Upravljanje pametnim ventilom s pomoću tehnologije Z-Wave

Pavelić, Ivan

SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

ZAVRŠNI RAD br. 1549

UPRAVLJANJE PAMETNIM VENTILOM S POMOĆU TEHNOLOGIJE Z-WAVE

Ivan Pavelić

Zagreb, lipanj 2024.

SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

ZAVRŠNI RAD br. 1549

UPRAVLJANJE PAMETNIM VENTILOM S POMOĆU TEHNOLOGIJE Z-WAVE

Ivan Pavelić

Zagreb, lipanj 2024.

SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

Zagreb, 4. ožujka 2024.

ZAVRŠNI ZADATAK br. 1549

Pristupnik:	**Ivan Pavelić (0036543754)**
Studij:	Elektrotehnika i informacijska tehnologija i Računarstvo
Modul:	Računarstvo
Mentor:	prof. dr. sc. Mario Kušek

Zadatak: **Upravljanje pametnim ventilom s pomoću tehnologije Z-Wave**

Opis zadatka:

U današnjem svijetu raste broj uređaja spojenih na Internet. To su uređaji poput senzorskih čvorova u pametnom domu i pametnih kućanskih uređaja kojima je moguće upravljati i/ili očitavati stanje putem Interneta. Posebno područje upravljanja pametnim domom je upravljanje temperaturom. Vaš zadatak je istražiti komunikaciju tehnologijom Z-Wave za potrebe pametnog doma. Praktično je potrebno napraviti aplikaciju za upravljanje pametnim ventilom na radijatoru koji komunicira tehnologijom Z-Wave. Aplikacija se izvršava na računalu na kojem je USB kartica za komunikaciju s pomoću Z-Wavea. Ona prima komande preko REST-a i/ili preko protokola MQTT te ih prosljeđuje pametnom ventilu. Svu potrebnu literaturu i uvjete za rad osigurat će Vam Zavod za telekomunikacije.

Rok za predaju rada: 14. lipnja 2024.

Izražavam duboku zahvalnost prof. dr. sc. Mariu Kušeku za nesebičnu pomoć, podršku i stručne savjete tijekom izrade ovog završnog rada.

Sadržaj

Uvod

U današnjem dobu ubrzanog tehnološkog napretka, broj uređaja povezanih na internet eksponencijalno raste, stvarajući temelje za razvoj pametnih domova. Pametni domovi koriste razne senzorske čvorove i kućanske uređaje koji omogućavaju automatizaciju i daljinsko upravljanje putem interneta [1]. Jedno od ključnih područja primjene pametnih domova je upravljanje temperaturom, što omogućava korisnicima veću udobnost i energetsku učinkovitost.

Pametni ventili na radijatorima predstavljaju značajan korak prema postizanju optimalne kontrole temperature u prostoru. Oni omogućavaju precizno podešavanje protoka topline u radijatorima, čime se postiže željena temperatura u različitim dijelovima doma. Tehnologija koja omogućava komunikaciju između pametnih uređaja u domu igra ključnu ulogu u njihovoj funkcionalnosti i interoperabilnosti.

Jedna od najčešće korištenih tehnologija u području pametnih domova je Z-Wave. Z-Wave je bežična komunikacijska tehnologija posebno dizajnirana za aplikacije pametnog doma, koja omogućava međusobnu komunikaciju različitih uređaja kao što su senzori, aktuatori i kontroleri. Glavna prednost Z-Wave tehnologije je njena pouzdanost, niska potrošnja energije i jednostavna instalacija, što je čini idealnim izborom za pametne domove.

U okviru ovog završnog rada, istražit će se mogućnosti komunikacije tehnologijom Z-Wave za potrebe pametnog doma. Konkretno, cilj je razviti aplikaciju za upravljanje pametnim ventilom na radijatoru koji komunicira tehnologijom Z-Wave. Aplikacija će se izvršavati na računalu opremljenom USB karticom za komunikaciju putem Z-Wavea te će primati komande putem REST API-ja i/ili protokola MQTT, koje će zatim prosljeđivati pametnom ventilu.

Realizacija ovog projekta zahtijeva duboko razumijevanje protokola Z-Wave, razvojne alate i okvire potrebne za implementaciju komunikacije, te sposobnost integracije različitih softverskih i hardverskih komponenti. S obzirom na to da se sve veći broj pametnih uređaja uvodi na tržište, očekuje se da će rezultati ovog rada doprinijeti daljnjem razvoju i primjeni pametnih tehnologija u svakodnevnom životu.

Cilj rada je pružiti sveobuhvatan uvid u tehnologiju Z-Wave, njenu primjenu u upravljanju pametnim ventilima i potencijalne benefite koje ovakvi sustavi donose korisnicima u kontekstu pametnog doma.

1 Pregled korištenih tehnologija

1.1 Pametni ventili u pametnim kućama

Pametni ventili su inovativna tehnologija koja značajno doprinosi upravljanju temperaturom i energetskom učinkovitošću u modernim pametnim kućama (Sl. 1.1). Oni omogućuju automatizirano i precizno upravljanje protokom topline kroz radijatore, čime se optimizira potrošnja energije i poboljšava udobnost korisnika.

Sl. 1.1 Pametni ventil za regulaciju temperature

Pametni ventili se postavljaju na radijatore umjesto klasičnih termostatskih ventila. Njihova osnovna funkcija je regulacija protoka tople vode kroz radijator, čime se kontrolira temperatura u prostoriji [2]. Korištenjem senzora temperature i vlage, pametni ventili mogu automatski prilagoditi protok vode kako bi održali željenu temperaturu. Pametni ventili utječu na sljedeće:

1. energetsku učinkovitost: Pametni ventili omogućuju preciznu kontrolu temperature u svakoj prostoriji pojedinačno. To znači da se toplina isporučuje samo tamo gdje je potrebna, čime se smanjuje nepotrebna potrošnja energije i optimizira ukupna energetska učinkovitost doma;
2. udobnost i prilagodljivost: Korisnici mogu postaviti različite temperature za različite prostorije prema svojim preferencijama i potrebama. Pametni ventili omogućuju stvaranje personaliziranih vremenskih rasporeda grijanja koji se prilagođavaju dnevnoj rutini korisnika;
3. daljinsko upravljanje: Pomoću mobilnih aplikacija ili centralnih sustava za upravljanje pametnim domom, korisnici mogu daljinski upravljati ventilima. Ovo je posebno korisno kada korisnici žele prilagoditi temperaturu prije dolaska kući ili tijekom odsutnosti;
4. integraciju s drugim pametnim sustavima: Pametni ventili se lako integriraju s ostalim pametnim uređajima i sustavima unutar doma, poput pametnih termostata, senzora prozora i vrata, te centralnih sustava za upravljanje energijom. Ova integracija omogućuje holistički pristup upravljanju energijom i poboljšava učinkovitost sustava.

Pametni ventili koriste razne komunikacijske protokole za povezivanje s drugim uređajima i sustavima unutar pametnog doma. Jedan od najčešće korištenih protokola je Z-Wave, koji omogućuje pouzdanu i sigurnu bežičnu komunikaciju s niskom potrošnjom energije. Osim Z-

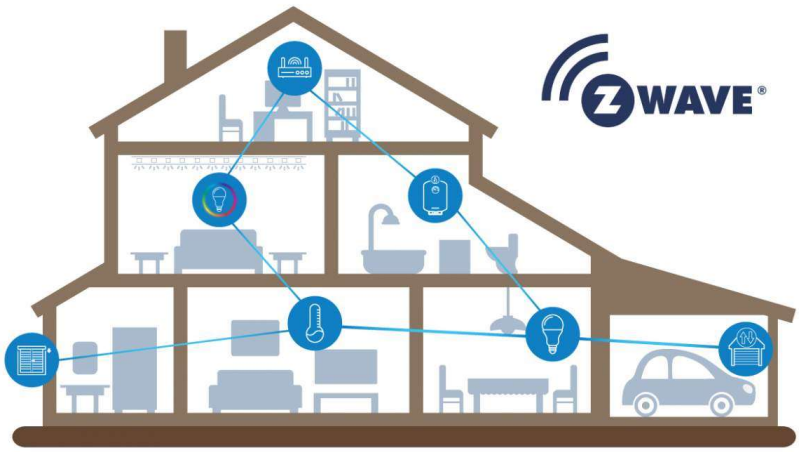
Wavea, koriste se i drugi protokoli poput Zigbeea [3] i Wi-Fi-a, ovisno o specifičnim zahtjevima i konfiguraciji sustava.

Na tržištu je dostupno nekoliko renomiranih proizvođača pametnih ventila, uključujući Tado, Netatmo, i Danfoss. Ovi uređaji nude različite funkcionalnosti i mogućnosti integracije, omogućujući korisnicima da odaberu rješenje koje najbolje odgovara njihovim potrebama i preferencijama.

Pametni ventili su ključna komponenta pametnih kuća koja omogućava učinkovito upravljanje temperaturom i poboljšanje energetske učinkovitosti. Njihova sposobnost precizne kontrole, prilagodljivost korisničkim potrebama, mogućnost daljinskog upravljanja i integracija s drugim pametnim sustavima čine ih nezamjenjivim dijelom modernih rješenja za pametne domove.

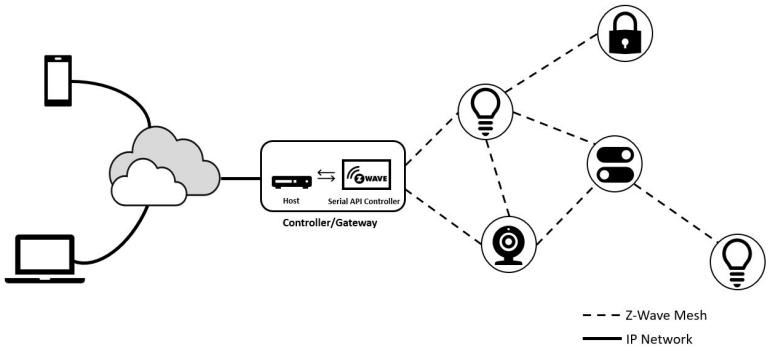
1.2 Protokol Z-Wave

Z-Wave je bežični komunikacijski protokol dizajniran posebno za automatizaciju pametnih domova (Sl. 1.2). Ovaj protokol omogućava međusobnu komunikaciju raznih pametnih uređaja, uključujući senzore, aktuatore, termostate, pametne brave i rasvjetne sustave. Protokol Z-Wave je poznat po svojoj pouzdanosti, sigurnosti i jednostavnoj implementaciji [4].



Sl. 1.2 Z-Wave integracija pametnih domova

Protokol Z-Wave razvila je danska tvrtka Zensys 2001. godine, a kasnije ga je preuzela Sigma Designs. Danas je protokol standardiziran i podržavan od strane Z-Wave Alliance, konzorcija kompanija koje promoviraju i razvijaju tehnologiju Z-Wave. Z-Wave Alliance osigurava interoperabilnost među uređajima različitih proizvođača kroz certificiranje.



Sl. 1.3 Međusobno povezivanje Z-Wave uređaja

Tehničke karakteristike:

1. frekvencijski pojas: Z-Wave koristi niskofrekvencijski pojas (najčešće oko 908.42 MHz u SAD-u i 868.42 MHz u Europi), što omogućuje manju interferenciju s drugim bežičnim uređajima poput Wi-Fi mreža i Bluetootha;
2. mesh mreža: Z-Wave uređaji koriste mesh mrežnu arhitekturu, što znači da svaki uređaj može komunicirati s drugim uređajima u mreži (Sl. 1.3), direktno ili putem drugih uređaja koji prenose signal. Ova struktura povećava domet i pouzdanost mreže;
3. niska potrošnja energije: Z-Wave uređaji su dizajnirani za nisku potrošnju energije, što ih čini idealnima za baterijski pogonjene uređaje poput senzora i pametnih bravica;
4. sigurnost: protokol Z-Wave koristi AES-128 enkripciju za osiguranje komunikacije između uređaja, pružajući visok stupanj sigurnosti i zaštitu od hakiranja;
5. brzina i domet: protokol Z-Wave omogućava brzine prijenosa podataka do 100 kbps s dometom do 100 metara na otvorenom prostoru, što je dovoljno za većinu aplikacija u kućanstvima.

Prednosti protokola Z-Wave:

- interoperabilnost: Jedna od ključnih prednosti Z-Wavea je interoperabilnost među uređajima različitih proizvođača. Svi Z-Wave certificirani uređaji mogu međusobno komunicirati, omogućujući korisnicima fleksibilnost pri odabiru i proširivanju svojih pametnih sustava;
- pouzdanost: Zahvaljujući djelomično povezanoj (engl. mesh) mrežnoj arhitekturi, Z-Wave sustavi su vrlo pouzdani. Ako jedan uređaj ne može direktno komunicirati s kontrolerom, signal će biti proslijeđen putem drugih uređaja, osiguravajući stabilnu komunikaciju;
- jednostavna Instalacija: Z-Wave uređaji su dizajnirani za jednostavnu instalaciju i konfiguraciju. Većina uređaja se može brzo dodati u mrežu putem funkcije “plug and play”, čime se smanjuje potreba za tehničkim znanjem korisnika;
- širok Raspon Uređaja: Z-Wave ekosustav obuhvaća tisuće uređaja, uključujući rasvjetu, termostate, senzore, pametne brave, sigurnosne kamere i mnoge druge, omogućujući korisnicima sveobuhvatno rješenje za pametni dom.

Primjena protkola Z-Wave

Z-Wave se koristi u raznim aplikacijama unutar pametnih domova (Sl. 1.4), uključujući:

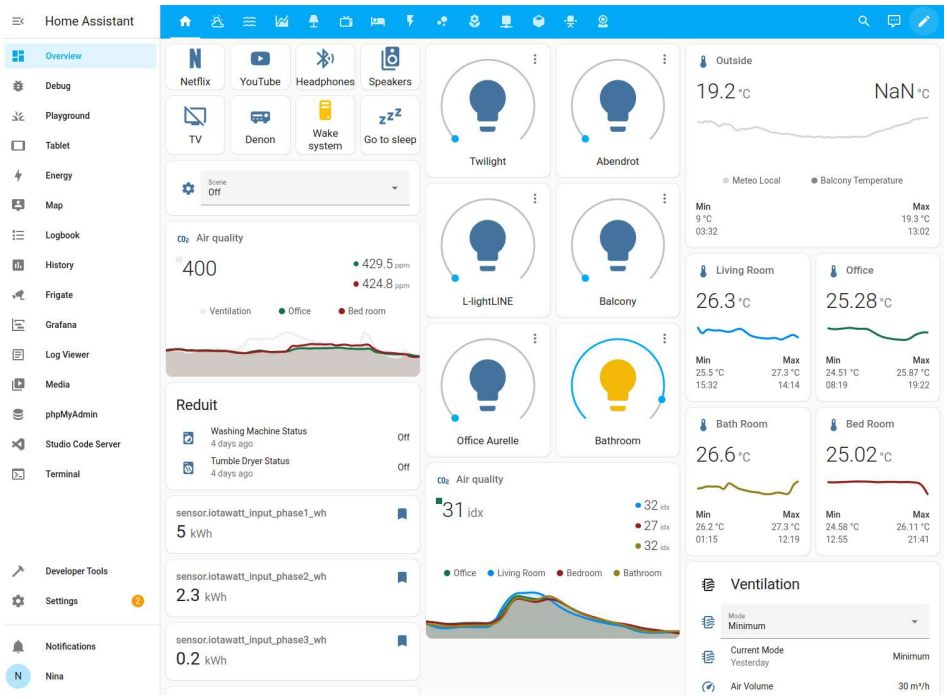
- kontrolu rasvjete: Pametne žarulje, prekidači i dimmeri omogućavaju daljinsko upravljanje i automatizaciju rasvjete;
- sigurnosni sustavi: Senzori pokreta, pametne brave i sigurnosne kamere integrirane sa Z-Waveom poboljšavaju sigurnost doma;
- upravljanje energijom: Pametni termostati, utičnice i uređaji za mjerenje potrošnje energije pomažu u optimizaciji potrošnje energije i smanjenju troškova;
- kontrola pristupa: Pametne brave i sustavi za praćenje ulaza omogućuju sigurnu kontrolu pristupa i nadzor nad kretanjem unutar doma.

Sl. 1.4 Primjena protokola Z-Wave

Protokol Z-Wave predstavlja pouzdan, siguran i fleksibilan način za povezivanje i upravljanje pametnim uređajima unutar doma. Njegova interoperabilnost, niska potrošnja energije i jednostavna instalacija čine ga idealnim izborom za korisnike koji žele unaprijediti svoj dom s pametnim tehnologijama.

1.3 Platforma Home Assistant

Home Assistant je otvorena platforma za automatizaciju pametnih kuća koja omogućava centralizirano upravljanje i integraciju širokog spektra pametnih uređaja i protokola (Sl. 1.5**Pogreška! Izvor reference nije pronađen.**). Razvijena je kao open-source projekt i pruža korisnicima fleksibilnost i prilagodljivost za stvaranje prilagođenih rješenja za upravljanje pametnim domom.

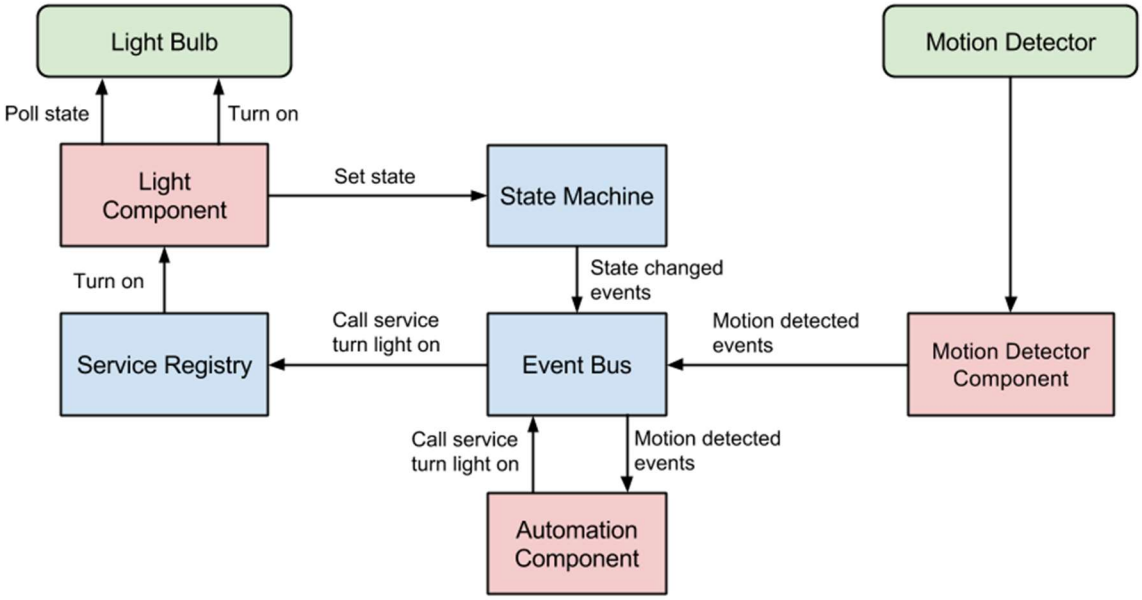


Sl. 1.5 Home Assistant upravljačka ploča

Home Assistant projekt je započeo 2013. godine, a njegov osnivač je Paulus Schoutsen [5]. Cilj projekta bio je stvoriti platformu koja omogućuje jednostavno upravljanje različitim pametnim uređajima bez obzira na proizvođača. Kroz godine, Home Assistant je postao jedan od najpopularnijih alata za automatizaciju pametnih kuća zahvaljujući aktivnoj zajednici i kontinuiranom razvoju.

Tehničke Karakteristike

1. open-source platforma: Home Assistant je potpuno besplatan i open-source, što znači da je izvorni kod dostupan svima. Korisnici mogu prilagođavati i proširivati funkcionalnosti prema svojim potrebama;
2. integracija različitih protokola i uređaja: Home Assistant podržava preko 1700 različitih integracija, uključujući popularne protokole kao što su Z-Wave, Zigbee, MQTT, Wi-Fi, Bluetooth i mnogi drugi. Ova široka podrška omogućava korisnicima da spoje gotovo bilo koji pametni uređaj na svoju mrežu (Sl. 1.6);
3. lokalno pokretanje: Home Assistant se može pokretati lokalno na uređajima kao što su Raspberry Pi, serveri ili virtualne mašine. Lokalno pokretanje povećava privatnost i sigurnost, jer podaci ne moraju prolaziti kroz oblačne servise;
4. automatizacija: Platforma omogućuje složene automatizacije korištenjem jednostavnog sučelja za pravila ili skripti pisanih u YAML formatu. Automatizacije mogu biti pokrenute na temelju različitih uvjeta kao što su vrijeme, senzori ili korisnički unosi;
5. korisničko sučelje: Home Assistant nudi intuitivno web sučelje koje omogućava korisnicima jednostavno upravljanje svim pametnim uređajima. Sučelje je prilagodljivo i može se prilagoditi specifičnim potrebama korisnika.



Sl. 1.6 Arhitektura Home Assistant platforme

Prednosti Home Assistant Platforme

- fleksibilnost i prilagodljivost: Zahvaljujući open-source prirodi, korisnici imaju potpunu kontrolu nad sustavom i mogu ga prilagoditi prema svojim specifičnim potrebama. Moguće je stvarati prilagođena korisnička sučelja, automatizacije i integracije;
- privatnost i sigurnost: Budući da se Home Assistant može pokretati lokalno, korisnici imaju potpunu kontrolu nad svojim podacima. Nema potrebe za slanjem podataka na oblačne servise, što smanjuje rizik od narušavanja privatnosti;
- aktivna zajednica: Home Assistant ima veliku i aktivnu zajednicu koja kontinuirano doprinosi razvoju platforme. Korisnici mogu pronaći pomoć, dijeliti svoja rješenja i doprinositi razvoju kroz forume, društvene mreže i GitHub;
- široka podrška za uređaje: S podrškom za preko 1700 integracija, korisnici mogu spojiti i upravljati gotovo bilo kojim pametnim uređajem, bez obzira na proizvođača. Ovo omogućava stvaranje interoperabilnog i kohezivnog sustava pametnog doma.

Primjena Home Assistant Platforme

Home Assistant može se koristiti u raznim scenarijima unutar pametnog doma, uključujući:

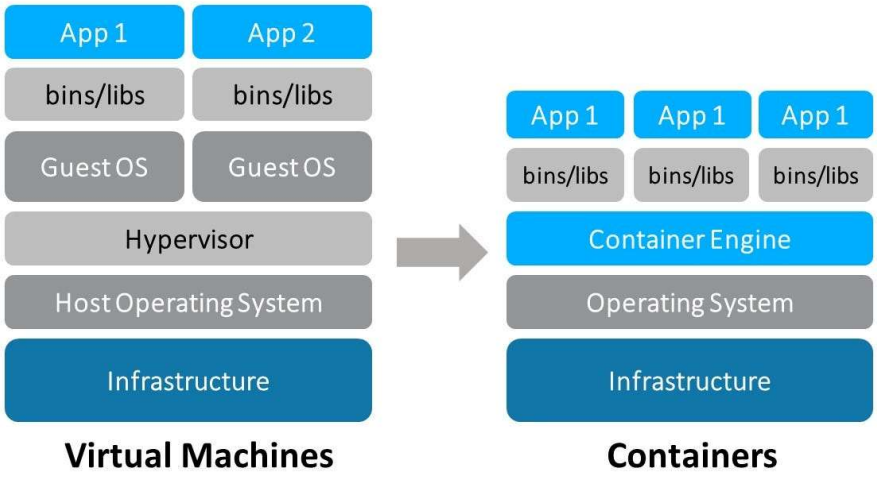
- upravljanje rasvjetom: Automatizacija rasvjete na temelju prisutnosti, vremena dana ili korisničkih komandi;
- kontrola klime: Upravljanje grijanjem, hlađenjem i ventilacijom kako bi se održala optimalna temperatura i kvaliteta zraka;
- sigurnosni sustavi: Integracija sigurnosnih kamera, senzora pokreta, pametnih bravica i alarmnih sustava za poboljšanje sigurnosti doma;
- upravljanje energijom: Praćenje potrošnje energije i upravljanje uređajima za optimizaciju energetske učinkovitosti.

Home Assistant platforma pruža sveobuhvatno rješenje za automatizaciju pametnih kuća, omogućavajući korisnicima fleksibilnost, sigurnost i kontrolu. Njena open-source priroda i

široka podrška za različite uređaje i protokole čine je idealnim izborom za entuzijaste pametnih kuća i one koji žele imati potpunu kontrolu nad svojim pametnim domom.

1.4 Docker kontenjeri

Docker kontejneri predstavljaju tehnologiju koja omogućuje izolaciju i pokretanje aplikacija u laganim virtualiziranim okruženjima (Sl. 1.7). Ova tehnologija revolucionirala je način na koji razvijamo, isporučujemo i pokrećemo softver, pružajući brojne prednosti u pogledu konzistentnosti, skalabilnosti i efikasnosti [6].



Sl. 1.7 Sličnost Docker kontejnera sa virtualnim mašinama

Docker kontejneri su lagane, prenosive, izvršne komponente koje uključuju sve što je potrebno za pokretanje aplikacije, uključujući kod, runtime, sistemske alate, biblioteke i postavke. Docker koristi tehnologiju virtualizacije na razini operativnog sustava, što omogućava dijeljenje jezgre operativnog sustava među više izoliranih kontejnera, čime se smanjuje potreba za resursima u usporedbi s tradicionalnim virtualnim mašinama.

Ključne Karakteristike Docker Kontejnera

1. izolacija: Kontejneri pružaju izolirano okruženje za aplikacije, što znači da svaka aplikacija radi neovisno od drugih. To omogućava sigurnost i stabilnost jer kvar u jednom kontejneru ne utječe na druge kontejnere;
2. prenosivost: Docker kontejneri mogu se pokretati na bilo kojem sustavu koji podržava Docker, uključujući lokalne strojeve, podatkovne centre i oblak. Ovo omogućava dosljedno ponašanje aplikacija bez obzira na okruženje;
3. efikasnost: Za razliku od virtualnih mašina koje uključuju cijeli operativni sustav, kontejneri dijele jezgru host operativnog sustava. To smanjuje troškove resursa kao što su memorija i CPU, omogućavajući pokretanje više kontejnera na istom hardveru;
4. brzo pokretanje i skaliranje: Kontejneri se brzo pokreću i zaustavljaju, što omogućava agilnost u razvoju i implementaciji aplikacija. Skaliranje aplikacija postaje jednostavno jer se novi kontejneri mogu brzo dodati ili ukloniti prema potrebi.

Arhitektura i Komponente Docker Ekosustava

Docker Engine: Temelj Docker ekosustava, Docker Engine, je aplikacija koja pokreće i upravlja kontejnerima. Sastoji se od Docker demona, REST API-ja i CLI (Command Line Interface).

Docker Images: Slike (images) su nepromjenjivi predlošci koji se koriste za stvaranje kontejnera. Svaka slika sadrži sve što je potrebno za pokretanje aplikacije, uključujući kod, runtime i biblioteke. Docker slike se mogu preuzeti iz Docker Hub-a ili drugih repozitorija.

Docker Containers: Kontejneri su pokrenute instance Docker slika. Oni su izolirani procesi koji dijele jezgru operativnog sustava s hostom i drugim kontejnerima.

Docker Compose: Alat koji omogućava definiranje i pokretanje aplikacija koje koriste više kontejnera. Konfiguracija se definira u YAML datoteci, što olakšava orkestraciju složenih aplikacija.

Docker Swarm i Kubernetes: Rješenja za orkestraciju kontejnera koja omogućuju upravljanje velikim brojem kontejnera u grozdu računala. Docker Swarm je integrirano rješenje, dok je Kubernetes otvoreni standard za orkestraciju kontejnera.

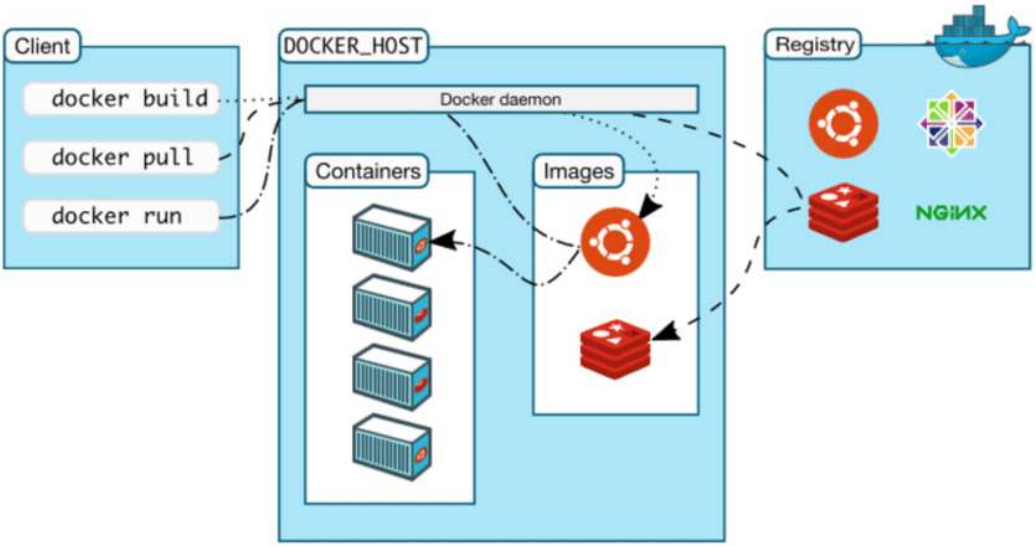
Prednosti Docker Kontejnera

- konzistentnost: Razvojni, testni i produkcijski okoliši mogu biti identični, što smanjuje probleme povezane s razlikama u konfiguraciji i okruženju;
- brza implementacija: Docker omogućava brzu implementaciju aplikacija jer se kontejneri mogu brzo izgraditi, pokrenuti i replicirati;
- skalabilnost: Docker olakšava horizontalno skaliranje aplikacija dodavanjem ili uklanjanjem kontejnera prema potrebi, čime se osigurava optimalno korištenje resursa;
- sigurnost: Izolacija između kontejnera pruža dodatnu sigurnost jer aplikacije rade u odvojenim okruženjima, što smanjuje rizik od neovlaštenog pristupa i kompromitacije sustava.

Primjena Docker Kontejnera

Docker kontejneri se koriste u različitim scenarijima, uključujući:

- razvoj i testiranje: Programeri koriste Docker za stvaranje konzistentnih razvojnih i testnih okruženja koja su identična produkcijskim okruženjima;
- mikroservisne arhitekture: Docker omogućava jednostavno razdvajanje aplikacija u mikroservise, gdje svaki servis radi u vlastitom kontejneru;
- CI/CD (eng. Continuous Integration/Continuous Delivery): Docker se integrira s alatima za kontinuiranu integraciju i isporuku (CI/CD), omogućavajući automatsko testiranje, izgradnju i implementaciju aplikacija;
- računrstvo na rubu mreže (eng. Edge Computing): Docker kontejneri omogućavaju implementaciju aplikacija na rubnim uređajima gdje su resursi ograničeni.



Sl. 1.8 Način komunikacije unutar Docker kontejnera

Docker kontejneri predstavljaju moćnu tehnologiju za razvoj, isporuku i pokretanje aplikacija u izoliranim i prenosivim okruženjima (Sl. 1.8). Njihova sposobnost pružanja konzistentnosti, efikasnosti i skalabilnosti čini ih ključnim alatom u modernom softverskom inženjerstvu. Sa širokim rasponom primjena i podrškom za različite platforme, Docker kontejneri nastavljaju oblikovati način na koji razvijamo i implementiramo softver u današnjem brzom tehnološkom okruženju.

1.5 Samostalno programiranje pametnih uređaja

Samostalno programiranje pametnih uređaja omogućava korisnicima prilagodbu funkcionalnosti i automatizaciju prema specifičnim potrebama i preferencijama. Ovaj pristup zahtijeva osnovno poznavanje programskih jezika, razvojnih alata i protokola, ali pruža visoku razinu fleksibilnosti i kontrolu nad pametnim sustavima.

Samostalno programiranje pametnih uređaja uključuje pisanje vlastitog koda ili korištenje skriptnih jezika za konfiguraciju, kontrolu i automatizaciju pametnih uređaja. To može uključivati kreiranje prilagođenih automatizacija, integraciju s drugim uređajima i uslugama, te proširenje postojećih funkcionalnosti uređaja.

Ključne Komponente i Tehnologije

1. razvojni alati i okviri: Razvoj pametnih uređaja često koristi specifične razvojne alate i okvire poput Arduino IDE-a, PlatformIO-a ili Visual Studio Code-a s odgovarajućim proširenjima. Ovi alati olakšavaju pisanje, testiranje i implementaciju koda na uređaje;
2. programski jezici: Najčešće korišteni programski jezici za programiranje pametnih uređaja uključuju C/C++, Python, JavaScript (Node.js) i Lua. Izbor jezika ovisi o platformi i specifičnim zahtjevima projekta;
3. komunikacijski protokoli: Pametni uređaji često komuniciraju putem različitih protokola kao što su MQTT [7], HTTP/HTTPS, WebSockets, Zigbee, Z-Wave i Bluetooth. Razumijevanje ovih protokola ključno je za integraciju i međusobnu komunikaciju uređaja;

4. razvojne ploče i moduli: Razvojne ploče kao što su Arduino [8], Raspberry Pi [9], ESP8266/ESP32 i druge često se koriste za razvoj i programiranje pametnih uređaja. Ove ploče pružaju potrebne ulazno-izlazne (I/O) pinove, komunikacijske module i razvojna okruženja.

Prednosti Samostalnog Programiranja

fleksibilnost i prilagodljivost: Korisnici mogu razviti rješenja koja su potpuno prilagođena njihovim specifičnim potrebama. Mogu se kreirati prilagođene automatizacije, dodavati nove funkcionalnosti i integrirati uređaje s različitim platformama;

kontrola: Samostalno programiranje pruža potpunu kontrolu nad uređajem i njegovim ponašanjem. Korisnici mogu optimizirati performanse, sigurnost i pouzdanost sustava prema vlastitim standardima;

učenje i razvoj vještina: Proces programiranja pametnih uređaja omogućava korisnicima stjecanje novih vještina i znanja u područjima programiranja, elektronike i mrežnih komunikacija.

Primjeri Primjene

- pametna rasvjeta: Programiranje pametnih svjetiljki ili prekidača za prilagodbu osvjetljenja prema vremenskom rasporedu, prisutnosti ili senzorskim podacima;
- kućna automatizacija: Kreiranje složenih scenarija automatizacije koji uključuju više uređaja, kao što su senzori pokreta, termostati, sigurnosne kamere i pametni ventili;
- upravljanje energijom: Razvoj sustava za praćenje i optimizaciju potrošnje energije, uključujući integraciju s pametnim brojilima i sustavima za upravljanje energijom;
- sigurnosni sustavi: Programiranje prilagođenih sigurnosnih rješenja koja uključuju senzore, alarme i kamere, te njihovu integraciju s obavijesnim sustavima i daljinskim pristupom.

Proces Samostalnog Programiranja

1. odabir hardvera: Izbor odgovarajućih razvojnih ploča, senzora i aktuatora prema potrebama projekta;
2. postavljanje razvojnog okruženja: Instalacija potrebnih alata i okvira za razvoj, kao što su Arduino IDE, Python okruženja ili Node.js;
3. pisanje koda: Razvoj softvera za kontrolu i automatizaciju uređaja koristeći odgovarajuće programske jezike i biblioteke;
4. testiranje i debugging: Provođenje opsežnih testiranja i ispravljanje pogrešaka kako bi se osigurala pouzdanost i funkcionalnost sustava;
5. implementacija: Prenošenje koda na uređaje i njihova integracija u postojeći sustav pametnog doma;
6. održavanje i ažuriranje: Redovito održavanje i ažuriranje softvera kako bi se osigurala optimalna funkcionalnost i sigurnost.

Izazovi i Prepreke

- tehnička složenost: Programiranje pametnih uređaja može biti tehnički izazovno, posebno za korisnike bez prethodnog iskustva u programiranju i elektronici;

- kompatibilnost: Osiguravanje kompatibilnosti među različitim uređajima i protokolima može biti zahtjevno, posebno u heterogenim sustavima;
- sigurnost: Zaštita pametnih uređaja od sigurnosnih prijetnji i osiguranje privatnosti korisničkih podataka zahtijeva dodatne mjere i pažnju.

Samostalno programiranje pametnih uređaja nudi značajne prednosti u smislu fleksibilnosti, kontrole i prilagodljivosti. Iako može biti tehnički zahtjevno, pruža mogućnost razvoja inovativnih i personaliziranih rješenja koja najbolje odgovaraju specifičnim potrebama korisnika. Kroz razumijevanje osnovnih komponenti, tehnologija i procesa, korisnici mogu uspješno implementirati i održavati složene sustave pametnih kuća.

2 Metodologija

Metodologija korištenja Danfoss Living Connect Z pametnog ventila obuhvaća nekoliko ključnih koraka, uključujući odabir i pripremu opreme, instalaciju, konfiguraciju, programiranje i testiranje.

1. Oprema
    - Danfoss Living Connect Z pametni ventil
    - Z-Wave USB stick za računalo
    - Računalo s instaliranim operativnim sustavom (npr. Windows, macOS, Linux)
    - Home Assistant platforma ili druga Z-Wave kompatibilna platforma
    - Internet veza za preuzimanje softvera i ažuriranja
2. Instalacija softvera
    - Instalacija Z-Wave upravljačkog softvera na računalo (npr. Z-Wave PC Controller ili Home Assistant).
    - Koristeći Home Assistant, instalirati potrebne dodatke za Z-Wave integraciju.
3. Instalacija i konfiguracija pametnog ventila
    - Zamjena postojećeg termostatskog ventila na radijatoru s Danfoss Living Connect Z ventilom.
    - Povezivanje s Z-Wave Mrežom
    - Priključivanje Z-Wave USB sticka u računalo.
    - Otvoriti Z-Wave upravljački softver ili Home Assistant platformu.
    - Uključiti Z-Wave mrežu i dodati novi uređaj (Danfoss Living Connect Z ventil). Slijeditu upute za uparivanje uređaja.
    - Provjeriti je li ventil uspješno dodan u Z-Wave mrežu i može li komunicirati s upravljačkim softverom.
4. Programiranje i automatizacija
    - Konfigurirati osnovne postavke ventila kroz Z-Wave softver ili Home Assistant.
    - Postaviti naziv uređaja i dodati ga u odgovarajuće sobe ili zone unutar aplikacije za lakše upravljanje.
    - Kreirati automatizacije pomoću Home Assistant platforme ili drugog Z-Wave kompatibilnog softvera.

Korištenje Danfoss Living Connect Z pametnog ventila u kombinaciji sa Z-Wave tehnologijom i Home Assistant platformom omogućava precizno i automatizirano upravljanje grijanjem u pametnim kućama. Slijedeći ovu metodologiju, možemo osigurati pravilnu instalaciju, konfiguraciju i optimizaciju ventila, što će rezultirati poboljšanom energetskom učinkovitošću i udobnošću za korisnike.

3 Implementacija

3.1 Konfiguracija Home Assistanta i Z-Wave integracija

Postavili smo i konfigurirali Home Assistant na Windows operacijskom sustavu koristeći VirtualBox.

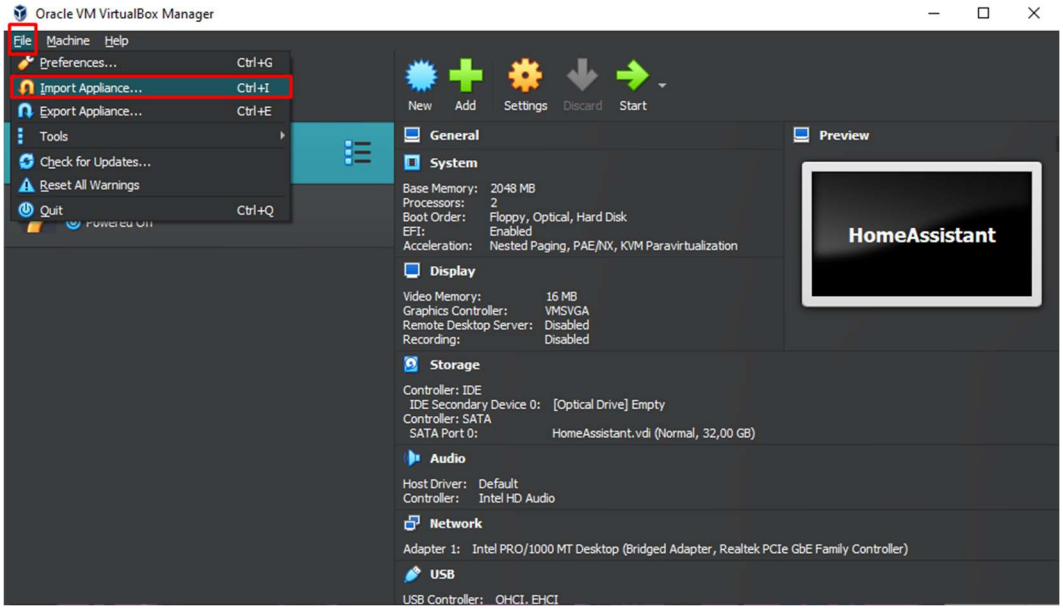
Priprema

Prvo je potrebno preuzeti i instalirati VirtualBox. To možemo učiniti tako da posjetitimo VirtualBox službenu stranicu i preuzmemo instalacijski paket za Windows. Nakon što preuzmemo instalacijski paket, instaliravamo VirtualBox prateći upute na ekranu.

Nakon toga, preuzimamo Home Assistant OVA datoteku. Posjetimo Home Assistant službenu stranicu i pronađemo dio koji se odnosi na instalaciju pomoću VirtualBoxa. Kada pronađemo odgovarajući dio, preuzmimamo Home Assistant OVA datoteku.

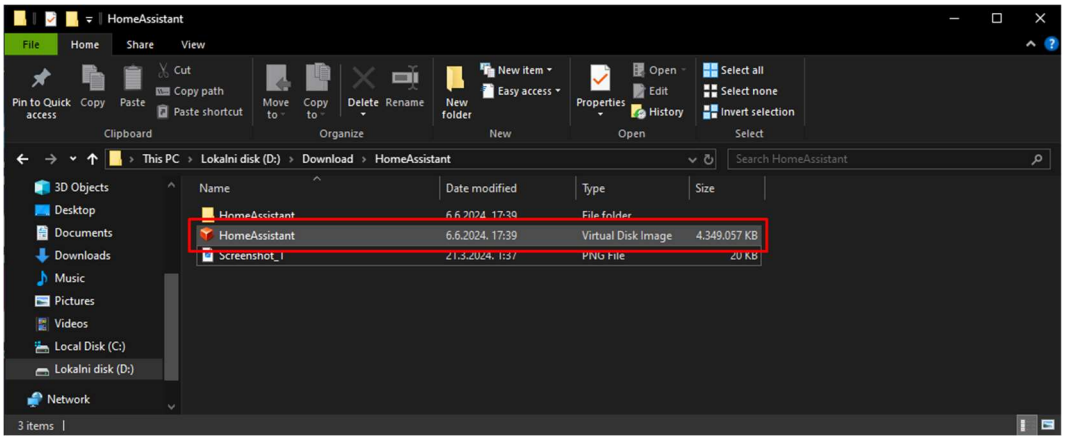
Instalacija Home Assistanta u VirtualBoxu

Prvo pokrećemo VirtualBox tako da otvorimo VirtualBox aplikaciju. Zatim uvozimo Home Assistant OVA datoteku. Kliknemo na "File" u gornjem izborniku, zatim odaberemo "Import Appliance" (Sl. 3.1).



Sl. 3.1 Uvoz Home Assistant .OVA datoteke

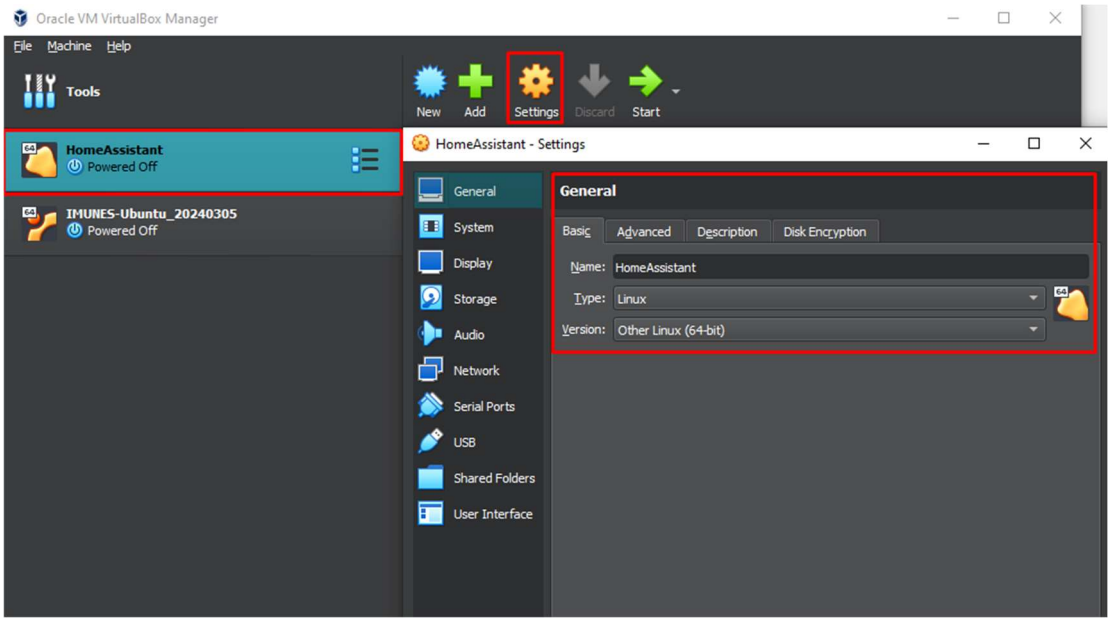
U dijaloškom prozoru koji se otvori, kliknemo na Choose a file i odaberemo prethodno preuzetu Home Assistant OVA datoteku (Sl. 3.2).



Sl. 3.2 Pronalazak .OVA datoteke na računalu

Kliknemo Next, a zatim Import da bismo započeli proces uvoza.

Nakon što je uvoz dovršen, nova virtualna mašina će biti prikazana u popisu VirtualBoxa. Odaberemo tu virtualnu mašinu i kliknemo na "Settings" (Sl. 3.3).



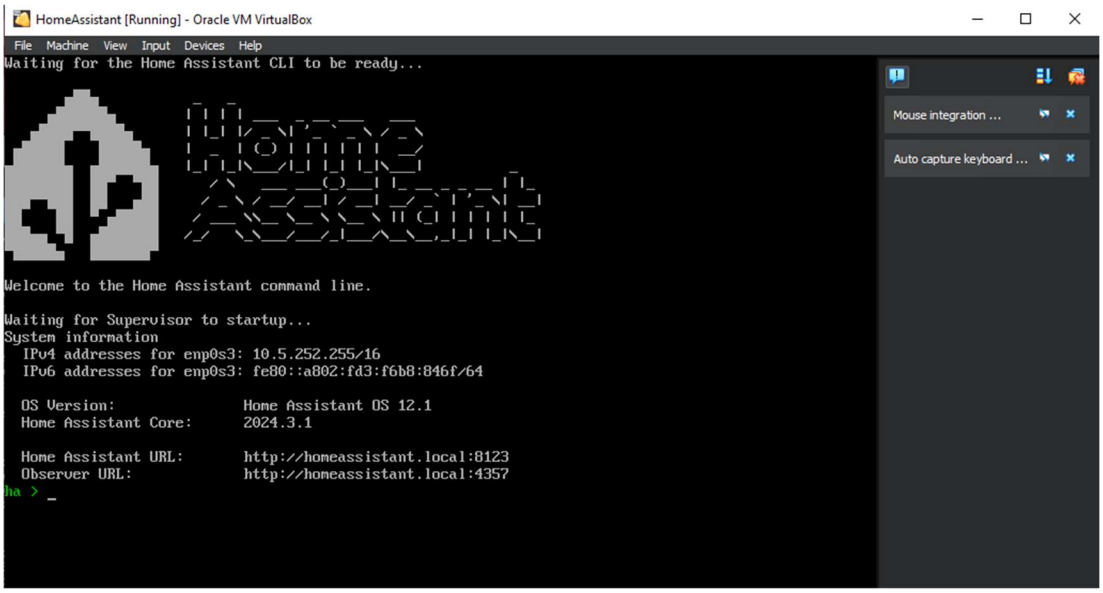
Sl. 3.3 Postavke Virtualne mašine

General: Provjeravamo naziv i OS tip (Linux 64-bit).

System: Provjeravamo količinu dodijeljene RAM memorije (preporučeno barem 2 GB).

Network: Osiguravamo da je mrežna kartica postavljena na Bridged Adapter kako bi VM mogla pristupiti mreži.

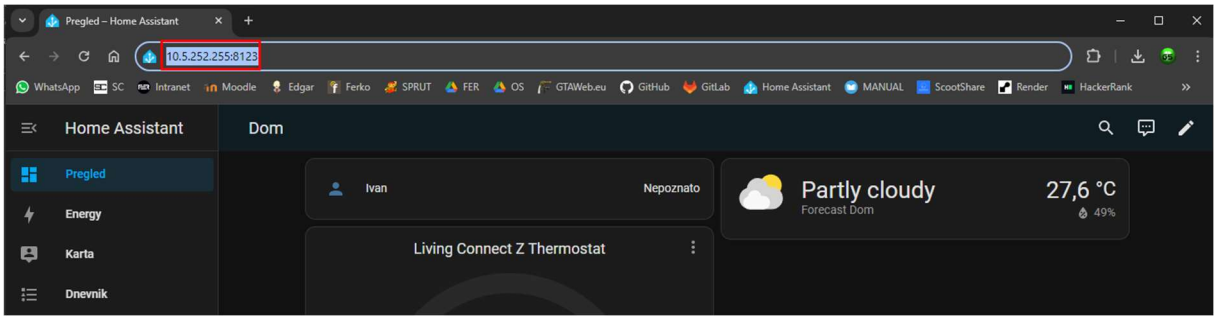
Pokrećemo Home Assistant virtualnu mašinu klikom na "Start". Pričekamo da se Home Assistant OS pokrene, što može potrajati nekoliko minuta (Sl. 3.4).



Sl. 3.4 Pokretanje Home Assistant u virtualnoj mašini

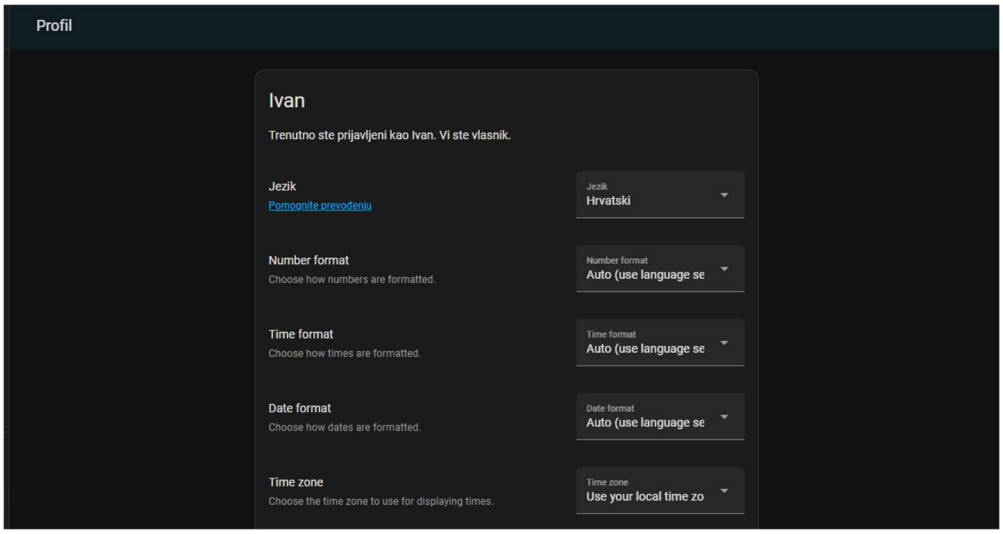
Konfiguracija Home Assistanta

Nakon što se Home Assistant pokrene, zabilježimo IP adresu (10.5.252.255) prikazanu na ekranu VirtualBox konzole. Zatim otvorimo web-preglednik na Windows računalu i upišemo IP adresu Home Assistanta u adresnu traku, koristeći port 8123 (http://10.5.252.255:8123/) (Sl. 3.5).



Sl. 3.5 Pristupanje Home Assistant kontrolnoj ploči

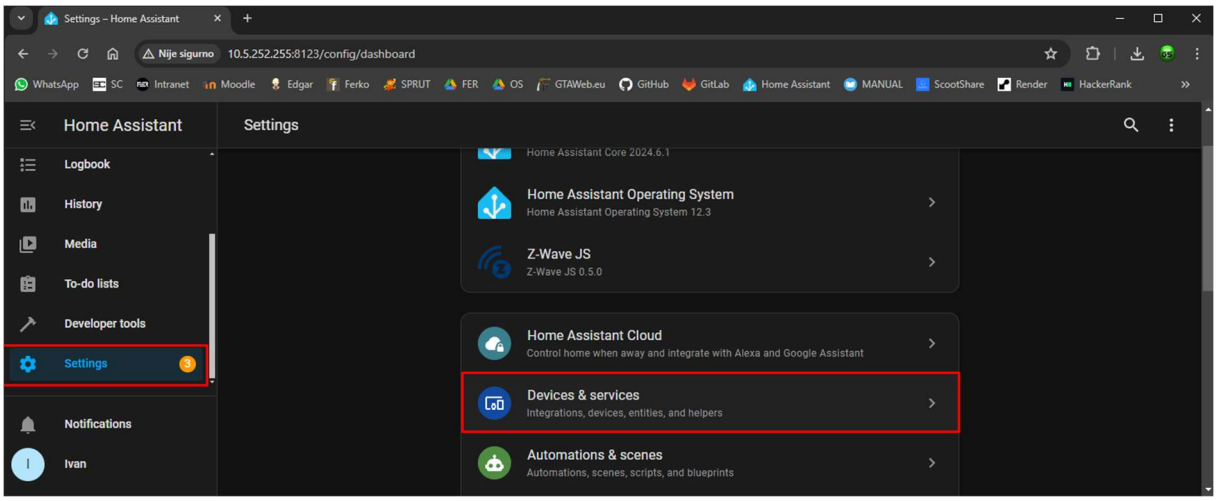
Prilikom prvog pristupa Home Assistantu, slijedimo upute na ekranu za postavljanje korisničkog računa (Sl. 3.6). Postavljamo naziv doma, lokaciju, vremensku zonu i druge osnovne postavke.



Sl. 3.6 Registracija u Home Assistant platformu

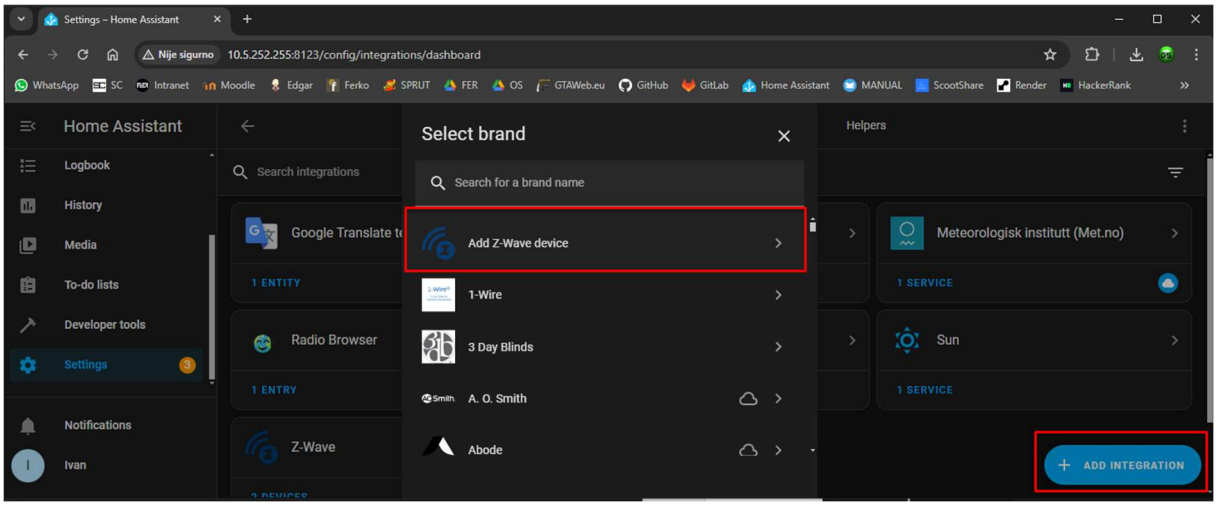
Home Assistant će automatski otkriti neke uređaje na mreži koji se mogu odmah integrirati.

Za dodavanje Z-Wave integracije, pozicioniramo se u "Settings" -> "Devices and Services" (Sl. 3.7).



Sl. 3.7 Dodavanje Z-Wave integracije

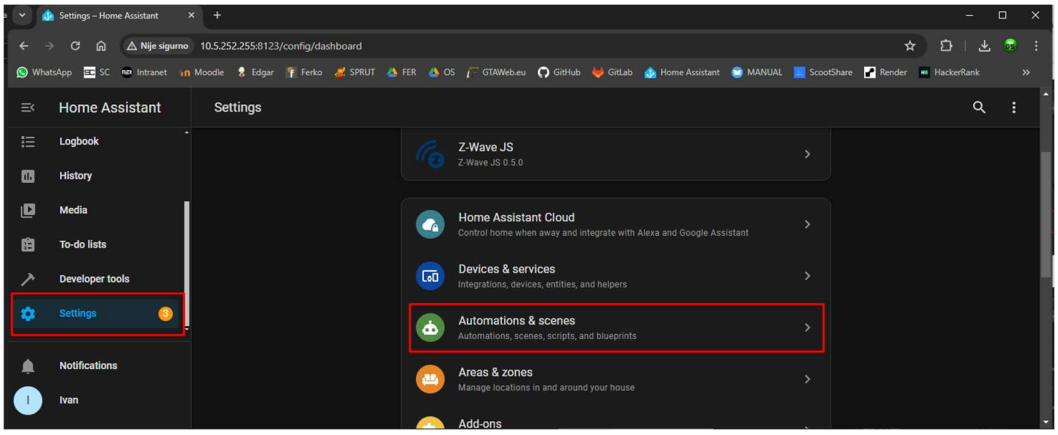
Kliknemo na + Add Integration i potražimo Z-Wave (Sl. 3.8).



Sl. 3.8 Dodavanje Z-Wave uređaja

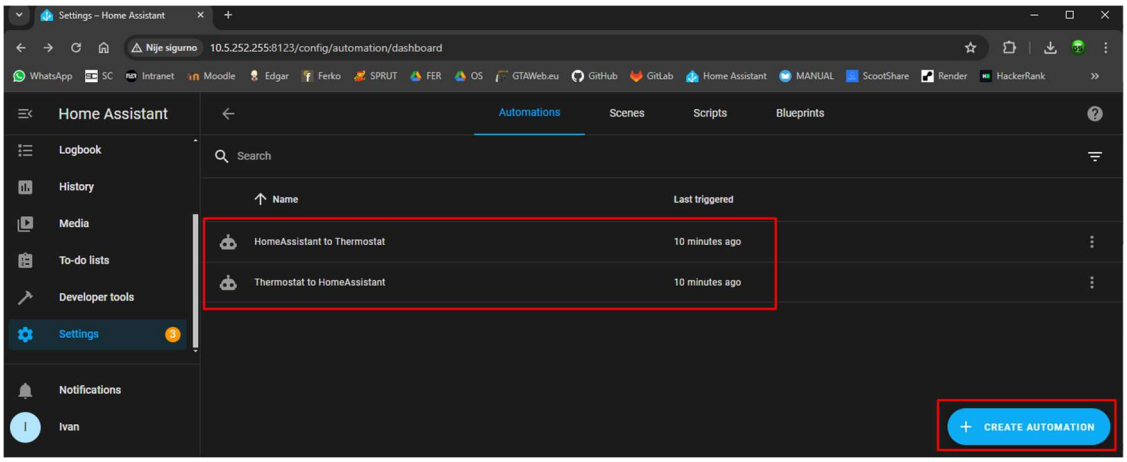
Slijedimo upute za dodavanje Z-Wave USB sticka i postavljanje Z-Wave mreže.

Za kreiranje i konfiguraciju automatizacija, u izborniku odaberemo "Settings" -> "Automations and Scenes" (Sl. 3.9).



Sl. 3.9 Postavljanje automatizacije u HA

Kliknemo na + Add Automation i koristimo jednostavno sučelje za kreiranje automatizacija (Sl. 3.10).



Sl. 3.10 Pregled dodanih automatizacija

Postavljamo uvjete, radnje i okidače za automatizaciju pametnih uređaja.

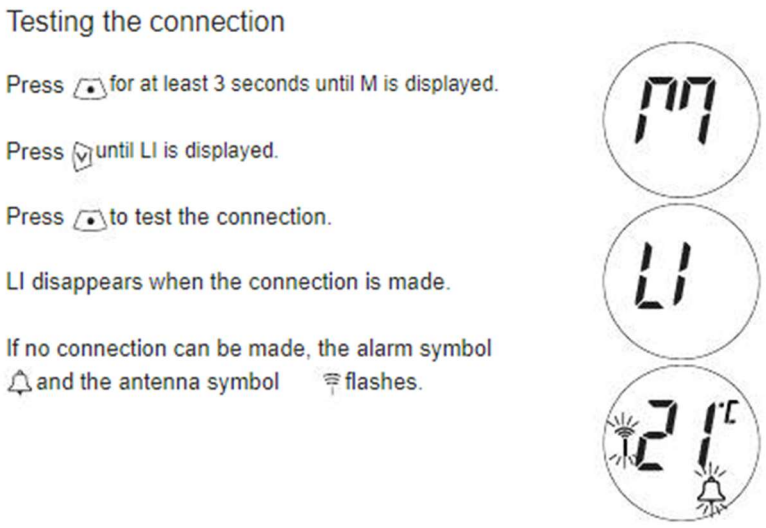
Prateći ove korake, uspješno smo postavili i konfigurirali Home Assistant na našem Windows računalu koristeći VirtualBox.

3.2 Konfiguracija pametnog ventila u Home Assistantu

Konfigurirali smo Danfoss Living Connect Z pametni ventil u Home Assistant platformi na sljedeći način.

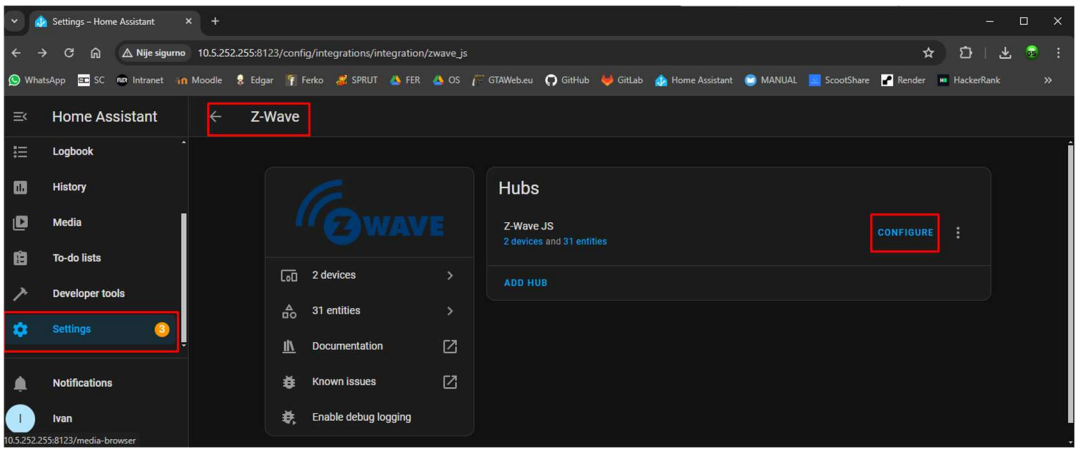
Postavljamo Z-Wave integraciju slijedeći upute na ekranu za konfiguraciju Z-Wave integracije. Ako se zatraži, unosimo put do Z-Wave USB sticka (npr. /dev/ttyS0 za Linux ili odgovarajući COM port za Windows).

Za uparivanje Danfoss Living Connect Z pametnog ventila, prebacujemo ventil u način uparivanja prema uputama proizvođača (držanjem određenog gumba na uređaju) (Sl. 3.11).



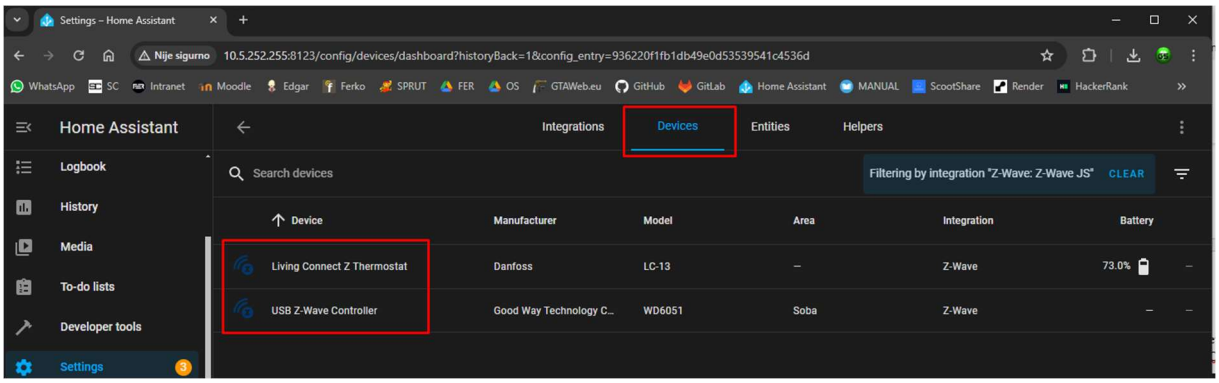
Sl. 3.11 Upute za uparivanje Danfoss Living Connect Z pametnog ventila

U Home Assistantu, pozicinoiramo se u "Settings" -> "Devices & Services". Pronađemo Z-Wave integraciju i kliknemo na "Configure" (Sl. 3.12).



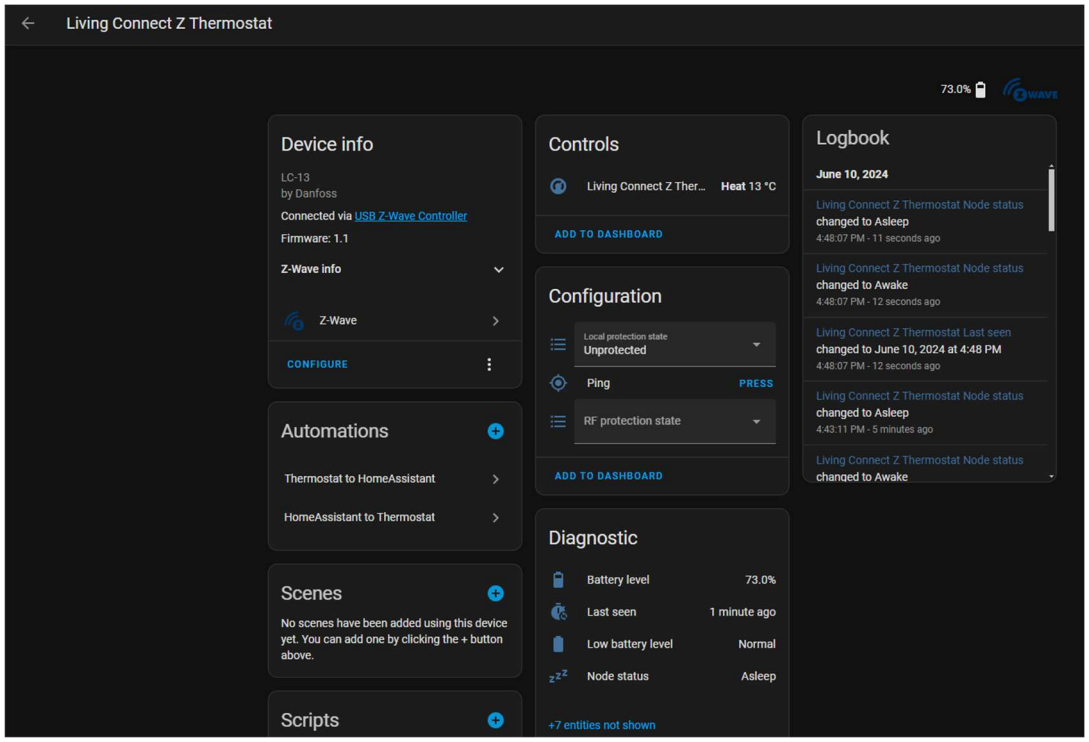
Sl. 3.12 Konfiguracija dodanog uređaja kroz HA

Odaberemo "Add Device" kako bismo dodali novi uređaj u Z-Wave mrežu. Slijedimo upute za uparivanje Danfoss ventila. Nakon uspješnog uparivanja, uređaj bi trebao biti vidljiv u Home Assistantu (Sl. 3.13).



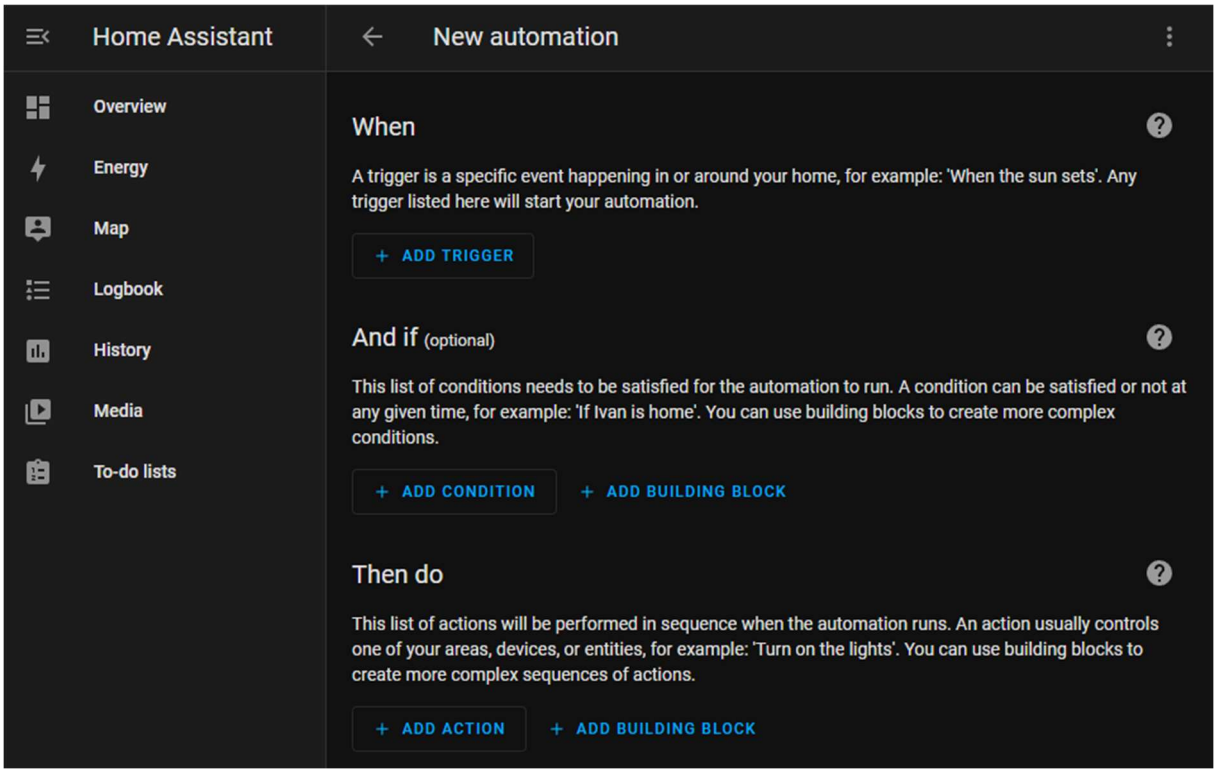
Sl. 3.13 Dodavanje uparenog uređaja u upravljačku ploču HA

Za konfiguraciju pametnog ventila, pozicioniramo se u "Settings" -> "Devices & Services". Pronađemo Danfoss Living Connect Z ventil u popisu uređaja i kliknemo na uređaj za prikaz detalja. Postavimo osnovne parametre kao što su naziv uređaja, soba u kojoj se nalazi i osnovne postavke poput željene temperature (Sl. 3.14).



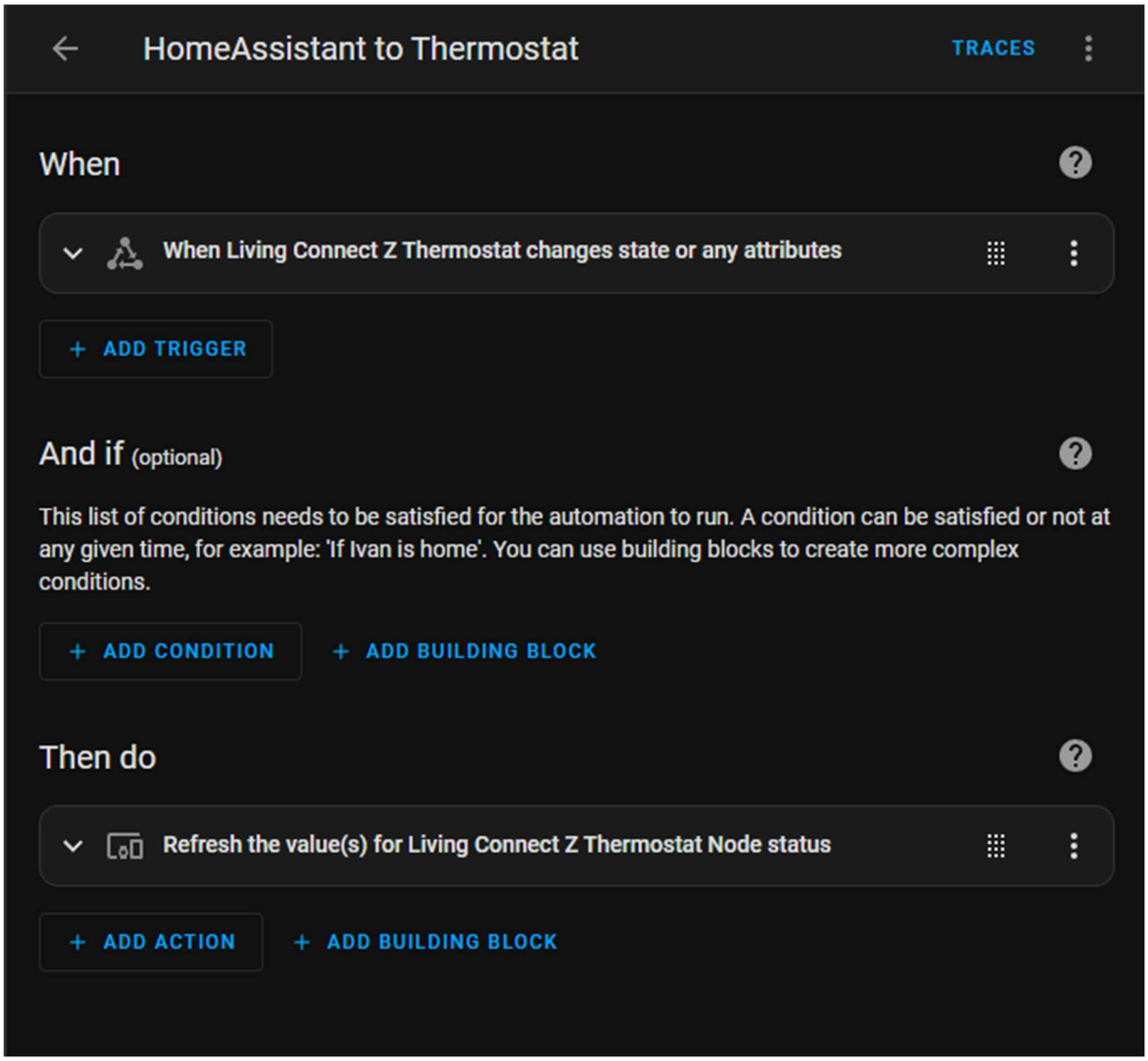
Sl. 3.14 Konfiguracija osnovnih parametara dodanog uređaja

Uređaj ima opcije za kontrolu temperature, način rada (npr. grijanje, hlađenje, isključeno) i ostale relevantne postavke. Pozicioniramo se u "Settings" -> "Automations". Kliknemo na "+ Add Automation" kako bismo kreirali novu automatizaciju. Odaberemo "Create new automation" (Sl. 3.15).



Sl. 3.15 Postavljanje automatizacije za pametni ventil

Dodajemo Trigger (Okidač): Kada Living Connect Z termostat promijeni stanje ili bilo koji atribut. Dodajemo Action (Akcija): Osvježiti vrijednosti za status čvora termostata Living Connect Z (Sl. 3.16).



Sl. 3.16 Definiranje uvjeta i akcija za automatizaciju

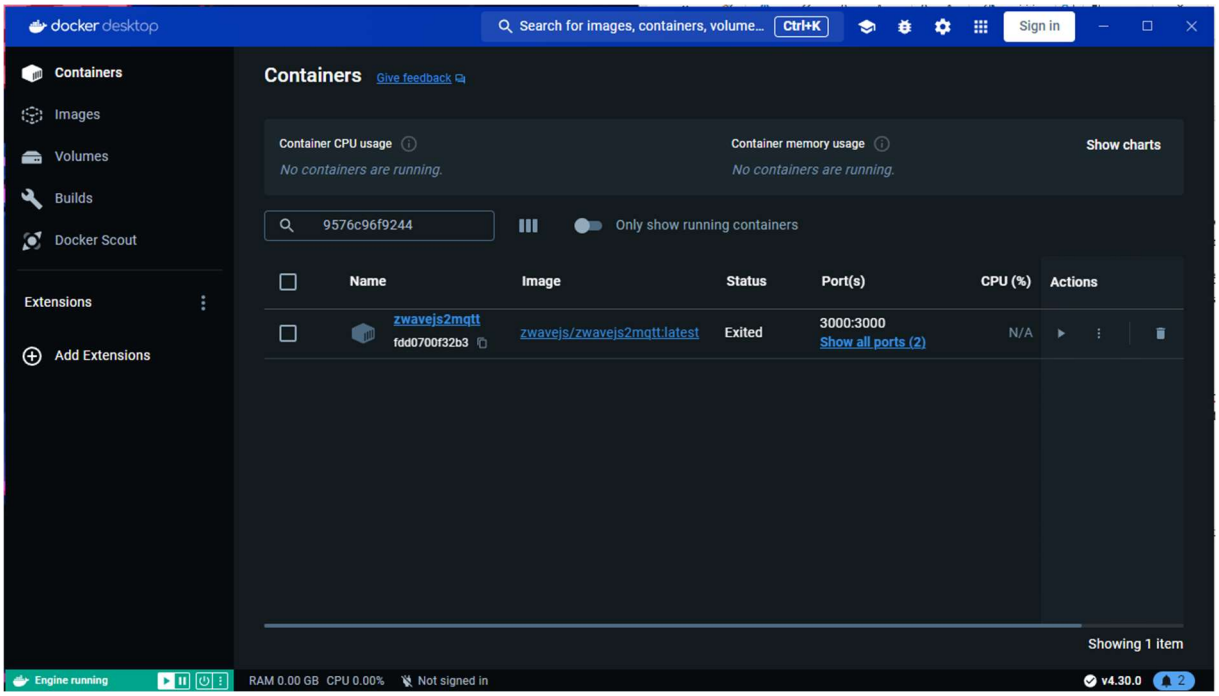
Nakon definiranja uvjeta i akcija, kliknemo na Save kako bi spremili automatizaciju.

Prateći ove korake, uparili smo uređaj s Home Assistantom, konfigurirali osnovne postavke i kreirali prilagođene automatizacije.

3.3 Konfiguracija Docker kontenjera i zwavejs2mqtt konfiguracija

Instaliracija Docker na Windows operacijski sustav i pokretanje zwavejs2mqtt Docker kontenjera uključuje preuzimanje i instalaciju Docker-a, konfiguraciju Docker-a na Windows sustavu te pokretanje i konfiguraciju zwavejs2mqtt [10] Docker container-a.

Za instalaciju Docker-a za Windows, posjećujemo Docker Desktop za Windows službenu stranicu. Kliknemo na "Download Docker Desktop for Windows" i preuzimamo instalacijski paket. Pokrećemo preuzeti instalacijski paket i slijedimo upute na ekranu za instalaciju Docker Desktop-a. Tijekom instalacije, Docker će zatražiti da se omogući WSL 2 (Windows Subsystem for Linux 2). Ako WSL 2 nije već instaliran, Docker instalacijski program će ga omogućiti. Nakon instalacije, pokrećemo Docker Desktop aplikaciju (Sl. 3.17).



Sl. 3.17 Pokretanje Docker aplikacije

Za pokretanje zwavejs2mqtt Docker container-a, u Command Promptu ili PowerShellu pokrećemo sljedeću naredbu kako bismo preuzeli najnoviju Docker sliku za zwavejs2mqtt (Sl. 3.18):

- docker pull zwavejs/zwavejs2mqtt:latest

```
PS C:\Users\G5\Desktop\ZavršniRad> docker pull zwavejs/zwavejs2mqtt:latest
latest: Pulling from zwavejs/zwavejs2mqtt
Digest: sha256:634a51ba806f49e1453405342685b28ce69c34379305296f717978ca75d8fa7
Status: Image is up to date for zwavejs/zwavejs2mqtt:latest
docker.io/zwavejs/zwavejs2mqtt:latest
```

Sl. 3.18 Preuzimanje i pokretanje najnovije Docker slike

Nakon što je slika preuzeta, pokrećemo sljedeću naredbu za pokretanje zwavejs2mqtt Docker container-a:

- docker run -d \

  --name zwavejs2mqtt \

  --restart unless-stopped \

  -p 8091:8091 \

  -p 3000:3000 \

  -v ~/zwavejs2mqtt/store:/usr/src/app/store \

  -v /run/udev:/run/udev:ro \

  --device=/dev/ttyS0 \

  zwavejs/zwavejs2mqtt:latest

Ovdje su objašnjeni parametri korišteni u naredbi:

-d: Pokreće container u pozadini.

--name zwavejs2mqtt: Naziv Docker container-a.

--restart unless-stopped: Automatski ponovno pokreće container osim ako nije ručno zaustavljen.

-p 8091:8091: Preslikava port 8091 na domaćinu (host) na vratima (port) 8091 u kontejneru (za zwavejs2mqtt upravljačko sučelje).

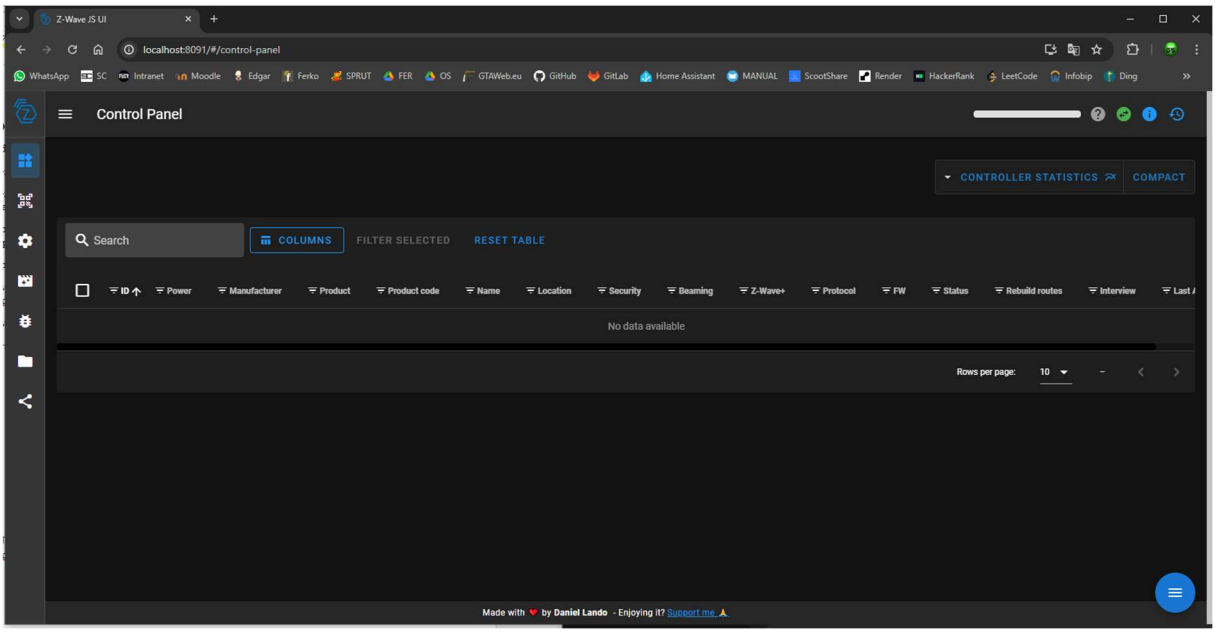
-p 3000:3000: Mapira port 3000 na hostu na port 3000 u containeru (za Z-Wave JS).

-v ~/zwavejs2mqtt/store:/usr/src/app/store: Montira direktorij na domaćinu za pohranu postavki kontenjera.

-v /run/udev:/run/udev:ro: Montira udev direktorij za prepoznavanje USB uređaja.

--device=/dev/ttyS0: Montira Z-Wave USB stick.

Za konfiguraciju zwavejs2mqtt, otvorimo web preglednik i upisujemo http://localhost:8091 kako bismo pristupili upravljačkom sučelju zwavejs2mqtt (Sl. 3.19).



Sl. 3.19 Konfiguracija zwavejs2mqtt upravljačkog sučelja

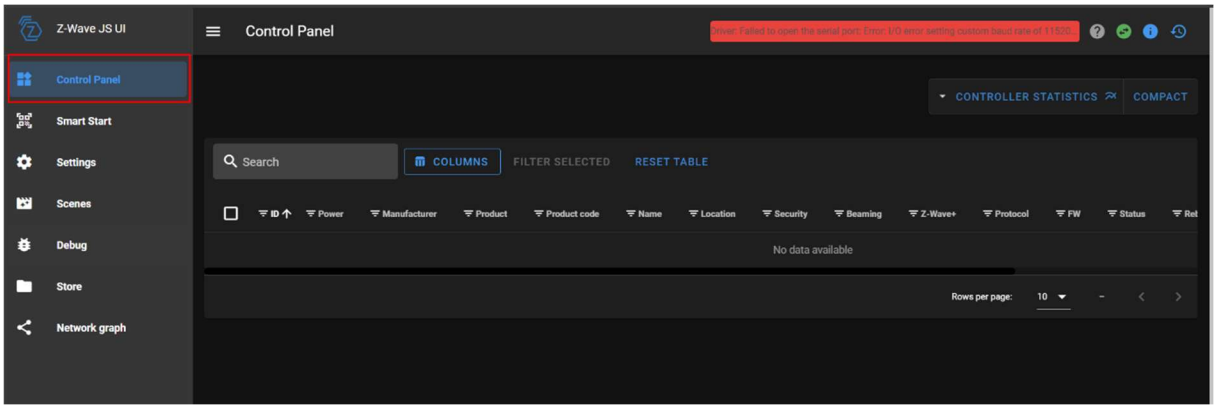
Prvi put kada pristupimo sučelju, potrebno je slijediti upute za početnu konfiguraciju. Odaberemo Z-Wave JS kao server za integraciju. Konfiguriramo port za serijsku komunikaciju s Z-Wave USB stickom (npr. /dev/ttyS0).

Po uputama smo uspješno instalirali Docker na Windows sustavu i pokrenuli zwavejs2mqtt Docker container.

3.4 Konfiguracija Danfoss Living Connect Z u zwavejs2mqtt Control Panelu

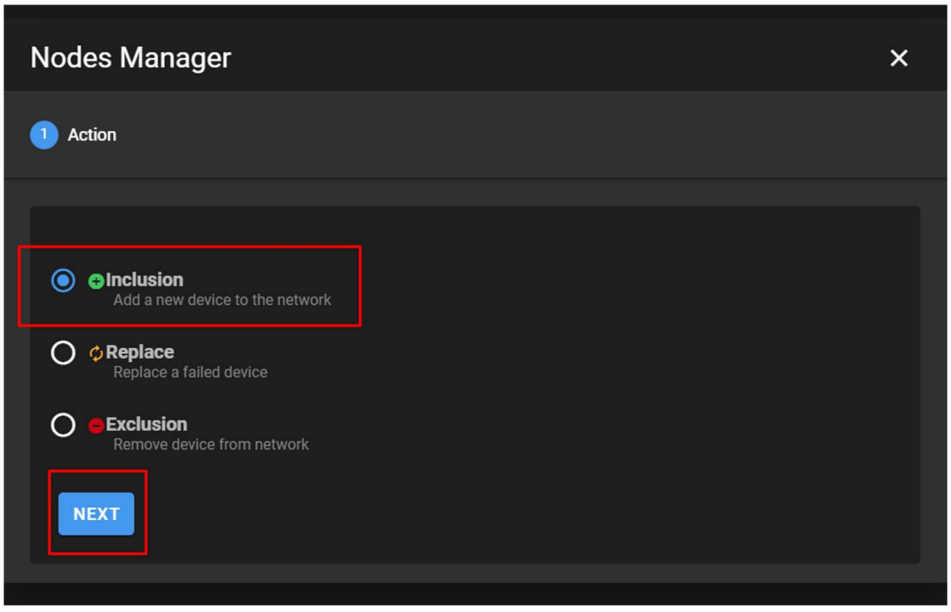
Nakon postavljanja Z-wave sučelja potrebno je konfigurirati Danfoss Living Connect Z pametni ventil u zwavejs2mqtt Control Panelu:

Da bismo dodali uređaj u Z-Wave mrežu, u zwavejs2mqtt sučelju, kliknemo na "Control Panel" (Sl. 3.20).



Sl. 3.20 Dodavanje Z-Wave uređaja u mrežu

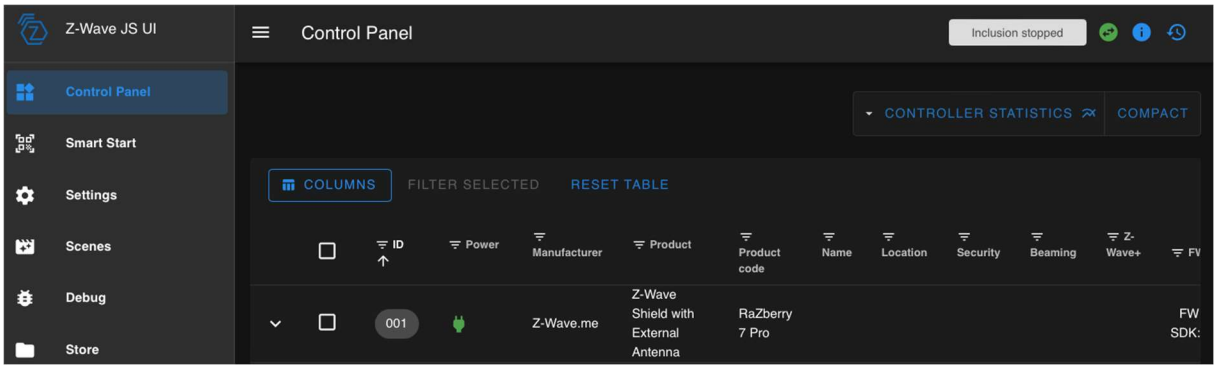
Kliknemo na Manage Nodes -> Inclusion (Sl. 3.21).



Sl. 3.21 Stvaranje nove niti za uređaj

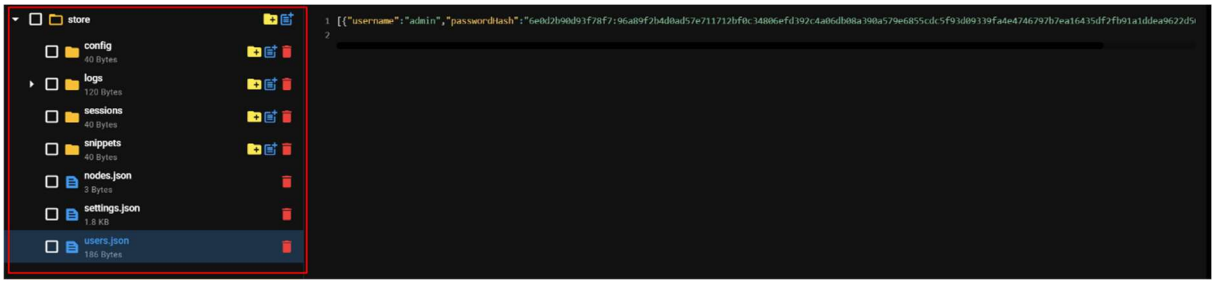
Uključujemo Danfoss Living Connect Z pametni ventil u način uparivanja: slijedimo upute za uparivanje Danfoss ventila.

Za uparivanje uređaja, nakon što je ventil u načinu uparivanja, zwavejs2mqtt sučelje će automatski otkriti uređaj. Potvrđujemo dodavanje uređaja u Z-Wave mrežu (Sl. 3.22).



Sl. 3.22 Pregled dodanog uređaja u Z-Wave mrežu

Za konfiguraciju uređaja, kliknemo na uređaj u zwavejs2mqtt Control Panelu kako bismo otvorili detalje konfiguracije (Sl. 3.23). Postavimo osnovne parametre uređaja kao što su naziv, soba u kojoj se nalazi, željena temperatura i način rada (npr. grijanje, hlađenje, isključeno).



Sl. 3.23 Detalji konfiguracije dodanog uređaja

3.5 Programiranje i postavljanje vlastite skripte u zwavejs2mqtt

Za kraj je potrebno postaviti i koristiti vlastitu skriptu u zwavejs2mqtt upravljačkoj ploči za čitanje, ispisivanje i mijenjanje temperature na pametnom ventilu.

Za pripremu skripte, napisali smo svoju skriptu u Pythonu koja će čitati trenutnu temperaturu s pametnog ventila i omogućiti korisnicima da postave novu željenu temperaturu (Sl. 3.24).

```python
import asyncio
import aiohttp
from zwave_js_server.client import Client

async def get_temperature(client, node_id):
    await client.connect()
    node = client.driver.controller.nodes[node_id]
    for value in node.values.values():
        if value.command_class == 49 and value.property_name == "Air temperature":
            print(f"Trenutna temperatura: {value.value}°C")
    await client.disconnect()

async def set_temperature(client, node_id, temperature):
    await client.connect()
    node = client.driver.controller.nodes[node_id]
    for endpoint in node.endpoints.values():
        for value in endpoint.values.values():
            if value.command_class == 67 and value.property_name == "Heating":
                await endpoint.set_value(value, temperature)
                print(f"Temperatura postavljena na {temperature}°C za čvor {node_id}")
    await client.disconnect()

async def main():
    ws_server_url = "ws://localhost:8091"
    node_id = 0

    async with aiohttp.ClientSession() as session:
        client = Client(ws_server_url, session)

        await get_temperature(client, node_id)

        new_temperature = float(input("Unesite željenu temperaturu: "))
        await set_temperature(client, node_id, new_temperature)

if __name__ == "__main__":
    asyncio.run(main())
```

Sl. 3.24 Skripta control_temperature.py

Spremamo skriptu u direktorij na računalu s imenom control_temperature.py.

Za upload skripte na server, moramo svoju skriptu pohraniti unutar Docker container-a pomoću naredbe docker cp:

docker cp control_temperature.py zwavejs2mqtt:/usr/src/app/control_temperature.py

Za pokretanje skripte unutar Docker container-a, povežemo se na Docker container koristeći naredbu docker exec:

docker exec -it zwavejs2mqtt /bin/sh

Pokrećemo skriptu koristeći Python:

python /usr/src/app/control_temperature.py

Skripta će ispisati trenutnu temperaturu i tražiti od korisnika da unese željenu temperaturu (Sl. 3.25).

```
Temperatura postavljena na 4.0°C
Trenutna temperatura: 22.5°C
Unesite željenu temperaturu: 7
Temperatura postavljena na 7.0°C
Trenutna temperatura: 22.5°C
Unesite željenu temperaturu: 3
Temperatura postavljena na 3.0°C
Trenutna temperatura: 22.5°C
Unesite željenu temperaturu: 24
Temperatura postavljena na 24.0°C
Trenutna temperatura: 22.5°C
Unesite željenu temperaturu:
```

Sl. 3.25 Vlasiti ispis dodane skripte

Za izlaganje API-ja u Home Assistant-u, postavljamo RESTful API u Home Assistant-u koji će primati željenu temperaturu i mijenjati postavke termostata (Sl. 3.26).

Za konfiguraciju API-ja, u configuration.yaml datoteci Home Assistant-a, dodajemo sljedeći kod za izlaganje API-ja:

configuration.yaml

ZavršniRad > configuration.yaml

```yaml
rest_command:
  set_thermostat_temperature:
    url: "http://localhost:8123/api/services/climate/set_temperature"
    method: post
    headers:
      content-type: "application/json"
    payload: '{"entity_id": "climate.living_room_thermostat", "temperature": {{ temperature }} }'

```

Sl. 3.26 RESTful API koji povezuje uređaj s Home Assistantom

Za korištenje API-ja iz Python skripte, ažuriramo Python skriptu kako bi koristila HTTP API za postavljanje temperature (Sl. 3.27):

httpApi.py

ZavršniRad > httpApi.py

```python
import requests

def set_temperature_via_api(temperature):
    url = "http://localhost:8123/api/services/climate/set_temperature"
    headers = {
        "Content-Type": "application/json"
    }
    payload = {
        "entity_id": "climate.thermostat",
        "temperature": temperature
    }
    response = requests.post(url, headers=headers, json=payload)
    print(response.json())

if __name__ == "__main__":
    current_temperature = 22.0
    print(f"Trenutna temperatura: {current_temperature}°C")

    new_temperature = float(input("Unesite željenu temperaturu: "))
    set_temperature_via_api(new_temperature)

```

Sl. 3.27 Skripta koja koristi HTTP za pristup API-ju

S ovim koracima, uspješno postavljamo i pokrećemo vlastitu skriptu unutar zwavejs2mqtt Docker container-a, te koristimo skriptu za čitanje i postavljanje temperature na pametnom ventilu.

Zaključak

Tijekom ovog završnog rada, uspješno smo istražili i implementirali ključne komponente za upravljanje pametnim ventilom Danfoss Living Connect Z pomoću Z-Wave tehnologije. Proces je obuhvaćao instalaciju i konfiguraciju Home Assistant platforme, korištenje Docker kontejnera za integraciju Z-Wave uređaja, te detaljno programiranje i upravljanje pametnim ventilom kroz zwavejs2mqtt Control Panel.

Jedan od glavnih ciljeva bio je omogućiti precizno i automatizirano upravljanje temperaturom. Kroz zwavejs2mqtt Control Panel kreirali smo automatizacije koje su omogućile postavljanje i prilagođavanje temperature prema unaprijed definiranim uvjetima. Također smo razvili vlastiti kod za upravljanje pametnim ventilom pomoću MQTT protokola, čime smo dodatno povećali fleksibilnost i kontrolu nad uređajem.

Tijekom projekta suočili smo se s nekoliko tehničkih izazova, uključujući probleme s prepoznavanjem serijskog porta i postavljanjem odgovarajućih komunikacijskih parametara. Kroz detaljno istraživanje i testiranje različitih pristupa, uspjeli smo riješiti ove probleme i osigurati stabilan rad sustava.

Implementacija pametnog ventila i integracija s Home Assistant platformom donosi brojne prednosti, uključujući poboljšanu energetsku učinkovitost, povećanu udobnost korisnika te mogućnost daljinskog upravljanja i automatizacije. Ovaj projekt demonstrira kako kombinacija različitih tehnologija i alata može rezultirati naprednim rješenjima za pametne domove.

Uspješna realizacija ovog završnog rada potvrđuje važnost integracije i interoperabilnosti različitih pametnih tehnologija u stvaranju učinkovitih i prilagodljivih sustava za upravljanje domom. Stečeno znanje i iskustvo predstavljaju čvrstu osnovu za buduće projekte u području pametnih kuća i IoT tehnologija. Kroz kontinuirano istraživanje i inovacije, nastavit ćemo doprinositi razvoju tehnologija koje poboljšavaju kvalitetu života korisnika u njihovim svakodnevnim okruženjima.

Literatura

[1] Home Assistant. (2024). Installation - Home Assistant. URL: https://www.home-assistant.io/installation/windows/ (pristupljeno 14. ožujka 2024.)

[2] Danfoss. (2024). Danfoss Living Connect Z - Product Documentation. URL: https://www.danfoss.com/ (pristupljeno 14. ožujka 2024.)

[3] Zigbee Alliance. (2024). About Zigbee Technology. URL: https://zigbeealliance.org/ (pristupljeno 11. ožujka 2024.)

[4] Z-Wave Alliance. (2024). What is Z-Wave? URL: https://z-wavealliance.org/ (pristupljeno 11. ožujka 2024.)

[5] Paulus Schoutsen. (2013). Home Assistant - A Look Back and Forward. URL: https://www.home-assistant.io/blog/ (pristupljeno 22. svibnja 2024.)

[6] Docker. (2024). Docker Documentation. URL: https://docs.docker.com/ (pristupljeno 17. travnja 2024.)

[7] MQTT. (2024). MQTT - The Standard for IoT Messaging. URL: https://mqtt.org/ (pristupljeno 17. travnja 2024.)

[8] Arduino. (2024). Arduino Documentation. URL: https://www.arduino.cc/en/Guide (pristupljeno 5. lipnja 2024.)

[9] Raspberry Pi Foundation. (2024). Getting Started with Raspberry Pi. URL: https://www.raspberrypi.org/documentation/ (pristupljeno 5. lipnja 2024.)

[10] zwavejs2mqtt. (2024). Home - zwavejs2mqtt. URL: https://zwave-js.github.io/zwavejs2mqtt/ (pristupljeno 17. travnja 2024.)

Upravljanje pametnim ventilom s pomoću tehnologije Z-Wave

Sažetak

Cilj ovog rada je bio istražiti i implementirati tehnologije potrebne za upravljanje pametnim ventilom Danfoss Living Connect Z koristeći protokol Z-Wave i integrirati ga s Home Assistant platformom. U radu su predstavljeni osnovni pojmovi i koncepti vezani uz pametne kuće, uključujući protokol Z-Wave, Home Assistant platformu, te Docker kontejnere.

Provedena je instalacija i konfiguracija Home Assistanta na Windows operativnom sustavu putem VirtualBoxa, čime je omogućeno centralizirano upravljanje pametnim uređajima. Nadalje, pokrenut je zwavejs2mqtt Docker container koji omogućava upravljanje Z-Wave uređajima putem jednostavnog web sučelja. Poseban fokus bio je na uparivanju i konfiguraciji Danfoss Living Connect Z pametnog ventila, pri čemu su se rješavali izazovi vezani uz postavljanje serijskih portova i baudrate postavki.

Također kreirane automatizacije za upravljanje temperaturom, uključujući razvoj vlastitog koda za upravljanje ventilom putem MQTT protokola. U završnom dijelu rada, analizirani su dobiveni rezultati, a izvedeni su zaključci o prednostima i izazovima implementiranih rješenja te mogućnostima za buduća poboljšanja.

Ključne riječi: pametni ventil, Z-Wave, Home Assistant, Docker, zwavejs2mqtt, MQTT, automatizacija, pametne kuće.

Smart valve control using Z-Wave technology

Abstract

The aim of this thesis was to explore and implement the technologies required to manage the Danfoss Living Connect Z smart valve using the Z-Wave protocol and to integrate it with the Home Assistant platform. The paper presents the basic concepts and notions related to smart homes, including the Z-Wave protocol, Home Assistant platform, and Docker containers.

The installation and configuration of Home Assistant on the Windows operating system via VirtualBox were conducted, enabling centralized management of smart devices. Furthermore, a zwavejs2mqtt Docker container was launched, allowing the management of Z-Wave devices through a simple web interface. Special focus was placed on pairing and configuring the Danfoss Living Connect Z smart valve, addressing challenges related to serial port settings and baud rate configurations.

Automation for temperature control was also created, including the development of custom code for controlling the valve via the MQTT protocol. In the final part of the thesis, the obtained results were analyzed, and conclusions were drawn about the advantages and challenges of the implemented solutions and the possibilities for future improvements.

Keywords: smart valve, Z-Wave, Home Assistant, Docker, zwavejs2mqtt, MQTT, automation, smart homes.