Pedagogická fakulta
Faculty of Education
Jihočeská univerzita v Českých Budějovicích
University of South Bohemia in České Budějovice

Jihočeská univerzita v Českých Budějovicích
Pedagogická fakulta
Katedra aplikované fyziky a techniky

Diplomová práce

IoT systém sledování hladiny vody a sběr dat – soubor motivačních úloh využitelných ve vzdělávání

Vypracoval: Bc. Karel Beránek
Vedoucí práce: Ing. Michal Šerý, Ph.D.

České Budějovice 2022

Prohlášení

„Prohlašuji, že jsem autorem této kvalifikační práce a že jsem ji vypracoval pouze s použitím pramenů a literatury uvedených v seznamu použitých zdrojů.“

Prohlašuji, že v souladu s § 47b zákona č. 111/1998 Sb. v platném znění souhlasím se zveřejněním své diplomové práce, a to v nezkrácené podobě, elektronickou cestou ve veřejně přístupné části databáze STAG provozované Jihočeskou univerzitou v Českých Budějovicích na jejich internetových stránkách.

V českých Budějovicích dne 7. července 2022 Karel Beránek

Anotace

Cílem práce je navrhnout, vytvořit a otestovat komplexní řídící a měřící systém zásobníku vody a vytvořit ucelený systém pro sběr, archivaci a prezentaci naměřených dat. Reálná naměřená data budou využita v navržených motivačních úlohách pro studenty, kteří tím získají praktickou zkušenost s využitím IoT technologií a s problematikou správného hospodaření s vodou.

Klíčová slova

IoT, ESP32, Arduino, ultrazvuk, akumulace vody, Zabbix

Abstract

The aim of the work is to design, create and test a comprehensive control and measuring system of a water tank and to create a comprehensive system for the collection, archiving and presentation of measured data. Real measured data will be used in the proposed motivational tasks for students, who will gain practical experience with the use of IoT technology and the issue of proper water management.

Keywords

IoT, ESP32, Arduino, ultrasound, water storage, Zabbix

Poděkování

Rád bych touto cestou velice poděkoval panu Ing. Michalovi Šerému, Ph.D. za odborné vedení, trpělivost a cenné rady, ochotu a pomoc, které mi v průběhu zpracování Diplomové práce věnoval.

Obsah

1 Úvod

Zachycení vody, její sledování a její hospodárné využití je předmětem diskuzí na celém světě. Tato diplomová práce se zabývá specifickou problematikou systému pro sledování a hospodárné využití užitkové vody s minimálními pořizovacími a provozními náklady. Realizace mnou popsaného technického řešení je cenově dostupná a je vhodná nejen k výuce, ale i k běžnému provozu.

Popisované řešení je určeno pro studenty základních a středních škol, které mají o popisovanou problematiku zájem, především v zájmových kroužcích nebo při práci s nadanými žáky. Předpokládám, že každý pedagog si sám určí vhodnost jednotlivých témat dle schopností svých studentů a případně si přizpůsobí obsah nebo navržené motivační úlohy. Komplexní pojetí této problematiky propojilo znalosti a zkušenosti z několika samostatných oborů, proto je pro studenty pochopení jednotlivých komponent systému různě obtížné a značně závislé na jejich schopnostech. Například hardwarová část může sloužit kroužkům elektrotechniky, případně informačním technologiím či fyzice, softwarová část je určena spíše pro pokročilé studenty informatiky nebo pedagogy. Sběr a prezentaci dat je možné vhodně použít pro fyzikální úlohy. V oblasti přírodovědy a zeměpisu narazíme na ekologii, získání povědomí o hospodárném využívání cenné komodity, jakou voda dnes je. Všechny obory dohromady spojuje ekonomika, poměr vynaložených nákladů a možných úspor. V předmětech fyziky či matematiky je možné využívat reálných údajů pro vytvoření početních úloh na konkrétních datech, které je možno následně ověřit, či na datech simulovat různé situace, jež by studenty více vtáhly do reality, kde by si uvědomili, že teoretická fyzika má i praktické využití.

Diplomová práce záměrně není podrobným návodem pro realizaci celého řešení, protože aplikace v ní použité se rychle vyvíjejí a mění, v průběhu několika měsíců by byl podrobný návod nepřesný, proto je kladen důraz na neměnné principy fungování, jejich vysvětlení a pochopení, práce uvádí důvody, které vedly ke konkrétnímu řešení, včetně rozborů nevhodných řešení, kterých je nutno se vyvarovat, a naopak obsahuje doporučení i pro možný další rozvoj.

Cílem diplomové práce je navrhnout soubor motivačních úloh použitelných v různých formách vzdělávání mládeže. Úlohy jsou navrženy na základě praktického systému pro monitorování hladiny nádrže na dešťovou vodu s využitím IoT technologií, sběru a prezentaci naměřených dat.

2 Teoretický úvod do problematiky IoT

Definice IoT („Internet of Things“) se zdá být poměrně jednoduchá, český doslovný překlad zní „Internet věcí“, jenže pod tímto překladem je možné si představit neuvěřitelně širokou škálu zařízení, mnohdy ani nejsme schopni určit, co je a co není IoT.

Internet chápeme jako globální síť, ke které se běžně připojujeme, bez které si dnes neumíme představit fungování současného, moderního světa. Kdo by se dnes z nás dobrovolně vzdal možnosti připojení k internetu? Dokáže někdo říci, že internet je zbytečný v civilizovaném, vyspělém světě? Bez internetu jsme před čtyřiceti lety normálně žili a pracovali, na neustálou potřebu být „online“ nikdo tehdy nemyslel, a ano i dnes jsou místa na zemi, kde internet není, a kde se čas možná zastavil, i u nás najdeme ještě poměrně velkou část starší generace, kterou tato technologie neoslovila, a přesto jsou tito lidé šťastní a spokojení, na rozdíl od mnohem mladší generace, která se hroutí s každým přetížením, výpadkem a nedostupností internetové sítě. Kdybychom dokázali vypnout internet, zhroutila by se většina vyspělých ekonomik světa. Důležitost výměny dat po celosvětové globální síti už není v současném světě pro nikoho žádným překvapením, internet je běžnou součástí našich životů. Pro potřeby IoT je internet jen prostředek ke komunikaci s okolním světem, lidmi nebo zařízeními navzájem. Ve skutečnosti lze podstatu definice chápat jako svět fyzických zařízení připojených do informační sítě, aby mohly komunikovat s okolím. Technicky vzato, přímé připojení k internetu není nutnou podmínkou pro jejich správné fungování, přestože je přímo slovo internet obsaženo v názvu. Jakmile nějaké zařízení samo dokáže komunikovat se svým okolím, je možné jej považovat za IoT zařízení. Neexistuje definice, jak má IoT zařízení vypadat, jaká je jeho velikost nebo jaký má maximální výpočetní výkon a podobně. Všeobecně se předpokládá zařízení s vlastním procesorem, pamětí, s nízkou spotřebou energie a schopné dlouhodobého provozu, dokáže sbírat a odesílat informace, přijímat informace a reagovat na ně.

Sběr informací

Sběr informací v technickém světě je postaven na fyzikálním měření, které nutně ke své činnosti potřebuje čidla a senzory, díky nim měříme například teplotu, tlak, zvuk, napětí, proud, hmotnost, chemické vlastnosti, rychlost, intenzitu osvětlení atd.

Zároveň je možné sbírat již zpracované informace z jiných zdrojů, například předpověď počasí je možné získat z jiného zařízení nebo serveru – sbírají se tedy informace, data.

Reakce

Na získané informace je možné reagovat prostřednictvím akčních členů - řídících obvodů, které ovládají další elektrická zařízení jako například motory, elektromagnety, světelné zdroje, spínače a tím dojde k skutečné „akci“, nebo je možné informace předávat dalšímu systému, například odeslat email, uložit data do databáze, cloudu nebo předat dalšímu IoT systému.

Za posledních dvacet let se svět IoT hodně změnil, už není pouze pro bohaté firmy a instituce, výroba elektroniky je stále levnější a výkon zařízení větší, klesá spotřeba a rozměry. V řádech stokorun je dnes možné pořídit komerční zařízení pro „chytrou domácnost“ jako například dálkově ovládané zásuvky nebo žárovky, komunikující s hlasovým asistentem v domácnosti nebo v mobilním telefonu. Do stejné kategorie, nazývající se „spotřební“, patří například chytré spotřebiče jako jsou ledničky, pračky, mikrovlnné trouby, dále okenní rolety a žaluzie, zavlažovací systémy, osvětlení, řízení vytápění nebo klimatizace. Mezi průmyslové IoT je možné zahrnout zařízení využívané ve výrobních podnicích v samotné výrobě, ke sledování kvality, produkce, v logistice a následných analýzách dat v oblasti efektivity, ekonomiky, a to i v zemědělství, v dopravě, a podobně. Obě kategorie se vzájemně prolínají, odlišnost spočívá především v ceně, v požadavcích na obsluhu nebo instalaci a zprovoznění.

2.1 Historie zadržování vody

První vodní, technická díla, která přiváděla potřebnou vodu do míst, kde jí byl nedostatek evidujeme v Čechách již ve 12. století, tehdejší přivaděče měly formu otevřených koryt či vodovodních řádů vyrobených z různých materiálů (dřevo, kámen, pálená hlína, olovo), fungovaly na principu samospádu, tedy gravitace bez čerpací techniky. První projektovaný vodovod byl vybudován pro potřeby Strahovského kláštera, přiváděl kvalitní vodu z pramenů Petřína pomocí několika kanálků do klášterních budov a do kašny, voda stékala do vytesané nádrže a odtud do velkého bazénu v rajském dvoře, tento románský vodovod sloužil Strahovu až do konce 16. století, následovala modernizace, a i dnes tato voda stále zásobuje Strahovský klášter. Zmínit můžeme i jeden z nejstarších přivaděčů vody pro samotný Vyšehrad, který nechal vybudovat český kníže Vladislav II. po roce 1140. V roce 1333 byly zahájeny práce na Zbraslavském olověném vodovodu, v malém výčtu technických nejstarších vodovodních památek nesmíme zapomenout zmínit přívody vody pro samotný Pražský hrad a jeho zahrady, k jehož

rozvoji došlo s nástupem Habsburků, nový vodovod přiváděl užitkovou vodu z rybníků u obcí Chýně a Hostivice přes tzv. Královský reservoár, dnes Libocký rybník. Dnešní Praha čerpá vodu ze Želivky a z reservoárů v Jesenici, k distribuci vody v Praze slouží 65 vodojemů, 48 čerpacích stanic a přibližně 3500 km vodovodních řádů. Zajištění užitkové vody, úprava vody a její doprava ke spotřebitelům je problematikou s dlouhou tradicí. [1]

Ačkoli vodu dnes zadržuje 21 velkých přehrad s rozlohou větší jak 2,5 km2, dále 136 přehrad s menší rozlohou a zhruba na 23.000 malých vodních nádrží tedy rybníků, tak v současné době se nejen užitková ale i dešťová voda stává hlavním tématem zpráv, slýcháme o nedostatku užitkové vody, nedostatku srážek, Česko se dlouhodobě potýká s velkým suchem, proto v posledních letech bylo schváleno mnoho projektů, jejichž úkolem je pomoci zadržovat vodu v krajině a posílit tak lokální vodní ekosystémy. [2] Vracíme se zpět do minulosti, kdy v průběhu několika let jsme zlikvidovali třetinu všech vodních ploch v České republice, mnohé takto zrušené rybníky byly následně využity jako zemědělská půda a dnes rybníky čistíme, revitalizujeme a obnovuje, do krajiny vracíme zpět mokřady a rašeliniště, vystavujeme remízky atd. Stát v poslední době vytipoval 65 území, která jsou vhodná pro potenciální vybudování přehradních nádrží, v roce 2015 byly zahájeny práce na prvních dvou nádržích u Pěčína a Vlachovic a buduje i lokální nádrže: Senomaty, Šanov atd. [3] Vodu stále považujeme za běžnou věc, v mnohých částech světa bohužel běžná není, téměř jedna miliarda lidí nemá přístup k nezávadné vodě, 2,5 miliardy lidí nemají lepší hygienické zařízení. Zdravotní a ekonomické dopady jsou ohromující, každým rokem zemře 1,4 milionu dětí na průjmy způsobené znečištěnou vodou, evidujeme klimatické změny. Pod tíhou všech těchto výše uvedených okolností se mění i legislativa v České republice. Voda se stává cennou a chráněnou komoditou.

Při porovnání s výše uvedenými stavbami snadno můžeme nabýt dojmu, že malé nádrže budované soukromými subjekty nemohou nijak pomoci se současným způsobem hospodaření s vodou. To ovšem není pravda! Jedním důkazem takové podpory je například dotační program „Dešťovka“ pravidelně vypisovaný Státním fondem pro životní prostředí ČR, jehož hlavním cílem je motivovat vlastníky a stavebníky rodinných a bytových domů v celé republice k udržitelnému a efektivnímu hospodaření s vodou a snížit tak množství odebírané pitné vody z povrchových a podzemních zdrojů. [4] Dotace může pokrýt až 50 % nákladů na pořízení celého systému. V závislosti na zvoleném způsobu využití to může být až 65 tisíc korun. [5]

3 Návrh systému pro měření, sběr a dlouhodobé uchování fyzikálních veličin

Navržený systém umožňuje sledování množství vody v podzemní nádrži s dešťovou vodou a sám aktivně hlídá provozní stavy a dokáže reagovat na nedostatek vody, vzniklé poruchy nebo provozní odchylky a informovat různými způsoby uživatele, případně zablokovat spuštění čerpadla, které z nádrže vodu odebírá a zabrání tím škodám. Informace je možné zobrazit na displeji připojenému k tomuto zařízení, sledovat na počítači, mobilním telefonu a podobně, zároveň uchovává historii naměřených hodnot a dokáže je prezentovat v libovolném čase zpětně. Systém umožňuje udělit různé přístupy úrovně sledování různým uživatelům, například zvláštní přístupy pro administrátory a jiné pro studenty, je koncipován modulárně, s možností rozšíření nebo výměny jednotlivých částí. Naměřená data jsou předávána prostřednictvím WIFI sítě do lokálního nezávislého serverového systému, navržený systém není proto omezen jen na jednu konkrétní činnost - měření hladiny, ale umožňuje připojení prakticky libovolného počtu IoT zařízení a měřit libovolné veličiny.

Důležitou částí celého sytému je nádrž na dešťovou vodu, bez které by nebylo možné nic měřit, je stavebně zasazená pod úroveň terénu s pomocnou šachtou s tepelně izolovaným víkem, zabraňujícím promrzání v zimním období, tak aby byl umožněn bezproblémový celoroční provoz. Uvnitř nádrže je ve vhodné výšce nad hladinou umístěno vodotěsné ultrazvukové čidlo, které měří vzdálenost k hladině vody, jeho elektronika je situována vně nádrže, v utěsněné elektroinstalační krabičce, ale mimo vlhké prostředí. Čidlo komunikuje s jednočipovým počítačem, který se nachází u čerpadla uvnitř budovy a je spolu s dalšími periferiemi součástí elektroinstalace, potřebné k jeho sledování a ovládání. Ze znalosti rozměrů nádrže, umístění ultrazvukového čidla a jeho aktuální vzdálenosti od hladiny vody je následně vypočítán stav zaplnění nádrže. Všechna naměřená data jsou přenášena pomocí WIFI do centrálního systému (serveru), kde je umožněna jejich další prezentace, archivace a zpracování, samotná prezentace je možná v číselné formě, graficky nebo textově. Serverový systém umožňuje pro potřeby výuky i uživatelský přístup přes webové rozhraní, studenti tak mohou sledovat naměřená data v reálném čase, rozebírat historii a na základě těchto dat dokonce predikovat jejich vývoj.

Při návrhu systému jsem se snažil co nejvíce aplikovat moderní technologie, které jsou v dnešní době běžně označované pod společným názvem „IoT“ avšak za přijatelnou

cenu, modulární a snadno dostupné. Celý systém vznikal několik let a postupně se rozšiřoval o různé komponenty a funkční bloky. V první fázi systém pouze sledoval hladinu a tím množství vody v nádrži. Postupně bylo nutné přidat blokaci třífázového čerpadla při nedostatku vody nebo poruše. Dalším logickým rozšířením byl systém pro archivaci naměřených hodnot v souvislosti s automatickým zavlažováním. V praxi se ukázalo, že znalost historických dat, spotřebovaného i dostupného množství vody včetně predikce dalšího vývoje je zcela zásadní pro správné hospodaření s omezenými zdroji, jakou bezpochyby dešťová voda je. Není bohužel možné vybudovat neomezeně velkou nádrž. Potřebné stavební úpravy a mnohdy i složité místní podmínky často znemožňují dodatečné vybudování dostatečně objemné nádrže. To vše se promítá do pořizovacích nákladů a je samozřejmě nutné uvažovat o poměru nutných investic vzhledem k úsporám, které toto řešení v praxi přináší.

3.1 Nádrž na vodu

Z výše uvedeného textu vyplývá, že nádrž na dešťovou vodu je bezesporu klíčovou součástí celého systému, přesto se jí budu věnovat jen zčásti, protože její vybudování souvisí s konkrétními podmínkami, které patří do kategorie stavebních úprav v místě realizace podle možností budovatele. Zmínit však mohu několik variant, které je možné popsat a zvážit jejich jednotlivé výhody a nevýhody.

3.1.1 Podzemní nádrž

Podzemní nádrž je nejběžnější í variantu, často je nazývána „akumulační“, protože slouží převážně k dlouhodobému zadržení vody, nebo je označována jako „retenční“, jelikož převážně zpomaluje odtok dešťových vod a zabraňuje možnému přetížení kanalizační soustavy v době přívalových nebo intenzivních dešťů. Samotná retenční nádrž není vhodná pro akumulaci vody, je však možné vybudovat retenční nádrž, která zároveň část vody akumuluje a část zpomaluje, je tedy kombinací obou uvedených typů.

Mezi nesporné výhody podzemní nádrže patří:

- dlouhá životnost – není vystavena nadměrné zátěži způsobené UV zářením, mrazem a teplem,
- možnost celoročního provozu – při vhodném umístění potrubí,
- redukce růstu řas a jiných mikroorganismů – nízká teplota vody a slunečního záření,

- neomezuje plochu pozemku – nad nádrží může být například trávník nebo zpevněná plocha.

Při realizaci podzemní nádrže je nutné počítat s možnými administrativními i technickými potížemi. Pokud vyřešíme podmínky, jakými jsou rozměry a umístění na pozemku, zůstane pouze jeden problém a tím je spodní voda. Fyzikální zákony platí pro všechny, Archimédův zákon nevyjímaje. Na prázdnou nádrž působí obrovské vztlakové síly, které mohou zdeformovat její konstrukci, mohou nadzvednout povrch nad ní, nebo dokonce může celá nádrž vyplavat nad úroveň terénu, přesto je podzemní nádrž v současné době jedna z nejčastěji realizovaných staveb a u novostaveb je často nutnou podmínkou pro stavební povolení, v takovém případě je stavba posuzována v rámci celého projektu a je jeho součástí. Při dodatečné pozdější realizaci není dle § 103 Zákona o územním plánování a stavebním řádu (stavební zákon) 183/2006 Sb potřeba stavební povolení do objemu 50 m^3 a do výšky 3 m, i tak je třeba získat územní souhlas příslušného stavebního úřadu. Při využívání dešťové vody v domě (například pro splachování nebo praní) je nutné kontaktovat svého provozovatele kanalizace a upravit smlouvu o ceně stočného, většinou se jedná o instalaci podružného vodoměru nebo paušální navýšení ceny, pro účely zavlažování není úprava smlouvy nutná.

3.1.2 Nadzemní nádrž

Mezi nadzemní nádrže je možné zařadit sudy, barely, plastové jímky, konve, ale i samostatné stavby. Při dodržení správných výškových poloh je možné kombinovat a spojovat různé nadzemní nádrže do větších celků a tím zvětšit celkový objem zadržované vody.

Výhody:

- Snadná realizace s možností rozšíření o další kapacitu.
- Cenová dostupnost.

Nevýhody:

- Nemožnost celoročního provozu, nutné odstavení v zimním období.
- Větší znečištění vlivem vyšších letních teplot.
- Kratší životnost plastových nádrží způsobená povětrnostními vlivy (UV záření, střídání teplot, mráz).

Nadzemní nádrže větších rozměrů jsou brány jako stavby a postupuje se podle platné legislativy.

3.1.3 Studna

Studna je důležitým zdrojem vody, nejen užitkové, ale i pitné. Původně tyto technické stavby sloužily především k jímání a odběru podzemní vody, bohužel dnes však existují studny bez vody, proto je možné je po úpravě využít pro akumulaci dešťové vody, takže vlastně vznikne svislá nádrž, nebo si takovou nádrž ve tvaru studny můžeme nově vybudovat. Zákon studny definuje pojmem „vodní dílo“, legislativně se jedná o jiný proces, při stavbě se postupuje odlišně než v případě akumulační nádrže.

Výhody:

- Není nutná žádná dodatečná filtrace a následná úprava vody.
- Možný celoroční provoz.
- Množství dostupné vody přímo nesouvisí s aktuálními dešťovými srážkami, nejedná se o zdroj povrchové vody.
- Dlouhá životnost.
- Estetika.

Nevýhody:

- Technicky náročnější realizace.
- Složitější legislativa při nové realizaci.
- U vrtané studny (malý průměr, velká hloubka) není možné sledování hladiny ultrazvukem.
- Pokud se z nějakého důvodu ztratí voda nebo je jí nedostatek, není zpravidla možné zvýšit její kapacitu.

3.1.4 Retenční nádrže

Posledním typem, který zmíním jsou přírodní, otevřené retenční nádrže, ty na rozdíl od rybníků nedisponují samostatným přívodem vody, jsou tedy plně závislé na dešťové vodě, jsou zasazeny do krajiny, existují v různých velikostech a mohou mít i další funkce, jelikož se jedná o stavbu, tak je nutné postupovat v souladu s příslušnými zákony a povoleními.

Výhody:

- Zdroj vody slouží k zavlažování.
- Estetika, klidový režim.
- Možnost využití rázu krajiny a tím minimalizovat náklady.

Nevýhody:

- Technicky náročnější realizace.
- Vyšší pořizovací náklady.
- Je nutná filtrace.
- Není možný celoroční provoz.
- Stavební povolení.

3.2 Technické zázemí

Při návrhu technického řešení je nutné počítat s umístěním nejen elektrických součástí, ale i vodoinstalačních jako: čerpadlo, filtrace, rozvaděč, elektronika a podobně. Jednotlivé varianty provozu jsou závislé na způsobu využití a jejich ceně. Pro letní provoz, kde převažuje zavlažování zahrad, je možné veškerou technickou infrastrukturu umístit mimo budovu, postačí dřevěný přístřešek, případně mělká šachta pod úrovní terénu. Při celoročním provozu je nutné technické zázemí umístit v budově nebo v šachtě v nezámrzné hloubce.

Návrh technického zázemí lze posuzovat dle různých kritérií, například:

- Ponorné čerpadlo nebo externí čerpadlo se sacím potrubím:
  - ponorné čerpadlo má napájecí kabel ponořený do vody a je náchylnější na nečistoty,
  - externí čerpadlo má sací potrubí omezené svou délkou a hloubkou čerpání.
- Dostupnost napájení, jednofázové nebo třífázové:
  - větší vodárny výrobce „Sigma“ jsou převážně třífázové,
  - elektroinstalace pro třífázové čerpadlo je složitější,
- Prostor pro umístění IoT zařízení:
  - nejlépe samostatný rozvaděč,
  - prostor pro akční a řídící prvky (stykače, proudové ochrany, SSR relé, EP ventily).
- Filtrace:
  - nutný dostatečný prostor pro umístění a následnou manipulaci, dimenzovat dle znečištění.
- Přirozený odtok vody - kanalizace:
  - při servisních zásazích nebo poruše musí nežádoucí voda někam odtéct, jinak může dojít například k zaplavení čerpadla nebo elektroinstalace.

- Hlučnost čerpadla je nutné řešit v případě jeho umístění v budově.
- Požadovaný výkon v oblasti průtoku a tlaku.
- Dostupná vzdálenost pro komunikaci mezi čidly a IoT.
- Dostupnost sítě WIFI nebo LAN.

3.3 IoT zařízení

Stěžejní částí technického řešení, ale i mé práce, je vytvoření a popis vlastního IoT zařízení, jeho propojení a vazby s dalšími systémy, infrastrukturou a systémem pro sběr dat s návazností na edukaci studentů. V průběhu realizace se ukázalo, že návrh vlastního řídícího systému je nezbytný, protože komerční produkty podobného typu používají jen základní možnosti nastavení, nepřipouští rozšiřitelnost a neexistuje k nim podrobná technická dokumentace určená veřejnosti. Na trhu najdeme systémy, které měří množství vody v nádrži a například při jejím nedostatku doplní automaticky minimální množství pitnou vodou nebo zablokují čerpadlo, případně jen měří výšku hladiny a zobrazí ji pomocí sloupce led diod, ovšem ne vždy na vhodném místě, takovým řešením je například: „Hladinoměr - RAINMASTER ECO-FS“ (https://eshop.destovka.eu/as-rainmaster-eco-10). Komerční, řídící systémy pro domácnost nepočítají s možností ukládání a uchovávání naměřených dat, nezabývají se historií a její návaznosti na další systémy jako například automatická korekce množství zavlažování v závislosti na dostupném objemu vody nebo automatické vypínání zahradního okruhu v závislosti na čase nebo různých provozních stavech, to je prakticky nemožné. Existují komerční systémy vytvořené na zakázku, ty jsou bohužel cenově velmi nákladné, a pro použití při výuce k demonstraci možností IoT technologií a sběru dat se vůbec nehodí. Řešení IoT sice chápeme jako celek, ale pro větší přehlednost a možnost porovnávat jednotlivé komponenty jsem dále čidla a akční členy od řídícího modulu oddělil.

Stanovené podmínky pro technický návrh byly následující:

- Výška hladiny je měřena způsobem nepodléhajícímu korozi, znečištění nebo zamrznutí.
- Jednoduchost a cenová dostupnost součástí.
- Snadná rozšiřitelnost a možná aktualizace v budoucím čase.
- Připojení k počítačové síti pro zobrazení dat a synchronizaci času.
- Zobrazení potřebných dat na displeji.

- Dostatečný počet vstupních/výstupních portů (pinů) pro připojení čidel a periférií:
  - čidlo hladiny,
  - teplotní čidlo,
  - LCD displej,
  - relé pro blokaci čerpadla,
  - tlakové čidlo,
  - ovládání ventilů pro zavlažovací systém nebo dopouštění nádrže pitnou vodou.
- Automatické nastavení času z internetu.
- Ovládání a konfigurace na dálku pomocí mobilního telefonu, minimum fyzických tlačítek.
- Ukládání historie naměřených dat.
- Jednoduché napájení celého systému, nejlépe pouze 5 V a nízká spotřeba.

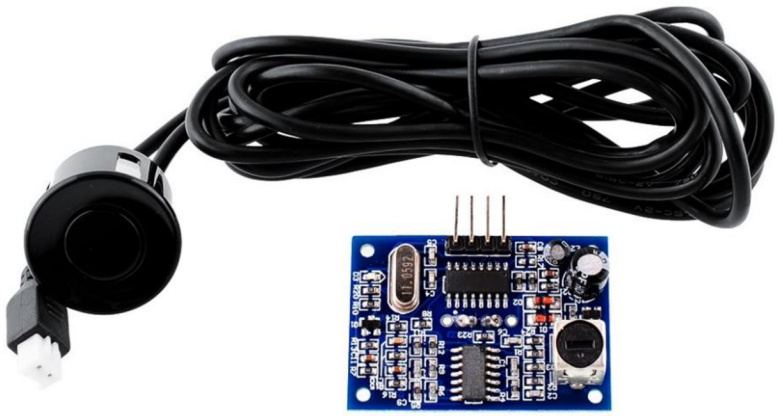
3.3.1 Čidlo hladiny

U úvodu této kapitoly musím konstatovat, že všechny způsoby měření pomocí plováků jsem automaticky vyloučil, důvodem je znečištění mechanizmu a s ním spojená, komplikovaná instalace a problematická údržba, u hlubokých podzemních nádrží prakticky neumožňuje takový systém použít. Vyloučil jsem také měření pomocí tlakových čidel, protože jsou vhodná jen pro případ nadzemní nádrže, při umístění do otvoru u dna, toto použití sebou nese technické problémy s kalibrací, znečištěním a korozí, u podzemní nádrže je použití prakticky nemožné. Vyloučil jsem i měření izolovaným kabelem ponořeným do vody, pracujícím na fyzikálním principu změny kapacity z důvodu možné koroze při kontaktu s vodou, zároveň z praktické zkušenosti vím, že na stěnách nádrže se časem vytváří povlak nečistot a to znamená, že na kabelu by tyto nečistoty byly také, z to vyplívá, že znečištění kabelu by ovlivňovalo výsledky měření, musel by být pravidelně čištěn a kalibrován při minimální a maximální hladině. V případě velkých nádrží to představuje značnou komplikaci. Samotné měření optickým přístrojem (laserem) je nákladné a do vlhkého prostředí nevhodné.

Z výše uvedených negativních důvodů a s ohledem na stanovené technické podmínky mě jako jediná možná varianta řešení vyšlo měření pomocí ultrazvuku, protože čidla tohoto typu jsou běžně dostupná, můžeme je najít například v náraznících osobních automobilů, pod názvem „parkovací senzory“, jsou opravdu vodotěsná a drobné znečištění pro ně nepředstavuje problém.

Zvolil jsem vodotěsný ultrazvukový modul pro měření vzdálenosti, dostupný pod označením JSN-SR04T (Tento modul najdete například na webových stránkách https://dratek.cz/arduino/1306-ultrazvukovy-vodotesny-modul-pro-mereni-vzdalenosti-pro-arduino.html za cenu 289 Kč).

Obrázek 1 - Fotografie vodotěsného ultrazvukového čidla JSN-SR04T [6]



Technické parametry čidla:

- Typ: JSN-SR04T.
- Provozní napětí: DC 5 V.
- Klidový proud: 5 mA.
- Při měření cca: 30 mA.
- Frekvence: 40 khz.
- Nejvzdálenější bod měření: 4,5 m.
- Nejbližší bod měření: 25 cm.
- Rozměry modulu: 41 mm * 28,5 mm.
- Rozlišení: 0,5 cm.
- Pozorovací úhel: 45–75°.
- Provozní teplota: -10 ~ 70 °C.
- Skladovací teplota: -20 ~ 80 °C.

Vlastnosti:

- malé rozměry, snadné použití,
- nízké napětí, nízká spotřeba energie,
- vysoká přesnost,

- silné anti-rušení,
- vodotěsné pouzdro senzoru.

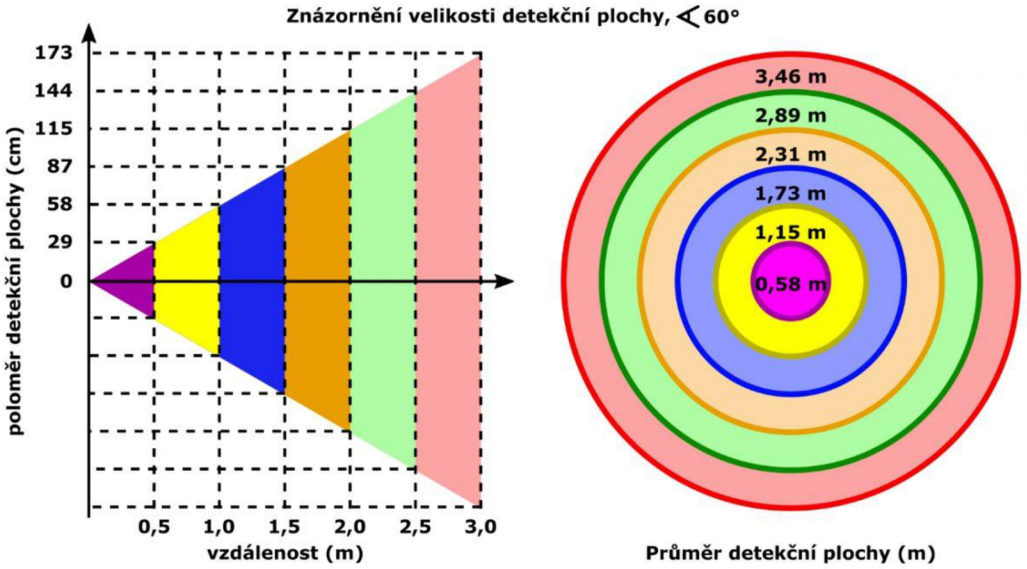
3.3.1.1.1 Princip měření pomocí čidla:

Na řídící pin modulu s označením „Trig“ je přiveden krátký impuls (10 µs), který v modulu spustí proces měření, ultrazvukový reproduktor vyšle signál o frekvenci 40 kHz a přepne elektroniku tak, že je reproduktor zapojen k vstupním obvodům ve funkci mikrofonu, dále samotný modul přepne výstup „Echo“ na logickou 1 a čeká na odraz zvuku stejné frekvence, jakmile je zachycen první odraz, elektronika okamžitě změní signál logické úrovně pinu „Echo“ na hodnotu logická 0 a ukončí měření. Samotné měření není nijak číselně vyjádřeno, programem sami musíme změřit čas mezi sestupnou hranou impulzu „Trig“ a sestupnou hranou na pinu „Echo“, výsledná vzdálenost je vypočítána dle vzorce jako součin času a rychlosti zvuku ve vzduchu dělený dvěma, protože délka trasy zvukové vlny je dvojnásobná – počítá se odraz od měřené plochy ($s = vt/2$).

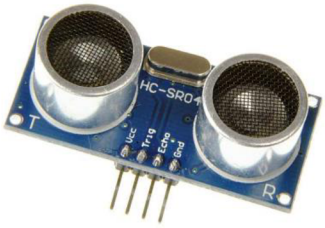
Důsledkem přepnutí režimu z reproduktoru na mikrofon a délkou generovaného zvukového impulzu je nemožnost měřit po krátkou dobu odražený signál, to se projeví minimální možnou naměřenou vzdáleností, která je 21 cm, s touto vlastností je nutné počítat při umístění čidla nad hladinu vody, nesmí být blíže než 21 cm od maximální možné hladiny vody.

Ultrazvuk se z čidla šíří jako vlna, ve tvaru kužele, která je tlumena s rostoucí vzdáleností, takže zpětný odraz vlny zachycený čidlem musí mít určitou intenzitu, pokud je detekovaná plocha příliš daleko, v tomto případě více než 4,5 metru, není intenzita zpětného odrazu dostatečně silná a čidlo už neměří. Překážky umístěné v nádrži, v prostoru vyzařovaného kužele ultrazvukových vln, mohou negativně ovlivnit měření, dokonce jej mohou znemožnit. Pro každé vlnění platí, že může být odraženo, pohlceno nebo jen částečně pohlceno, proto mohou být i ultrazvukové vlny odráženy, je možné ovlivňovat směr a tvar jejich šíření, například vhodnou ozvučnicí. Umístěním čidla do plastové trubky můžeme využít jeho limitující oblasti v rozsahu 0 až 21 cm, kde čidlo neměří a částečně tím upravit velikosti detekční plochy a vyzařovaného kužele. Obrázek 2 ukazuje velikost kužele a velikosti detekční plochy při vyzařovacím úhlu 60°.

Obrázek 2 - Detekční vzdálenosti ultrazvukového čidla [6]



Pro výukové účely je možné také použít výrazně levnější modul ultrazvukového čidla HC-SR04, který není sice vodotěsný, nicméně pro demonstraci a oživení systému na pracovním stole je naprosto dostačující, je softwarově i elektricky shodný s JSN-SR04T, nejsou nutné žádné úpravy, jeho cena se se pohybuje okolo 40 Kč, a je běžnou součástí výukových setů s Arduinem.



Obrázek 3 - Výukové ultrazvukové čidlo HC-SR04 [6]

3.3.2 Jednočipový počítač

Jednočipový počítač je standardně označován jako SoC (System on a Chip). Rozdíl mezi klasickým procesorem v běžném počítači je především v implementaci všech součásti jako například: procesor, operační paměť, úložná paměť, grafický procesor, komunikační rozhraní do jedné součástky, jednoho pouzdra, jednoho čipu. Dále se v mém textu používají další označení - čip nebo procesor a představují stejnou součástku, jednočipový počítač SoC. Běžně se takové typy procesorů používají v mobilních telefonech, tabletech, spotřební elektronice, mini počítačích, které jsou ve světě IT velice populární (například Raspberry PI). Každý typ SoC má své zaměření a je

vhodný pro konkrétní účely, jsou vyráběny na zakázku, přesně podle zadání zákazníka, k takovým bohužel není možné získat dokumentaci, protože se může jednat o patentové či obchodní tajemství, použití jinou fyzickou či právnickou osobou může být vázáno autorskými a licenčními právy. Na trhu můžeme získat i takové, které je možné programovat bez speciálních a velmi nákladných pomůcek, vývojového prostředí a podobně.

Při řešení technického návrhu jsem uvažoval o jednočipovém počítači, dostupném ve školách, vycházel jsem z předpokladu, že stavebnice Arduino a k ní odpovídající vývojové prostředí jsou dnes již standardním vybavením. Při porovnání dokumentací a veřejně dostupných informací různých desek z této kategorie jsem vybral dva možné kandidáty:

- ATMEL AVR MEGA 2560
- Espressif ESP-WROOM-32

Oba procesory po výkonové stránce naprosto vyhovují, jsou podporovány přímo v Arduino IDE vývojovém prostředí a jsou dostupné na českém trhu. V následující tabulce uvádím v porovnání obou procesorů pouze ty parametry, které mají vliv na navrhované řešení a je vůbec možné je mezi sebou porovnávat:

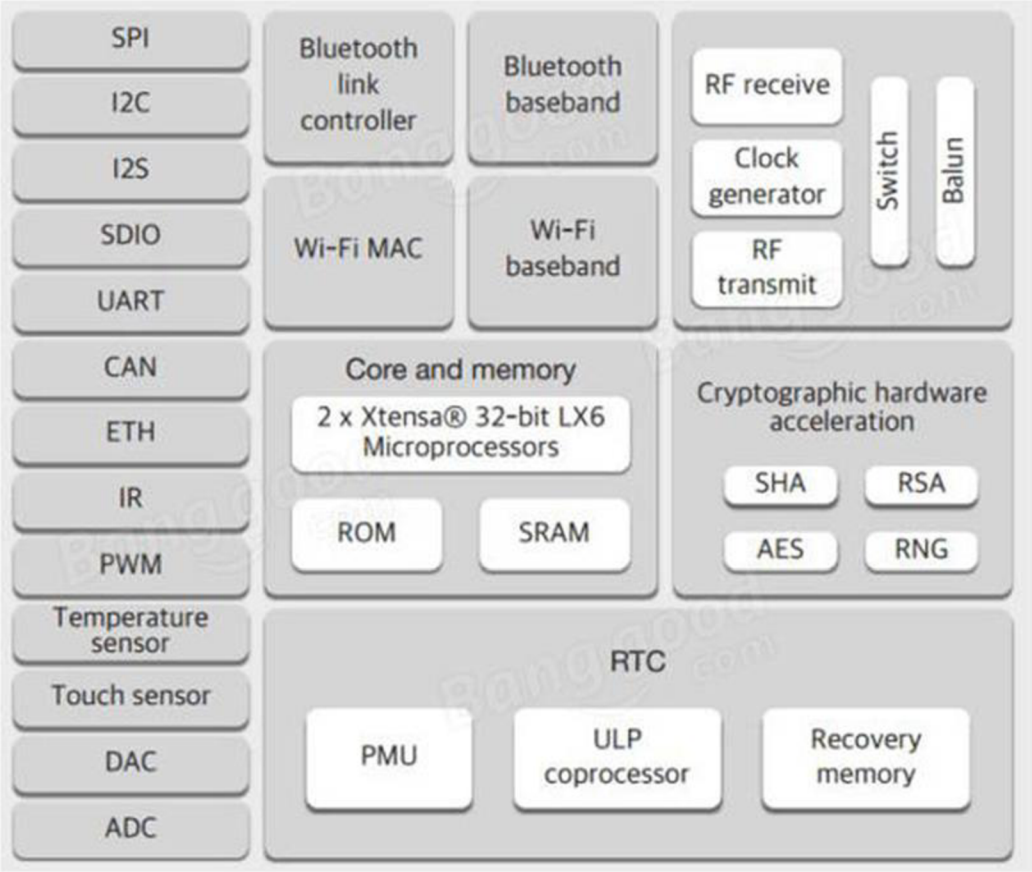
Tabulka 1 - Porovnání jednočipových počítačů

	MEGA 2560	***ESP-WROOM-32***
architektura	CMOS 8-bit Advanced RISC Architecture	32-bit RISC
počet jader	1	2
paměť Flash	256 kB (z toho 4 kB bootloader) životnost max. 10 000 zápisů	448 kB
paměť EEPROM	4 kB životnost max. 100 000 zápisů	4 MB
Paměť SRAM	8 kB	520 kB + 8 kB fast + 8 kB slow
Externí paměť	až 64 kB	16 MB flash 8 MB SRAM
frekvence	1-16Mhz	40 MHz až 240 MHz
MIPS	16	

počet instrukcí	135	-
registry	32x8bit	16x32 4x16
Digitální I/O piny	86	34 GPIO Matrix IO_MUX
PWM piny	12 (16-bitové)	3+16 (16-bitové)
Analogové piny (ADC)	16 (10-bitové)	2 (12-bitové)
UART (hardwarové seriové linky)	4	3
SPI (Serial Peripheral Interface)	1	3
Rozsah pracovní teploty	-40°C až 85°C	-40°C až 85°C
Napájení	4,5 V – 5,5 V	2,7 V - 3,6V
Spotřeba max	-	500 mA
Spotřeba průměr	20 mA	80 mA
Spotřeba sleep	od 0,1 uA	od 0,1 uA
I2C	1	2
SD/SDIO/MMC Host Controller	ano, SW	ano, HW
WIFI	ne	Ano
Bluetooth	ne	Ano
cache	ne	32 kB pro externí úložiště

Vzhledem k absenci WIFI modulu u procesoru ATMEL je nutné počítat s dokoupením dalšího hardware, což komplikuje řešení s ohledem na cenu a rozměry, proto byl dále vybrán pouze čip ESP-WROOM-32. Tento čip byl také vybrán proto, že jeho cena je vzhledem k procesoru ATMEL dokonce nižší, samotný čip bez vývojové desky a regulátoru napětí se pohybuje přibližně od 60 do 130 Kč, zároveň jej najdeme v mnoha variantách a na nepřeberném množství vývojových desek a stavebnic, dokonce rovnou obsahuje WIFI a Bluetooth modul, který extrémně zjednodušuje další použití pro IoT technologie. Výkonové parametry ESP-WROOM-32 jsou i při srovnání s nejvýkonnějším čipem ATMEGA 2560 několikanásobně lepší, vítězí proto procesor řady ESP32. Zmínit u něj můžeme i hardwarovou akceleraci šifrování, 4 MB paměť flash, kterou lze dělit na více částí a tím ji využít nejen pro vlastní programový kód, ale i pro souborový systém, dvě procesorová jádra, rozdělené podle typu - aplikační a komunikační, více operační paměti i možnost adresace většího rozsahu externích paměťových modulů, výrazně větší rychlost…

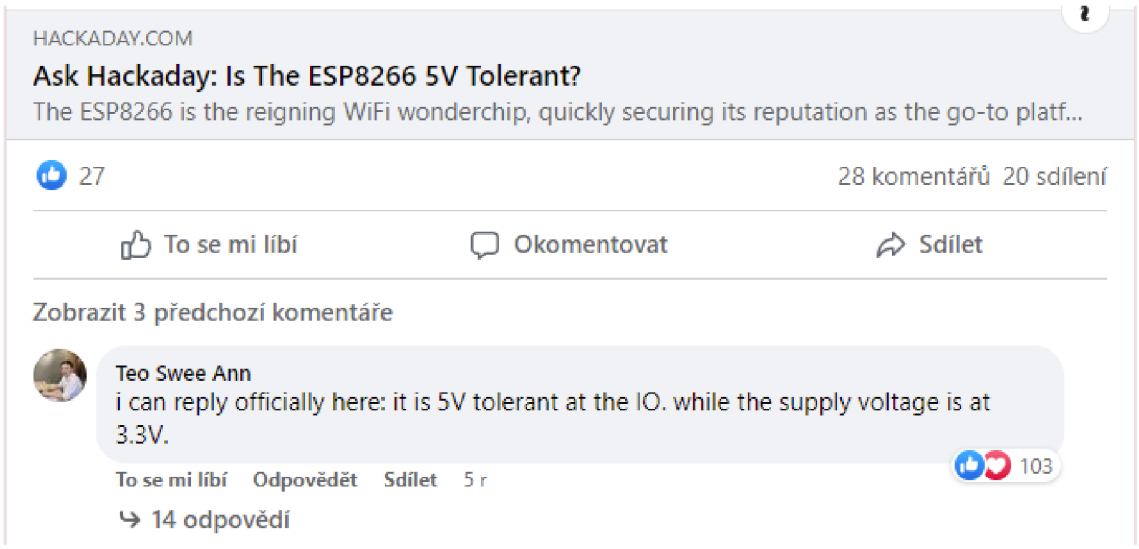
Obrázek 4 - Blokové schéma čipu ESP32 [7]



Nevýhodu může v tomto navrhovaném případě představovat pouze menší počet I/O pinů nebo napájení 3,3 V nebo nekompatibilita periférií mezi 5 V a 3,3 V logikou.

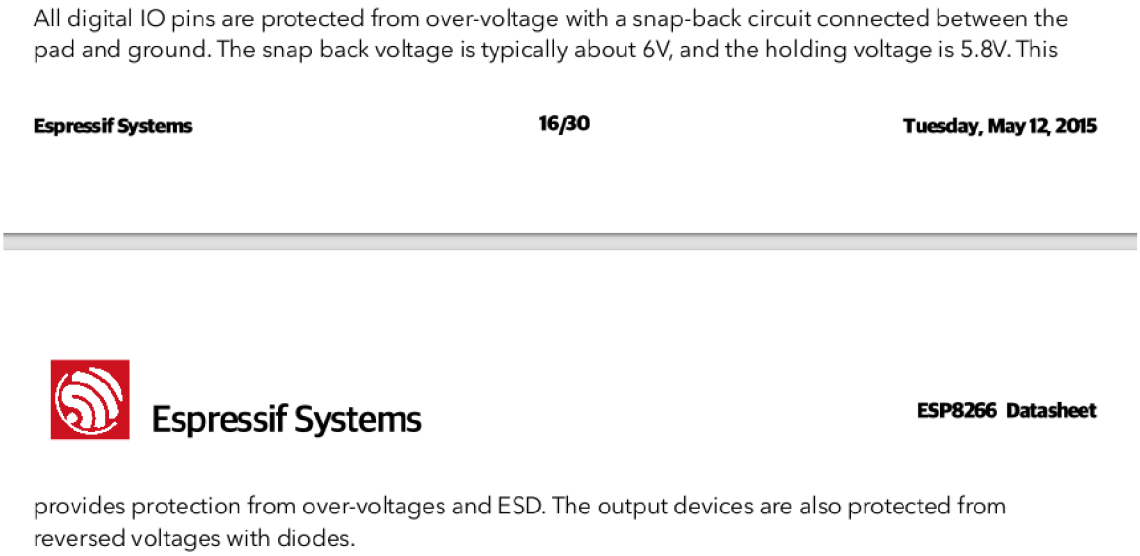
Dle oficiální dokumentace je maximální vstupní napětí na pinech procesoru 3,6 V, přesto je právě kolem této vlastnosti mnoho diskuzí, část uživatelů tvrdí, že vstupní piny ESP32 jsou 5 V tolerantní, logická úroveň 5 V je nezničí. Jiní, s odvoláním na oficiální dokumentaci toto vylučují. Přesto pan Teo Swee Ann, generální ředitel společnosti Espressif Systems napsal jako oficiální vyjádření pro uživatele hackday.com na jejich Facebook, že opravdu tolerantní jsou.

Obrázek 5 - Vyjádření k toleranci vstupních pinů na 5 V [8]



Samotné výstupní piny z čipu ESP32 s 3,3 V logikou mohou bez problémů spínat relé modul nebo spouštět trigger ultrazvukového čidla, napětí 3,3 V je dostatečné pro spolehlivé sepnutí i v TTL logice 5 V. Obráceně je situace problematičtější, dle mé zkušenosti opravdu vstupní piny jsou tolerantní, již několik let mám v provozu zařízení bez jakékoliv závady. Pravděpodobně jde o ochranou proti ESD (electrostatic discharges) a reverznímu napětí, která je v procesoru implementována. V dokumentaci z roku 2015 byla dokonce věta, která byla bohužel později odstraněna. Nutno dodat, že dokumentace je k čipu ESP8266, který je levnější varianta a předchůdce čipu ESP32.

Obrázek 6 - Dokumentace se zmínkou ochrany vstupních pinů proti ESD [7]



Je nutné upozornit na skutečnost, že tolerance je pouze na vstupních GPIO pinech, nikoliv napájení! Samotné napájení procesoru musí být stále 3,3 V. Ani ADC piny tolerantní nejsou. Bohužel se nedá garantovat, že ve všech možných variantách, v nichž se procesory vyrábí, bude 5 V tolerance stále zachována. V případě mého projektu jsem nesnižoval napětí z modulu ultrazvukového čidla, přesto předpokládám, že dvěma jednoduchými odpory lze snadno realizovat odporový dělič napětí a snížit tím výstupní napětí pinu „Echo“.

3.3.3 Fyzické zobrazování hodnot

V názvu kapitoly je záměrně uvedeno slovo fyzické, protože v projektu je zároveň realizováno i zobrazování virtuální, pomocí webového rozhraní. Lokální zobrazení hodnoty, zpravidla v místě fyzického zapojení IoT zařízení, bude ukazovat pouze informativní hodnotu aktuálního zaplnění nádrže, případně servisní informace, nebude se používat sloupec LED diod, ale displej, připojený minimálním počtem vodičů, nejlépe sběrnicí I2C, SPI nebo sériovou linkou. Sloupec LED diod byl vyloučen z několika důvodů, jedním z nich je, že počet LED diod určuje přesnost, s jakou jsme schopni zjistit stav zaplnění, a zároveň platí, že zobrazovat méně než tři úrovně nemá význam, smysluplné zobrazení je z mého pohledu minimálně po čtvrtinách, optimální rozdělení je zobrazení do deseti LED diod. Zapojení tímto způsobem vyžaduje stejné množství vodičů mezi procesorem a LED diodami, k tomu jeden společný, buď napájení (VCC) nebo zem (GND) a stejný počet rezistorů, zapojených v sérii s každou LED diodou. Samotné zapojení více LED diod bez desky plošných spojů je technicky náročnější, počet výstupních pinů čipu je omezen, obsazením deseti pinů znemožníme jejich budoucí využití, možným řešením, jak omezit počet obsazených pinů procesoru, je použití například osmibitového expandéru sběrnice I2C, např. PCF8574 a zapojit osm led diod pouze k němu a pouze dvěma datovými vodiči k procesoru.

Pro účely navrhovaného technického řešení je proto vhodnější použít displej, který umožní zobrazení přesného, číslem vyjádřeného stavu zaplnění nádrže, ale i podrobnějších informací jako například stav připojení k WIFI síti, IP adresu zařízení, chybové stavy a podobně. Výsledné zobrazovací hodnoty se mohou v pravidelných intervalech střídat a na jednom malém displeji tím lze docílit zobrazení většího množství informací.

Návrh displeje byl limitován následujícími požadavky:

- Rozměry: umístění do rozvaděče nebo elektroinstalační krabičky je prostorově omezené, displej je pouze informativní, veškeré údaje budou dostupné prostřednictvím webového rozhraní.
- Jednoduchost zapojení: minimální počet vodičů mezi displejem a SoC, nejlépe pomocí I2C sběrnice.
- Spotřeba elektrické energie pro případ bateriového a solárního napájení.
- Cena a dostupnost.

Na základě výše stanovených parametrů jsem vyloučil všechny dotykové displeje pro jejich cenu a složitost programování. Nebude potřeba zařízení ovládat přímo, veškeré nastavení bude umožněno pomocí webového rozhraní, přímým přístupem k jeho IP adrese, je proto naprosto zbytečné vytvářet dvě rozhraní pro ovládání a nastavování zařízení, prioritou je webové rozhraní. Realizace kompletního nastavení přímo pomocí fyzických tlačítek nebo dotykového displeje jen duplikuje funkci webového rozhraní a zbytečně komplikuje následný zdrojový kód, hardware i náročnost celého projektu.

Do úzkého výběru byly zahrnuty pouze displeje z českého e-shopu dratek.cz (uvedené ceny pocházejí z června 2022), jejich parametry jsou uvedené v tabulce:

Tabulka 2 - Přehled možných displejů [6]

	IIC I2C Displej OLED 0.91" 128x32 Modrý 3,3 V 5V Pro IOT Arduino Raspberry Rozměry 38 x 12 x 4 mm	Dostupný na www.dratek.cz Cena: 94 Kč
	IIC I2C Display LCD 1602 16X2 Znaků LCD Modul Modrý Rozměry 82 x 35 x 18 mm	Dostupný na www.dratek.cz Cena: 97 Kč

	IIC I2C OLED displej 0,96" 128x64 Bílý Rozměry: 27 x 27 x 4,1 mm	Dostupný na www.dratek.cz Cena: 115 Kč

Po zvážení všech limitů jsem vybral jako nejvhodnějšího kandidáta OLED displej 0,96". S ohledem na jeho rozměry, možnosti uchycení a parametry. Pro úplnost uvádím, že všechny výše uvedené moduly je možné použít, jen u modelu LCD 16x2 je třeba počítat s větší spotřebou energie na podsvícení displeje, v noci dokonce s nepříjemným svitem a samozřejmě s odlišnostmi ve zdrojovém kódu.

3.3.4 Základní / vývojová deska

Samotný čip pro vlastní realizaci projektu nestačí, nutně ke svému provozu potřebuje další obvodové součástky. Výběrem vhodné vývojové desky lze docílit velkého zjednodušení technického návrhu a následného použití celého systému v provozu, proto jsem vybral kompletní modul WeMos ESP32 OLED s čipem ESP-WROOM-32 popsaným v kapitole 3.3.2, obsahující regulátor napětí z 5 V na 3,3 V, resetovací tlačítko, integrovaný OLED displej připojený ke sběrnici I2C, napájecí a programovací microUSB konektor a integrovaný obvod CP2012 s funkcí převodníku USB na sériovou linku, nutný pro naprogramování modulu [9]. Obrovskou výhodou tohoto modulu je jeho kompaktnost, obsahuje všechny důležité části včetně již zapojeného OLED displeje 0,96".

Obrázek 7 - Vývojová deska jednočipového počítače WeMos ESP32 OLED [9]

Další možná varianta vývojové desky by byla deska „ESP32-DevKitC Development Board 38pin“ (v ceně 227 korun), je sice bez OLED displeje, je však možné použít displej navržený v kapitole 3.3.3, pouze je nutné ve zdrojovém kódu správně definovat výstupní piny displeje a počítat s jiným umístěním výstupních pinů. Pokud bude zařízení umístěné na takovém místě, kde by displej nedával smysl, je možné nepoužívat displej vůbec a použít tuto vývojovou desku bez integrovaného displeje.

3.3.5 Napájení

Zvolená vývojová deska je napájená přímo přes microUSB konektor standartním napětím 5 V nebo přímo přes pin s označením 5 V, procesor i displej jsou napájeny napětím

3,3 V, které zajišťuje již zmíněný integrovaný stabilizátor, tímto tak odpadá potřeba řešit samostatné napájení pro procesor a jiné periferie pracující s 3,3 V logikou. Obě uvedená napětí jsou k dispozici na samostatných pinech a to znamená, že je možné využít běžnou nabíječku k mobilnímu telefonu a s ní bez problémů napájet celý modul včetně dalších periférií. Standartních nabíječek s USB výstupem 5 V / 1 A je na trhu k dispozici celá řada, v ceně stokorun, považuji tento způsob napájení za velice bezpečný a praktický, neboť je možné využít již nepoužívané nabíječky nebo např. powerbanku pro experimentální použití.

3.4 Monitorovací systém

Základním principem IoT zařízení je komunikace a předávání informací, aby bylo možné analyzovat naměřená data ze senzorů a čidel, je nutné je někam ukládat, nejlépe na společné místo, kde se dále analyzují a případně archivují. V současnosti je to právě internet, který nabízí mnoho virtuálních služeb, které dokáží data přijmout, analyzovat a prezentovat, reagovat na běžné i nenadálé události nebo odchylky v získaných datech, obecně se takovým virtuálním službám říká „cloud“, ve schématech a vizualizacích jsou často tyto služby zjednodušeny do podoby mráčku, jenž v sobě ukrývá ohromné množství serverů, infrastruktury a software, právě proto je důležité si uvědomit, že taková infrastruktura není zadarmo. Existující cloudové služby různých společností mají pro případ bezplatného využití velmi omezené možnosti, je omezen počet záznamů za určitý čas, a zároveň je omezena i historie jejich uchování, což je pro průmyslové a komerční využití pochopitelné, pro vzdělávání mládeže a pro domácnosti je s ohledem na cenu jejich využití velice omezené, proto jsem záměrně od takových služeb upustil, nejen z důvodu ceny, ale i nutných registrací při vytváření účtů, ale také z praktických důvodů, aby nebyl systém závislý na připojení k internetu a bylo možné jej provozovat v rámci uzavřené výukové laboratoře nebo domácnosti. Zohlednil jsem i možnost provozovat systém pro několik desítek IoT zařízení současně, například pokud bude mít každý student svoje IoT zařízení, budeme potřebovat server ve vlastní vnitřní síti, na němž budeme schopni přijímat data od různých IoT, ukládat různé datové typy, zobrazovat je a vizualizovat, umožnit jejich efektivní archivaci, a také umožnit přístup k získaným informacím různým osobám.

Jak již bylo zmíněno, IoT nepotřebuje internet, může komunikovat v prostředí lokální sítě LAN a server, na který se informace posílají může být její součástí, proto je pro sběr dat navržen centrální server, jehož instalace a následný provoz není cenově nákladný, ve školách postačí dokonce starší, ale funkční počítač, případně virtuální server, v domácích podmínkách je možné použít například Raspberry PI nebo se obejdeme i bez něho, v každém případě pro vzdělávání studentů a návazností na motivační úlohy a zpracování dat je server nutnou podmínkou.

Navržený serverový systém

Do technického řešení jsem vybral „Dohledový systém Zabbix“ ve verzi 5.4, která byla v době realizace tohoto projektu aktuální, jedná se o velmi komplexní, multiplatformní, univerzální monitorovací řešení pro různé případy použití. Zabbix

systém je open source, to znamená, že je 100 % zdarma, vydaný pod GNU General Public License (GPL) verze 2., sloužící především pro:

- dohled dalších systémů,
- sběr metrik (informací),
- detekci problémů,
- varování,
- vizualizaci dat.

Umožňuje sbírat metriky z jakéhokoliv zdroje a není limitován ani:

- síťovými zařízeními,
- cloudovými službami, kontejnery, virtuálními stroji,
- monitoringem úrovně OS,
- logy,
- databázemi,
- aplikacemi,
- službami,
- IoT sensory,
- monitoringem webových stránek,
- HTTP/HTTPS monitoringem koncových zařízení,
- podporou celé řady standardních průmyslových protokolů,
- sběrem dat z externích koncových API zařízení.

Instalace Zabbixu je možná jak na fyzický, tak virtuální počítač, požadavky na konfiguraci závisí na počtu sledovaných zařízení, na prostoru potřebném pro uchování historie dat a na skutečném zatížení serveru samotnými uživateli (studenty), pro základní instalaci pro monitorování do 100 zařízení je doporučen systém se dvěma CPU a 2 GB RAM, čím více fyzické paměti bude mít systém Zabbixu k dispozici, tím rychlejší bude databáze a zároveň celý monitorovací systém.

Pro začátek je vhodný SSD disk o kapacitě 128 GB, hrubý výpočet potřebného místa na disku se provede dle následujícího modelového příkladu: uchovávat budeme historii 3 naměřených hodnot, každou minutu po dobu 1 roku, budeme mít tedy 3 * (60 * 24 * 365) = 1576800 hodnot, zaokrouhleno na 1,6 milionu, v závislosti na datovém typu (float, integer, string) a použitého databázového systému je i velikost obsazeného diskového prostoru jedním záznamem různá, obecně mohu napsat, že se pohybuje od 40

bajtů do stovek bajtů, dokumentace uvádí průměrnou hodnotu 90 bajtů na záznam, v našem případě pro 1,6 milionu záznamů počítáme 1,6 * 90 = 144 a vychází potřebný prostor 142 MB. Další prostor lze ušetřit použitím historie trendů, které jsou uchovány jako průměrné hodnoty za jednu hodinu, skutečná data pak nemusí být uchovávána dlouhodobě, stačí například jen 30 dní. Dále je možné využít odstranění duplicitních hodnot a do databáze ukládat jen data, která se změnila. Podrobnější příklady výpočtu potřebného diskového prostoru jsou popsány v dokumentaci v části instalace. (https://www.zabbix.com/documentation/5.4/en/manual/installation/requirements)

Oblast zabezpečení je řešena na několika úrovních, fyzické síťové komunikaci, šifrování, omezení přístupu jen určitých zařízení a podobně, dále na uživatelské úrovni v omezení přístupu k datům nebo konfiguraci, možnosti rozdělení oprávnění jen k určitým zařízením a jejich datům, správu a ověřování uživatelů je možné zajistit různými způsoby, například interní správou účtů přímo v Zabbixu, ověření LDAP nebo Radius z jiného systému, integrací s Active Directory, Single sign-on autentizací a podobně [10]. Celý monitorovací systém je možné integrovat do školního prostředí, tak aby pro přístup studenti používali svá uživatelská jména a hesla.

3.5 Infrastruktura

Základem navrženého technického systému je komunikační síť WIFI s přímým přístupem do vnitřní LAN sítě, ke které jsou připojeny jak IoT zařízení, tak server Zabbix a počítače sloužící pro programování a konfiguraci zařízení nebo pro monitoring provozu. Celá navržená síť je pomocí routeru připojena do internetu nebo další nadřazené sítě. Předpokladem je pro administrátora nebo pedagoga přístup k tomuto routeru, znalost jeho konfigurace, schopnost nastavit firewall, DHCP server, zjistit IP adresy připojených zařízení, možnost konfigurace rezervací v DHCP atd. Samozřejmě je možné využít již stávající síťovou infrastrukturu, v místech předpokládaného umístění lze ověřit sílu WIFI signálu například mobilním telefonem. Samotné připojení sítě k internetu je doporučeno, není však vyžadováno, záleží na variantě konfigurace, připojení do internetu je použito především pro synchronizaci času IoT zařízení, čas je možný však získat i z funkčního NTP serveru ve vnitřní síti, nebo čas můžeme použít ze serveru Zabbix a zprovoznit na něm zmíněnou službu, nebo v příslušné části ve zdrojových kódech vynechat celou část synchronizace a zobrazení času, a tak fungovat v uzavřeném prostředí. V navrženém technickém řešení jsem použil stávající router MikroTik RB951G-2HnD, jeho parametry a konfigurace nejsou dále popisovány.

3.6 Zabezpečení

„Útoky na IoT rostou rychleji než počet samotných připojených zařízení. Jakmile dojde k průniku do zařízení, stane se obvykle součástí botnetu, který lze následně zneužít ke spuštění DDoS nebo jiného typu útoku. Již jsme viděli několik takových malwarových útoků jako Mirai, Gafgyt, Hajime, Amnesia, Persirai, Remaiten, NyaDrop. Alternativně se malware na IoT zařízeních, jako jsou domácí kamery, používá ke shromažďování soukromých informací o spotřebitelích, které se pak zneužívají – například se prodávají na dark webu, nebo slouží k vydírání uživatele atd.“ [11]

Svět IoT byl ve svých počátcích provozován na zařízeních s malým výkonem a dostatečné zabezpečení bylo téměř nemožné realizovat, často se záměrně neřešilo. Počet zařízení raketově stoupá, nedá se přesně ani vyčíslit odhady hovoří o 10 miliardách, předpokládá se dokonce až 25 miliard zařízení v dalších pěti letech. IoT zařízení jsou oblíbeným cílem hackerů už jenom proto, že je jich na světě obrovské množství, v průměru čtyři zařízení na každého člověka na zemi. [11]

Navržený procesor v tomto projektu je dostatečně výkonný, obsahuje hardwarovou akceleraci pro různé kryptografické operace, zabezpečení spouštění, šifrování dat uložených v interní paměti, včetně kryptografického podepisování zkompilovaného programu a mnoho dalšího, přesto sám o sobě bezpečný není, dokonce jsou k dispozici informace o jeho chybách a zranitelnosti, je a bude tak bezpečný, jak ho zabezpečíme my, především kombinací několika opatření. V první řadě musíme pochopit, jak jsou IoT zařízení ohrožena, jak vlastně fungují, co dělají a jak se bránit případnému zneužití a předcházet škodám.

Zabezpečení je nutné chápat jako souhrn mnoha opatření, která dohromady tvoří neustále se vyvíjející celek. Prostředí, ve kterém fungovaly IoT zařízení v období vývoje prvního prototypu, se už za pár měsíců může zásadně změnit a zabezpečení již nemusí být dostatečné, o všechna zařízení se musíme starat, pečovat o ně, neustále je monitorovat a vylepšovat, jedině tak je možné se bránit hrozícím zranitelnostem. Mezi nejčastější ohrožení patří útoky hrubou silou, tedy metodou pokus omyl, kdy je přihlášení k zařízení chráněno jednoduchým uživatelským jménem a heslem, nebo veřejně známým běžným heslem, v horším případě není chráněno vůbec, případně je ve firmware nějaká známá chyba. Nelze se však soustředit pouze na IoT zařízení, ale na celou infrastrukturu, jako například: LAN, WAN a WIFI směrovače (routery a switche), servery, operační systémy ve stejné síti a podobně. V případě školy musíme předpokládat i zneužití pomocí hesel

získaných neoprávněně či nevědomky, verbální nebo vizuální cestou, zjednodušeně, pokud v dobré víře řekneme studentům přístupová hesla, nebo je před nimi opakovaně píšeme na klávesnici, zapamatují si to.

Nejběžnějším opatřením, jak zabránit výše popsaným zranitelnostem je:

- Změnit výchozí uživatelská jména a hesla k zařízením před prvním uvedením do provozu.
- Pravidelně aktualizovat firmware z důvěrného zdroje.
- Používat silná hesla nebo dvoufaktorové autentizační mechanismy.
- Používat šifrování pro přenos nebo ukládání citlivých informací.
- Zakázat nebo zabezpečit „zadní vrátka“, které výrobci často implementují do svých zařízení pro potřeby zákaznické podpory.
- Nastavit síťová pravidla komunikace pomocí firewallu, omezit komunikaci pouze na nutné služby, protokoly a zařízení.
- Monitorovat odchylky od provozních stavů, jakmile je něco jinak, zkontrolovat nebo restartovat zařízení a sledovat podrobněji jeho aktivitu.

V technickém projektu navrženého řešení není zohledněno zabezpečení WIFI sítě, routerů, operačních systémů, běžných počítačů a podobně, nutným předpokladem je již funkční a zabezpečená infrastruktura, do níž se IoT pouze připojí. Všechny důležité operace v tomto projektu jsou chráněny jménem, heslem nebo pinem, samotná komunikace není zabezpečena šifrováním z důvodu názornosti a s ohledem na zjednodušení zdrojového kódu pro potřeby výuky. Implementace HTTPS šifrování vyžaduje další činnosti ve smyslu správy certifikátů a privátních klíčů, důvěryhodnosti certifikačních autorit a zároveň nutnost instalovat takové certifikáty do klientských operačních systémů, v moderních mobilních telefonech je dokonce záměrně bráněno v instalaci certifikátů neautorizovaných (vlastních) kořenových certifikačních autorit nebo vlastních self-signed certifikátů, z těchto důvodů je šifrovaná komunikace záměrně vynechána, je však třeba upozornit na tuto skutečnost a v případě potřeby komunikace přes veřejnou nebo nezabezpečenou síť počítat s nutnou úpravou firmware, případně využít šifrování pomocí VPN sítě.

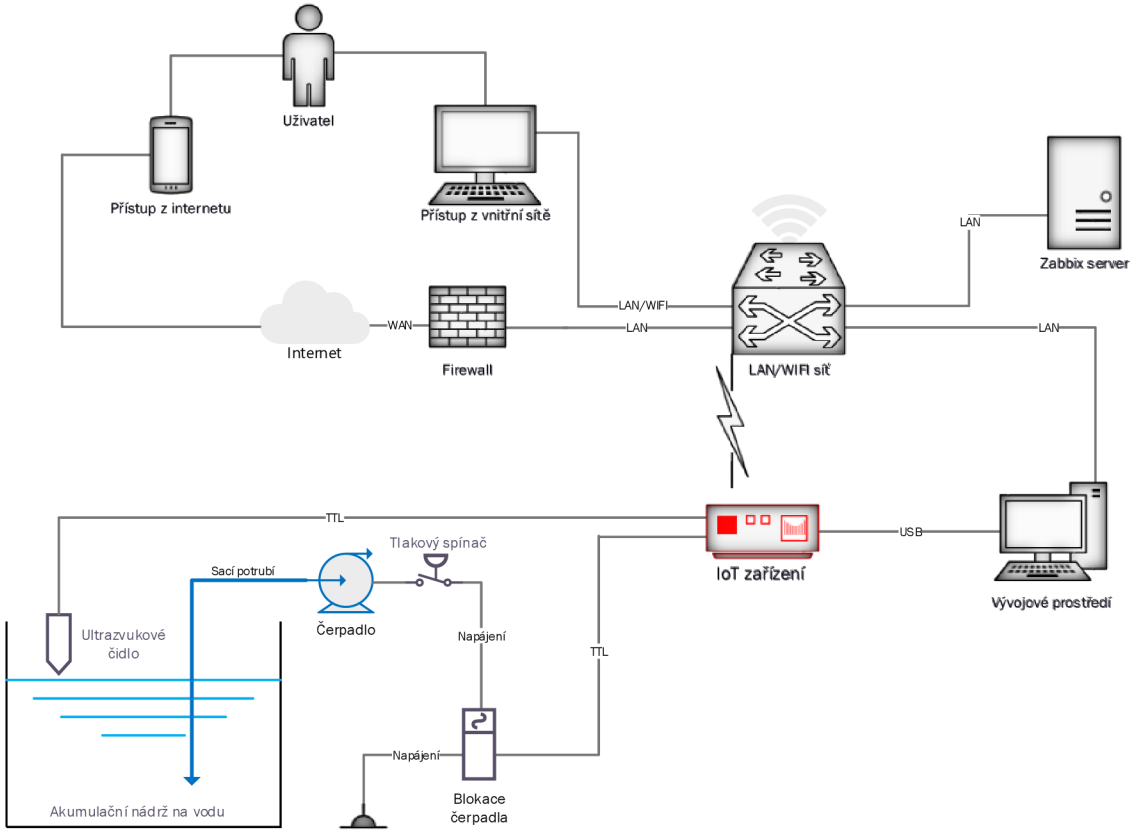
3.7 Přístup k datům z internetu

Zabbix server je umístěn ve vnitřní síti, není za normálních okolností přístupný z internetu, lze však využít alternativní metodu, přístup do vnitřní sítě pomocí virtuální privátní sítě (VPN), v například typu L2TP, která umožní přístup jak k samotnému Zabbix serveru, tak k IoT zařízením. Pro realizaci je nutná veřejná IP adresa od poskytovatele internetového připojení a router MikroTik, zmíněný v kapitole 3.5, samotná implementace VPN není cílem ani podmínkou této práce, není dále podrobněji popsána, předpokládá se její zprovoznění pouze v případě jejího možného využití, pokud to místní podmínky budou umožňovat a v součinnosti se správcem sítě.

4 Realizace jednotlivých částí

Zjednodušené blokové schéma celého navrženého systému, včetně základních komunikačních propojení mezi jednotlivými částmi, je znázorněno na následujícím obrázku.

Obrázek 8 – Blokové schéma celého systému



4.1 Předpoklady pro realizaci

Pro realizaci technického projektu diplomové práce jsou nutné požadavky kladené na vybavení, hardware a potřebné součástky, včetně stavebních úprav, úpravy elektroinstalace a podobně, popsané v kapitole 3. Pro kompletní realizaci celého systému je samozřejmě nutná znalost následujících okruhů, které nemohu podrobně v této práci popisovat:

- práce s Arduino IDE vývojovým prostředím,
- základní knihovny a funkce Arduino,
- elektrotechnické normy, součástky a jejich úplný popis,

- fyzikální principy: mechanika kapalin a plynů, akustika, vlnění, Ohmův zákon, výkon,
- matematiku a geometrii, goniometrické funkce (sinus),
- základy programování v jazyce C/C++,
- základy operačního systému Linux,
- operační systém Windows.

4.2 Nádrž na vodu a čerpadlo

Pro technické řešení v mé domácnosti, byla využita stávající nádrž s kruhovou podstavou, umístěná pod úrovní terénu, čidlo je umístěné 25 cm nad maximální možnou hladinou, nejnižší provozní hladina je ve vzdálenosti 145 cm, další komponenty jsou:

- čerpadlo Sigma Hranice (32-SVA-130-10-2-LM-90-1),
- motor 2,2 kW (MEZ Mohelnice 1AP100L-4s),
- expanzní nádoba (MAXIVAREM LS 100 l),
- filtrace na sacím potrubí (ALKO),
- tlakový spínač (VR21D 0,35MPa),
- motorový spouštěč (PKZMO-6,3),
- jistič 3x16A pro ochranu čerpadla,
- jistič 1x2A pro ovládací obvody,
- stykač třífázový (Schneider LCK1K09)
- SSR relé (CELDUC XKA 20420).

4.3 Instalace vývojového prostředí

Nad operačním systémem Microsoft Windows 10 použitém v technickém řešení jsou nainstalované následující komponenty:

- Arduino IDE 1.8.5 dostupné na https://www.arduino.cc/en/software,
- ovladač převodníku USB na virtuální COM port „*CP210x USB to UART Bridge VCP Drivers*“, dostupné na https://www.silabs.com/developers/usb-to-uart-bridge-vcp-drivers,
- doplněk ESP32 pro Arduino IDE,
- doplněk ESP Sketch Data Upload,

- doplněk ESP Exception Decoder,
- knihovny použité v projektu.

4.3.1 Instalace ESP32 pro Arduino IDE

Instalace doplňku, který přidá do vývojového prostředí podporu vývojových desek ESP32 byla provedena následujícím způsobem:

1. Ve vývojovém prostředí Arduino, v menu *Soubor -> Vlastnosti*

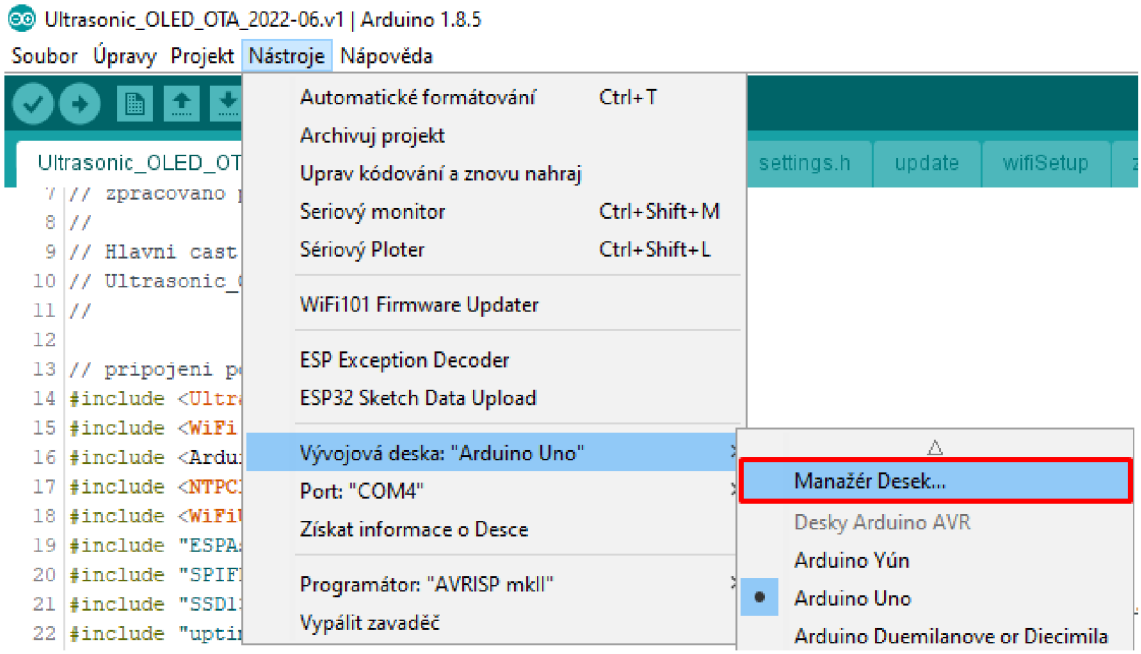
   do pole *Správce dalších desek URL:* vyplníme následující adresu:

```
https://raw.githubusercontent.com/espressif/arduino-esp32/gh-pages/package_esp32_index.json
```

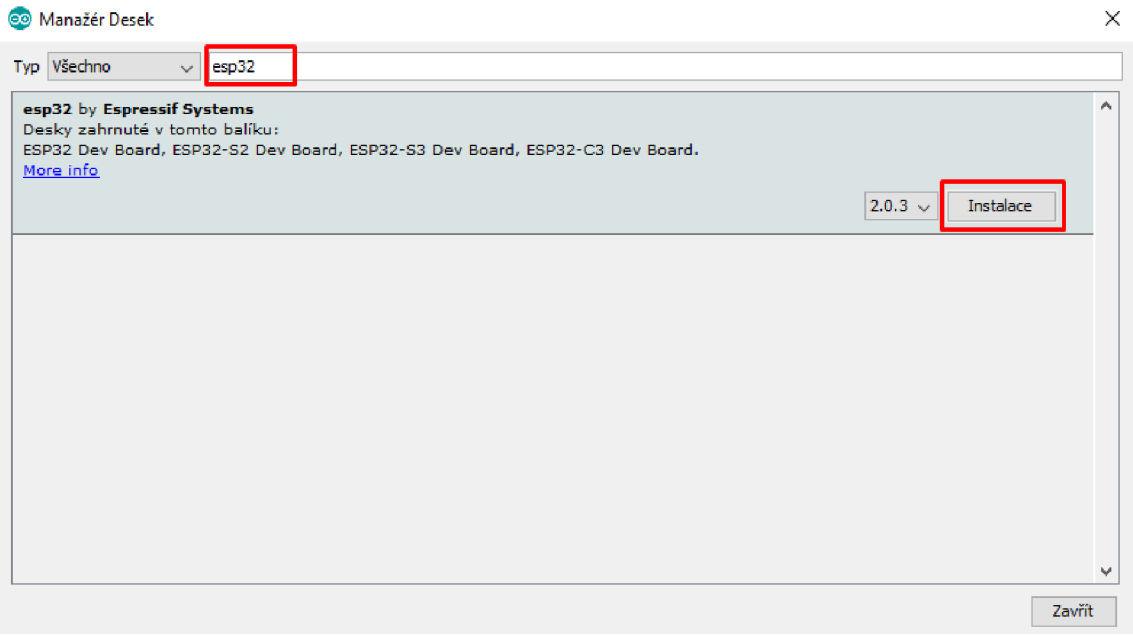
2. V menu *Nástroje -> Vývojová deska: .... -> Manažer desek*

   viz. Obrázek 9, napíšeme do vyhledávacího filtru text *ESP32* a vybereme *esp32 by espresssif* a volbu *Instalace*, viz. Obrázek 10. Instalace trvá poměrně dlouho.

Obrázek 9 - Arduino IDE – menu Manažer desek

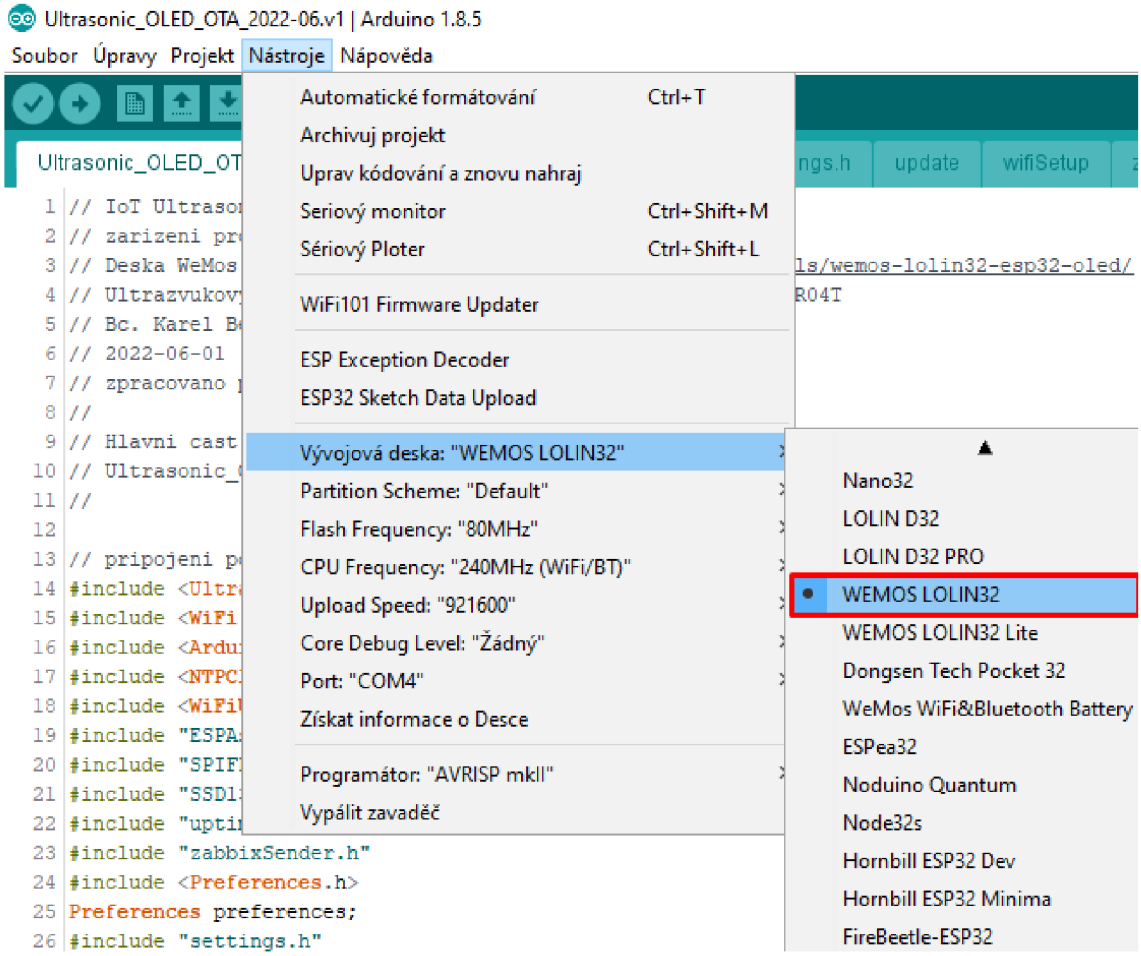


Obrázek 10 - Arduino IDE – Manažer desek, vyhledání esp32



3. Po instalaci je nutné zvolit správnou desku a správný port,
   v menu *Nástroje -> Vývojová deska: .... -> WEMOS LOLIN32*
   viz. Obrázek 11.
4. Vybereme správný port, v menu *Nástroje -> Port:*
   pokud je nainstalovaný driver USB, je možné jej zvolit, pokud jich vidíme více, je možné ve správci zařízení ve Windows zjistit, jaké porty jsou aktuálně k dispozici. Pozor, při používání více zařízení se může i v jednom USB portu měnit číslo COM portu, je nutné vybrat správný!

Obrázek 11 - Arduino IDE – menu výběru vývojové desky



Podrobné informace o instalaci podpory procesorů řady ESP v Arduino IDE vývojovém prostředí jsou na stránkách výrobce čipů, https://docs.espressif.com/projects/arduino-esp32/en/latest/installing.html

4.3.2 Instalace ESP Sketch Data Upload

V technickém projektu je využíván souborový systém SPIFFS, který je umístěn v části FLASH paměti procesoru ESP32, obsahuje html soubory, kaskádové styly a obrázky. Na počítači ve složce, kde je umístěn zdrojový kód projektu (Sketch), je podsložka s názvem data, všechny soubory v ní uložené se pomocí nástroje „ESP Sketch Data Upload“ nahrají přímo do FLASH paměti, zároveň se provede její inicializace a formátování.

Postup instalace:

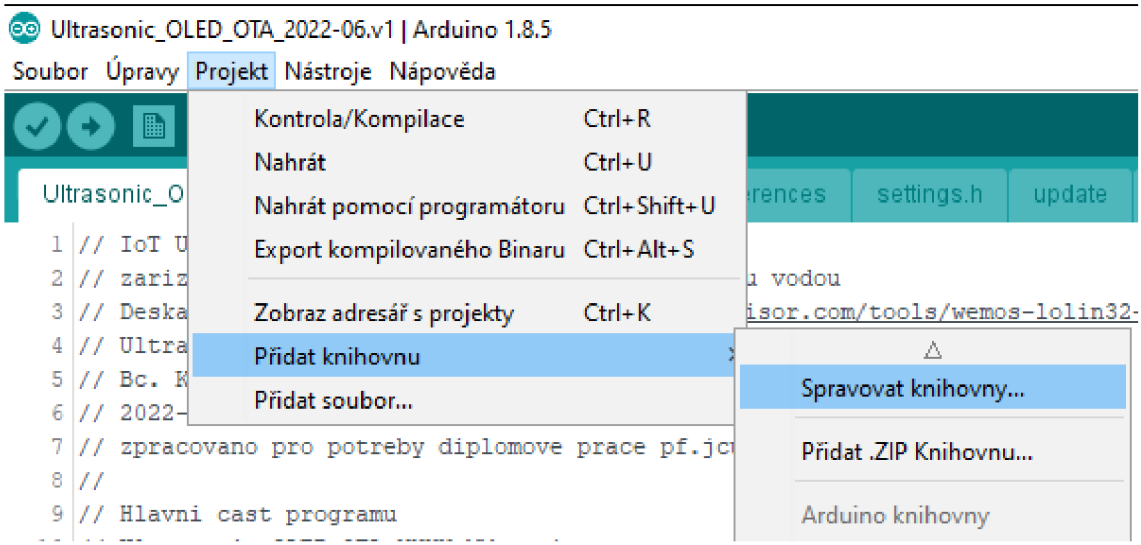
1. Zjistíme, kde je složka s instalací Arduino, v menu *Soubor -> Vlastnosti* v poli *Umístění projektů:* například C:\Users\karel\Documents\Arduino.
2. Vytvoříme podsložku *tools*.

3. Stáhneme doplněk z:
   `https://github.com/me-no-dev/arduino-esp32fs-plugin/releases/download/1.0/ESP32FS-1.0.zip`
   a rozbalíme ho do vytvořeného adresáře.
4. Uvnitř adresáře vznikne následující soubor, přesně s touto cestou:
   `<projekty>/tools/ESP32FS/tool/esp32fs.jar`
   kde <projekty> představuje cestu zjištěnou v bodu 1.

4.3.3 Instalace knihoven

Arduino IDE disponuje přímo nástrojem „Manažer Knihoven“ pro jejich snadné vyhledávání a instalaci, je v menu *Projekt -> Přidat knihovnu -> Spravovat knihovny…* . Podle klíčových slov nebo názvů knihoven z projektu je snadno najdeme a nainstalujeme, pokud při kompilaci nastane chyba, je ve výpisu kompilátoru snadno vše dohledatelné, která knihovna chybí a případně ji doinstalujeme, pokud není potřebná knihovna k dispozici, je v textu uveden zdroj, odkud je možno danou knihovnu stáhnout, stažené knihovny lze nainstalovat i přímo vložením staženého zip souboru.

Obrázek 12 - Arduino IDE – menu Spravovat knihovny

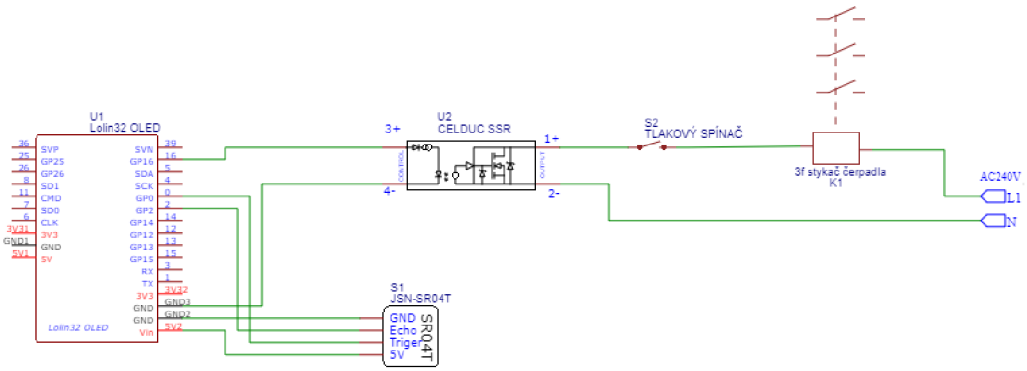


4.4 Zapojení

Schéma zapojení technického řešení obsahuje pouze slaboproudé části IoT, zejména modul vývojové desky WeMos ESP32 OLED (ve schématu „LOLIN32 OLED“), ultrazvukového čidla a SSR relé pro blokaci čerpadla, část ovládání blokace čerpadla s napájením 240 V je pro názornost zjednodušena, neobsahuje ochranné prvky, jističe, zapojení motoru čerpadla a podobně, uvedené součásti nebyly předmětem realizace, byla použita stávající zařízení, do původního zapojení se pouze přidalo SSR

relé. Samotné schéma bylo vytvořeno v online verzi programu EasyEDA. (https://easyeda.com/)

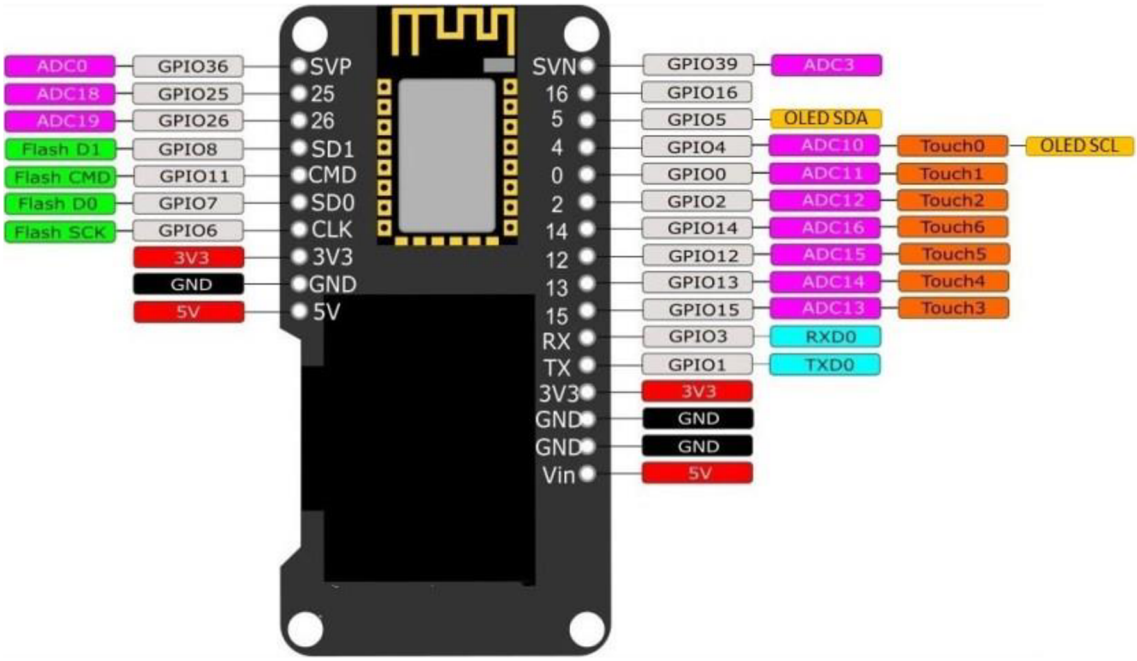
Obrázek 13 - Schéma zapojení vývojové desky ESP32 a periferií



Pro připojení periferií byly použity následující datové piny na vývojové desce:

- GPIO16 – připojení SSR relé pro blokaci čerpadla
- GPIO2 – Echo modulu ultrazvukového čidla
- GPIO0 – Trigger modulu ultrazvukového čidla
- GPIO4 – I2C sběrnice, SCL pro vestavěný OLED displej
- GPIO0 – I2C sběrnice, SDA pro vestavěný OLED displej

Obrázek 14 - Popis a zapojení pinů vývojové desky WEMOS ESP32 OLED [12]



4.5 Zdrojový kód

Zdrojový kód byl pro přehlednost rozdělen do několika souborů, aby bylo možné se v něm lépe orientovat (Poznámka: Všechny mnou vytvořené nebo upravené soubory naleznete v adresáři projektu, které jsou součástí elektronické přílohy této práce, v následujících kapitolách je proto uveden pouze popis funkcí programu z mého pohledu důležitých částí zdrojového kódu.). Vycházel jsem z dokumentace dostupné pouze v elektronické podobě, výrobce čipů řady ESP (Espressif Systems) [7], oficiální dokumentace Arduino [13], knihy ‚Electronics Projects with the ESP8266 and ESP32' [14] a výborného webu s množstvím zajímavých projektů Random Nerd Tutorials [12]. Výchozím souborem technického projektu diplomové práce je „Ultrasonic_OLED_OTA_2022-06.v1.ino". Pro rychlý přehled a orientaci, uvádím v tomto místě seznam mnou vytvořených a upravených souborů zdrojových kódů v adresáři projektu a jejich stručný význam:

```
preferences.ino                 Uloženi nebo načtení konfigurace
settings.h                      Definice konstant síťové komunikace
Ultrasonic_OLED_OTA_2022*.ino   Hlavní část programu
update.ino                      Zpracovaní uloženi běžného nastaveni
webProcessor.ino                Nahrazovaní textu v html
wifiSetup.ino                   Zpracovaní uloženi nastaveni wifi
zabbixSender.cpp                Příprava dat pro odeslání do Zabbixu
zabbixSender.h                  Hlavičkový soubor k upravené verzi
```

4.5.1 Měření hladiny

Skutečné měření vzdálenosti hladiny ultrazvukovým čidlem je prováděno pravidelně každé dvě sekundy, ukládáno do pole o velikosti 30 hodnot, které se cyklicky přepisují, z nich je vždy vypočtena průměrná hodnota naměřené vzdálenosti a zároveň procento zaplnění nádrže. Následující tři hodnoty jsou nejdůležitější, názvy jejich proměnných jsou: `zaplneni, prumernaVzdalenost, aktualniVzdalenost`. Vždy při dosažení posledního, třicátého cyklu, dojde k odeslání jejich hodnot na server Zabbix, interval odesílání je tedy 60 sekund (30 x 2).

Pro výpočet procentuálního zaplnění jsou v programu pevně stanovené konstanty, odpovídající parametrům nádrže a polohou ultrazvukového čidla.

```
#define maxVzdalenost 450
#define minVzdalenost 20
#define nizkaHladinaVzdalenost 145
#define nejvyssiHladinaVzdalenost 25
#define pocetPoslednichPrumerovanychHodnot 30

int arrPosledniHodnoty[pocetPoslednichPrumerovanychHodnot];
int indexPosledniMereneHodnoty = 0;
const long interval = 2000;

int prumernaVzdalenost = 0;
int aktualniVzdalenost = 0;
int zaplneni = 0;
```

Vlastní vzorec pro výpočet zaplnění je v samostatné funkci, aby jej bylo možné snadněji identifikovat a v případě jiného typu nádrže upravit.

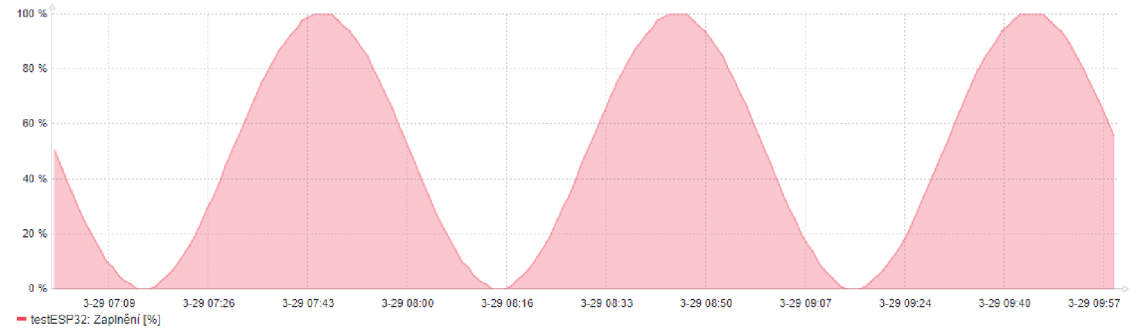
```
// vypocita skutecne zaplneni v procentech
int spocitejZaplneni(int vzdalenost) {
return  map(constrain(vzdalenost, nejvyssiHladinaVzdalenost,
nizkaHladinaVzdalenost), nizkaHladinaVzdalenost,
nejvyssiHladinaVzdalenost, 0, 100);
}
```

Pro případ, kdy není k systému připojeno žádné čidlo, to je, když je detekovaná vzdálenost mimo stanovené limity, přejde systém do „demo“ režimu, vypočítá aktuální vzdálenost pomocí funkce sinus a aktuálního času, jednu periodu funkce sinus vypočítá systém s intervalem jedné hodiny, s počátkem každé celé hodiny a amplitudou v mezích minimálních a maximálních hodnot.

```
// pouze pro testování, když není připojeno čidlo, tak to vrátí
hodnotu sinus s periodou jedne hodiny, pocatek ma v celou hodinu
if (aktualniVzdalenost == 0) {
       aktualniVzdalenost =
(int)(((sin((((float)timeClient.getMinutes() * 60.0) +
(float)timeClient.getSeconds()) / (3600.0 / (3.14159 * 2.0))) +
1.0) * (((double)nizkaHladinaVzdalenost -
(double)nejvyssiHladinaVzdalenost) / 2.0)) +
(double)nejvyssiHladinaVzdalenost);
}
```

Užitečnost této funkcionality je především v možnosti testování IoT na pracovním stole bez nutnosti mít připojené ultrazvukové čidlo a snažit se generovat smysluplné hodnoty, při poruše reálného zařízení je možné z grafů okamžitě odhadnout, že čidlo nefunguje správně, protože generuje graf ve tvaru sinusoidy.

Obrázek 15 - Zabbix, graf IoT bez připojeného ultrazvukového čidla



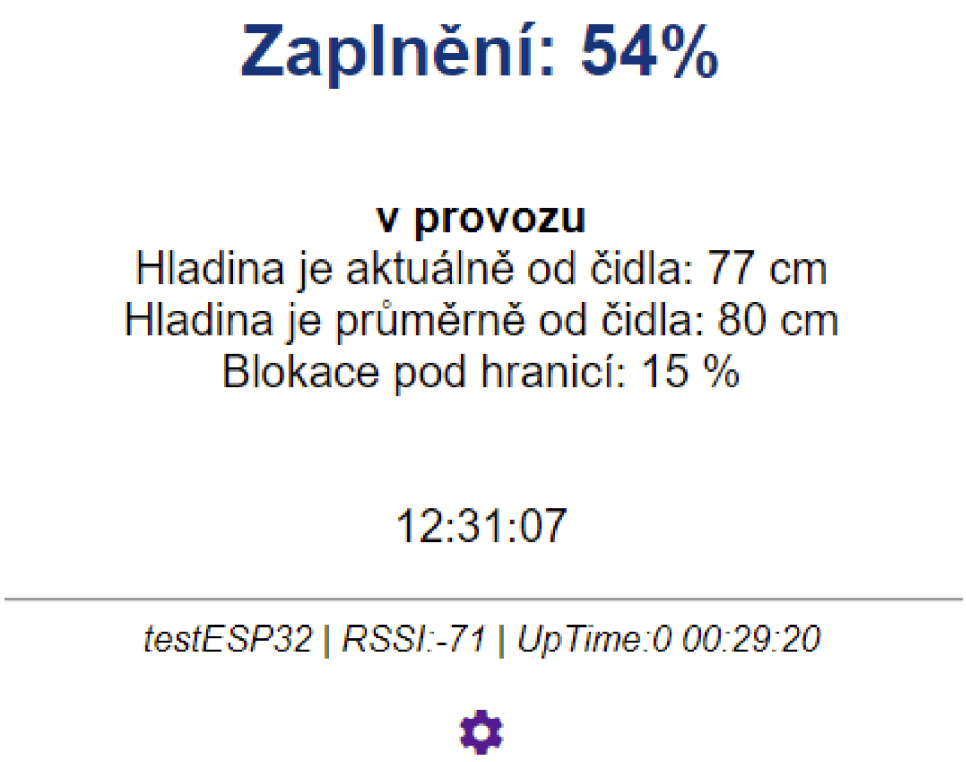
4.5.2 Webový server

Po otestování několika různých knihoven pro webový server jsem nakonec použil asynchronní webový server z knihovny `ESPAsyncWebServer.h` dostupný na https://github.com/me-no-dev/ESPAsyncWebServer, protože disponuje velmi širokou škálou možností, umožňuje například:

- zpracovávat požadavky nezávisle na chodu hlavní smyčky `loop(){}`, není třeba neustále v programu cyklicky zjišťovat, zda se připojil nějaký klient a věnovat se vyřízení jeho požadavku,
- zpracovávat požadavky více klientů najednou,
- automaticky uvolnit paměť po odpojení klienta, uzavřít síťové spojení,
- pracovat se souborovým systémem SPIFFS, načítat přímo soubory,
- používat „template processing“, nahrazovat v souborech proměnné přímo při zpracování před odesláním klientovi,
- jedním podprogramem (handle) obsloužit více požadavků,
- „Request“ a „Response“ jsou využitelné objekty s metodami a vlastnostmi,
- autentizaci jménem a heslem,
- přesměrování požadavků na jinou adresu.

Hlavní stránka webového zařízení je optimalizována a vyzkoušena v internetových prohlížečích ze zařízení Android 10, Windows 10, iOS a Ubuntu, obsahuje zřetelně stav zaplnění nádrže a menším písmem tučně provozní stav blokace čerpadla, menším písmem pak aktuální a průměrnou vzdálenost hladiny vody od čidla, přesný čas, pod čarou je zleva uveden název zařízení (hostname), síla signálu RSSI a doba běhu zařízení od restartu. Samotná síla signálu je tím větší, čím je hodnota blíže k nule, teoreticky může nabývat hodnot od 0 do -100, každé dvě minuty dojde k automatické aktualizaci stránky, poslední hodnotou hlavní stránky úplně dole je ikona ozubeného kolečka, odkazující na stránku konfigurace zařízení (setup.html).

Obrázek 16 - Hlavní webová stránka IoT zařízení



Ve zdrojovém kódu je obsluha konkrétní url adresy, požadavku klienta na nějakou stránku registrována pouze jednou, v části `setup(){}`, pomocí metody `server.on()`, hlavní smyčka `loop(){}` již neřeší vlastní obsluhu požadavků. Níže je popsán způsob a princip funkce webového serveru pro registraci události k vyřízení požadavku klienta na HTTP_GET s url adresou /.

Při požadavku klienta na uvedenou adresu načte server šablonu (soubor) jménem index.html ze souborového systému SPIFFS a zpracuje pomocí funkce `processor`.

```
// Obsluha pro /
server.on("/", HTTP_GET, [](AsyncWebServerRequest *request){
request->send(SPIFFS, "/index.html", String(), false, processor);
});
```

Funkce `processor` zpracuje obsah textové šablony, nahradí každý nalezený text ohraničený znaky `%%` odpovídající okamžitou hodnotou, v případě `index.html` souboru je `%MINIMUM%` nahrazeno hodnotou proměnné `minimalniHladina` převedenou na text, pro tuto operaci jsem použil pouze jednu funkci, v ní jsem definoval podmínky pro všechny využitelné hodnoty pro zpracování všech .html souborů,

```
String processor(const String& var){
 if(var == "STATE"){
  return getStatusText(blokaceCerpadla);
 }
 if(var == "MINIMUM"){
  return String(minimalniHladina);
 }
```

pokud text v šabloně neodpovídá žádné podmínce, je vymazán a není uživateli zobrazen, protože funkce na konci vrátí prázdný řetězec,

```
  return String();
}
```

pro zajímavost uvádím část souboru `index.html` s texty před nahrazením.

```
</head>
<body>
<h1>
Zaplnění: %ZAPLNENI%&#37;
</h1>
<p><b>%STATE%</b><br>
Hladina je aktuálně od čidla: %AKTUALNI% cm<br>
Hladina je průměrně od čidla: %PRUMERNA% cm<br>
Blokace pod hranicí: %MINIMUM% &#37;<br>
<br>
```

Stránka `setup.html` webového serveru slouží pro nastavení hodnoty minimální hladiny pro blokaci čerpadla a zároveň pro konfiguraci WIFI připojení, je rozdělena do dvou samostatných formulářů:

```
<form action="/update" method="POST">
```

a

```
<form action="/wifiSetup" method="POST">
```

z nichž je každý odeslán na jinou adresu a zpracován odlišně. Celou stránku mohou zobrazit všichni uživatelé, bez autentizace mohou i uložit konfiguraci minimální hladiny, samotnou konfiguraci WIFI lze provést jen po autentizaci uživatelským jménem a heslem, definovaným v souboru `settings.h` s prefixem `http_`.

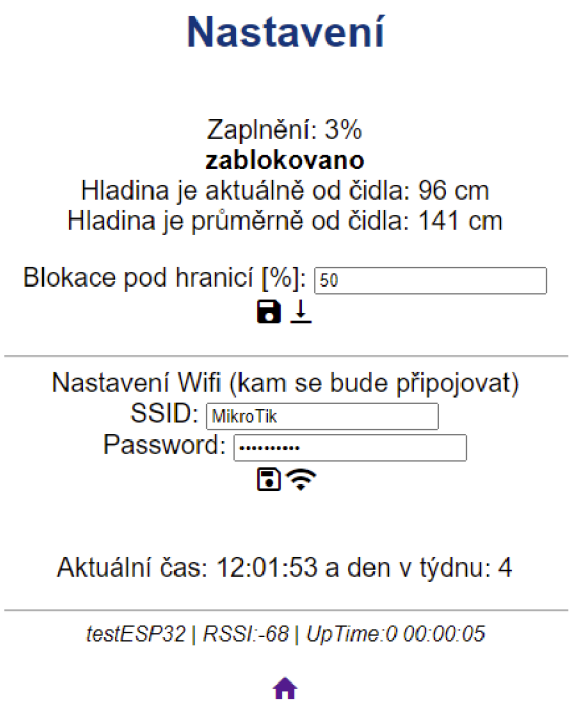
Zpracování odpovědí z konfiguračního formuláře webové stránky začíná odesláním dat metodou HTTP_POST na adresu `/wifiSetup` a na základě registrace události k této adrese se data dále automaticky zpracovávají funkcí `handlewifiSetup`,

```
server.on("/wifiSetup", HTTP_POST, handlewifiSetup);
```

část kódu vyžadující autentizaci uživatele je umístěná ihned na začátku funkce `handlewifiSetup` v souboru `wifiSetup.ino`, obsluhující odpovědi stránky `setup.html` z formuláře `wifiSetup`. Pokud není uživatel přihlášen požadovaným jménem a heslem, funkce nepokračuje dále a uživateli je vrácen požadavek na ověření.

```
void handlewifiSetup(AsyncWebServerRequest *request){
       if(!request->authenticate(http_username, http_password))
       return request->requestAuthentication();
```

Obrázek 17 - Konfigurační stránka webového rozhraní IoT zařízení



4.5.3 Připojení k WIFI

Běžný start zařízení probíhá připojením k WIFI síti v úvodní části funkce setup(){}, v cyklu while(){} je testován status připojení a dokud není stav WL_CONNECTED, není možné pokračovat dále. Systém je nastaven tak, že při třicátém prvním pokusu o připojení dojde k nucenému resetu zařízení a celý cyklus se opakuje, pokud není WIFI síť dostupná, probíhají restarty v pravidelných intervalech a systém se stává nedostupný po síti i nefunkční pro ovládání blokace čerpadla.

```
int wifiConnectionCounter = 0;
WiFi.begin(ssid.c_str(), password.c_str());
while (WiFi.status() != WL_CONNECTED) {
    Serial.print(".");
    Serial.println(wifiConnectionCounter);
    if (wifiConnectionCounter > 30) {
      ESP.restart();
    }
    wifiConnectionCounter++;
    feedLoopWDT();
    delay(1000);
}
```

Navržené zapojení s takovým chováním počítá a výstupní pin pro blokaci je proto nastaven na výchozí hodnotu ještě před inicializací připojení k WIFI síti, SSR relé tím blokuje běh čerpadla i při neúspěšném připojení. Ve zdrojovém kódu je výstupní pin nastaven ještě před připojením WIFI, za začátku funkce setup(){} na hodnotu LOW, protože blokace je provedena logickou nulou na výstupním pinu.

```
// nastaveni pinu pro rele blokace cerpadla
pinMode(RelayPin, OUTPUT);
digitalWrite(RelayPin,LOW);
```

Před prvním spuštěním IoT zařízení bylo nutné definovat správné jméno a heslo pro připojení k WIFI přímo ve zdrojovém kódu, další změny se již provádějí pomocí konfigurační stránky setup.htm, můžeme je provádět i během provozu. Odpadá tím nutná úprava a následná kompilace zdrojového kódu včetně nahrání nového firmware do IoT zařízení. Nově získané informace se uloží do flash paměti, kde jsou uchovány i po restartu. Více je princip ukládání popsán dále v kapitole 4.5.6.

Výchozí jméno a heslo WIFI sítě je definováno v souboru `settings.h` a je nutné jej změnit, pokud ale už došlo k uložení nastavení pomocí konfigurační stránky setup.htm, jsou primárně použity hodnoty uložené v zařízení a změna jejich výchozích hodnot ve zdrojovém kódu již nemá vliv na další připojení, ani aktualizace firmware tyto hodnoty nezmění. Navržený způsob je vhodný například tehdy, pokud je nežádoucí, aby heslo pro připojení k WIFI síti měli studenti přímo ve zdrojovém kódu.

4.5.4 Souborový systém

Vybraný čip ESP32 disponuje 4MB flash paměti, která je rozdělena na několik oddílů, jejichž přesné velikosti můžeme ovlivnit v Arduino IDE, menu *Nástroje -> Partition Scheme.*

Volba *„Default“* znamená výchozí rozložení oddílů (partitions), jejich přesná specifikace je v textovém souboru v místě instalace:

`"C:\Users\<user>\Documents\Arduino\hardware\espressif\esp32\tools\partitions\default.csv"`

```
# Name,   Type, SubType, Offset,  Size, Flags
nvs,      data, nvs,     0x9000,  0x5000,
otadata,  data, ota,     0xe000,  0x2000,
app0,     app,  ota_0,   0x10000, 0x140000,
app1,     app,  ota_1,   0x150000,0x140000,
spiffs,   data, spiffs,  0x290000,0x170000,
```

Souborový systém je uložen v oddílu `spiffs` o velikosti 1507328 bajtů (0x170000h) a je možné si jej představit jako miniaturní SD kartu vloženou přímo do procesoru, po naformátování správným souborovým systémem je využita k ukládání dat.

Po kompilaci hlavního zdrojového kódu jsem provedl sestavení binárního souboru souborového systému a jeho nahrání do oddílu `spiffs` pomocí *Nástroje -> ESP32 Sketch Data Upload*, volba zároveň naformátovala potřebný prostor a nebylo nutné jej formátovat jiným způsobem, automaticky došlo k vytvoření souborového systému, obsahující všechny soubory z adresáře `/data`, který je umístěn ve stejném adresáři jako zdrojové soubory projektu. Sestavení trvá výrazně kratší dobu a nahrává se do ESP32 nezávisle na hlavním programu, odeslání bylo provedeno standardně přes USB a zvolený COM port.

Výhodou popsaného principu je možnost vkládání html nebo textových souborů, šablon, binárních souborů, obrázků a podobně. Soubory na počítači v adresáři /data je možné libovolně měnit a upravovat oblíbenými programy, ladění vzhledu html stránek a kaskádových stylů je jednodušší, ušetří se hodně času programátora.

Seznam souborů v adresáři `/data` využívaný webovým serverem:

```
favicon.ico
index.html
setup.html
style.css
```

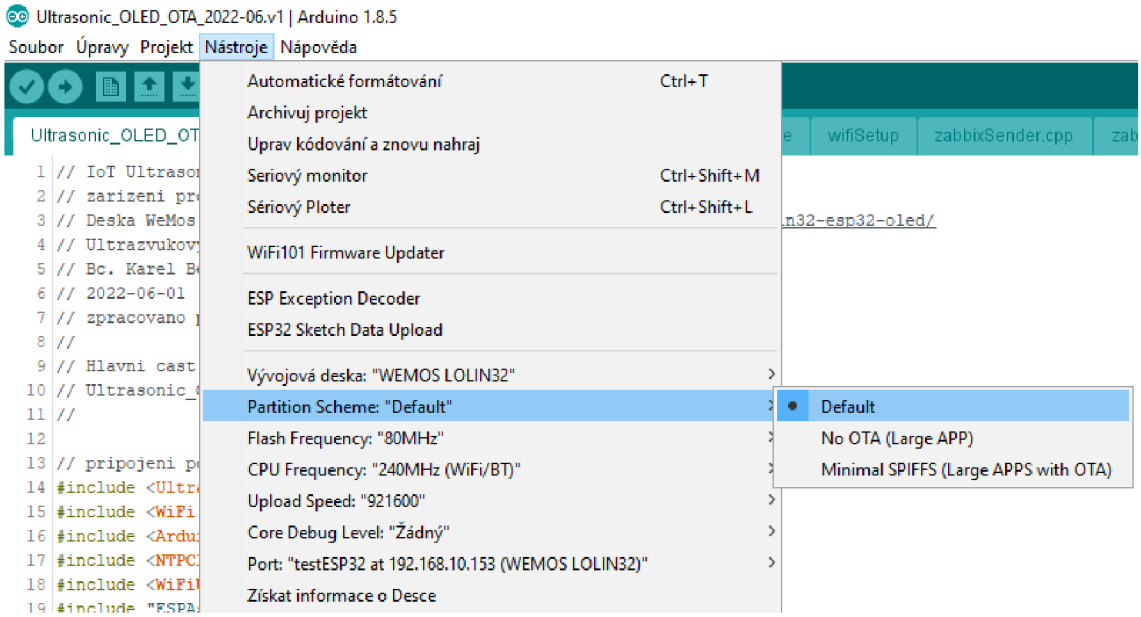
4.5.5 Aktualizace firmware po síti

IoT zařízení jsou často umístěna na obtížně dostupných místech, proto by byla aktualizace jejich firmware poměrně komplikovaná, samotný čip ESP32 však podporuje funkci aktualizace firmware přímo po síti, hovoříme o funkci OTA (Over the Air), tj. možnosti aktualizace bez nutnosti fyzického připojení prostřednictvím USB kabelu, to znamená, že v případě, kdy budeme potřebovat upravit firmware v zařízení, umístěnému na nepohodlném či vzdáleném místě, nemusíme kvůli tomu vůbec vstát od počítače [15].

Nutným předpokladem pro OTA je správné rozdělení interní flash paměti, musí obsahovat dvě stejně velké části (partition), `app0` a `app1` zmíněné v předchozí kapitole 4.5.4, v jedné je aktuální běžící systém a v druhé je prostor pro nahrání nového kódu, po úspěšné verifikaci je změněno pořadí spouštění tak, že po restartu se zavede systém z druhé partition.

Jak z výše uvedeného principu vyplývá, je v paměti alokován prostor pro vlastní program dvakrát, a tím je zmenšena jeho možná velikost, pokud bychom potřebovali více prostoru pro vlastní program, je možné se funkce OTA vzdát ve prospěch jiných oddílů, a tak zvětšit `app0` nebo `spiffs` nebo oba, nebo `spiffs` úplně vynechat.

Obrázek 18 - Arduino IDE - výběr rozdělení interní paměti a rozložení oddílů



Pro správnou funkci bylo nutné dodržet potřebné rozložení oddílů a samozřejmě přidat do projektu příslušný kód, z kterého uvádím pouze nejdůležitější části, především ve funkci setup():

```
#include <ArduinoOTA.h>

// Hostname pro OTA
ArduinoOTA.setHostname(hostname);

// Nastaveni hesla pro OTA
ArduinoOTA.setPassword(OTA_password);

ArduinoOTA.begin();
```

dále bylo nutné zajistit pravidelné dotazování ve funkci loop(),

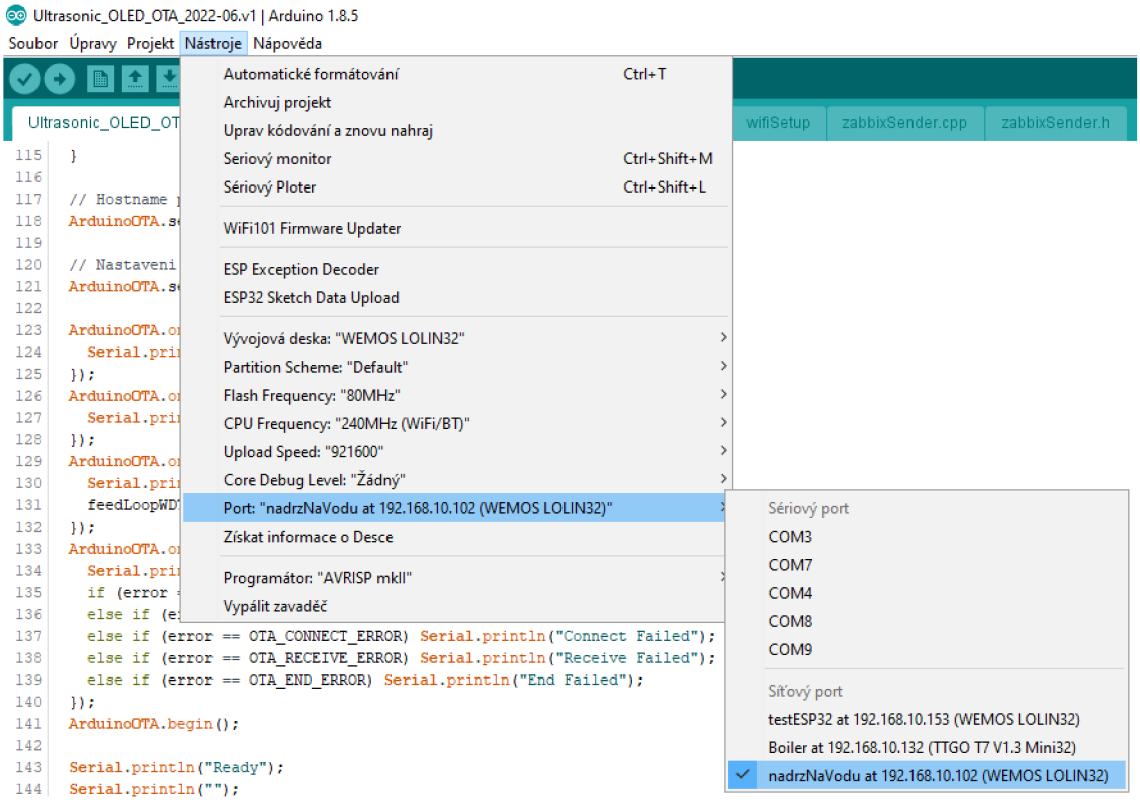
```
ArduinoOTA.handle();
```

Aktualizace po síti začínala stejným způsobem, jako běžná kompilace a nahrání programu přes USB kabel, pouze místo virtuálního sériového portu COM4 byl v menu *Nástroje -> Port:* zvolen název zařízení a jeho MAC adresa.

Pro identifikaci více zařízení v síti jsem nastavil jeho název v proměnné hostname, který se následně zobrazí přímo v menu Arduino IDE, jak je vidět na obrázku Obrázek 19, zároveň je použit i pro odesílání naměřených hodnot do systému Zabbix. Proti neoprávněné aktualizaci byl celý aktualizační proces zabezpečen heslem definovaným v souboru settings.h:

```
const char* OTA_password = "*****";
```

Obrázek 19 - Arduino IDE - menu pro výběr zařízení OTA



4.5.6 Ukládání hodnot do trvalé paměti

V technickém řešení byla pro trvalé uložení hodnot do paměti flash použita knihovna Preferences.h, která elegantně řeší původní, složité ukládání hodnot do EEPROM paměti, které se běžně používá u ATMEL procesorů v Arduino projektech. Instalace je dostupná z (https://github.com/espressif/arduino-esp32/tree/master/libraries/Preferences), její zprovoznění bylo provedeno následujícím zdrojovým kódem:

```
#include <Preferences.h>
Preferences preferences;
```

Ve skutečnosti jsou hodnoty uloženy stále v jediné 4 MB paměti, pouze v jiném (nvs) oddílu, který se nepřepisuje při nahrání nového programu, podrobněji je rozložení paměti (partition scheme) popsáno v kapitole 4.5.4

Každá ukládaná hodnota musí mít jméno, maximálně 15 znaků dlouhé a musí být přiřazena do společného jmenného prostoru, kterých může být více, v jednom jmenném prostoru může být více pojmenovaných hodnot. Níže uvedená funkce uloží do jmenného

prostoru `config` proměnnou `minimalniHladina`, ale pod názvem `minHladina`, protože název už přesahoval 15 znaků:

```
void save_conf() {
  preferences.begin("config");
  preferences.putInt("minHladina",minimalniHladina);
  preferences.end();
}
```

Následující funkce načte z jmenného prostoru `config` položku s názvem `minHladina` a přiřadí ji do proměnné `minimalniHladina`, pokud takový název v paměti ještě neexistuje, použije se výchozí hodnota `minimalniHladina`, protože je již inicializovaná přímo ve zdrojovém kódu výchozí hodnotu 5:

```
int minimalniHladina = 5;
```

```
void load_conf(){
  preferences.begin("config",true);
  minimalniHladina                                                    =
preferences.getInt("minHladina",minimalniHladina);
  preferences.end();
}
```

Dvě podobné funkce slouží pro uložení nebo načtení jména a hesla pro připojení k WIFI síti, používá se však jiný jmenný prostor „WifiClient“:

```
void saveWifi_conf(String newSSID, String newPassword) {}
void loadWifi_conf(){}
```

4.5.7 Ochrana před zatuhnutím programu

V každém programu se mohou vyskytovat chyby, nejvíce jich bývá při vývoji a ladění. V určitých případech je výhodné, pokud se systém sám z chyb vzpamatuje, zotaví se a začne zase normálně fungovat. Aby bylo možné zajistit plynulý chod zařízení, je nutné detekovat a minimalizovat nekonečné čekání a zacyklení programu v nekonečné smyčce, využil jsem k tomu málo zmiňovanou vlastnost, interní HW čítač přímo v procesoru, s názvem WDT (Watch Dog Timer), po jeho aktivaci se začne v pravidelných intervalech odečítat hodnota jeho vnitřní proměnné, která jakmile dosáhne

nuly, vyvolá restart systému. Aby se systém nerestartoval, musí se pravidelně, minimálně jednou za 3,5 sekundy, navyšovat hodnota této proměnné na maximum. Zapnutí WDT bylo realizováno následující funkcí:

```
void setup() {
  // zapnuti WatchDogTimer
  enableLoopWDT();
```

V programu je na několika místech vložena funkce pro oddálení restartu a navýšení hodnoty vnitřní proměnné WDT:

```
feedLoopWDT();
```

4.5.8 Odeslání dat do systému Zabbix

Systém Zabbix umožňuje přijímat data od klientů, nebo je může sám aktivně získávat, proto jsem vyzkoušel obě varianty a jako spolehlivější se ukázala varianta odesílání dat klientem na server, kde klient může mít libovolnou IP adresu v síti, server naopak musí mít pevnou. Data odesílaná na server jsou ve formátu JSON, originální verze použité knihovny "zabbixSender.h", publikovaná na https://github.com/intellitrend/zabbix-iot, musela být upravena, obsahuje totiž funkce ze staré a nepodporované verze ArduinoJson.h, která je již ve verzi 6 zásadně přepracovaná. V adresáři projektu jsou proto vloženy dva soubory, mnou upravené verze zabbixSender.h a zabbixSender.cpp. Pro správnou funkci, jak již bylo zmíněno, je nutné instalovat pomocí „Manažeru Knihoven“ správnou verzi JSON minimálně však verzi 6.

Samotná implementace využívá mnou upravenou knihovnu, kterou bylo nutné importovat následujícím způsobem:

```
      #include "zabbixSender.h"
```

Funkce pro odeslání dat je volána jednou za třicet měření v hlavní části programu loop(){}, tedy každých 60 sekund.

```
void SendDataToZabbix() {}
```

Velmi důležitou hodnotou je proměnná `hostname`, nutná k přiřazení odeslaných hodnot ke správnému zařízení definovanému v databázi Zabbix, tato hodnota je zároveň

využita pro jméno zařízení v ArduinoOTA. Níže uvádím jen důležité části kódu, které se v případě změny počtu a názvů odesílaných hodnot můžou v budoucnu měnit.

Vytvoření JSON datové struktury s potřebnými hodnotami:

```
jsonPayload = zs.createPayload(hostname, prumernaVzdalenost, zaplneni, aktualniVzdalenost);
```

Vytvoření zprávy `msg` ve formátu `ZabbixSender`, přidá k datům `jsonPayload` potřebnou binární hlavičku:

```
String msg = zs.createMessage(jsonPayload);
```

K odeslání zprávy byl použit standartní WifiClient, dále následují jen důležité metody:

```
WiFiClient zbxClient;
zbxClient.connect(zbxServerHostName, port)
zbxClient.print(msg);
zbxClient.stop();
```

Hodnoty a názvy proměnných jsou součástí funkce `createPayload`, případné úpravy je nutné udělat ještě v následujících částech obou souborů:

```
// zabbixSender.h
String createPayload(const char *hostname, float prumer, float zaplneni, float aktualni);
```

```
// zabbixSender.cpp
String ZabbixSender::createPayload(const char *hostname, float prumer, float zaplneni, float aktualni) {
```

Název hodnoty v Zabbixu je klíč ["key"], vlastní hodnota je ["value"]:

```
JsonObject data_0 = data.createNestedObject();
    data_0["host"] = hostname;
    data_0["key"] = "prumer";
```

```
data_0["value"] = prumer;

JsonObject data_1 = data.createNestedObject();
data_1["host"] = hostname;
data_1["key"] = "zaplneni";
data_1["value"] = zaplneni;

JsonObject data_2 = data.createNestedObject();
data_2["host"] = hostname;
data_2["key"] = "aktualni";
data_2["value"] = aktualni;
```

4.5.9 Použité knihovny

Kompletní přehled knihoven a jejich verzí, které byly použity při kompilaci zdrojového kódu, tabulka slouží zároveň pro seznam knihoven potřebných k instalaci:

Tabulka 3 - Přehled knihoven použitých v projektu

Název	Verze	Umístění
Ultrasonic-master	2.1.0	\libraries\Ultrasonic-master
WiFi	2.0.0	\hardware\espressif\esp32\libraries\WiFi
ArduinoOTA	2.0.0	\hardware\espressif\esp32\libraries\ArduinoOTA
Update	2.0.0	\hardware\espressif\esp32\libraries\Update
NTPClient-master	3.1.0	\libraries\NTPClient-master
ESPAsyncWebServer	1.2.3	\libraries\ESPAsyncWebServer
FS	2.0.0	\hardware\espressif\esp32\libraries\FS
AsyncTCP	1.1.1	\libraries\AsyncTCP
SPIFFS	2.0.0	\hardware\espressif\esp32\libraries\SPIFFS
ESP8266_and_ESP32_OLED_driver_for_SSD1306_displays	4.2.1	\libraries\ESP8266_and_ESP32_OLED_driver_for_SSD1306_displays
Wire	2.0.0	\hardware\espressif\esp32\libraries\Wire
Uptime_Library	1.0.0	\libraries\Uptime_Library
Preferences	2.0.0	\hardware\espressif\esp32\libraries\Preferences
ArduinoJson	6.19.1	\libraries\ArduinoJson
ESPmDNS	2.0.0	\hardware\espressif\esp32\libraries\ESPmDNS

4.6 Zabbix server

Instalace a konfigurace Zabbix serveru se skládá z několika dílčích částí:

- Instalace,
- Administrace,
- Prezentace dat.

4.6.1 Instalace

Instalace Zabbix serveru byla umístěna na připravený operační systém Ubuntu 18.0.4 (Bionic) a hardware o parametrech: základní deska ASUS AT5IONT-I DELUXE, procesor Intel Atom D525 Dual-Core, 4GB RAM, 128GB SSD disk, pro úložiště dat Zabbixu byla použitá MySQL databáze a Apache server pro webové rozhraní (frontend).

Instalace repozitáře do systému:

```
wget  https://repo.zabbix.com/zabbix/6.0/ubuntu/pool/main/z/zabbix-release/zabbix-release_6.0-3+ubuntu18.04_all.deb
dpkg -i zabbix-release_6.0-3+ubuntu18.04_all.deb
apt update
```

Instalace Zabbix serveru a souvisejících částí:

```
apt install zabbix-server-mysql zabbix-frontend-php zabbix-apache-conf zabbix-sql-scripts zabbix-agent
```

V případě, že není nainstalován MySQL server, je nutné zajistit jeho instalaci:

```
apt install apt mysql-server
```

Vytvoření MySQL databáze (MySQL) a přidělení oprávnění, text password byl samozřejmě nahrazen skutečným heslem:

```
mysql -uroot -p password
mysql> create database zabbix character set utf8mb4 collate utf8mb4_bin;
mysql> create user zabbix@localhost identified by 'password';
mysql> grant all privileges on zabbix.* to zabbix@localhost;
mysql> quit;
```

V tuto chvíli byla k dispozici prázdná databáze a její schéma, výchozí konfigurační data pro Zabbix server byly naimportovány příkazem:

```
zcat /usr/share/doc/zabbix-sql-scripts/mysql/server.sql.gz | mysql -uzabbix -p password
```

V souboru /etc/zabbix/zabbix_server.conf bylo změněno heslo pro přístup serveru k databázi:

```
DBPassword=password
```

Instalace podpory českého prostředí vyžadovala úpravu souboru `/etc/locale.gen`, odkomentováním `'cs_CZ.UTF-8 UTF-8'`.

Nastartování služeb bylo provedeno příkazy:

```
systemctl restart zabbix-server zabbix-agent apache2
systemctl enable zabbix-server zabbix-agent apache2
```

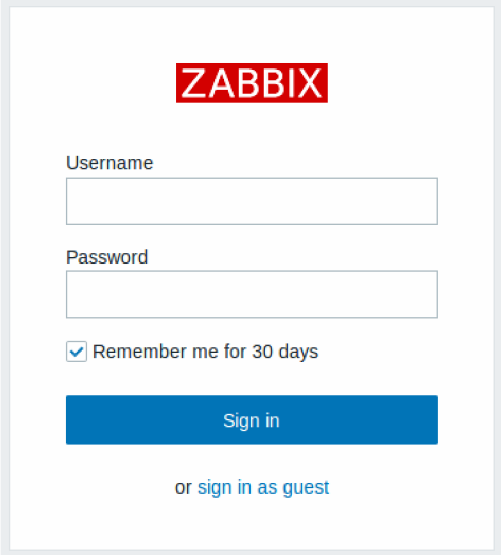
Další konfigurace pokračovala již na adrese frontend rozhraní, `http://ipadresaserveru/zabbix`.

Podrobný návod k instalaci Zabbix serveru je k dispozici na webových stránkách zabbix.com [16] (https://www.zabbix.com/documentation/5.4/en/manual/installation/frontend)

4.6.2 Administrace

Přihlášení k uživatelskému webovému rozhraní systému Zabbix bylo poprvé provedeno s výchozím uživatelským jménem *Admin* a heslem *zabbix*, které bylo následně z bezpečnostních důvodů změněno.

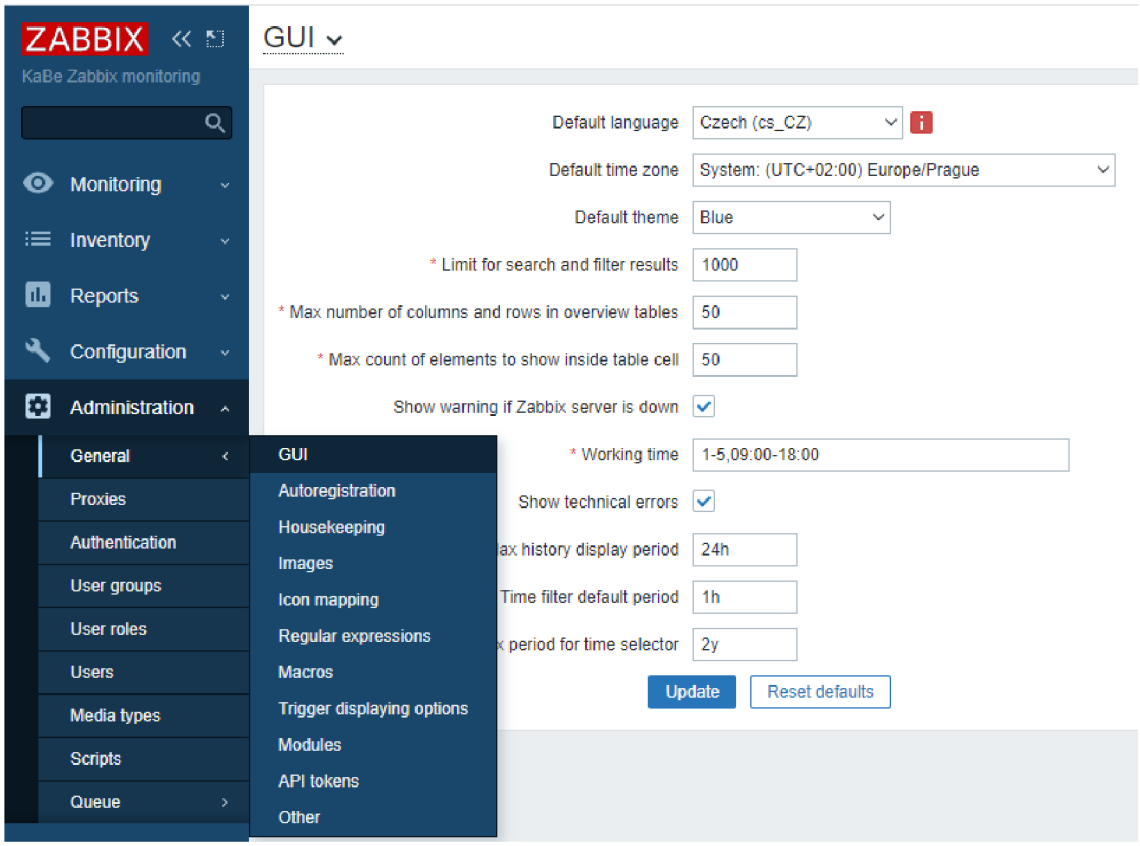
Obrázek 20 - Přihlašovací dialog Zabbix serveru



Systém Zabbix má v sobě integrovanou ochranu proti opakovaným neúspěšným pokusům o přihlášení, po pěti neúspěšných pokusech dalších 30 sekund brání případným slovníkovým útokům nebo útokům hrubou silou (brute force).

Pro Zabbix frontend byla nastavena globálně, pro všechny uživatele, česká jazyková verze, Obrázek 21 je ještě z anglického prostředí. Dále již bude následovat pouze popis české verze, pouze pokud nejsou v grafickém prostředí některé texty přeloženy, nebo se stejný text vyskytuje na různých místech česky i anglicky, použiji i původní anglický název.

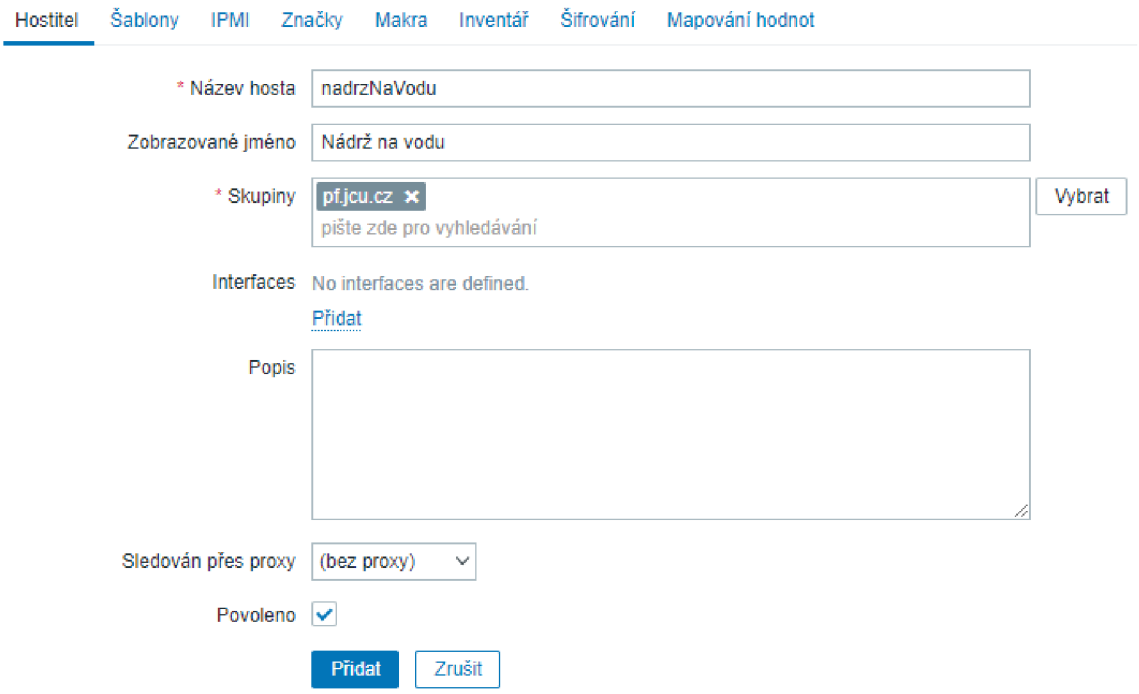
Obrázek 21 - Zabbix administrace, výběr výchozího jazykového prostředí



Vytvoření nového sledovaného zařízení (hosta) předcházelo nejprve vytvoření skupiny hostů, v menu *Nastavení – Skupiny hostů –> Vytvořit skupinu hostů* s názvem `pf.jcu.cz`

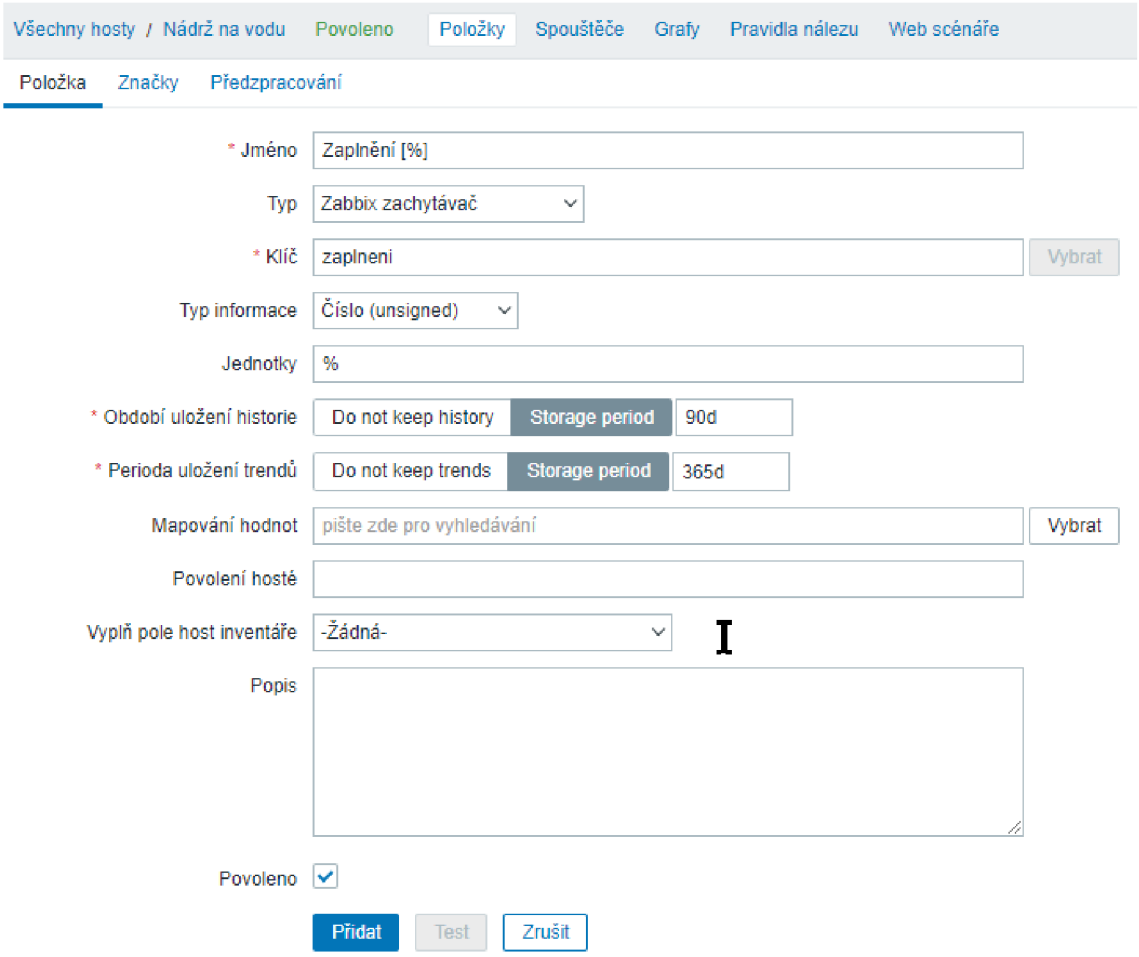
V menu *Nastavení –> Hosté –> Vytvořit hosta* jsem vytvořil první sledované zařízení, položka *Název hosta* musí přesně odpovídat proměnné `hostname` zařízení ve zdrojovém kódu, *Zobrazované jméno* je libovolné.

Obrázek 22 - Zabbix administrace, formulář přidání sledovaného zařízení



Následně jsem přidal sledované položky, jejichž klíč je shodný s názvy hodnot ve zdrojovém kódu, jedná se o tři položky s názvy `zaplneni`, `prumer`, `aktualni`. Trendy u všech položek jsou nastaveny na hodnotu zachovat 30 dní, historii `zaplneni` jsem záměrně nastavil na 1000 dní, položky `prumer` a `aktualni` stačí zachovat jen 7 dní. Důležitý je datový typ samotné hodnoty, *Číslo (unsigned)* a Typ *Zabbix zachytávač*, to znamená režim, kdy server čeká na data od hosta a sám aktivně nenavazuje žádné spojení.

Obrázek 23 - Zabbix administrace, formulář přidání položky



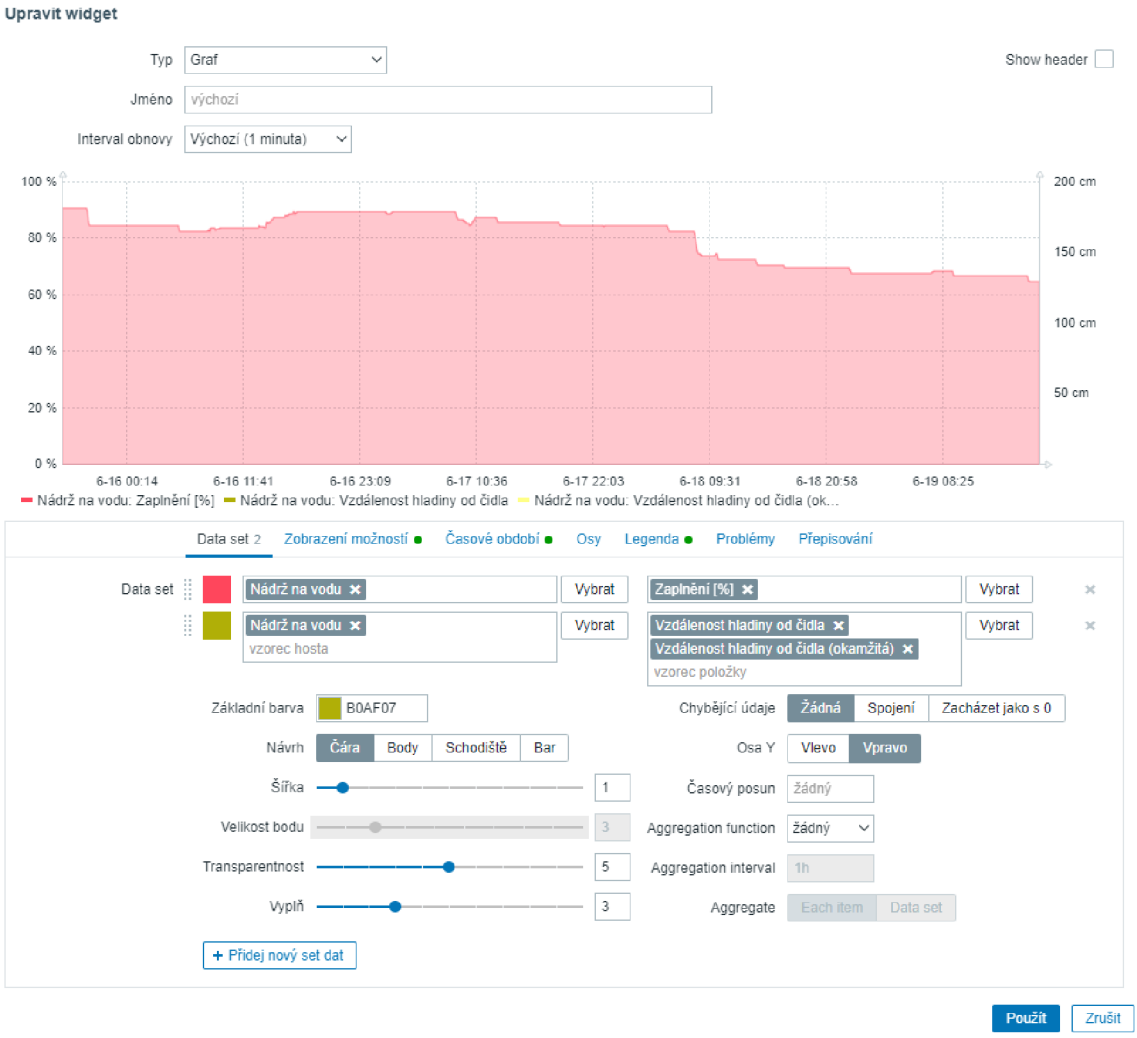
Obrázek 24 - Zabbix, přehled sledovaných položek

	Spouštěče	Klíč	Interval	Historie	Trendy	Typ
Vzdálenost hladiny od čidla		prumer		7d	30d	Zabbix zachytávač
Vzdálenost hladiny od čidla (okamžitá)		aktualni		7d	30d	Zabbix zachytávač
Zaplnění [%]		zaplneni		1000d	30d	Zabbix zachytávač

Grafy jsem vytvořil až v menu *Sledování –> Řídící panel –> Všechny panely –> Vytvořit dashboard*

Na řídící panel (dashboard) jsem přidal widget, *Typ:* Graf, dva datasety pro stejného hosta ale na každou stranu osy y jsem rozdělil položky podle stejných jednotek, nastavení os je definováno pevně, vlevo procenta v intervalu 0 až 100 %, vpravo centimetry v rozmezí 0 až 200 cm.

Obrázek 25 - Zabbix, vytvoření grafu pro řídící panel

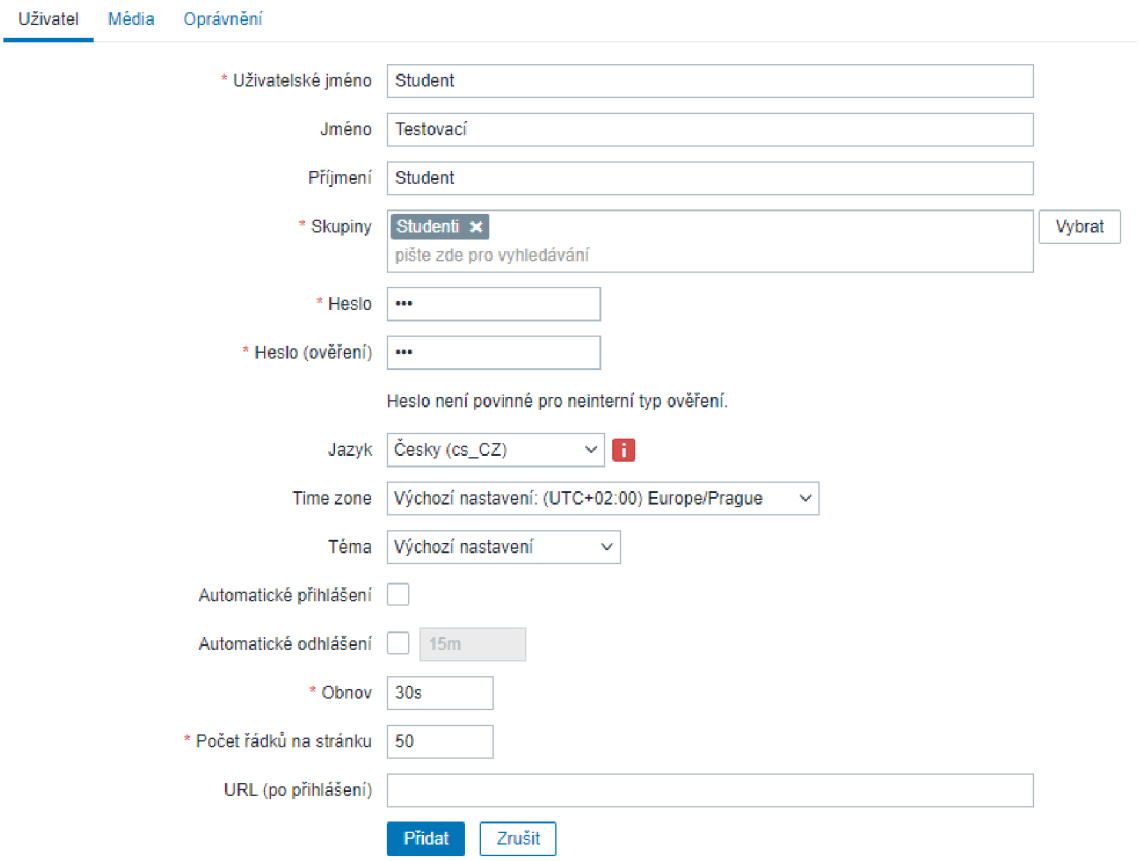


Na řídící panel lze přidat více typů zobrazovaných informací, nejen grafů, z různých hostů, mohou obsahovat textové informace nebo číselné hodnoty, různá varování a podobně, na závěr je nutné řídící panel uložit.

4.6.3 Prezentace dat

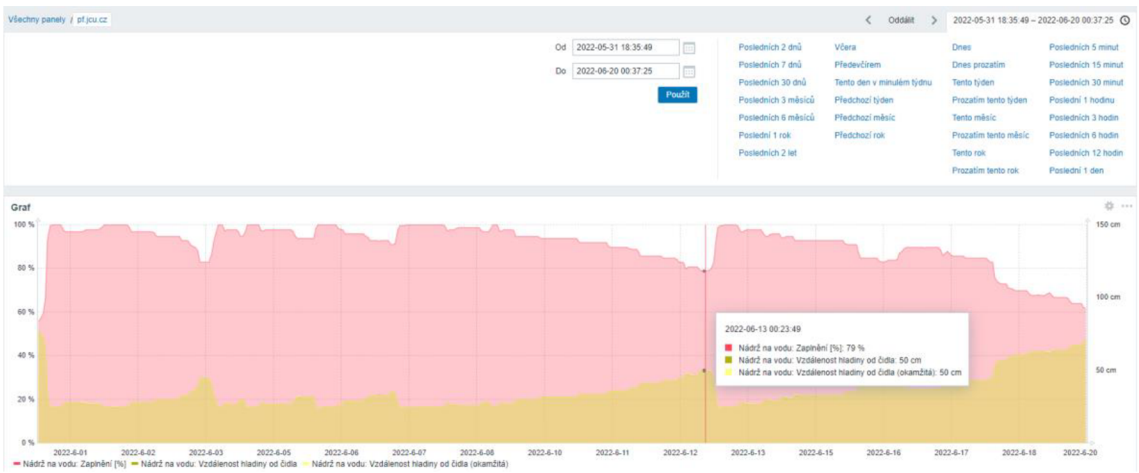
Vytvořené řídící panely jsou vstupní branou pro studenty, kteří s výslednými daty budou pracovat, je nežádoucí, aby studenti používali administrátorský účet pro celý systém Zabbix, proto jsem založil skupinu *Studenti*, přiřadil jí vzorovému uživateli s omezenými právy, přidal sdílení na vytvořený řídící panel skupině *Users* a tím připravil běžný uživatelský přístup pro jednoho studenta.

Obrázek 26 - Zabbix, formulář vytvoření nového uživatele

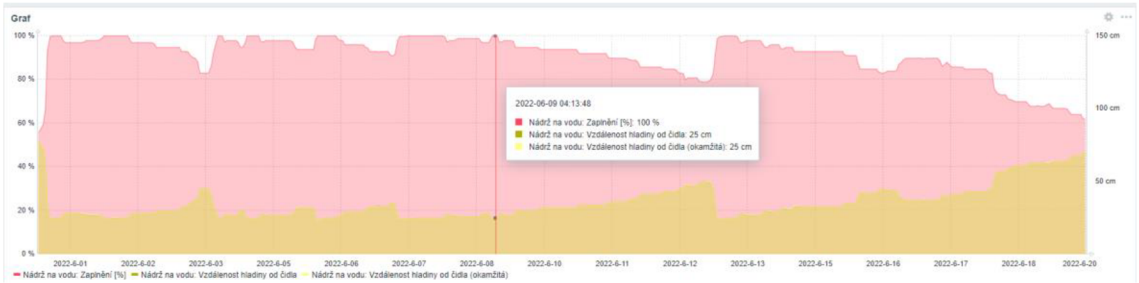


Pro čtení hodnot v určitém období je možné na řídícím panelu libovolně měnit zobrazovanou oblast, ve které již pouhým najetím myši nad křivku mohu studenti zjistit skutečné hodnoty, které se zobrazí v plovoucím obdélníku. Hodnoty před a po nějaké změně jsou pravděpodobně ty nejdůležitější data, které budou studenti hledat a odečítat z grafů.

Obrázek 27 - Zabbix, graf se zobrazením hodnoty ve zvoleném období



Obrázek 28 - Zabbix, graf s konkrétní hodnotou v místě ukazatele myši



Grafy na obrázcích ukazují, jak lze například zjistit pokles hladiny v období od 9.6.2022 do 13.6.2022 z hodnoty 100 % na 79 %, v motivačních úlohách je možné s rozdílovou hodnotou -21 % dále počítat, využít k výpočtu predikce prázdné nádrže, denní spotřeby a podobně.

5 Problémy

Během realizace jsem se potýkal s četnými problémy, které bylo nutné vyřešit a zdokumentovat pro případ jejich opětovného výskytu. Jednalo se především o problémy:

- **S instalací doplňku ESP Sketch Data Upload** do Arduino IDE prostředí

```
SPIFFS Error: esptool not found!
```

Řešení: prohledat disk, najít esptool.exe a nakopírovat do stejného umístění jako je esptool.py

```
copy
"C:\Users\<user>\Documents\Arduino\hardware\espressif\esp32\tools\e
sptool\esptool.exe"
"
C:\Users\<user>\Documents\Arduino\hardware\espressif\esp32\tools\"
```

- **Se vzorovými příklady Arduino**

asynchronní WebServer – zatuhává při vícenásobném připojení, moderní prohlížeče vždy pošlou dva požadavky současně, jeden pro samotnou požadovanou stránku a druhý pro /favicon.ico. Pokud není takový požadavek ošetřen, představuje /favicon.ico a synchronní přenos problém se zatucháváním serveru nebo internetového prohlížeče.

Řešení: používat synchronní webserver „ESPAsyncWebServer.h“, odpovídat na všechny dotazy klientů, dokonce i když taková stránka neexistuje.

```
server.onNotFound(notFound);
void notFound(AsyncWebServerRequest *request) {
    request->send(404, "text/plain", "Not found :(");
}
```

- **S úpravou knihoven**

Ultrasonic.h

- při větší vzdálenosti kabelu (15 m) mezi modulem ultrazvukového čidla a ESP32 vývojovou deskou nebyl 3,3 V spouštěcí impuls (Trigg) vždy správně vyhodnocen 5 V logikou modulu a docházelo k výpadkům měření.

Řešení: nutná úprava délky spouštěcího impulzu v knihovně Ultrasonic.cpp

ZabbixSender.cpp

- původní verze používala zastaralou knihovnu JSON 5, bylo nutné upravit pro Json 6.

Řešení: Upravené verze ZabbixSender.cpp a ZabbixSender.h jsou součástí projektu.

- **Se zatuháváním systému IoT**

Během testování a ladění nových verzí se občas stane, že chyba v kódu způsobí zatuhnutí systému, systém je bez odezvy a není bohužel na stole, je o dvě patra níže a nefunguje OTA.

Řešení: Použit wdt (WatchDogTimer) a feed (krmemí). Pokud není timer pravidelně resetován, provede se hardware reset zařízení.

- **Se zvlněnou hladinou**

Při přítoku většího množství vody dochází ke zvlnění hladiny a ovlivnění měření.

Řešení: Průměrování naměřených hodnot.

- **S nahráním firmware pomocí OTA**

Při aktualizaci firmware po síti se zastaví nahrávání s chybou *Connection Timeout*

Řešení: Vypnout firewall na operačním systému Windows.

Při aktualizaci SPIFFS po síti (Data Upload) skončí nahrávání chybou. Pravděpodobně je použito heslo pro OTA a Arduino IDE plugin neumožňuje jeho zadání.

Řešení 1: Připojit IoT zařízení USB kabelem a provést nahrání dat přes COM port.

Řešení 2: Použít příkazový řádek Windows a nahrát sestavený binární soubor příkazem

```
C:\Users\<user>\Documents\Arduino\hardware\espressif\esp32\tools\es
pota.exe -r -i <IPadresa> -p 3232 --auth=heslo -s -f
"C:\Users\<user>\AppData\Local\Temp\arduino_build_??????\Ultrasonic
_OLED_OTA_2022-06.v1.spiffs.bin"
```

6 Možnosti dalšího rozvoje

6.1 Doplnění systému o další čidla

Pouhou detekcí, uchováním a následnou analýzou dalších hodnot můžeme odhalit souvislosti s nenadálými událostmi a předejít poruše zařízení nebo vážným škodám. Rozšíření systému o další vstupní čidla nám poskytne cenná data, rozšířit systém je možné například o:

- teplotní čidlo
    - na motoru,
    - na tělese čerpadla,
    - na sacím potrubí v místě možného ohrožení mrazem,
- tlakové čidlo
    - na vzduchovém ventilku expanzní nádoby,
    - před a za filtrem nečistot ve vodovodním potrubí,
- detekci doby běhu motoru.

6.2 Automatické zavlažování

Navržená vývojová deska má dostatek výstupních pinů, které mohou být využity k ovládání ventilů automatického zavlažování, ve zdrojovém kódu je aktuální čas a den v týdnu již záměrně implementován, demonstrativně zobrazen ve webových stránkách a čeká jen na další využití. Jednoduchými kroky lze například sestavit potřebné podmínky pro optimální zalévání a ve spojení s čidlem vlhkosti půdy je možné doplnit podmínky zavlažování tak, aby se deaktivovalo, když je půda dostatečně vlhká.

6.3 Propojení více nádrží mezi sebou

Technicky není problém k jednomu IoT zařízení připojit další ultrazvukové čidlo a měřit s ním další nádrž, ovládat pomocí solenoid ventilů přepouštění vody mezi nimi, pokud jsou propojené do kaskády a správně výškově umístěné.

6.4 Webové rozhraní konfigurace

V budoucnu je vhodné uvažovat o rozšíření části konfigurace ve webovém rozhraní. Většina hodnot souvisejících s nastavením systému je stále napevno staticky definovaná ve zdrojovém kódu, změna je možná pouze jeho úpravou a opětovným

sestavením a nahráním do zařízení. Funkční aktualizace OTA je pro běžného uživatele těžko použitelná, vzhledem ke složitosti instalace celého vývojového prostředí a náročnosti jej udržet v aktuálním stavu a funkční, pokud by zařízení měly používat osoby bez programového vybavení a funkčního vývojového prostředí, mělo by být uživatelsky konfigurovatelné, a to znamená přidat následující konfigurační položky do webového rozhraní:

- Obecné nastavení:
    - název systému (hostname),
    - limitní hodnoty vzdáleností ultrazvukového čidla,
    - vzdálenost ultrazvukového čidla od hladiny.
- Nastavení Zabbix:
    - jméno nebo IP adresu a port Zabbix serveru, kam se odesílají data,
    - možnost úplně vypnout odesílání dat do Zabbix serveru.
- Zabezpečení:
    - nastavení hesla pro OTA,
    - jméno a heslo pro uložení konfigurace.

Další možnou úpravou pro pohodlnou konfiguraci by bylo, například po propojení dvou pinů na desce propojkou spustit IoT zařízení v režimu WIFI AccessPoint a umožnit tím připojení se k němu například mobilním telefonem a provést tak jeho nastavení, včetně WIFI klienta a umožnit případný reset uložených hesel.

6.5 Archivace dalších hodnot

Do systému Zabbix můžeme mimo již zmíněné rozšiřující hodnoty odesílat i sílu signálu WIFI RSSI, díky ní budeme schopni v případě problémů snáze detekovat možné problémy se sítí.

7 Motivační úlohy

Popsané motivační úlohy jsou pouze možným návrhem s případným zjednodušeným řešením, jejich použití je ve většině případů vázáno na funkční systém IoT, Zabbix a části popsané v předchozích kapitolách.

7.1 Informační technologie

7.1.1 Programování, bezpečnost

- Najděte ve zdrojovém kódu všechna hesla.
- Na kolika místech a jaká hesla jste našli?
- Která hesla před kompilací svého zařízení změníte?

Odpověď:

✓ Všechna hesla jsou pro přehlednost definovaná pouze v souboru settings.h, jsou celkem tři, pro:

WIFI připojení,

web server samotného IoT (pro stránky s uložením konfigurace),

aktualizace firmware OTA.

✓ Změníme samozřejmě všechny hesla.

7.1.2 Význam a použití „hostname“

- Najděte ve zdrojovém kódu *hostname* a vysvětlete, jeho účel a použití.
- Co se stane, když budou mít všichni studenti svá IoT zařízení se stejnou hodnotou proměnné *hostname*?

Odpověď:

✓ Pomocí CTRL+F najdeme klíčové slovo *hostname* ve všech zdrojových souborech projektu IoT. Používá se k identifikaci zařízení v systému Zabbix, pro název zařízení v menu vývojového prostředí a zobrazení názvu ve webovém rozhraní, podrobněji je vše popsáno v kapitolách 4.5.2, 4.5.5 a 4.5.8.

✓ Pokud budou mít všichni stejné *hostname*, budou v Zabbix systému všechny data ze všech zařízení smíchaná dohromady.

7.2 Elektrotechnika

7.2.1 Spínání výkonových součástek

- Najděte na internetu součástku nebo modul vhodný do tohoto projektu (např. dratek.cz, laskarduino.cz) pro spínání jednofázového spotřebiče s příkonem 1100 W a napětí 240 V.
- Pokud jste jich našli více, podle jakých parametrů je budete porovnávat? Jaké jste našli a jaké jsou v nich rozdíly?

Odpovědi:

✓ Musí být dimenzovány na proud větší než 4,6 A.

✓ Solid State Relay (SSR).

✓ Relé elektromagnetické (+ tranzistor nebo optočlen, pokud je součástí modulu).

Pozor: Pro zapojení těchto součástek je třeba osoba s potřebnou kvalifikací dle vyhlášky č. 50/1978 Sb. (Vyhláška Českého úřadu bezpečnosti práce a Českého báňského úřadu o odborné způsobilosti v elektrotechnice) nebo nově dle zákona č. 250/2021 Sb. (Zákon o bezpečnosti práce v souvislosti s provozem vyhrazených technických zařízení a o změně souvisejících zákonů).

7.2.2 Spotřeba zařízení

- Změřte klidový proud potřebný pro napájení IoT zařízení včetně modulu ultrazvukového čidla.
- Jaký je příkon zařízení?
- Kolik spotřebuje energie za 24 hodin?
- Jak dlouho je teoreticky možné napájet zařízení z powerbanky s kapacitou 10000 mAh?
- Jak lze snížit jeho příkon? Navrhněte možnosti a případně je vyzkoušejte a proveďte měření.

Odpověď:

- ✓ Proud je třeba skutečně změřit, například pomocí přístroje UM34C. Dle mých měření je proud průměrně 0,09 A, při komunikaci přes WIFI a zobrazení html stránek se špičkově dostane až k 0,14 A.
- ✓ Příkon zařízení je 0,45 W.
- ✓ Za 24 hodin zařízení spotřebuje 0,45 W x 24 h = 10 Wh.
- ✓ Udávaná kapacita powerbanky je většinou kapacita baterie, takže při jejím průměrném napětí 3,6 V je její kapacita 3,6 V x 10000 mAh = 36 Wh. Z powerbanky je možné napájet zařízení 3,6 dní.
- ✓ Příkon lze snížit například snížením frekvence procesoru, před kompilací zdrojového kódu v Arduino IDE, v menu *Nástroje -> CPU Frequency*, místo 240 MHz je systém funkční i s frekvencí 80 MHz. Příkon zařízení se tím sníží o více než polovinu. Dalšího snížení příkonu lze dosáhnout například prodloužením intervalu mezi každým měřením ultrazvukovým čidlem a prodloužením intervalu mezi každým odesíláním dat do systému Zabbix.

7.3 Fyzika

7.3.1 Výpočet potřebné velikosti nádrže

- Kolik vody naprší ročně ve Vaší lokalitě na 1 m^2 plochy?

Odpověď:

- ✓ 700 mm (lokalita České Budějovice)

- Pro střechu s půdorysnou plochou 150 m^2 potřebuji zadržet veškerou vodu, která průměrně naprší v období jednoho měsíce. Jak objemnou nádrž budu potřebovat?

Odpověď:

- ✓ 700 mm / 12 měsíců = 58,33 mm
- ✓ 58,33 mm * 150 m^2 = 8,75 m^3
- ✓ Potřebuji nádrž o objemu 8,75 m^3

Průměrně nejvíce srážek spadne ročně v lokalitě **Bílý potok** v Jizerských horách, a to **1705 mm**. Oblast leží na návětrné straně hor, a proto je srážkově velmi bohatá. Průměrný roční úhrn srážek v ČR je zhruba 600 - 800 mm. [17]

Tabulka 4 - Územní srážky v roce 2021 [18]

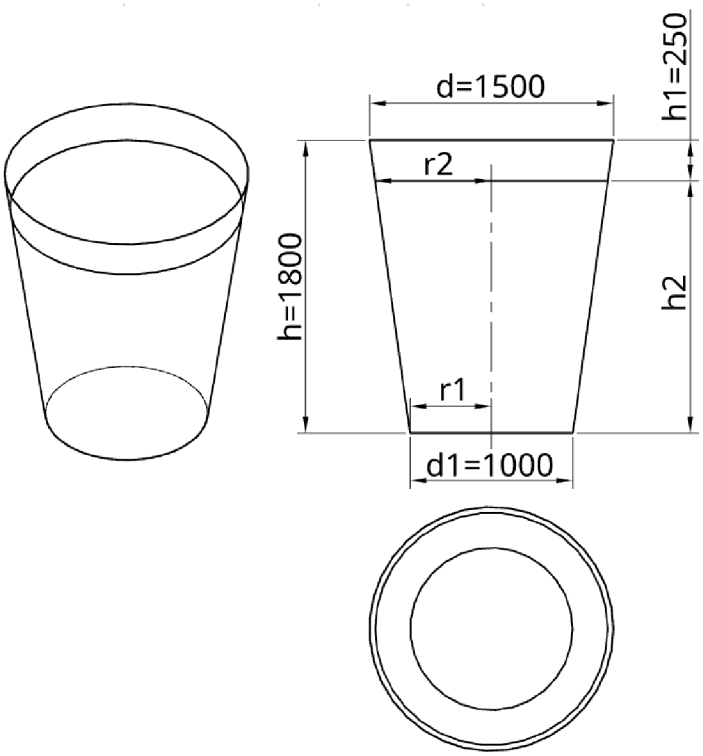
Kraj		Měsíc												Rok
		I.	II.	III.	IV.	V.	VI.	VII.	VIII.	IX.	X.	XI.	XII.	
Česká republika	S	55	38	28	32	99	88	107	106	23	19	46	42	683
	N	44	37	46	39	70	82	89	78	60	49	45	46	684
	%	125	103	61	82	141	107	120	136	38	39	102	91	100
Jihočeský	S	54	29	30	30	106	113	116	108	16	20	40	39	701
	N	42	33	47	39	75	92	94	85	56	48	41	41	694
	%	129	88	64	77	141	123	123	127	29	42	98	95	101

Vysvětlivky:
S = úhrn srážek [mm], N = dlouhodobý srážkový normál 9120 [mm], % = úhrn srážek v % normálu 9120

7.3.2 Výpočet objemu nádrže

- Vypočtěte objem kónické nádrže s kruhovou podstavou o průměru 1 m, výšce 1,8 m a horním otvorem s průměrem 1,5 m, pokud je maximální možná hladina vody ve výšce 250 mm od horní hrany nádrže.

Obrázek 29 - Výkres nádrže ve tvaru komolého kužele



Návrh řešení:

Musíme nejprve vypočítat skutečnou výšku hladiny a poloměr kružnice v její výšce r_2. Pomocné výpočty jsou například následující:

$$h_2 = h - h_1$$

$$r = \frac{d}{2}$$

$$r_1 = \frac{d_1}{2}$$

$$r_2 = \frac{r - r_1}{h} \cdot h_2 + r_1$$

Poté použijeme vzorec pro výpočet komolého kužele [19]:

$$V = \frac{1}{3} \cdot \pi \cdot h_2 \cdot \left(r_1^2 + r_1 \cdot r_2 + r_2^2\right)$$

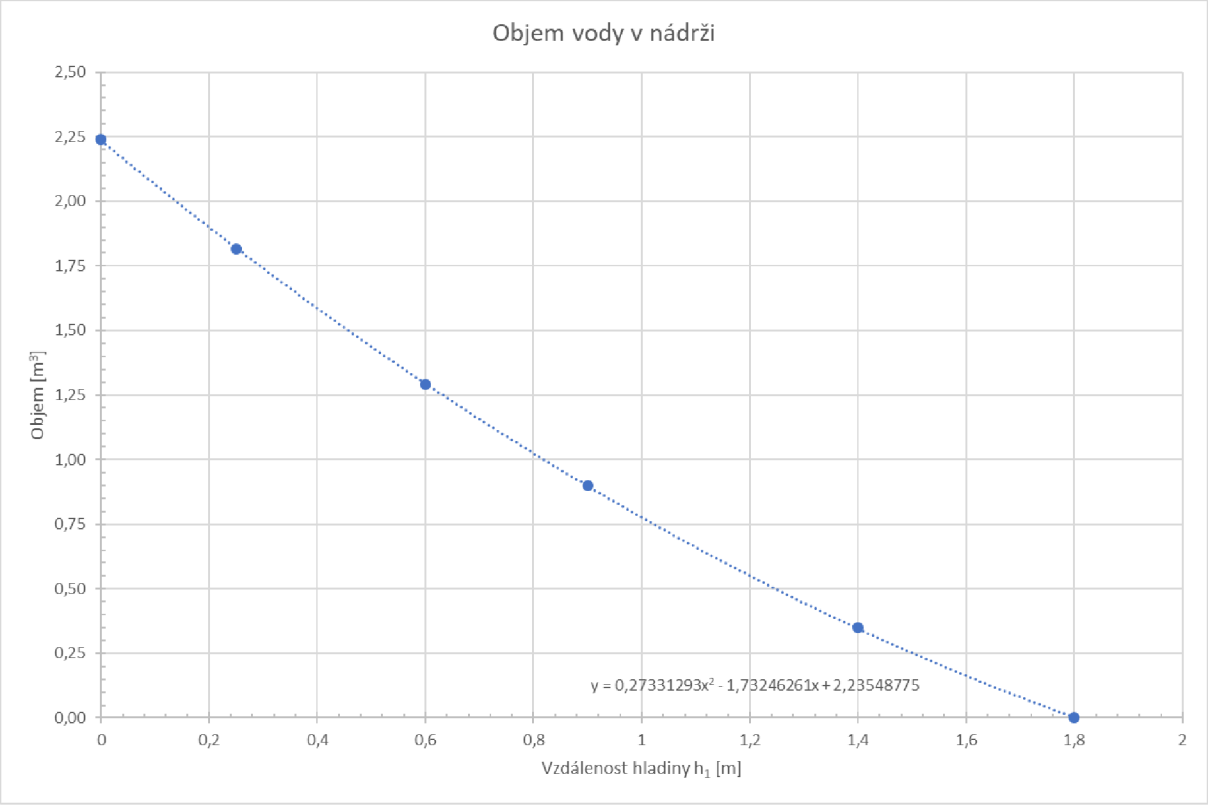
✓ Výsledný objem je V = 1,82 m^3.

- Sestrojte graf závislosti objemu vody v nádrži na vzdálenosti hladiny od horního okraje nádrže (h_1). Je možné využít rovnici spojnice trendu k výpočtu objemu?

Odpověď:

Je nutné vypočítat několik hodnot pro různé vzdálenosti a pak sestrojit graf, například v programu MS Excel.

Obrázek 30 – Graf závislosti objemu vody v nádrži na vzdálenosti hladiny



✓ Ano, rovnici je možné využít k jednoduššímu a rychlejšímu výpočtu, je možné sestrojit podobný graf s rovnicí pro různě složité a členité nádrže a následně ji použít v matematických výpočtech, které by jinak byly velmi náročné. Své uplatnění najde ve zjednodušení zdrojového kódu v IoT zařízení.

7.3.3 Rychlost zvuku

- Jaká je rychlost zvuku ve vzduchu?
- Vypočítejte čas, za který se vrátí odražená zvuková vlna od překážky vzdálené:
  a) 1,5 m
  b) 3 km

Odpověď:

✓ Rychlost zvuku je přibližně 340 ms^{-1}.

✓ Odražená vlna se vrátí za $t = 2s/v$

a) 8,82 ms

b) 17,6 s

Princip je podrobněji popsán v kapitole 3.3.1.

7.4 Ekologie

7.4.1 Porovnání srážkového úhrnu s naměřenými hodnotami

- Najděte na internetu denní srážkový úhrn ve své lokalitě za předchozích 14 dnů (pokud nepršelo, tak vyberte jiné období, ke kterému máte v systému Zabbix nasbíraná data).

  Vytvořte graf denního srážkového úhrnu za stejné období z naměřených hodnot zaplnění vaší nádrže v systému Zabbix, zapište na papír nebo do tabulky např. MS Excel a porovnejte je s hodnotami z internetu.

- Predikce nedostatku vody, z grafů v systému Zabbix:
  a) zjistěte úbytek objemu vody v nádrži za jeden den běžného zalévání,
  b) z předpovědi počasí víte, že následujících 14 dní pravděpodobně nebude pršet. Na jak dlouho vydrží zbývající objem v nádrži, pokud budete zalévat pravidelně stejným množstvím každý den?

Nápověda: Hodinové srážkové úhrny lze najít například na stránkách Českého hydrometeorologického ústavu, https://hydro.chmi.cz/hpps/hpps_act_rain.php.

7.4.2 Ušetřené náklady

- Zjistěte cenu vodného a stočného pitné vody ve vaší lokalitě.

Odpověď:

✓ Cena v lokalitě České Budějovice je 44,29 Kč za m^3.

(https://www.nase-voda.cz/vodne-a-stocne-2022-prehled-cen-jednotlivych-spolecnosti/)

- Z grafů v systému Zabbix zjistěte měsíční spotřebu na zalévání užitkovou vodou z akumulační nádrže (součet všech úbytků) a přepočítejte na objem v m^3.

  a) Jaké by byly náklady na zalévání, kdybychom nepoužívali užitkovou ale pitnou vodu?

  b) Jaké položky je třeba započítat do ceny pro výpočet nákladů?

Odpověď:

  a) Výpočet bude záviset na konkrétních datech v systému Zabbix.

  b) Náklady na pořízení systému rozpočítané do období předpokládané životnosti, dále spotřebovanou energii a cenu za údržbu.

- Na splachování toalety každý z nás potřebuje přibližně 24 litrů vody denně. Kolik vody spotřebuje průměrná čtyřčlenná domácnost ročně? Kolik peněž uspoříte, pokud budete toalety splachovat užitkovou vodou?

Odpověď:

✓ Ušetříme přibližně 35 m^3 vody v ceně 1552 Kč ročně.

8 Zhodnocení a závěr

Výsledkem diplomové práce bylo navržení, vytvoření a otestování IoT systému pro monitorování nádrže na dešťovou vodu, jeho praktické využití nejen v běžném provozu, ale i v různých formách vzdělávání, při výuce fyziky, technické výchovy nebo infomačních a komunikačních technologií, pomocí motivačních úloh pro studenty.

V úvodní části popisuji, co je IoT, historii a důležitost zadržování vody, její význam pro společnost a důvody, proč se tomuto tématu věnovat. V druhé části se dotýkám témat z oblasti stavebních úprav a variant řešení nádrže na dešťovou vodu a potřebné legislativy, dále se věnuji návrhu technického řešení v oblasti elektrotechniky, informačních a komunikačních technologií a bezpečnosti. V praktické části popisuji konkrétní postupy použité při realizaci, instalaci vývojového prostředí, části zdrojového kódu a jeho význam, instalaci a konfiguraci serverové části Zabbix a systém prezentace naměřených dat. Poslední část obsahuje návrh motivačních úloh rozdělených do čtyř oblastí, informační technologie, elektrotechnika, fyzika a ekologie.

Před vlastní prací bylo nutné nejdříve nashromáždit všechny potřebné informace k danému tématu napříč mnoha obory a seznámit se s obecnou problematikou a principy fungování všech použitých součástí, definovat požadavky na nový systém, analyzovat fyzikální měřitelné veličiny a teprve za pomocí nich navrhnout konkrétní vhodné technické řešení, včetně rozborů nevhodných řešení, kterých je nutno se naopak vyvarovat. Výsledky měření jednotlivých fyzikálních veličin jsou ukládány, archivovány a prezentovány systémem Zabbix. Práce obsahuje možná doporučení pro další rozvoj, jsou v ní rozepsány kroky pro sestavení vlastního systému včetně částí zdrojových kódů, hardwarových komponent, popisu jednotlivých částí programu, popisu ukládání dat, možnosti jejich prezentace v lokálních sítích, nebo na mobilních zařízeních. Celý systém je modulární a snadno rozšiřitelný, byl navržen tak, aby mohl být využit pro vzdělávání studentů různých věkových kategorií s rozdílnými znalostmi z pohledu ekologie, ale i fyziky a techniky a samozřejmě s ohledem na minimalizaci vstupních nákladů. Poslední částí diplomové práce je navržený soubor motivačních úloh pro různé formy vzdělávání mládeže, kterými jsou studenti více vtaženi do celé problematiky, mohou sami monitorovat hladinu nádrže na dešťovou vodou s využitím IoT technologií, včetně sběru, vyhodnocení a prezentace naměřeních dat, predikovat vývoj stavu zaplnění nádrže, přímo zasahovat do zdrojového kódu programu, modifikovat a rozšiřovat jeho funkce, nejen softwarové ale i hardwarové. K tomu bylo zapotřebí mít data z funkčního, sestaveného a

nakonfigurovaného systému, IoT zařízení, Zabbix serveru, infrastruktury a mít zkušenosti s vlastním provozem, proto byl systém nejdříve implementován do provozu mojí vlastní domácnosti a jeho funkčnost tak byla ověřena v reálném provozu. Na základě dlouhodobého provozu, měření a zajištěných dat mohu konstatovat, že navržený systém je funkční a lze jej využívat ke svému účelu a tím tak byly všechny stanovené cíle a podmínky diplomové práce splněny.

Literatura a zdroje

[1] JANÁSEK, Jaroslav. MILPO MÉDIA S.R.O. *HISTORIE VODÁRENSTVÍ: úryvky z knihy Vodárenství v Čechách, na Moravě a ve Slezsku* [online]. In: . [cit. 2022-06-24]. Dostupné z: https://www.pvs.cz/historie/historie-vodarenstvi/

[2] Seznam přehradních nádrží v Česku. In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001 [cit. 2022-03-26]. Dostupné z: https://cs.wikipedia.org/wiki/Seznam_přehradních_nádrží_v_Česku

[3] *Plochy rybníků v ČR byly v minulosti oproti dnešku trojnásobné* [online]. Nature media, s.r.o., 2017 [cit. 2022-06-2]. Dostupné z: https://www.nase-voda.cz/plochy-rybniku-cr-byly-minulosti-oproti-dnesku-trojnasobne/

[4] SFŽP. Dotace Dešťovka: Víte, kolik dostanete?. In: *Státní fond životního prostředí ČR* [online]. Praha [cit. 2022-06-4]. Dostupné z: https://www.sfzp.cz/dotace-destovka-vite-kolik-dostanete/

[5] GYMNÁZIUM KARLA ČAPKA. Důsledky nedostatku pitné vody. In: *Gymnázium Karla Čapka* [online]. [cit. 2022-05-20]. Dostupné z: https://www.gymkc.cz/files/dusledky-nedostatku-pitne-vody.pdf

[6] *Dratek.cz: VELKOOBCHOD, MALOOBCHOD S ARDUINEM* [online]. 2022 [cit. 2022-05-11]. Dostupné z: https://dratek.cz/

[7] ESPRESSIF SYSTEMS. *Wi-Fi & Bluetooth MCUs and AIoT Solutions I Espressif Systems* [online]. 2022 [cit. 2022-04-24]. Dostupné z: https://www.espressif.com

[8] (20+) ESP8266 | http://hackaday.com/2016/07/28/ask-hackaday-is-the-esp8266-5v-tolerant/ | Facebook. In: *Facebook* [online]. [cit. 2022-07-01]. Dostupné z: https://www.facebook.com/groups/1499045113679103/posts/1731855033731442/?comment_id=1732364133680532

[9] Geekcreit esp32 oled module esp32 oled wifi module + bluetooth dual esp-32. In: *Banggood.com* [online]. [cit. 2022-06-01]. Dostupné z: https://www.banggood.com/Geekcreit-ESP32-OLED-Module-ESP32-OLED-WiFi-Module-+-bluetooth-Dual-ESP-32-p-1181297.html?imageAb=1&akmClientCountry=CZ&a=1656635253.7795&cur_warehouse=CN&DCC=CZ&currency=CZK&akmClientCountry=CZ

[10] ZABBIX LLC. *Zabbix :: The Enterprise-Class Open Source Network Monitoring Solution*. 2022. Dostupné také z: https://www.zabbix.com/

[11] RAVI RAJ BHAT, . Jak zabezpečit IoT a jak před ním ochránit svou síť. In: *ChannelWorld* [online]. Praha: Internet Info, s.r.o., 2019 [cit. 2022-06-22]. Dostupné z: https://www.channelworld.cz/press-room/jak-zabezpecit-iot-a-jak-pred-nim-ochranit-svou-sit

[12] *160+ ESP32 Projects, Tutorials and Guides with Arduino IDE* [online]. RandomNerdTutorials.com, 2022 [cit. 2022-06-01]. Dostupné z: https://randomnerdtutorials.com/projects-esp32/

[13] *Arduino* [online]. 2022 [cit. 2022-04-04]. Dostupné z: https://www.arduino.cc/

[14] CAMERON, Neil. *Electronics Projects with the ESP8266 and ESP32: Building web pages, applications, and WiFi enabled devices*. Edinburgh: Apress, 2021. ISBN 978-1-4842-63358.

[15] ŠRAJER, Milan. *OTA programování ESP8266 | Milan Šrajer*. 2022. Dostupné také z: http://milansrajer.cz/ota-programovani-esp8266/

[16] Documentation. *Zabbix*. 2022. Dostupné také z: https://www.zabbix.com/manuals

[17] Nejvyšší průměrné roční srážky v ČR (Rekordy a kuriozity - počasí, srážky) • Mapy.cz. *Mapy.cz*. 2022. Dostupné také z: https://mapy.cz/zakladni?x=15.2733788&y=50.8443377&z=17&source=base&id=1913837

[18] ČESKÝ HYDROMETEOROLOGICKÝ ÚSTAV. Portál ČHMÚ : Historická data : Počasí : Územní srážky. *Portál ČHMÚ*. 2022. Dostupné také z: https://www.chmi.cz/historicka-data/pocasi/uzemni-srazky#

[19] DELVENTHAL, Katka, Alfred KISSNER a Malte KULICK. *Kompendium matematiky: vzorce a pravidla : četné příklady včetně řešení : od základních operací po vyšší matematiku*. V Praze: Knižní klub, 2004. Universum (Knižní klub). ISBN 80-242-1227-7.

Seznam obrázků

Všechny použité obrázky bez uvedeného zdroje jsou mé vlastní tvorby, mnou nakreslené, nebo se jedná o mé pořízené snímky obrazovky z nainstalovaného systému v průběhu realizace.

Seznam tabulek

Přílohy

Přílohou tištěné verze je CD-ROM, který obsahuje zdrojové kódy popsané v části 4.5., schéma zapojení slaboproudé části ve větším rozlišení a protokol z kompilace zdrojového kódu. Elektronická verze obsahuje výše zmíněné přílohy v jednom komprimovaném souboru, který je nahrán do informačního systému STAG provozovaném Jihočeskou univerzitou v Českých Budějovicích.