SEN – Inteligentní senzory
Implementace meteostanice s autonomním řízením

Bc. Petr Stehlík <xstehl14@stud.fit.vutbr.cz>

1 Úvod

Cílem projektu je navrhnout a implementovat meteostanici využívající MQTT protokol pro zasílání naměřených hodnot serveru. Server implementuje databázi a webové grafické uživatelské rozhraní (GUI), analyzuje historické hodnoty a na jejich základě řídí akce aktuátorů.

Meteostanice měří teplotu, vlhkost a tlak vzduchu; vlhkost v květináči a světelnou intenzitu. Navržené aktuátory jsou ovládání žaluzií, klimatizace, topení a zavlažování rostlin. Vše je autonomně ovládáno na základě předem stanovených pravidel.

GUI zobrazuje aktuální a historické naměřené hodnoty jednotlivých senzorů, trendy a naposledy vykonané akce aktuátorů společně s krátkodobou předpovědí počasí získanou z volně dostupných zdrojů.

2 Návrh řešení

Pro řešení byla zvolena kombinace Angular pro klientskou část webové aplikace a Flask (Python) s SQLite3 pro backend. Pro distribuci zpráv od senzorů k serveru a mezi aktuátory a serverem byl vybrán MQTT broker Mosquitto.

Rozšířeným frameworkem pro tvorbu klientské části webových aplikací je Angular[1]. Angular umožňuje snadnou integraci různých knihoven pro zobrazování grafů a tvorbu interaktivních single-page aplikací (SPA).

K implementaci serverové části webové aplikace je možné použít framework Flask[2] napsaný v jazyce Python. Flask umožňuje rychlou a snadnou tvorbu REST API na propojení serverové a klientské části webové aplikace. Výhodou použití jazyku Python pro tvorbu serverové části aplikace je, že obsahuje moduly pro obsluhu databázového systému a zprotředkování síťové komunikace s dalšími prvky celého řídícího systému.

K vyčítání hodnot ze všech zvolených senzorů byl také zvolen Python, protože existují volně dostupné knihovny pro všechny senzory.

3 Architektura a implementace

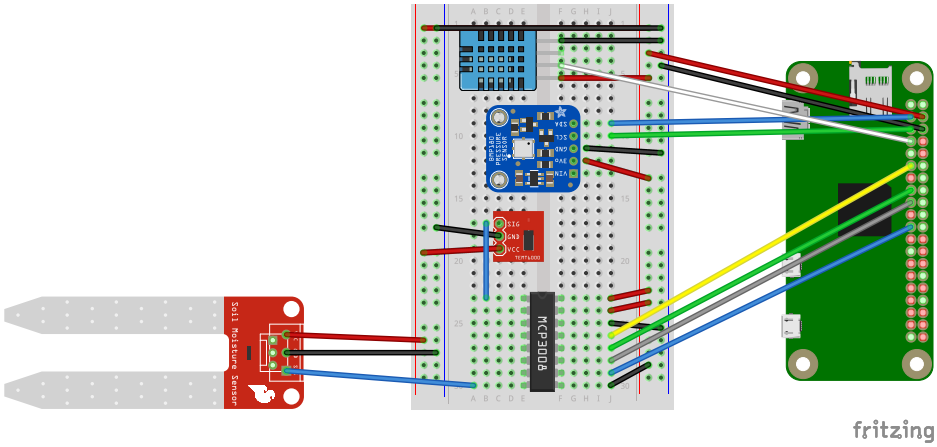
Meteostanice obsahuje senzory, z nichž jsou data přenášena na server. Server data ukládá do databáze, řídí činnost aktuátorů podle pravidel, které jsou také uloženy v databázi a obsluhuje webovou aplikaci, která poskytuje grafické uživatelské rozhraní k celému systému.

Činnost aktuátorů je simulována a pro účely testování byl také sestrojen modul pro simulování činnosti senzorů.

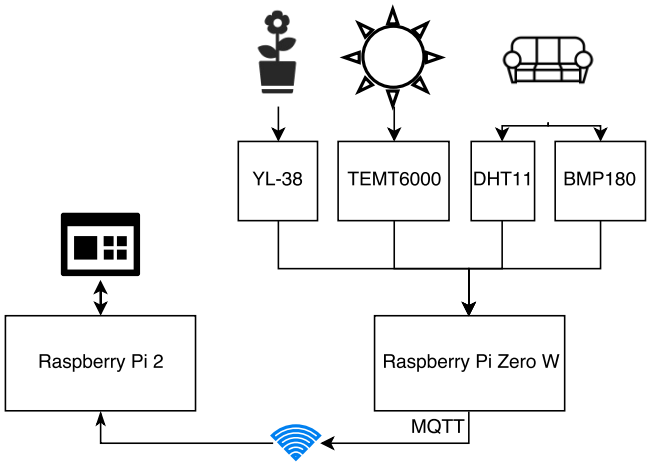
3.1 Hardware

Základem pro hardware byly zvoleny Raspberry Pi 2 jako server a Raspberry Pi Zero W jako samotná meteostanice. Server realizován jako Raspberry Pi 2 zpřístupňuje uživateli minimalistické webové rozhraní a slouží jako MQTT broker pro meteostanici. Také jsou zde ukládány všechny naměřené hodnoty do SQLite3 databáze a spuštěny simulované aktuátory. Na obrázku 1 je schéma zapojení jednotlivých senzorů k Raspberry Pi Zero W a na obrázku 2celkové schéma zapojení hardware.

[1] `https://angular.io/`
[2] `http://flask.pocoo.org/`



Obrázek 1: Zapojení jednotlivých senzorů na Raspberry Pi Zero W.



Obrázek 2: Schéma propojení hardware.

K Raspberry Pi Zero W jsou připojeny následující senzory, které posílají hodnoty každých 30 sekund:

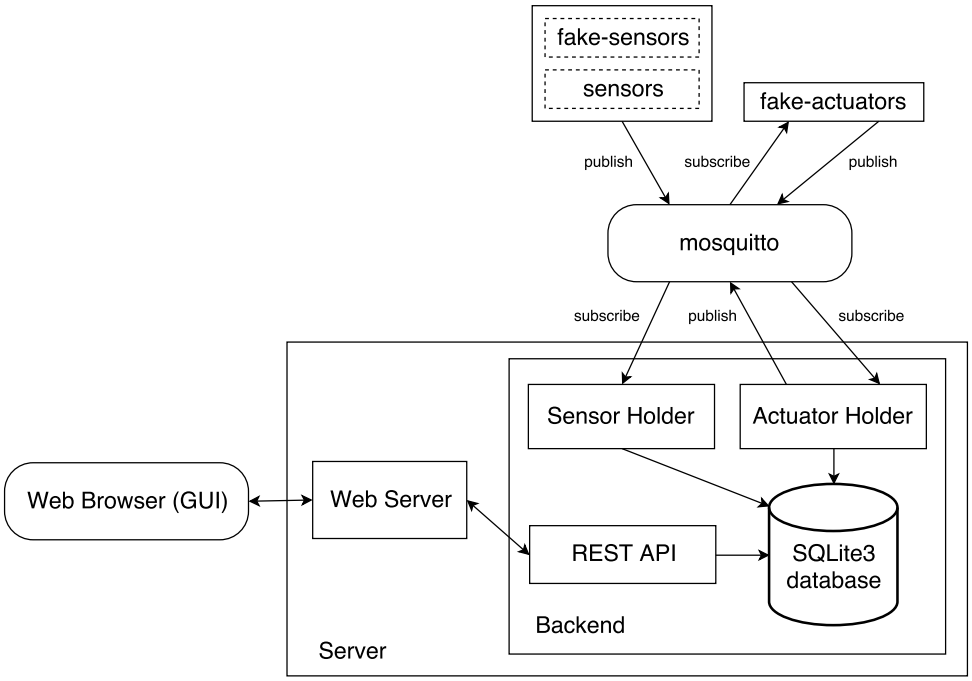
- DHT11 - digitální senzor pro měření teploty a vlhkosti vzduchu
- BMP180 - digitální senzor pro měření barometrického tlaku, teploty a nadmořské výšky
- TEMT6000 - analogový senzor pro měření intenzity okolního světla
- YL-69 společně s YL-38 - analogový senzor pro měření vlhkosti půdy

Pro konverzi analogových senzorů je použit AD převodník MCP3008, který komunikuje přes protokol SPI. Získávání dat ze senzorů probíhá tak, že se vyčte 5 hodnot, seřadí se od nejmenší po největší a první a poslední hodnota se zahodí. Ze zbylých 3 hodnot je spočítán průměr a ten je odeslán přes MQTT.

3.2 Software

Na obrázku 4 je znázorněné propojení jednotlivých modulů a způsob jejich komunikace.

Interakci s uživatelem zprostředkovává webová aplikace, která se skládá z grafického uživatelského rozhraní zobrazovaného ve webovém prohlížeči a serverové části.



Obrázek 3: Schéma propojení software modulů.

Webové GUI je implementované ve frameworku Angular. Webová aplikace dále využívá knihovnu Dygraphs pro zobrazování čárových grafů pro jednotlivé naměřené metriky. GUI získává data pomocí REST API ze serverové části aplikace. Při prvním načtení se získají historická data za posledních 24 hodin a dále si GUI získává nová data každých 5 sekund.

K implementaci REST API je použit framework Flask. Serverová část webové aplikace (backend) přijímá data ze senzorů a ukládá je do databáze. Na základě dat ze senzorů a nastavených pravidel určuje činnost a řídí aktuátory.

REST API umožňuje:

- získat poslední vzorek dat ze senzorů včetně časové známky spolu s trendem vypočítaným pro každou měřenou metriku
- získat informace o aktuálním stavu aktuátorů a pravidlech, kterými jsou řízeny
- nastavit nové prahové hodnoty jednotlivým aktuátorům
- získat předpověď počasí na 5 dní pomocí API OpenWeatherMap

Výpočet trendů probíhá na základě lineární aproximace z určitého počtu posledních vzorků dané veličiny. Pro každou veličinu je tak určena funkce ve tvaru:

$$y = ax + b$$

Koeficient a udává jako prudce funkce klesá nebo stoupá. Koeficient b udává posunutí funkce po ose x. Toto posunutí nemá na změnu veličiny vliv a proto může být z dalších výpočtů vypuštěno. Za x je dosazen počet vzorků (120 vzorků), z kterých sa počítá trend. Hodnota y udáva o kolik sa daná veličina za posledních x vzorků změnila.

Pro každou veličinu byly empirickým pozorováním určené hodnoty prahů. Pokud je absolutní hodnota změny veličiny větší než práh, tak podle znaménka změny veličiny je určen stoupající nebo klesající trend, jinak je určen trend rovnoměrného průběhu.

V databázi SQLite3 sa nachází 3 tabulky:

- `weather_data` - data získaná ze senzorů

- `actuators` - informace o stavu aktuátorů
- `thresholds` - informace o pravidlech řídících činnost aktuátorů

Data ze senzorů získává, agreguje a následně odesílá backendu modul `sensors`. Pro testovací účely byl připraven i modul `fake-sensors`, který simuluje činnost modulu `sensors` a odesílá umělá data ve stejném formátu.

Modul `fake-actuators` simuluje činnost aktuátorů. Z backendu přijímá příkazy na změnu stavu aktuátorů, které simuluje. Po provedení činnosti aktuátorů je odeslána informace o stavu, ve kterém se aktuátor nachází backendu.

Vytvořené simulované aktuátory jsou následující: klimatizace a topení–řízené na základě údajů o teplotě, žaluzie–řízené na základě údajů o intenzitě světla, zavlažování rostlin–řízené na základě údajů o vlhkosti půdy.

MQTT topic pro zprávy zasílané senzory má tvar `home/ws/sensor/<jméno senzoru>` a obsah dané zprávy ve formátu `<unix časové razítko>;<hodnota>`.

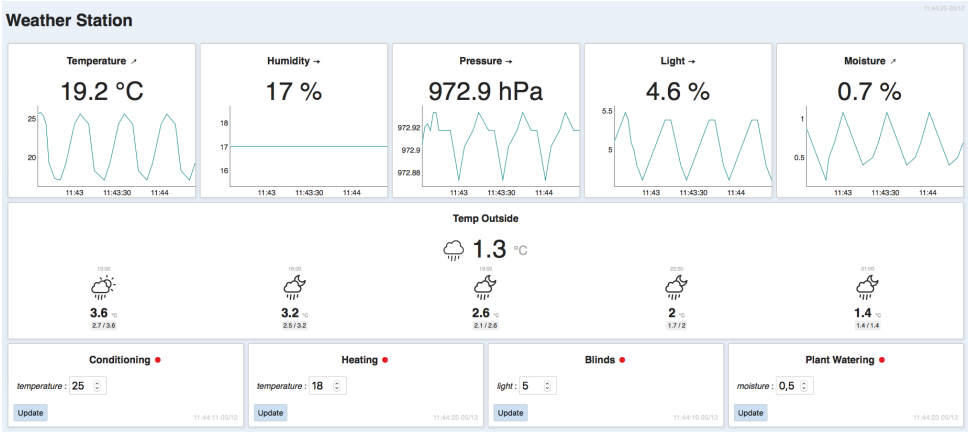
MQTT topic pro zprávy zasílané aktuátory má tvar `home/ws/actuator/<jméno senzoru>` a obsah dané zprávy ve stejném formátu jako u senzorů.

Aktuátory přijímají MQTT zprávy od backendu pro nastavení stavu s MQTT topic ve tvaru: `home/ws/actuator/<jméno senzoru>/state` a obsahem zprávy je pouze nově nastavovaná hodnota v podobě hodnoty 0 nebo 1.

4 Demonstrace systému

Celý systém lze spustit v demonstračním režimu, v kterém je simulována činnost aktuátorů i senzorů. Prerekvizity pro spuštení demostrace systémů pomocí skriptu `run.sh` jsou: Python 2.7 s moduly pip a virtualenv, NodeJS 8.7 a npm 5.4, Angular CLI 1.4.1[3], spuštěný Mosquitto broker na výchozím portu 1883 a dostupné a volné porty 4200 a 8080.

Postup pro demonstraci systému je následující: (1) spustit bash skript `run.sh`, (2) otevřít webový prohlížeč na adrese `http://localhost:4200`.



Obrázek 4: Webové rozhraní v demonstračním režimu.

5 Závěr

Webová aplikace pro vytvořený systém, která obsahuje data naměřená z reálných aktivních senzorů a simulovaných aktuátorů, je dostupná z adresy `https://ws.petrstehlik.cz`. Pro demonstrační účely byl vytvořen skript, který nainstaluje potřebné nástroje pro backend i frontend a spustí frontend, backend a moduly pro simulaci činnosti aktuátorů a senzorů.

[3]Balík Angular CLI vyžaduje globalní instalaci příkazem npm i `@angular/cli@1.4.1 -g`