoru a proměnné nepoužívám, ale z názvů vidíte, že se dá měnit jejich hodnota.

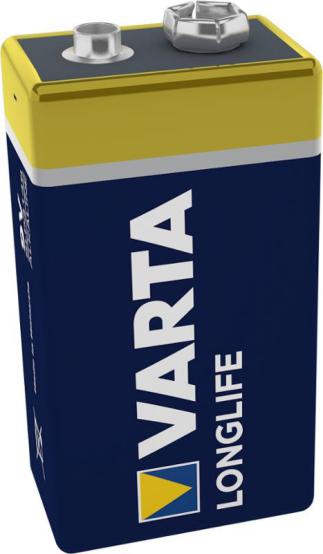
Rezistory mám v projektu přesně 3. Jeden s hodnotou 100 Ω a dva s hodnotou 220 Ω. V mém případě složí k tomu, aby do součástek nešlo moc napětí a nezničily se tak.

## 2.2 Napájení 9V baterie

Devíti-voltovka nebo 9V baterie, je obyčejná velikost baterie, která byla zavedena počátkem [tranzistorov](https://en.wikipedia.org/wiki/Transistor_radio" \o "Tranzistorové rádio)é doby. Má obdélníkový hranolový tvar se zaoblenými hranami a nahoře má polarizovaný snap konektor. Formát devíti voltových baterií je běžně dostupný v primární uhlík-zinkové podobě.

Většina 9V alkalických baterií je vyrobena ze šesti samostatných článků 1,5 V LR61 uzavřených v obalu. Tyto články jsou o něco menší než články [AAAA](https://en.wikipedia.org/wiki/AAAA_battery" \o "Baterie AAAA) LR8D425 a lze je místo nich použít pro některá zařízení, i když jsou o 3,5 mm kratší. Uhlík-zinkové typy jsou vyrobeny se šesti plochými buňkami pohromadě, uzavřené v obalu odolném proti vlhkosti, aby se zabránilo vysychání.

A přesně tohle je moje baterie. Jediný problém, co jsem měl, byl ten, že jsem musel koupit konektor, abych s ní mohl napájet součástky v nepájivém poli.



*Obrázek č. 7 9V baterie*

## 2.3 Software

### 2.3.1 VS Code

VS Code je [editor zdrojového kódu](https://cs.wikipedia.org/w/index.php?title=Editor_zdrojov%C3%A9ho_k%C3%B3du&action=edit&redlink=1" \o "Editor zdrojového kódu (stránka neexistuje)) vyvíjený společností [Microsoft](https://cs.wikipedia.org/wiki/Microsoft" \o "Microsoft) pro [operační systémy](https://cs.wikipedia.org/wiki/Opera%C4%8Dn%C3%AD_syst%C3%A9m" \o "Operační systém) [Windows](https://cs.wikipedia.org/wiki/Windows" \o "Windows), [Linux](https://cs.wikipedia.org/wiki/Linux" \o "Linux) a [macOS](https://cs.wikipedia.org/wiki/MacOS" \o "MacOS). Obsahuje podporu pro [Git](https://cs.wikipedia.org/wiki/Git" \o "Git) (a pro [GitHub](https://cs.wikipedia.org/wiki/GitHub" \o "GitHub)), [zvýraznění syntaxe](https://cs.wikipedia.org/wiki/Zv%C3%BDrazn%C4%9Bn%C3%AD_syntaxe" \o "Zvýraznění syntaxe), kontextový [našeptávač](https://cs.wikipedia.org/wiki/Na%C5%A1ept%C3%A1va%C4%8D" \o "Našeptávač) a podporu pro [ladění](https://cs.wikipedia.org/wiki/Lad%C4%9Bn%C3%AD_(programov%C3%A1n%C3%AD)" \o "Ladění (programování)) a [refaktorizaci](https://cs.wikipedia.org/wiki/Refaktorizace" \o "Refaktorizace). Tento zdrojový kód je [svobodný software](https://cs.wikipedia.org/wiki/Svobodn%C3%BD_software" \o "Svobodný software) pod [licencí MIT](https://cs.wikipedia.org/wiki/Licence_MIT" \o "Licence MIT). Sestavené binárky nabízené přímo Microsoftem jsou [freewarem](https://cs.wikipedia.org/wiki/Freeware" \o "Freeware) obsahujícím [telemetrii](https://cs.wikipedia.org/wiki/Telemetrie" \o "Telemetrie), ale existuje i komunitně sestavovaná varianta VSCodium.

VS Code jsem vybral proto, že už vím, jak se s ním pracuje z minulých ročníků, a taky jsem potřeboval extension Platformio IDE. Ve VS Code se dělají především webové stránky, ale může se zde i programovat v C++, čehož jsem využil.

Jak už jsem se zmiňoval, tak jsem neměl moc na výběr, kde můžu programovat. Práce by měla být na úrovni maturanta, proto jsem uposlechl slov pana učitele Grussmanna a pracoval ve vývojovém prostředí Platformio IDE. Takže o Arduinu IDE jsem si mohl jen nechat zdát.

### **2.3.2 Platformio IDE**

Platformio IDE je nejlepší vývojové IDE pro Microsoft VS Code. Platformio je uživatelsky přívětivé a rozšířitelné integrované vývojové prostředí se sadou nástrojů pro profesionální i amatérský vývoj. Poskytuje nám moderní a výkonné funkce pro zrychlení a zjednodušení vytváření kódu.

Platformio jsem si vybral, protože už mám s ním zkušenost z minulých let.

**Proč zrovna Platformio IDE?**

Je nejen lehký, ale i výkonný editor zdrojového kódu pro různé platformy.

Slouží jako inteligentní dokončení kódu na základě typů proměnných, definic, funkcí a závislostí knihoven.

Pracovní postupy se snadno navigují dle kódových základen jednotlivých projektů, které jsou ve více podoknech a podporují mnoho motivů.

# Způsoby řešení a použité postupy

## 3.1 Funkčnost součástek

Funkčnost součástek jsem ověřoval tak, jak by to řešil člověk zabývající se touto problematikou. Většinu jsem změřil volt metrem, jestli jimi vůbec prochází nějaký proud. Hned tedy ze začátku jsem se potýkal s problémy, že nějaké součástky nefungovaly. Tak jsem musel objednat další a naštěstí mi přišly včas, takže jsem mohl rychle zase pracovat.

Projekt jsem si rozdělil na 3 části.

* Senzorová
* Displayová
* Bzučáková

### 3.1.1 Senzorová část

V této části jsem se zabýval hlavně tím, zda senzor bude fungovat samostatně a jestli Serial monitor bude vypisovat hodnoty měřené senzorem. Hned na začátek se musím přiznat, jsem neměl nejmenší ponětí, jak jím měřit. Musel jsem zhlédnout vice než pár videí a gihubů, než jsem našel to, co jsem hledal. Nakonec se mi podařilo najít efektivní měřící metodu, která byla rychlá a spolehlivá.

void **mereni**()

{

**digitalWrite**(zapis, **LOW**);

**delayMicroseconds**(2);

**digitalWrite**(zapis, **HIGH**);

**delayMicroseconds**(10);

**digitalWrite**(zapis, **LOW**);

  dobaTrvani = **pulseIn**(vypis, **HIGH**);

  vzdalenost = dobaTrvani \* 0.034 / 2;

Měření funguje na principu, že když je status HIGH, provede se operace pulseIn a vypíše měřenou hodnotu.

### 3.1.2 Displayová část

Tato část se zabývá výpisem na LCD display. K tomuto jsem využil knihovnu LiquidCrystal\_I2C, která vlastně umožňuje i jednodušší zápis v kódu. Místo asi 8 řádků jsem napsal jen jeden, ve kterém je napsáno všechno potřebné k fungování displaye.

**LiquidCrystal\_I2C** lcd = **LiquidCrystal\_I2C**(0x27, 16, 2);

Na výpis jsem přišel sám, protože tato část kódu byla jednoduchá. Stačilo přidat funkci lcd a jen vypsat pomocí printu a ještě nastavit, kde se má hodnota vypsat na display. V tomto mi pomohla funkce setCursor.

void **lcdDisplay**()

{

  lcd.**setCursor**(0, 0);

  lcd.**print**("Vypis hodnoty: ");

  lcd.**setCursor**(10, 1);

  lcd.**print**(vzdalenost);

}

Na instalaci displaye bylo nutné, aby se nezaměnily jednotlivé dráty a správně se zapojily. Poté jsem napsal část kódu, který bezchybně fungoval.

### 3.1.3 Bzučáková část

V poslední části jsem zkoušel, zda funguje bzučák společně s LED diodami. Tudíž jsem vše zapojil a zkusil jsem napsat část kódu. Tady jsem dlouho vymýšlel podmínku, kdy bude bzučák bzučet a jak budou svítit či blikat ledky. Nakonec jsem vymyslel, že ledky budou jen svítit, a to zase statusem HIGH nebo LOW. Bzučák je vyřešený statusem tone a notone. A to všechno v jedné podmínce, která je zároveň jednoduchá a zároveň řídí skoro všechno.

  if (vzdalenost < maxVzdalenost)

  {

**tone**(bzucak, tonyFrekvence);

**digitalWrite**(cLed, **HIGH**);

**digitalWrite**(zLed, **LOW**);

  }

  else

  {

**noTone**(bzucak);

**digitalWrite**(cLed, **LOW**);

**digitalWrite**(zLed, **HIGH**);

  }

Zkoušel jsem o dost sofistikovanější podmínky, ale buď nefungovaly tak, jak jsem chtěl, nebo zkrátka nefungovaly vůbec. Tak jsem se dal na zlatou střední cestu a tato pracuje zcela bezchybně.

## 3.2 Kompletní řešení projektu

Když jsem viděl, že všechno funguje, jak má, tak jsem části spojil. Projekt se mi zdál hotov, ale ne nadarmo se říká: „Neříkej hop, dokud nepřeskočíš.” Ne vše fungovalo, jak mělo, a já jsem musel vymýšlet, co s tím. Jednak senzor měřil jen do té doby, než bzučák začal bzučet a display vypisoval tím pádem jen 0 nebo strašně nízké hodnoty. Pár dní jsem přemýšlel, čím by to mohlo být, a přišel jsem na to.

### 3.2.1 Problém s napájením

Do projektu jsem vstupoval s tím, že budu napájet všechno pomocí Arduino konektoru, který má chabých 5 V. Proč chabých? Protože 5 V nestačí napájet naráz LCD display, ultrazvukový senzor, bzučák a ještě LED diody. Proto jsem musel jít do opatření a poradit se s panem učitelem Miekischem, který mi řekl, ať použiju baterii, a to nejlépe 9V. To jsem také udělal a z nefunkčního projektu se stal najednou funkční. Nevyřešilo to všechny problémy, co jsem měl, ale vyřešilo to ten základní, za což panu Miekischovi ještě jednou děkuji.

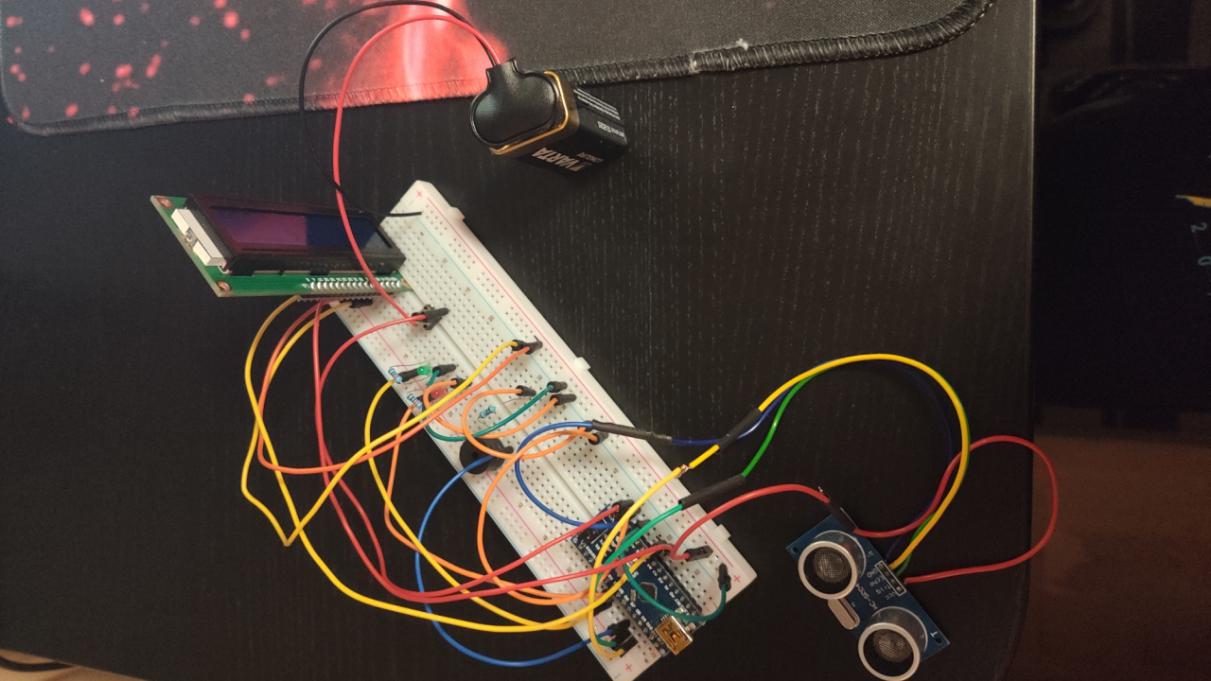
### 3.2.2 Problémy s knihovnami

Problémy se neobjevovaly jen v hardwarové části, ale i v softwarové. Několikrát jsem totiž musel měnit knihovny. Některé z toho důvodu, že neobsahovaly to, co jsem potřeboval a některé zase nebyly na tento projekt stavěné. Takže jsem velkou část smazal a nechal jen jednu jedinou, a to LiquidCrystal\_I2C na zprovoznění displaye.

# Výsledky řešení

## Finální podoba hardwaru

Finální podoba hardwaru je důležitá na zanechání dobrého dojmu z produktu na hodnotitele a samozřejmě zákazníka, který by si chtěl výsledný produkt třeba zakoupit. Já bohužel dobrý dojem opravdu udělat nemohu, protože můj celý projekt je na nepájivém poli a slouží pouze na ukázku. Ale i přes to jsem s hardwarovou částí spokojen, protože zapojení celého projektu jsem řešil samostatně.



*Obrázek č. 8 Závěrečná podoba hardwaru*

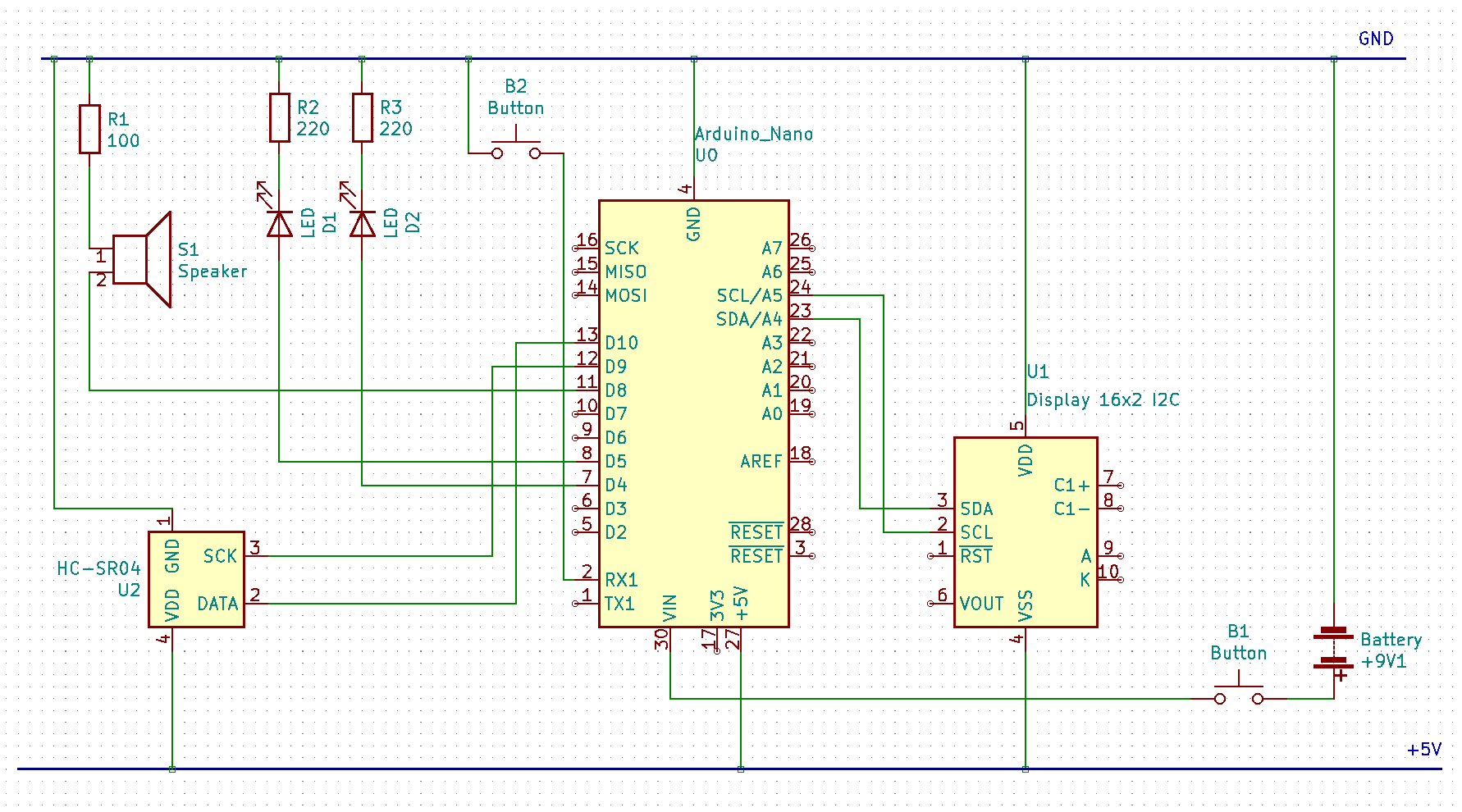
## Finální podoba kódu

S finální podobou kódu jsem taktéž spokojený. Až na pár nedostatků není co vytknout. Kód jsem se snažil správně oddělovat a dávat důležité části do funkcí.

|  |  |
| --- | --- |
| #include <Arduino.h>  #include <Wire.h>  #include <LiquidCrystal\_I2C.h>  #define **DEBUG**  const **byte** buttonInterrupt = 2;  const **byte** cLed = 4;  const **byte** zLed = 5;  const **byte** bzucak = 8;  const **byte** zapis = 9;  const **byte** vypis = 10;  const int tonyFrekvence = 500;  int maxVzdalenost = 100;  int vzdalenost;  **LiquidCrystal\_I2C** lcd =  **LiquidCrystal\_I2C**(0x27, 16, 2);  long dobaTrvani;  void **increment**()  {   maxVzdalenost = maxVzdalenost + 10;   if (maxVzdalenost>200){maxVzdalenost = 150;}  }  void **setup**()  {    Serial.**begin**(9600);  **pinMode**(buttonInterrupt, **INPUT\_PULLUP**);  **attachInterrupt**(**digitalPinToInterrupt**  (buttonInterrupt),**increment**, **FALLING**);  **pinMode**(cLed, **OUTPUT**);  **pinMode**(zLed, **OUTPUT**);  **pinMode**(bzucak, **OUTPUT**);  **pinMode**(zapis, **OUTPUT**);  **pinMode**(vypis, **INPUT**);    lcd.**init**();    lcd.**backlight**();  } | void **mereni**()  {  **digitalWrite**(zapis, **LOW**);  **delayMicroseconds**(2);  **digitalWrite**(zapis, **HIGH**);  **delayMicroseconds**(10);  **digitalWrite**(zapis, **LOW**);    dobaTrvani = **pulseIn**(vypis, **HIGH**);    vzdalenost = dobaTrvani \* 0.034 / 2;    if (vzdalenost < maxVzdalenost)    {  **tone**(bzucak, tonyFrekvence);  **digitalWrite**(cLed, **HIGH**);  **digitalWrite**(zLed, **LOW**);    }    else    {  **noTone**(bzucak);  **digitalWrite**(cLed, **LOW**);  **digitalWrite**(zLed, **HIGH**);    }  }  void **lcdDisplay**()  {    lcd.**setCursor**(0, 0);    lcd.**print**("Vypis hodnoty: ");    lcd.**setCursor**(10, 1);    lcd.**print**(vzdalenost);  }  void **loop**()  {    #if defined(**DEBUG**)    Serial.**println**("Vzdalenost:");    Serial.**println**(vzdalenost);    Serial.**println**(maxVzdalenost);    #endif  **mereni**();  **lcdDisplay**();  **delay**(500);  } |

## Schéma

Schéma je na přání pana učitele Grussmanna udělané v KiCadu. Jsem rád, že jsem si mohl vyzkoušet nějaké schéma vytvořit sám, protože to byla pro mě nová zkušenost, jak s KiCadem, tak se schématy obecně. Trvalo mi nějakou dobu, než jsem se to naučil, ale myslím si, že kdybych takových projektů udělal v budoucnu více, šlo by mi to časem od ruky.



*Obrázek č. 10 Schéma v KiCadu*

# Závěr

Projekt bych označil za dokončený a hlavně funkční. Bylo zde hodně věcí, nad kterými jsem se pozastavil a nad kterými budu přemýšlet ještě hodně dlouho, např. u 9V napájení celého projektu, protože v úvodu jsem využíval 5V napájení, které nefungovalo. Nasbíral jsem nové zkušenosti, co se týče psaní kódu v Platformiu IDE, vytváření schémat v KiCadu, spojování hardwaru a softwaru, pracování s Githubem a v neposlední řadě psaní dokumentace.

Myslím, že tímto si musí projít každý maturant, ať už slabší, nebo jedničkář. Já jsem se naučil jedno: Když chcete něco vytvořit, musíte do toho vložit spoustu času.

Do projektu je stále co přidávat. Napadlo mě i ukládat hodnotu do EEPROM od Arduina, ale už jsem neměl tolik času na realizaci, tak jsem radši od toho ustoupil, než abych se potýkal s dalšími problémy.

Dalším vylepšením by byl větší display, kde by se dalo udělat LiquidMenu a ovládal by se pomocí tlačítek nebo malé klávesnice.

A nakonec by možná mohlo být praktické, dát celý projekt do pěkné krabičky, aby mohl využít svůj plný potenciál.

# Seznam použitýCH INFORMAČNÍCH ZDROJů

[1] DEL SOLE, Alessandro. *Visual Studio Code Distilled: Evolved Code Editing for Windows, macOS, and Linux*. Imprint: Apress, 2019. ISBN 9781484242247.

[2] BELL, Peter a Brent BEER. *Introducing GitHub: a non-technical guide*. Beijing: O'Reilly, [2014]. ISBN 9781491949818.

[3] MATOUŠEK, David. *C++: výukový kurz*. Brno: Computer Press, 2018. ISBN 978-80-251-4906-5.

[4] VIRIUS, Miroslav. *Programování v C++: od základů k profesionálnímu použití*. Praha: Grada Publishing, 2018. Myslíme v.. ISBN 978-80-271-0502-1.

[5] ESPE, Werner a Rudolf REINBACH. *Pájky a pájení*. Přeložil František KAŠPAR. Praha: Vědecko-technické nakladatelství, 1950. Praktické příručky Elektrotechnického svazu českomoravského.

[6] SELECKÝ, Matúš. *Arduino: uživatelská příručka*. Přeložil Martin HERODEK. Brno: Computer Press, 2016. ISBN 978-80-251-4840-2.

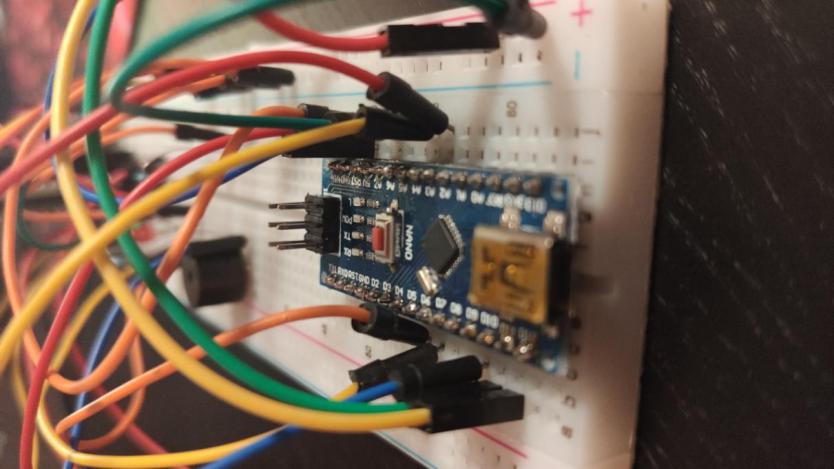
[7] BELL, Peter a Brent BEER. *Introducing GitHub: a non-technical guide*. Beijing: O'Reilly, [2014]. ISBN 9781491949818.

[8] MATOUŠEK, David. *Práce s inteligentními displeji LCD*. Praha: BEN - technická literatura, 2006-. ISBN 80-7300-121-7.

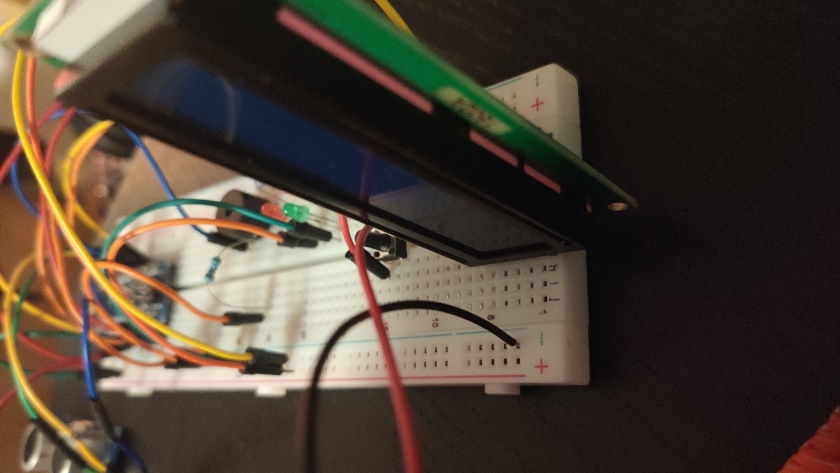
# Seznam příloh

č. 1 Fotodokumentace projektu

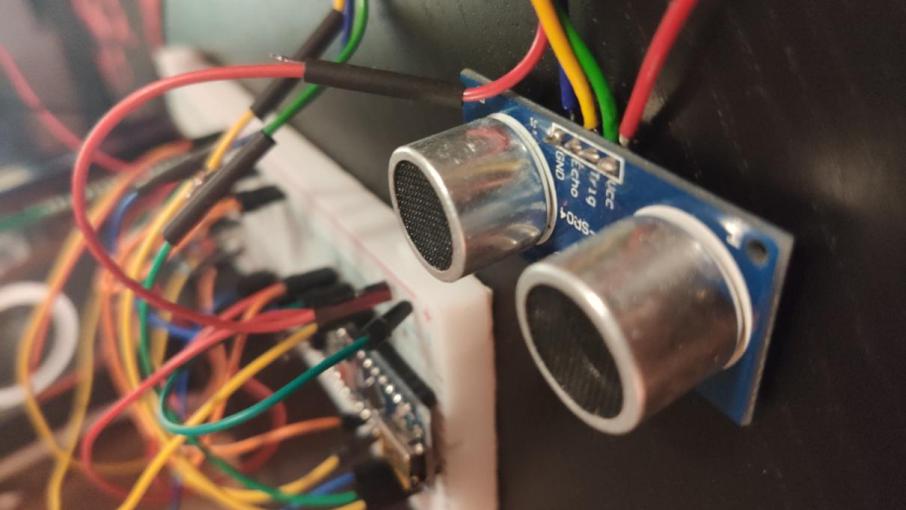
**Příloha č. 1:** Fotodokumentace projektu

**

*Obrázek arduina\_nano*

**

*Obrázek LCD displaye*

**

*Obrázek bzučáku*



*Obrázek napájení 9V baterií*