Web sustav za upravljanje mikroklimom plastenika na temelju podatkovne fuzije

Džijan, Perica

Master's thesis / Diplomski rad

2017

Degree Grantor / Ustanova koja je dodijelila akademski / stručni stupanj: **Josip Juraj Strossmayer University of Osijek, Faculty of Electrical Engineering, Computer Science and Information Technology Osijek / Sveučilište Josipa Jurja Strossmayera u Osijeku, Fakultet elektrotehnike, računarstva i informacijskih tehnologija Osijek**

Permanent link / Trajna poveznica: https://urn.nsk.hr/urn:nbn:hr:200:213850

Rights / Prava: In copyright / Zaštićeno autorskim pravom.

Download date / Datum preuzimanja: **2024-03-28**

Repository / Repozitorij:

Faculty of Electrical Engineering, Computer Science and Information Technology Osijek

zir.nsk.hr

Image not found or type unknown

SVEUČILIŠTE JOSIPA JURJA STROSSMAYERA U OSIJEKU
FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA I INFORMACIJSKIH TEHNOLOGIJA OSIJEK

Sveučilišni diplomski studij

WEB SUSTAV ZA UPRAVLJANJE MIKROKLIMOM PLASTENIKA NA TEMELJU PODATKOVNE FUZIJE

Diplomski rad

Perica Džijan

Osijek, 2017.

FERIT
FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA I INFORMACIJSKIH TEHNOLOGIJA OSIJEK

Obrazac D1: Obrazac za imenovanje Povjerenstva za obranu diplomskog rada

Osijek, 09.10.2017.

Odboru za završne i diplomske ispite

Imenovanje Povjerenstva za obranu diplomskog rada

Ime i prezime studenta:	Perica Džijan
Studij, smjer:	Diplomski sveučilišni studij Računarstvo, smjer Procesno računarstvo
Mat. br. studenta, godina upisa:	D 729 R, 14.10.2014.
OIB studenta:	86036413232
Mentor:	Prof.dr.sc. Goran Martinović
Sumentor:	
Sumentor iz tvrtke:	
Predsjednik Povjerenstva:	Prof.dr.sc. Željko Hocenski
Član Povjerenstva:	Doc.dr.sc. Ivica Lukić
Naslov diplomskog rada:	Web sustav za upravljanje mikroklimom plastenika na temelju podatkovne fuzije
Znanstvena grana rada:	**Programsko inženjerstvo (zn. polje računarstvo)**
Zadatak diplomskog rada:	Potrebno je opisati okolinu upravljanja mikoklimom plastenika s više povrtnih kultura, te koristeći načela podatkovne fuzije osmisliti model upravljanja pogodan za implementiranje u web sustav s bazom podataka. Koristeći ugradbeni računalni sustav s odgovarajućim senzorima, potrebno je sklopovski omogućiti prikupljanje više tipova podataka, te sustav odgovarajućih postavnih sprava. Programski treba omogućiti unos općenitih i za povrtne kulture željenih vrijednosti preko web sučelja, pohranu prikupljanih vrijednosti u bazu podataka, analizu prikupljenih podataka s predviđanjem, upravljanje i prikaz podataka. Ostavreno rješenje treba prikladno testirati.
Prijedlog ocjene pismenog dijela ispita (diplomskog rada):	Izvrstan (5)
Kratko obrazloženje ocjene prema Kriterijima za ocjenjivanje završnih i diplomskih radova:	Primjena znanja stečenih na fakultetu: 3 bod/boda Postignuti rezultati u odnosu na složenost zadatka: 3 bod/boda Jasnoća pismenog izražavanja: 2 bod/boda Razina samostalnosti: 3 razina
Datum prijedloga ocjene mentora:	09.10.2017.

Potpis mentora za predaju konačne verzije rada u Studentsku službu pri završetku studija:	Potpis:
	Datum:

FERIT
FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA I INFORMACIJSKIH TEHNOLOGIJA OSIJEK

IZJAVA O ORIGINALNOSTI RADA

Osijek, 16.10.2017.

Ime i prezime studenta:	Perica Džijan
Studij:	Diplomski sveučilišni studij Računarstvo, smjer Procesno računarstvo
Mat. br. studenta, godina upisa:	D 729 R, 14.10.2014.
Ephorus podudaranje [%]:	2

Ovom izjavom izjavljujem da je rad pod nazivom**: Web sustav za upravljanje mikroklimom plastenika na temelju podatkovne fuzije**

izrađen pod vodstvom mentora Prof.dr.sc. Goran Martinović

i sumentora

moj vlastiti rad i prema mom najboljem znanju ne sadrži prethodno objavljene ili neobjavljene pisane materijale drugih osoba, osim onih koji su izričito priznati navođenjem literature i drugih izvora informacija.
Izjavljujem da je intelektualni sadržaj navedenog rada proizvod mog vlastitog rada, osim u onom dijelu za koji mi je bila potrebna pomoć mentora, sumentora i drugih osoba, a što je izričito navedeno u radu.

Potpis studenta:

Sadržaj

1. UVOD 1
2. SUSTAVI ZA KONTROLIRANU PROIZVODNJU POVRĆA U PLASTENICIMA 2
2.1. Konstrukcija plastenika 2
2.2. Mikroklima plastenika 4
2.2.1 Temperatura 4
2.2.2. Svjetlost 6
2.2.3. Voda 6
2.2.4. Vlažnost zraka 7
3. PRIKAZ POSTOJEĆIH RJEŠENJA I VLASTITO IDEJNO RJEŠENJE 8
3.1. Prikaz postojećih rješenja upravljanja plasteničkom proizvodnjom 8
3.2. Idejno rješenje vlastite okoline plastenika 9
4. KORIŠTENO SKLOPOVLJE, PROGRAMSKE TEHNOLOGIJE I POTPORA PODATKOVNOJ FUZIJI 11
4.1. Sklopovski dio sustava 11
4.1.1. Prototipna platforma Arduino 11
4.1.2. Komunikacija i senzori 12
4.2. Programske okoline i tehnologije 13
4.2.1. NET Framework 13
4.2.2. ASP.NET MVC 14
4.2.3. ASP.NET Web API 14
4.2.4. Entity Framework 15
4.2.5. jQuery 15
4.2.6. Morris.js 16
4.2.7. Microsoft Azure 16
4.3. Osnove podatkovne fuzije 17

4.3.1. Izračun uvjetne vjerojatnosti ... 18
5. PROGRAMSKO RJEŠENJE SUSTAVA ... 20
5.1. Baza podataka i oblak računala ... 20
5.2. Struktura aplikacije ... 23
5.3. Modeli ... 23
5.4. Upravljači ... 26
5.5. Pogledi ... 29
5.6. Implementacija podatkovne fuzije ... 32
5.7. Programsko rješenje klijenta na Arduinu ... 33
6. PRIKAZ RADA SUSTAVA S TESTIRANJEM I ANALIZOM ... 36
6.1. Unos podataka ... 36
6.2. Prikaz trenutnih vrijednosti parametara sustava ... 37
6.3. Postavke testiranja ... 38
6.4. Analiza rezultata ... 39
7. ZAKLJUČAK ... 43
LITERATURA ... 44
SAŽETAK ... 46
ABSTRACT ... 47
ŽIVOTOPIS ... 48
PRILOZI (na CD-u) ... 49

1. UVOD

Proizvodnja povrća u zaštićenim prostorima jako je bitan dio prehrambene industrije. Kontrola mikroklimatskih uvjeta u zaštićenom prostoru uvelike pridonosi učinkovitosti proizvodnje povrća. Vrijeme uzgoja se povećava na cijelu godinu, povećava se prinos i kvaliteta proizvoda, te se smanjuje pojava bolesti i štetnika. Neki od najvažnijih mikroklimatskih uvjeta su temperatura, relativna vlažnost zemlje i intenzitet svjetlosti.

U ovom radu su objašnjeni sustavi za kontroliranu proizvodnju povrća u plastenicima. Opisana je konstrukcija plastenika i osnovni mikroklimatski uvjeti. Dan je primjer postojećeg rješenja upravljanja mikroklimom plastenika, kao i idejno rješenje vlastitog sustava. Prikazana je izrada potpuno funkcionalnog web sustava za upravljanje mikroklimom plastenika. Web sustav se sastoji od web aplikacije i ugradbenog računalnog sustava. Web aplikacija omogućuje korisniku unos željenih povrtnih kultura zajedno s optimalnim vrijednostima u bazu podataka. Ugradbeni računalni sustav pomoću odgovarajućih senzora prikuplja podatke te ih šalje web poslužitelju. S obzirom na referentne i trenutne vrijednosti, preko izvršnih uređaja, ugradbeni računalni sustav regulira mikroklimu u plasteniku. Trenutne vrijednosti se prikazuju na glavnoj stranici web aplikacije.

S obzirom da niti jedan sustav nije savršen, može se dogoditi da dođe do pogreške u komunikaciji između poslužitelja i senzora ili izvršnih uređaja. U radu je na primjeru kada poslužitelj ne dobije vrijednost trenutne temperature implementiran alat koji omogućuje da se pomoću podatkovne fuzije zaključi trenutno stanje u plasteniku i prema tome djeluje na izvršni uređaj. Podatkovna fuzija kombinira podatke iz više izvora da bi se dobile točnije i korisnije informacije od onih koje pruža samo jedan izvor. Na kraju rada, opisan je rad aplikacije, napravljena su testiranja sustava i analizirani su rezultati.

U poglavlju 2 je opisana konstrukcija plastenika i mikroklima plastenika. U poglavlju 3 je prikazan postojeći sustav upravljanja mikroklimom plastenika i idejno rješenje vlastite okoline plastenika. U poglavlju 4 su objašnjene programske tehnologije i korišteno sklopovlje, te je opisan postupak podatkovne fuzije. Poglavlje 5 prikazuje programsko rješenje sustava. U poglavlju 6 je prikazan rad sustava, te je napravljeno testiranje i analiza rezultata testiranja.

2. SUSTAVI ZA KONTROLIRANU PROIZVODNJU POVRĆA U PLASTENICIMA

Uzgoj u zaštićenom prostoru ne razlikuje se puno od uzgoja na otvorenom, osim što kod zaštićenog prostora postoji mogućnost kontroliranja mikroklimatskih uvjeta, što rezultira produžavanjem razdoblja plodonošenja te povećanjem prinosa. Zaštićene prostore čine plastenici i staklenici. S obzirom da je tema ovog diplomskog rada vezana za plastenike, u nastavku će biti opisana konstrukcija i izvedbe plastenika.

2.1. Konstrukcija plastenika

Prema [1], osnova plastenika je statički stabilna nosiva konstrukcija koja se uglavnom izrađuje od čeličnih, trajno zaštićenih tankostjenih cijevi. Raspored temeljnih stopa je uvjetovan veličinom i rasponom glavnih lučnih nosača. Stope kod plastenika većih raspona su u pravilu betonske, promjera 450 - 500 mm i dubine 700 – 800 mm u kojoj je ugrađen temeljni stup. Ovisno o namjeni i veličini, izvedba plastenika može biti realizirana iz jednog tunela (jednolađni) ili više tunela međusobno povezanih u cjelinu (višelađni). Prema [2], s obzirom na način izvedbe zaštićeni prostori se mogu podijeliti na nekoliko tipova:

- Niski tunel – najčešće u kućnim povrtnjacima. Konstrukcijski vrlo jednostavan i nije predviđen za instalaciju sustava za grijanje. Koristi se uglavnom za proizvodnju sadnica
- Visoki tunel – koristi se zbog niske cijene i jednostavne konstrukcije, za uzgoj na manjim površinama. Najčešće se koristi za predsezonsku proizvodnju bez grijanja. Ventilacija na ovakvom tipu plastenika je pasivna, obavlja se otvaranjem vrata i bočnih stranica.
- Plastenik – jednolađni plastenik s kružnim krovom. Pokrivač je najčešće polietilenska folija, a okomite stranice od polikarbonatnog panela. U krajevima gdje ima puno snijega plastenik se izvodi s „gotskim krovom“, koji ima lučni profil s vrhom na sredini. Ventilacija se obavlja otvaranjem vertikalnih stranica ili ventilatorima. Pogodan je za ugradnju sustava za grijanje te razdoblje proizvodnje traje gotovo cijelu godinu.
- Jednolađni staklenik/plastenik – za pokrov se koristi nesavitljiva plastika ili staklo (slika 2.1) Pogodan za cjelogodišnju proizvodnju uz mogućnost ugradnje različitih vrsta instalacija.
- Višelađni plastenik – sličan kao jednolađni plastenik. Više međusobno povezanih jednolađnih plastenika u cjelinu (slika 2.2). Koristi se na većim površinama.

- Višelađni staklenik/plastenik – više povezanih staklenika/plastenika u cjelinu. Koriste se na većim površinama uzgoja.
- Venlo staklenik – naziv je dobio po gradu u Nizozemskoj. Može imati i dvije središnje provodnice po jednoj lađi.

Slika 2.1. Jednolađni plastenik

Slika 2.2. Višelađni plastenik [3]

Prema [4], za pokrovni materijali plastenika koriste se materijali propusni za svjetlo, što su uglavnom polimerni materijali koji se proizvode u obliku folije ili ploča (panela). Najveći nedostatak polimera u odnosu na staklo je ograničeno vrijeme trajanja zbog ultraljubičastog

zračenja koje mijenja kemijsku strukturu polimera, čime se smanjuje propusnost za svjetlo potrebno za fotosintezu biljaka. Polietilenske folije su najraširenji polimeri za pokrivanje plastenika u svijetu (više od 90%). Kod manjeg dijela plastenika za pokrov je korištena kopolimerna folija koja osim polietilena sadrži i polimer etilenvinilacetat (EVA) koji smanjuje prolaznost topline. Folije koje sadrže EVA bolje čuvaju toplinu u plasteniku, smanjuju propusnost topline i za više od 50%. Polietilenske folije mogu biti jednoslojne ili dvoslojne. Kod dvoslojnih se između slojeva upuhuje zrak, te se tako smanjuju gubici topline, ali je propusnost svjetla manja. Za pokrivanje plastenika još se koriste i polikarbonatne i poliesterske ploče. Polikarbonatne ploče su ravne i dvoslojne. Debljina jednog sloja iznosi oko 0.5 mm, a zajedno sa zrakom između dva sloja ukupna debljina je od 6 do 12 mm. Poliesterske ploče su valovite, te zbog boljih mehaničkih svojstava sadrže vlakna polimetilmetaakrilata. Debljine su oko 1mm.

2.2. Mikroklima plastenika

Mikroklima plastenika je jedan od najvažnijih čimbenika kvalitetne i ekonomski opravdane proizvodnje. Direktno utječe na rast biljke i njen prinos, te pojavu bolesti i štetnika. Senzori za mjerenje mikroklimatskih čimbenika su uglavnom nepomični i žično povezani s upravljačkom jedinicom. Prema [15], postoje i rješenja u kojima se koriste pokretni roboti za prikupljanje informacija o mikroklimatskim uvjetima. Roboti se samostalno kreću kroz okolinu i prikupljene podatke bežičnom vezom šalju upravljačkoj jedinici.

2.2.1 Temperatura

Možda i najvažniji mikroklimatski uvjet je temperatura, koja direktno utječe na rast biljke, ranu zriobu te kvalitetu i količinu prinosa. Odstupanje od granica optimalnih vrijednosti rezultira usporenim rastom biljke, a trajna izloženost takvim uvjetima dovodi do prekida rasta biljke. Da bi se izbjegle nagle promjene temperature i razlike između dnevnih i noćnih temperatura potrebno je plastenike opremiti sustavom grijanja. Danas u plasteničkoj proizvodnji vrlo široku primjenu imaju najjednostavniji sustavi koji se sastoje od uređaja za proizvodnju toplog zraka s dimovodnim cijevima (slika 2.3). Ovi uređaji se postavljaju na nosivu konstrukciju u gornjoj zoni plastenika.

Slika 2.3. Uređaj za proizvodnju toplog zraka

Najučinkovitiji sustavi za grijanje plastenika su sustavi toplovodnog grijanja (slika 2.4), posebno ako je riječ o hidroponskom načinu uzgoja. Kod ovakvog sustava, sustav cijevi se postavlja na pod. Temperatura vode regulira se elektromotornim ventilima koji se nadziru središnjom upravljačkom jedinicom. Prednost ovog sustava je najviše izražena kod primjene na velikim površinama.

Slika 2.4. Sustav podnog grijanja s toplovodnim cijevima tvrtke GIS IMPRO d.o.o. [5]

2.2.2. Svjetlost

Za sve biljke neophodni su određeni intenzitet i kvaliteta svjetlosti, te određena dužina dana. Kod uzgoja povrtnih kultura najznačajniji je vidljivi dio spektra, fotosintetska aktivna radijacija (FAR), pri kojoj se normalno odvija fotosinteza. Ovisno o duljini dnevne svjetlosti, biljke dijelimo na biljke dugog i kratkog dana. Većina povrtnih kultura koje se uzgajaju u zaštićenim prostorima zahtijevaju dosta svjetlosti. Pri nedostatku svjetlosti biljke kasnije cvjetaju i formiraju plod. Za optimalnu osvijetljenost plastenika važno je da maksimalna količina sunčevih zraka pada pod kutem od 90°, te je zbog toga optimalan polukružni oblik krova. Sunčeve zrake koje padaju pod drugim kutem reflektiraju se izvan plastenika. Za fotosintezu najznačajniji dio sunčevog spektra su zrake valne duljine od 360 do 760 um, koje čine oko 50% sunčeve insolacije. Pri nedostatku dovoljne količine sunčeve svjetlosti koriste se asimilacijske lampe, kao na slici 2.5.

Slika 2.5. Asimilacijske svjetiljke u plasteniku Poljoprivredno šumarske škole u Vinkovcima

2.2.3. Voda

Kapacitet i izbor opreme za navodnjavanje ovisi o potrebama biljaka za vodom i tehnologije uzgoja. Osnovni tipovi navodnjavanja su orošavanjem i kapanjem. Kod navodnjavanja kapanjem, cijevi se postavljaju na zemlju i navodnjavanje se odvija ispod lista. U slučaju ovog načina navodnjavanja, zbog neravnog terena, može doći do stvaranja ulegnuća, koja se ispunjavaju vodom, čime se povećava vlažnost u zoni biljke i mogućnost pojave bolesti. Proizvodnja presadnica zahtijeva malo složeniji sustav navodnjavanja. Navodnjavanje se vrši fleksibilnim sustavom navodnjavanja, odnosno cijevi se podižu iznad biljke i rotacijskim rasprskivačima se vrši takozvano orošavanje. Ovakav sustav je vidljiv na slici 2.5. Prema [6], često se u plastenike

ugrađuje i drenažni sustav. On predstavlja vrlo korisnu instalaciju jer omogućuje da se ispiranjem zemljišta smanji prekomjerna koncentracija soli koja nastaje uslijed obilnog prehranjivanja poljoprivrednih kultura mineralnim gnojivom. Sustav se sastoji od drenažnih cijevi i kolektora, smještenim oko 75 cm ispod površine zemlje, iznad kojih se obično nalazi sloj šljunka.

2.2.4. Vlažnost zraka

Jako bitnu ulogu u rastu biljke ima i relativna vlažnost zraka. Ona utječe na pojavu bolesti, oplodnju, fotosintezu i transpiraciju. Transpiracija je isparavanje vode iz biljaka kroz puči lista, čime se smanjuje hidrostatski tlak u gornjem dijelu biljke i time se omogućuje dotok hranjivih tvari od korijena do gornjeg dijela biljke. Manja relativna vlažnost zraka povećava stopu transpiracije. Ovisno o potrebama biljke, vrši se dodatno orošavanje, ukoliko biljka zahtjeva visoku relativnu vlažnost zraka, dok kod biljaka koje ne podnose visoku vlažnost zraka potrebno je provjetravati plastenik nakon svakog zalijevanja.

3. PRIKAZ POSTOJEĆIH RJEŠENJA I VLASTITO IDEJNO RJEŠENJE

3.1. Prikaz postojećih rješenja upravljanja plasteničkom proizvodnjom

U sklopu Poljoprivredno šumarske škole u Vinkovcima nalaze se i plastenici u kojima se kao primarna kultura uzgaja rajčica. Jedan plastenik služi za uzgoj sadnica rajčice, dok drugi služi za proizvodnju gotovog proizvoda. Proizvodnja je hidroponska, što je vidljivo na slici 3.1 gdje je prikazan eksperimentalni uzgoj zelene salate.

Slika 3.1. Hidroponska proizvodnja zelene salate

Plastenici su potpuno automatizirani pomoću sustava NETAFIM NMC-Junior. Ovaj sustav se sastoji od upravljačke jedinice (slika 3.2), senzora i izvršnih uređaja. Sustavom se regulira količina vode, hranjivih tvari, temperatura i pH-vrijednost hranjive tekućine.

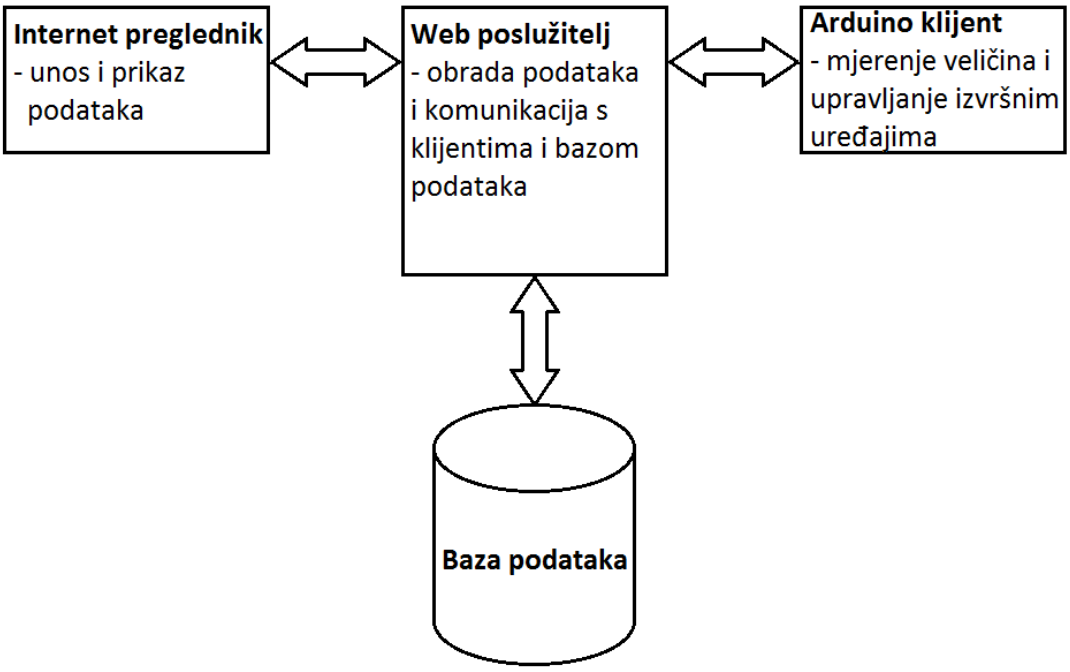
Slika 3.2. Upravljačka jedinica sustava NETAFIM NMC-Junior

Prema[14], osnovne sklopovske značajke upravljačke jedinice su: petnaest izlaza izmjenične struje od 24V, šest digitalnih, pet analognih ulaza za senzore, LCD zaslon. Od programske podrške sadrži: petnaest programa zalijevanja, kontrolu protoka vode, program za hlađenje i ovlaživanje, povijest zalijevanja i prihrane. Zalijevanje se vrši tako da se odabere jedan od postojećih programa ili se unosom preko LCD zaslona definira novi program. U slučaju Poljoprivredne i šumarske škole u Vinkovcima zalijevanje se obavlja pomoću šest programa, po jedan za svaki mjesec proizvodnje. U prvom mjesecu se zalijeva dvanaest puta dnevno po dvije minute, te se za svaki sljedeći mjesec trajanje jednog zalijevanja produžuje za jednu minutu, osim u zadnjem mjesecu, kada se zalijevanje smanjuje za jednu minutu. Ukupna potrošnja vode za jedan ciklus uzgoja rajčice je 477180 litara. Osim spremnika za vodu, sustav sadrži i još dva spremnika. Jedan služi za miješanje hranjivih tvari, dok je u drugom dušična kiselina za regulaciju pH-vrijednosti.

3.2. Idejno rješenje vlastite okoline plastenika

Rješenje sustava se sastoji od web aplikacije za unos, prikaz i obradu podataka, baze podataka u koju će se spremati podaci o povrtnim kulturama, stanju plastenika, te izmjerenim vrijednostima temperature, vlažnosti zraka, vlažnosti zemlje i intenziteta svjetlosti i makete plastenika s pripadajućim sustavom mikro-računala, senzora i izvršnih uređaja. Maketa plastenika je izrađena od akrilnog stakla na drvenom postolju. Za mikro računalo se koristi Arduino (inačica Croduino). Na Arduino su spojeni senzor za temperaturu i vlažnost zraka, senzor intenziteta svjetlosti i senzor za vlažnost zraka. Očitavanje vrijednosti se vrši svakih 5 sekundi i podaci se šalju na poslužitelj. Poslužitelj nakon obrade podataka vraća podatak koji sadrži informacije o tome koji izvršni uređaji trebaju biti uključeni ili isključeni. Uključivanje/isključivanje uređaja se vrši preko releja. Za upravljanje temperaturom se koristi žarulja za zagrijavanje prostora i ventilator za rashlađivanje prostora. Ukoliko je nedovoljna količina svjetlosti, pali se LED žarulja. Sustav navodnjavanja je realiziran pomoću potopne pumpe koja se nalazi u posudi s vodom i polimernog crijeva koji je doveden do posude sa zemljom. Web aplikacija se sastoji od tri prozora. Glavni prozor je kontrolna ploča i na njemu su prikazani grafovi trenutnih vrijednosti temperature, vlažnosti zraka, vlažnosti zemlje i intenziteta svjetlosti. Grafovi prikazuju vrijednosti u stvarnom vremenu. Ispod grafova je područje za podatkovnu fuziju. Pritiskom na gumb može se uključiti ili isključiti podatkovna fuzija. U slučaju da je uključena, prikazuju se tri dodatne kućice u kojima se nalaze podaci o vjerojatnosti da je temperatura niska, u redu ili previsoka, a koji su dobiveni uvjetnom vjerojatnosti temperature s obzirom na količinu svjetlosti. U drugom prozoru se nalaze informacije o kulturama koje se nalaze u plasteniku. U ovom prozoru je moguće iz baze podataka povrtnih kultura, preko

padajućeg izbornika, odabrati povrtnu kulturu koja se nalazi u plasteniku. U trećem prozoru je tablični prikaz svih povrtnih kultura koje su unesene u bazu podataka. Ovdje je moguće dodavati novu kulturu ili brisati postojeću kulturu iz baze podataka. Tablica sadrži podatke o nazivu i vrsti biljke i optimalnim vrijednostima temperature, vlažnosti zraka, vlažnosti zemlje i intenziteta svjetlosti. Na slici 3.3 je prikazana idejna shema sustava.



Slika 3.3. Idejna shema sustava

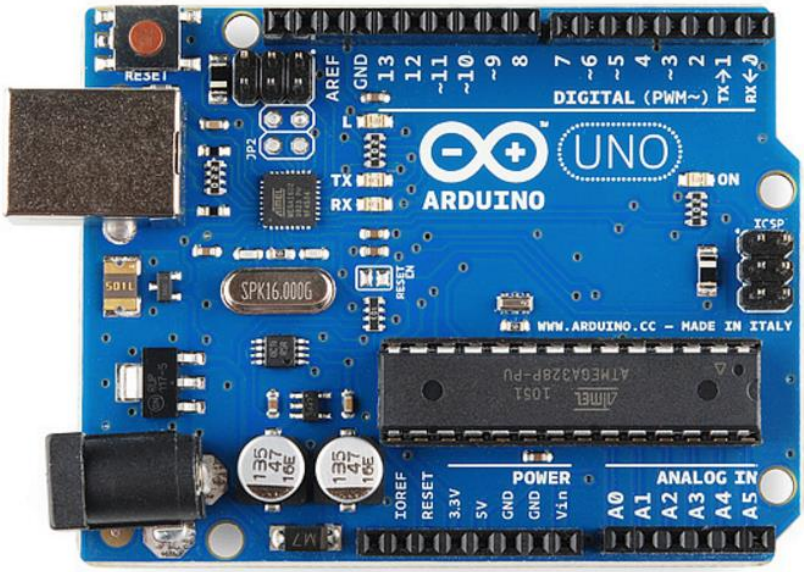
4. KORIŠTENO SKLOPOVLJE, PROGRAMSKE TEHNOLOGIJE I POTPORA PODATKOVNOJ FUZIJI

Za izradu web aplikacije korištene su tehnologije tvrtke Microsoft i tehnologije otvorenog koda. Za izradu ugradbenog računalnog sustava korišten je Arduino s pripadajućim senzorima i postavnim spravama. U nastavku će biti objašnjene korištene tehnologije pri izradi sustava.

4.1. Sklopovski dio sustava

4.1.1. Prototipna platforma Arduino

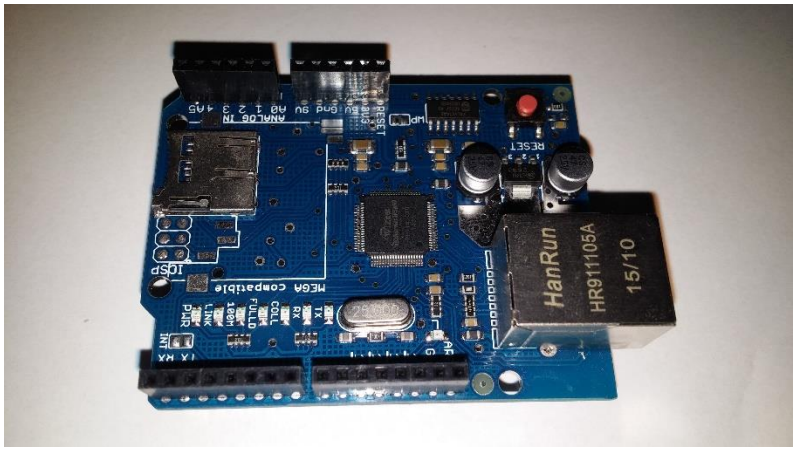
Arduino (slika 4.1) je prototipna platforma otvorenog koda koja omogućuje konstruktorima stvaranje uređaja i naprava koje omogućuju spajanje računala sa stvarnim svijetom. Prema [7], sastoji se od fizičkog programabilnog strujnog kruga (mikroupravljač) i programskog dijela koji se naziva IDE (*Integrated Development Environment*) koji se pokreće na računalu i koristi za pisanje i upload programskog koda na fizičku ploču. Programiranje se obavlja u prilagođenoj inačici programskog jezika C++.



Slika 4.1. Arduino pločica [7]

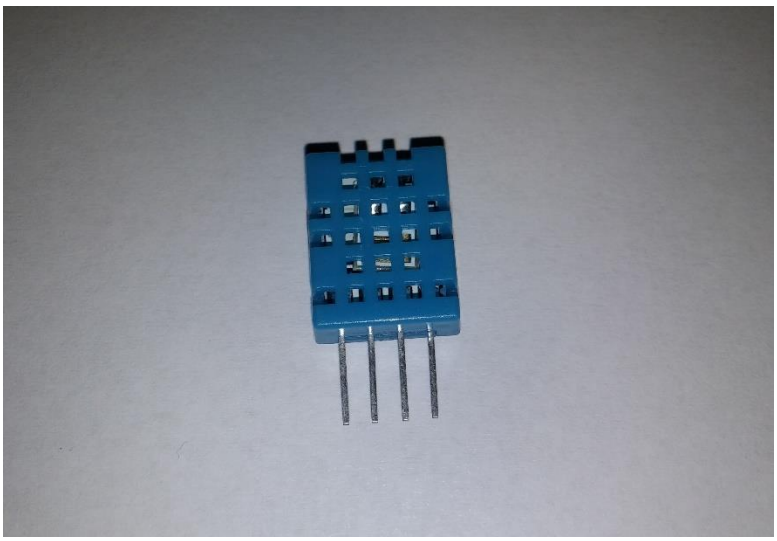
4.1.2. Komunikacija i senzori

Za komunikaciju je korišten Arduino ethernet shield S W5100 (slika 4.2). Prema [8], W5100 je potpuno opremljen, jednokomponentni 10/100 Ethernet upravljač dizajniran za ugradbene sustave gdje je potrebna jednostavnost integracije, stabilnost i performanse uz male troškove. Dizajniran je za lakše implementiranje internetske veze bez dodatnog operacijskog sustava. Za SPI komunikaciju s Arduinom koristi ICSP priključak. Na mrežu se spaja pomoću RJ-45 priključka.

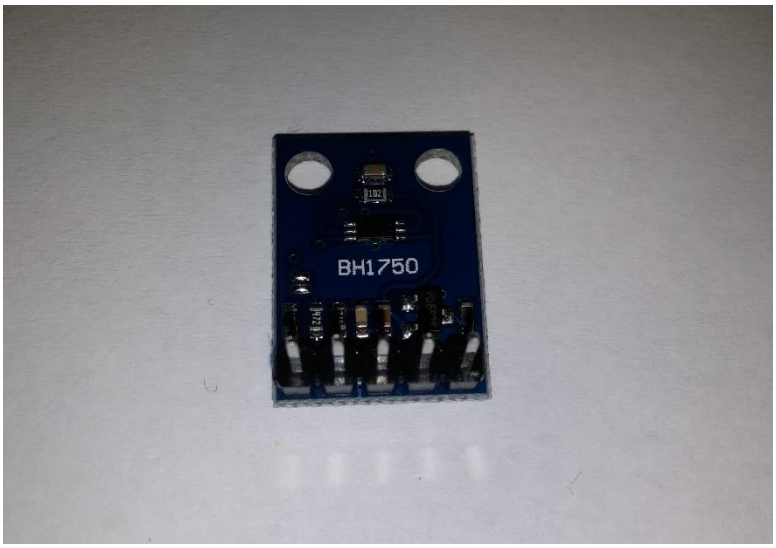


Slika 4.2. *Ethernet shield* S W5100

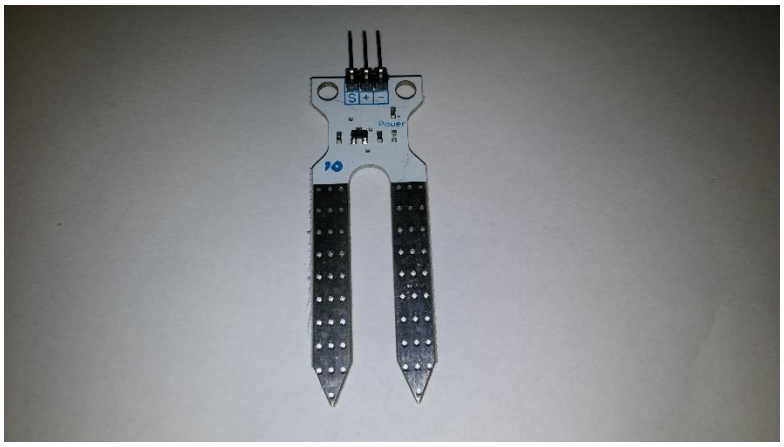
Od senzora su korišteni senzor za temperaturu i vlažnost zraka DHT11 (slika 4.3), senzor za intenzitet svjetlosti BH1750 (slika 4.4) i senzor za vlažnost zemlje VMA303 (slika 4.5).



Slika 4.3. Senzor za temperaturu i vlažnost zraka DHT11



Slika 4.4. Senzor intenziteta svjetlosti BH1750



Slika 4.5. Senzor vlažnosti zemlje VMA303

4.2. Programske okoline i tehnologije

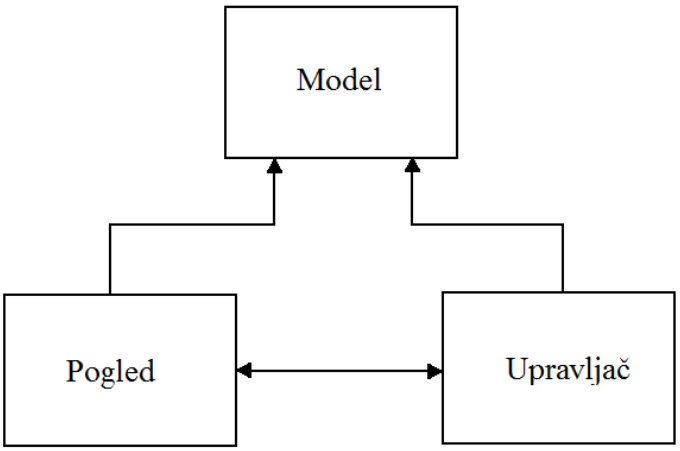
4.2.1. NET Framework

.NET Framework je Microsoftov okvir za programiranje. Omogućuje programerima lakše razvijanje i održavanje web stranica, usluga i aplikacija. Prvenstveno je namijenjen pokretanju na operacijskom sustavu Microsoft Windows.

Microsoft je .NET Framework počeo razvijati u kasnim 1990-im godinama pod originalnim imenom *Next Generation Windows Services (NGWS)*. Prva inačica .NET Frameworka se pojavljuje 2002. godine. Microsoft je nastavio s daljnjim razvojem, te je najnovija inačica 4.7 iz 2017. godine.

4.2.2. ASP.NET MVC

Prema [9], ASP.NET MVC je okvir za razvoj web aplikacija razvijen na temeljima Microsoftovog popularnog i zrelog kostura .NET Framework. Čvrsto se oslanja na dokazane predloške i praksu kojima je naglasak na slabo povezanoj arhitekturi aplikacije i kodu koji se vrlo lako održava. 2008. Microsoft je ponudio prvu inačicu platforme ASP.NET MVC. Koristi se za odvajanje aplikacije na tri osnovne komponente: model, pogled (*View*) i upravljač (*Controller*). Model služi za vezu s bazom podataka, te izravno upravlja podacima, logikom i pravilima aplikacije. Učahuruje svojstva i ponašanje entiteta te izlaže svojstva koja ga opisuju. Pogled služi za prikaz podataka. On je „ono što korisnik vidi“. Odgovoran je za pretvaranje jednog ili više modela u vizualnu reprezentaciju. Upravljač upravlja korisničkim zahtjevima, radi s modelima i na kraju odabire pogled koji će se prikazati na korisničkom sučelju. Prima ulazne informacije, surađuje s modelom i izvršava određene operacije te rezultate prosljeđuje pogledu. Na slici 4.6, prema [9], je prikazana arhitektura ASP.NET MVC okvira.



Slika 4.6. MVC oblikovni obrazac

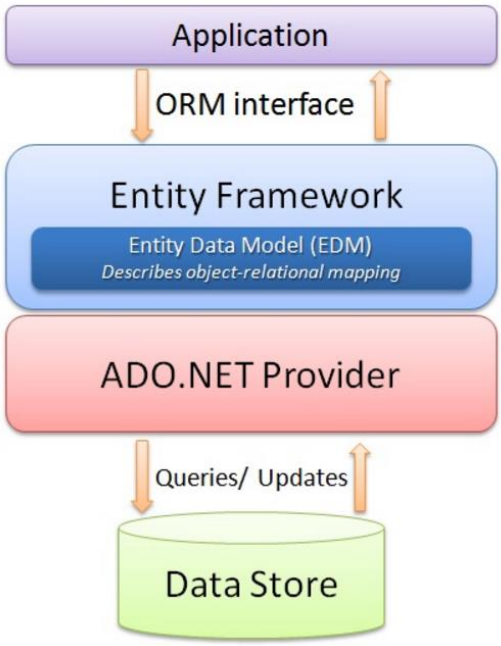
4.2.3. ASP.NET Web API

ASP.NET Web API je okvir za razvijanje HTTP usluga koje mogu biti korištene od strane različitih klijenata, kao što su mobilni uređaji, tableti, internet preglednici. Koristi HTTP metode: GET, PUT, POST, DELETE za osnovne operacije s podacima (*CRUD metode*). Vrlo je sličan ASP.NET MVC okviru, jer se oslanja na iste temeljne koncepte, kao što su usmjeravanje, upravljači i rezultati akcija upravljača. Za razliku od ASP.NET MVC okvira, WEB.API te

koncepte koristi na malo drugačiji način i to kako bi podržavao bitno drugačiji skup situacija, koje podrazumijevaju rad s podacima a ne samo generiranje HTML koda.

4.2.4. Entity Framework

Entity Framework je, prema [12], okvir za objektno relacijsko mapiranje (ORM), koji je otvorenog koda i koristi ADO.NET. ADO.NET je vrlo jak okvir za pristup podacima, koji je razvila tvrtka Microsoft. Prva inačica Entity Frameworka je uključena u .NET Framework 3.5 SP1 2008. godine. Od 2013. godine i izlaska inačice 6.0 postaje otvorenog koda i nije više dio .NET Framework-a. Entity Framework olakšava rad s podacima i pojednostavljuje programerima korištenje osnovnih (*CRUD*) operacija s podacima. Vrlo se jednostavno rukovodi „1 na 1“, „1 na više“ i „više na više“ vezama. Na slici 4.7 je prikazana arhitektura Entity Framework-a.



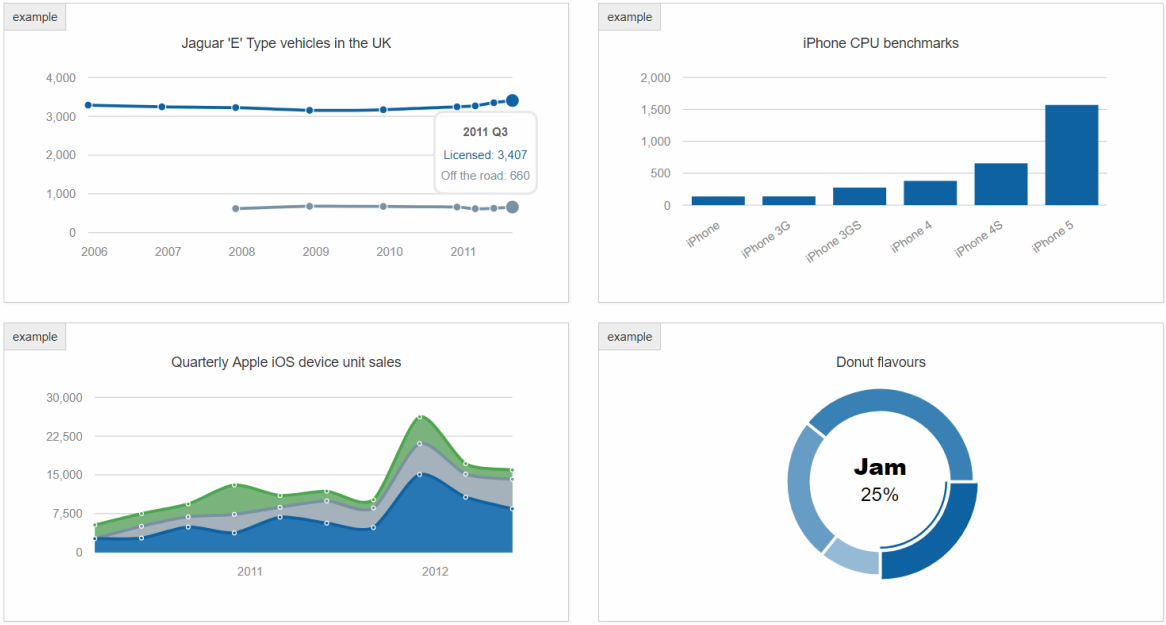
Slika 4.7. Arhitektura Entity Framework-a [12]

4.2.5. jQuery

jQuery je višeplatformska JavaScript biblioteka otvorenog koda dizajnirana tako da pojednostavi skriptiranje HTML-a na klijentskoj strani. Sintaksa je dizajnirana tako da olakša odabiranje modela za prikaz i interakciju s objektima u HTML dokumentu (DOM), kreiranje animacija, rukovođenje događajima i razvoj Ajax aplikacija. jQuery je objavljen 2006. godine na BarCamp konferenciji.

4.2.6. Morris.js

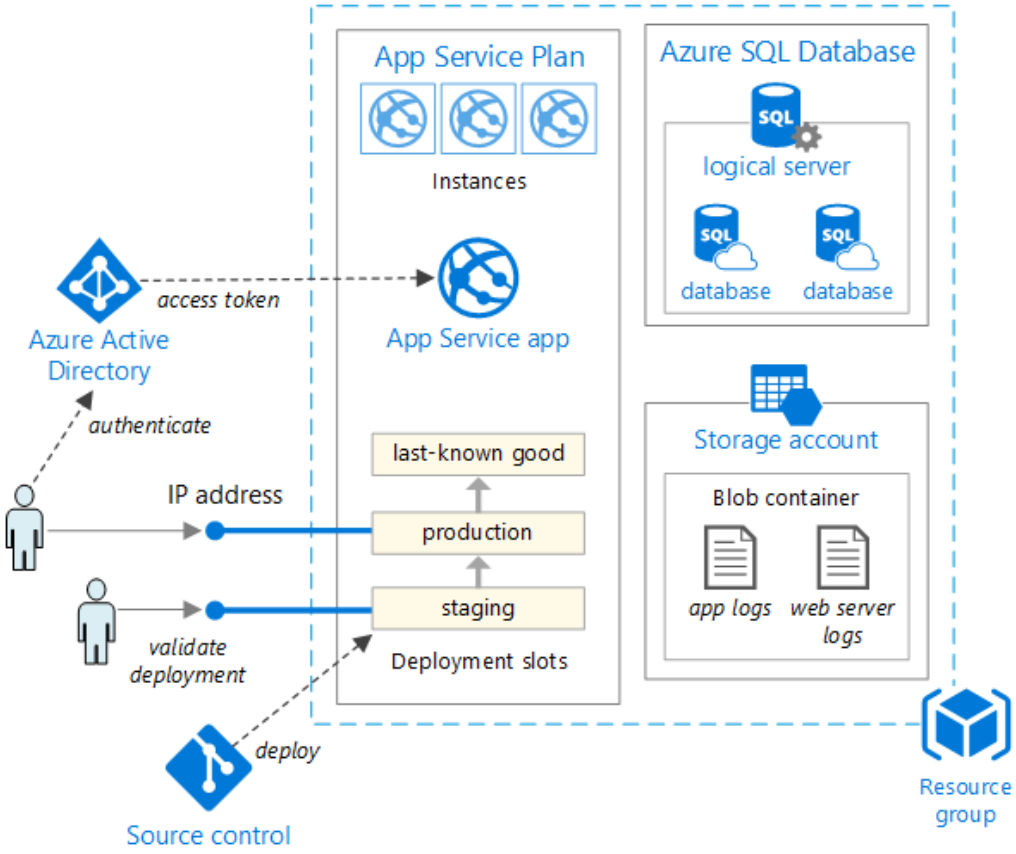
Morris.js je biblioteka za crtanje grafova. Temelji se na Raphaël biblioteci za crtanje vektorskih grafova, koja za prikaz podataka korist SVG. Morris.js je napravljen kako bi olakšao korištenje jednostavnijih grafova. Na slici 4.8 su prikazani primjeri korištenja različitih Morris.js grafova.



Slika 4.8. Primjeri korištenja Morris.js

4.2.7. Microsoft Azure

Microsoft Azure je platforma za oblak računala koja programerima omogućuje lakšu izradu, implementaciju i upravljanje aplikacijama. Pruža infrastrukturu kao uslugu (eng. Infrastructure as a Service – IaaS), platformu kao uslugu (eng. Platform as a Service – PaaS) i programsku podršku kao uslugu (Software as a Service – SaaS), te alate i okvire, uključujući programe i sustave specifične za Microsoft, ali i programe i sustave treće strane. Azure je najavljen u listopadu 2008. godine, a objavljen 1. veljače 2010. godine pod nazivom Windows Azure. 25. ožujka 2014. godine mijenja ime u Microsoft Azure. Na slici 4.9 je prikazana arhitektura web aplikacije koja koristi Azure App Service i Azure SQL Database.



Slika 4.9. Arhitektura web aplikacije koja koristi Microsoft Azure platformu [10]

4.3. Osnove podatkovne fuzije

Podatkovna fuzija je postupak integriranja više izvora podataka radi dobivanja dosljednih, preciznih i korisnijih informacija od onog koji pruža samo jedan izvor podataka. Prema [11], kod fuzije podataka nije neophodno da podaci budu s identičnih tipova senzora. Postoji izravna fuzija, neizravna fuzija i fuzija izlaznih podataka prethodne dvije. Kod izravne fuzije koriste se podaci iz niza heterogenih i homogenih senzora i senzorskih vrijednosti u povijesti, dok kod neizravne fuzije kao izvor podataka se koristi „a priori“ znanje o okolini i ljudski unosi. U ovom diplomskom radu je korištena izravna podatkovna fuzija, koja kombinira podatke sa senzora temperature i intenziteta svjetlosti. Podatkovna fuzija se koristi u slučaju da se ne dobije vrijednost trenutne temperature sa senzora, te se pomoću podatkovne fuzije na temelju podataka iz baze podataka donosi odluka treba li uključiti ili isključiti grijanje ili ventilaciju.

4.3.1. Izračun uvjetne vjerojatnosti

U tablici 4.1 nalaze se ulazni podaci potrebni za primjenu podatkovne fuzije na osnovu kojih se računaju uvjetne vjerojatnosti za nisku temperaturu, dobru temperaturu i visoku temperaturu.

Tablica 4.1. Primjer podataka za podatkovnu fuziju

Temperatura (°C)	Intenzitet svjetlosti (lux)	Svjetlo uključeno
20	3200	Ne
22	3500	Ne
19	2800	Ne
21	3400	Ne
23	4200	Ne
23	4600	Da
25	5100	Da

Zadana optimalna temperatura je od 22°C do 24°C, a optimalni intenzitet svjetlosti je od 3000 luxa do 4500 luxa. Ulazni podaci su: intenzitet svjetlosti 3100 luxa i dodatno svjetlo je isključeno. Na osnovu zadanih vrijednosti i ulaznih podataka možemo formirati novu tablicu sa svim potrebnim podacima za primjenu Bayesovog teorema za uvjetnu vjerojatnost. Podaci su prikazani u tablici 4.2. Prema [13], izraz za uvjetnu vjerojatnost je:

$$P(A|B) = \frac{P(A \cap B)}{P(B)} \tag{4-1}$$

gdje je:

- P – vjerojatnost,
- A i B događaji.

Tablica 4.2. Proširena tablica podataka za podatkovnu fuziju

Temperatura	Intenzitet svjetlosti	Svjetlo uključeno	Razina temperature	Razina svjetlosti
20	3200	Ne	Niska	U redu
22	3500	Ne	U redu	U redu
19	2800	Ne	Niska	Niska
21	3400	Ne	Niska	U redu
25	4200	Ne	Visoka	U redu
23	4600	Da	U redu	Visoka
25	5100	Da	Visoka	Visoka

Uvrštavanjem vrijednosti u jednadžbu (4-1) dobiju se sljedeći izrazi:

Vjerojatnost da je temperatura niska:

$$P(Tn|\overline{Su}So) = \frac{P(Tn \cap \overline{Su} \cap So)}{P(\overline{Su} \cap So)} = \frac{|Tn \cap \overline{Su} \cap So|}{|\overline{Su} \cap So|} = \frac{2}{4} = 0.5$$

gdje je:

T_n - Temperatura niska

T_o - Temperatura u redu

T_v - Temperatura visoka

S_u - Svjetlo uključeno

S_o - Intenzitet svjetlosti u redu

Vjerojatnost da je temperatura u redu:

$$P(To|\overline{Su}So) = \frac{P(To \cap \overline{Su} \cap So)}{P(\overline{Su} \cap So)} = \frac{|To \cap \overline{Su} \cap So|}{|\overline{Su} \cap So|} = \frac{1}{4} = 0.25$$

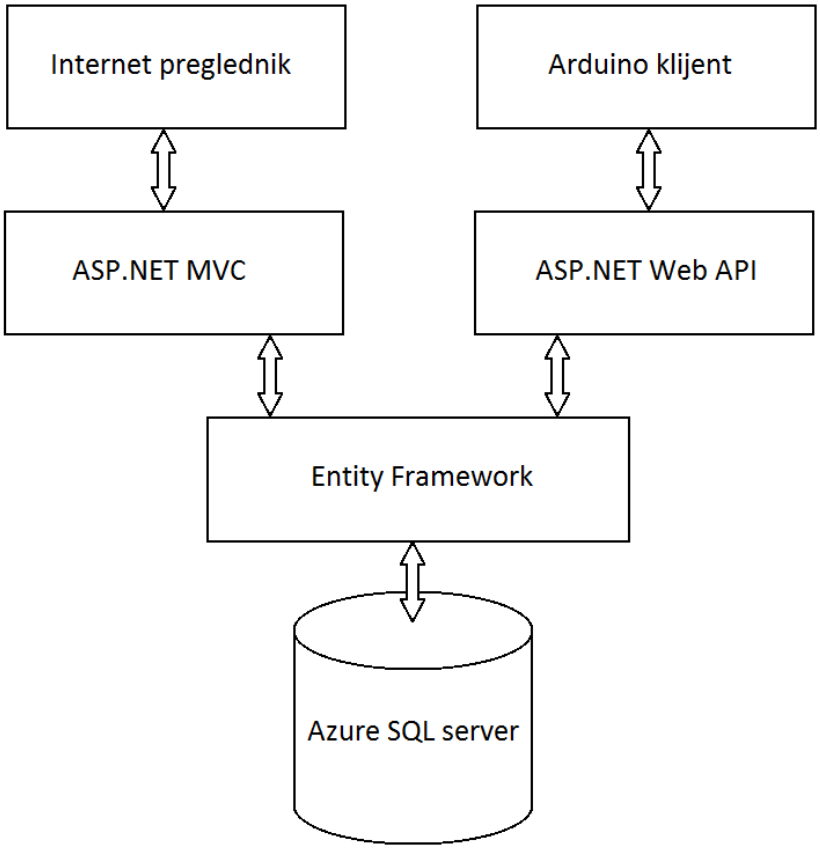
Vjerojatnost da je temperatura visoka:

$$P(Tv|\overline{Su}So) = \frac{P(Tv \cap \overline{Su} \cap So)}{P(\overline{Su} \cap So)} = \frac{|Tv \cap \overline{Su} \cap So|}{|\overline{Su} \cap So|} = \frac{1}{4} = 0.25$$

Prema dobivenim rezultatima vidljivo je da je za zadane vrijednosti i ulazne podatke najveća vjerojatnost da je temperatura niska, te će se prema tom rezultatu djelovati na sustav tako da će se uključiti grijanje.

5. PROGRAMSKO RJEŠENJE SUSTAVA

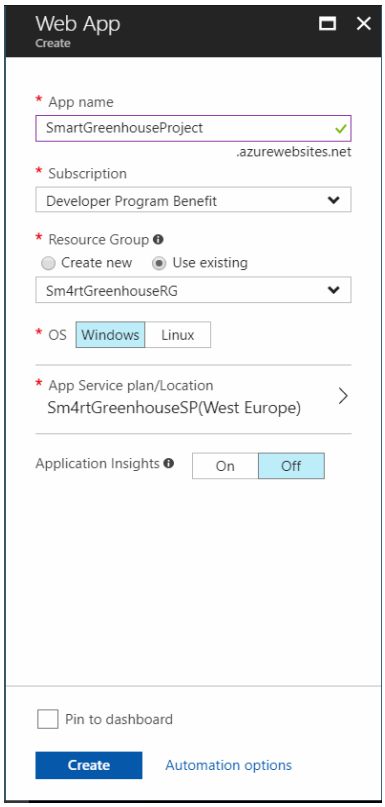
Sustav se sastoji od web aplikacije i Arduino uređaja za očitanje trenutnih vrijednosti i upravljanje izvršnim uređajima. Web aplikacija s bazom podataka komunicira preko ASP.NET MVC upravljača, dok je za komunikaciju Arduina s bazom podataka zadužen ASP.NET Web API. Na slici 5.1 je prikazana struktura sustava.



Slika 5.1. Struktura sustava

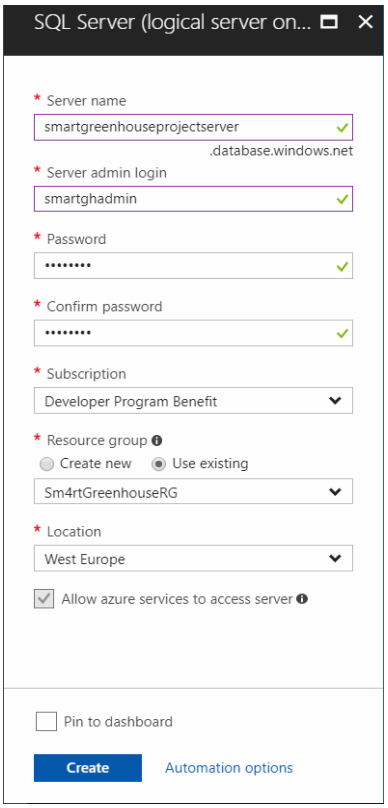
5.1. Baza podataka i oblak računala

Baza podataka je smještena u oblaku računala. Kod kreiranja baze podataka pomoću Azure-a potrebno je prvo unutar Azure portala kreirati *App servis*, *SQL poslužitelj* i tek onda se može napraviti *SQL baza podataka*. Na slici 5.2 je prikazan prvi korak, odnosno kreiranje *App servisa*.



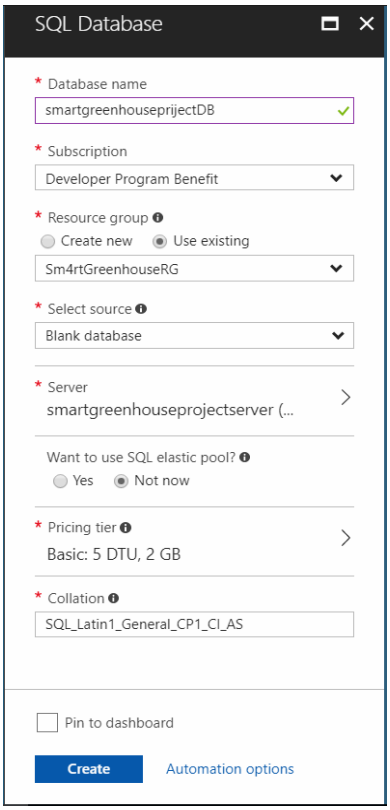
Slika 5.2. Kreiranje *App servisa*

Drugi korak je kreirati *SQL poslužitelj* (slika 5.3) na kojem će se poslije kreirati baza podataka.



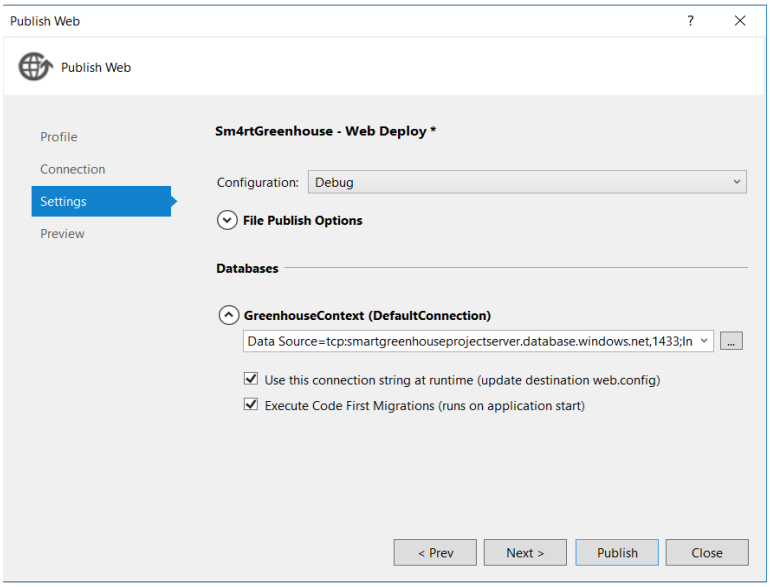
Slika 5.3. Kreiranje *SQL poslužitelja*

Zadnji korak je kreiranje baze podataka (slika 5.4).



Slika 5.4. Kreiranje *SQL baze podataka*

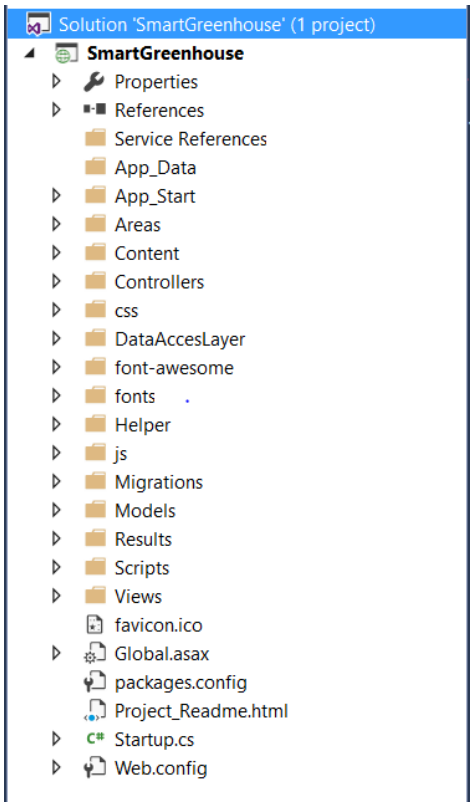
Kada je baza podataka spremna, prilikom objavljivanja aplikacije potrebno je kopirati *connection string* na bazu podataka i zalijepiti ga na predviđeno mjesto unutar alata za objavljivanje u Visual Studio 2015 (slika 5.5).



Slika 5.5. Visual Studio 2015 alat za objavljivanje aplikacije

5.2. Struktura aplikacije

Aplikacija je strukturirana prema MVC uzorku i sastoji se od modela, upravljača i pogleda. Sastoji se od šest modela, tri upravljača i tri pogleda. Povezana je na bazu podataka u koju se pohranjuju očitane trenutne vrijednosti, kulture s pripadajućim optimalnim vrijednostima, te kompozicija plastenika (kulture koje se uzgajaju u plasteniku). Na slici 5.6 je prikazana struktura projekta u razvojnom okruženju Visual Studio 2015.



Slika 5.6. Struktura projekta

5.3. Modeli

Model je klasa s atributima koje će tablica u bazi podataka sadržavati. S obzirom da je korišten pristup *code first*, prvo se definiraju modeli, koji zapravo predstavljaju tablice u bazi podataka. U programskom kodu 5.1 je prikazan model *Vegetable* koji sadrži sve atribute koje će tablica *Vegetables* imati.

```
public class Vegetable
{
    0 references
    public int ID { get; set; }
    0 references
    public string VegetableName { get; set; }
    0 references
    public string VegetableType { get; set; }
    1 reference
    public int OptimalTemperatureMax { get; set; }
    1 reference
    public int OptimalTemperatureMin { get; set; }
    1 reference
    public int OptimalHumidityMax { get; set; }
    1 reference
    public int OptimalHumidityMin { get; set; }
    1 reference
    public int OptimalSoilMoistureMax { get; set; }
    1 reference
    public int OptimalSoilMoistureMin { get; set; }
    1 reference
    public int OptimalLuminosityMax { get; set; }
    1 reference
    public int OptimalLuminosityMin { get; set; }
}
```

Programski kod 5.1. *Vegetable* model

Atributi *OptimalTemperatureMin*, *OptimalTemperatureMax*, *OptimalHumidityMin*, *OptimalHumidityMax*, *OptimalSoilMoisureMin*, *OptimalSoilMoisureMax*, *OptimalLuminosityMin* i *OptimalLuminosityMax* su tipa int (cijelobrojni tip podatka) i predstavljaju optimalne vrijednosti za određenu kulturu. Atributi *VegetableName* i *VegetableType* su tipa string (znakovni tip podatka) i odnose se na naziv i sortu kulture. U programskom kodu 5.2 je prikazan model *Greenhouse* koji je zapravo povezna tablica i sadrži poveznicu na tablicu *Vegetables*.

```
1   namespace SmartGreenhouse.Models
2   {
3       using System;
4       using System.Collections;
5       using System.Collections.Generic;
6       using System.ComponentModel.DataAnnotations.Schema;
7
        5 references
8       public class Greenhouse
9       {
            0 references
10          public int ID { get; set; }
11
12          [ForeignKey("Vegetable")]
            1 reference
13          public int VegetableId { get; set; }
            0 references
14          public virtual Vegetable Vegetable { get; set; }
15
16      }
17  }
```

Programski kod 5.2. *Greenhouse* model

U programskom kodu 5.3 je prikazan model *RecentValues* koji predstavlja tablicu *RecentValues* u bazi podataka. U ovu tablicu se spremaju podaci očitani sa senzora, te podaci o trenutnim postavkama izvršnih uređaja zajedno s točnim vremenom kada je očitanje sa senzora napravljeno. Ovi podaci se koriste za prikaz trenutnih stanja na kontrolnoj ploči te za prikaz povijesti u odabranom vremenskom intervalu.

```
public class RecentValue
{
    1 reference
    public int ID { get; set; }
    12 references
    public int Temperature { get; set; }
    1 reference
    public int Humidity { get; set; }
    2 references
    public int SoilMoisture { get; set; }
    24 references
    public int Luminosity { get; set; }
    7 references
    public bool Heating { get; set; }
    7 references
    public bool Ventilation { get; set; }
    11 references
    public bool Light { get; set; }
    2 references
    public bool Watering { get; set; }
    9 references
    public DateTime InsertDate { get; set; }
}
```

Programski kod 5.3. *RecentValues* model

Model *DataFusionResults* (programski kod 5.4) sadrži atribute *TemperatureLow, TemperatureOk* i *TemperatureHigh*. Ovaj model predstavlja podatke o rezultatu podatkovne fuzije koji se spremaju u bazu podataka.

```
public class DataFusionResults
{
    2 references
    public int ID { get; set; }
    6 references
    public double TemperatureLow { get; set; }
    6 references
    public double TemperatureOk { get; set; }
    6 references
    public double TemperatureHigh { get; set; }
}
```

Programski kod 5.4. *DataFusionResults* model

5.4. Upravljači

Aplikacija se sastoji od upravljača: *HomeController*, *GreenhouseController*, *VegetablesController* i *ArduinoController*. Upravljač *HomeControler* sadrži metodu *GetRecentValues()* koja vraća podatke za crtanje grafova na početnoj stranici. Kao parametar prima vrijednost koja označava o kojoj veličini se radi. Postoje četiri slučaja, te ovisno o parametru kojeg metoda prima dohvaća iz baze zadnjih 10 vrijednosti koje će se prikazati na grafu. U programskom kodu 5.5 je prikazan slučaj za dohvaćanje temperature. Na isti način se dohvaćaju i vrijednosti za ostala tri parametra. Ova metoda vraća listu objekata tipa *GraphData* koji sadrži atribute *Label* i *Value*. Atribut *Label* sadrži podatak o vremenu očitanja senzora a atribut *Value* sadrži podatak o vrijednosti očitane veličine.

```
public ActionResult GetRecentValues(string value)
{
    try
    {
        var data = Helper.Helper.GetGraphData(value, db);
        return Json(data, JsonRequestBehavior.AllowGet);
    }
    catch (Exception ex)
    {
        return View("Index");
    }
}
```

Programski kod 5.5. Metoda *GetRecentValues()*

Upravljač *GreenhouseController* sadrži metode: *Greenhouse()* koja vraća model pogleda *WrapperModel* koji sadrži modele *Greenhouse* i *Vegetable* i pogled na *GreenhouseView*. U programskom kodu 5.6 je prikazana metoda *Greenhouse()*. Metoda se poziva dolaskom na stranicu „Plastenik“ gdje se prikazuju povrtne kulture koje se nalaze u plasteniku.

```
public ActionResult Greenhouse()
{
    WrapperModel wrapperModel = new WrapperModel();
    ViewBag.Title = "Greenhouse Page";
    wrapperModel.Greenhouse = db.Greenhouse.ToList();
    wrapperModel.Vegetables = db.Vegetables.ToList();

    return View("Greenhouse", wrapperModel);
}
```

Programski kod 5.6. Metoda *Greenhouse()*

Metoda *AddVegetable()* kao parametar prima ID kulture koja se dodaje u plastenik i preusmjerava na metodu *Greenhouse()* što je prikazano u programskom kodu 5.7. Metoda se poziva nakon odabira kulture koja se želi dodati u plastenik iz padajućeg izbornika klikom na gumb *Dodaj* na

stranici „Plastenik“. Nakon dodavanja kulture u plastenik, vrši se izračun optimalnih vrijednosti i sprema se u tablicu *OptimalValues.*

```
public ActionResult AddVegetable(int id)
{
    try
    {
        Greenhouse greenhouse = new Greenhouse();
        greenhouse.VegetableId = id;
        db.Greenhouse.Add(greenhouse);
        db.SaveChanges();
        var optimalValues = Helper.Helper.GetOptimalValues(db);
        db.OptimalValues.Add(optimalValues);
        db.SaveChanges();
    }
    catch (DataException)
    {
        return RedirectToAction("AddVegetable", new { id = id, saveChangesError = true });
    }
    return RedirectToAction("Greenhouse");
}
```

Programski kod 5.7. Metoda *AddVegetable()*

Metoda *Delete()* kao parametar prima ID kulture koju je potrebno maknuti iz plastenika i također nakon brisanja preusmjerava na metodu *Greenhouse()* (programski kod 5.8). Poziva se klikom na gumb *Obriši* koji se nalazi unutar kućice koja označava pojedinu povrtnu kulturu.

```
public ActionResult Delete(int id)
{
    try
    {
        Greenhouse greenhouse = db.Greenhouse.Find(id);
        db.Greenhouse.Remove(greenhouse);
        db.SaveChanges();
    }
    catch (DataException)
    {
        return RedirectToAction("Delete", new { id = id, saveChangesError = true });
    }
    return RedirectToAction("Greenhouse");
}
```

Programski kod 5.8. Metoda *Delete()*

Upravljač *VegetablesController* sadrži tri metode:

Metoda *Vegetables()* vraća model *Vegetable* i pogled na *VegetablesView* (programski kod 5.9).

```
public ActionResult Vegetables()
{
    ViewBag.Title = "Vegetables Page";

    return View("Vegetables", db.Vegetables);
}
```

Programski kod 5.9. Metoda *Vegetables()*

Metoda *Create()* kao parametar prima model *Vegetable* koji se sprema u bazu i vraća model *Vegetable* i pogled na *VegetablesView* (programski kod 5.10).

```
public ActionResult Create(Vegetable vegetable)
{
    try
    {
        if (ModelState.IsValid)
        {
            db.Vegetables.Add(vegetable);
            db.SaveChanges();
        }
    }
    catch (DataException)
    {
        ModelState.AddModelError("", "Unable to save changes.");
    }
    return View("Vegetables", db.Vegetables.ToList());
}
```

Programski kod 5.10. Metoda *Create()*

Metoda *Delete()* isto kao i u *Greenhouse* upravljaču kao parametar prima ID kulture i nakon brisanja iz baze preusmjerava na metodu *Vegetables()*.

Upravljač *ArduinoController* sadrži HTTP post metodu *SaveRecentValues()* koja kao parametar prima model *RecentValue* i nakon provjere treba li raditi fuziju podataka sprema podatke u bazu i vraća znakovni niz nula i jedinica koji označava koji izvršni uređaj treba upaliti, odnosno ugasiti (programski kod 5.11).

```
public IHttpActionResult SaveRecentValues(int temperature, int humidity, int soilMoisture, int luminosity)
{
    try
    {
        RecentValue values = new RecentValue();
        values.Temperature = temperature;
        values.Humidity = humidity;
        values.SoilMoisture = soilMoisture;
        values.Luminosity = luminosity;
        string result = string.Empty;
        RecentValue recentValues = new RecentValue();
        var isDatafusion = db.DataFusionStatus.FirstOrDefault(d => d.ID == 1).Enabled;
        if (isDatafusion)
        {
            var fusionResult = Helper.Helper.GetDataFusionResults(db, values);
            db.DataFusionResults.Add(fusionResult);
            db.SaveChanges();
        }
        recentValues = Helper.Helper.SetOutputStates(db, values, isDatafusion);
        recentValues.InsertDate = DateTime.Now;
        db.RecentValues.Add(recentValues);
        db.SaveChanges();
        result += (recentValues.Heating ? "1" : "0") + ",";
        result += (recentValues.Ventilation ? "1" : "0") + ",";
        result += (recentValues.Watering ? "1" : "0") + ",";
        result += recentValues.Light ? "1" : "0";
        return Ok(result);
    }
    catch (Exception ex)
    {
        return BadRequest(ex.ToString());
    }
}
```

Programski kod 5.11. Metoda *SaveRecentValues()*

5.5. Pogledi

Aplikacija se sastoji od pogleda: *Index, Greenhouse, Vegetables.* Izgled aplikacije je definiran u pogledu *_Layout.cshtml* u kojemu je postavljen okvir s pripadajućim izbornicima. Sve ono što se prikazuje u glavnom prozoru aplikacije se nalazi u jednom od tih pogleda. U programskom kodu 5.12 je prikazan najvažniji dio *_Layout.cshtml* u kojemu su definirani izbornici s pripadajućim akcijama koje se obavljaju kod klika na određeni element u izborniku.

```
35  <body>
36      <div id="page-wrapper">
37          <div id="wrapper">
38              <!-- Navigation -->
39              <nav class="navbar navbar-inverse navbar-fixed-top" role="navigation">
40                  <!-- Brand and toggle get grouped for better mobile display -->
41                  <div class="navbar-header">
42                      <button type="button" class="navbar-toggle" data-toggle="collapse" data-target=".navbar-ex1-collapse">
43                          <span class="sr-only">Toggle navigation</span>
44                          <span class="icon-bar"></span>
45                          <span class="icon-bar"></span>
46                          <span class="icon-bar"></span>
47                      </button>
48                      <a class="navbar-brand" href="@Url.Action("Index", "Home")">Upravljanje plastenikom</a>
49                  </div>
50                  <!-- Sidebar Menu Items - These collapse to the responsive navigation menu on small screens -->
51                  <div class="collapse navbar-collapse navbar-ex1-collapse">
52                      <ul class="nav navbar-nav side-nav">
53                          <li id="IndexNav">
54                              <a href="@Url.Action("Index", "Home")"><i class="fa fa-fw fa-bar-chart-o"></i> Kontrolna ploča</a>
55                          </li>
56                          <li id="GreenhouseNav">
57                              <a href="@Url.Action("Greenhouse", "Greenhouse")"><i class="fa fa-fw fa-desktop"></i> Plastenik</a>
58                          </li>
59                          <li id="VegetablesNav">
60                              <a href="@Url.Action("Vegetables", "Vegetables")"><i class="fa fa-fw fa-table"></i> Povrtne kulture</a>
61                          </li>
62                      </ul>
63                  </div>
64                  <!-- /.navbar-collapse -->
65              </nav>
66              @RenderBody()
67          </div>
68      </div>
```

Programski kod 5.12. Pogled *_Layout.cshtml*

Unutar *_Layout.cshtml* se nalazi i Javascript kod unutar kojeg se pomoću jQuery-a obavljaju različite akcije prilikom pritiskanja gumbova unutar cijele aplikacije.

U pogledu *Index* su definirane kućice za grafove i prikaz trenutnih vrijednosti. U programskom kodu 5.13 je prikazan primjer za temperaturu, ali i za ostale veličine kućice se definiraju na jednak način.

```
13      <div class="row">
14          <div class="col-lg-4">
15              <div class="panel panel-default">
16                  <div class="panel-heading">
17                      <h3 class="panel-title"><i class="fa fa-bar-chart-o fa-fw"></i> Temperatura</h3>
18                  </div>
19                  <div class="panel-body">
20                      <div id="temperature"></div>
21                  </div>
22              </div>
23          </div>
24          <div class="col-lg-2">
25              <div class="panel panel-default">
26                  <div class="panel-heading">
27                      <h3 class="panel-title">Trenutna temperatura</h3>
28                  </div>
29                  <div class="panel-body">
30                      <span id="temperatureSpan"></span>
31                  </div>
32              </div>
33          </div>
```

Programski kod 5.13. Pogled *Index*

Kao što je vidljivo u kodu, kućica u kojoj se nalazi graf za prikaz temperature ima *id* atribut postavljen na „temperature“ koji se u javascript kodu za crtanje grafa za temperaturu postavlja kao referenti element.

U pogledu *Greenhouse* se nalazi prikaz poljoprivrednih kultura koje se nalaze u plasteniku. Za svaku kulturu se formira kućica s osnovnim podacima o kulturi i gumbom za brisanje, odnosno za izbacivanje kulture iz plastenika. Karakteristični dio koda koji za svaku kulturu u modelu dodaje novu kućicu je prikazana u programskom kodu 5.14.

```
<div class="row">
    @foreach (var item in Model.Greenhouse)
    {
        <div class="col-lg-4">
            <div class="panel panel-default">
                <div class="panel-heading">
                    <h3 class="panel-title">@item.Vegetable.VegetableName</h3>
                </div>
                <div class="panel-body">
                    @item.Vegetable.VegetableType
                    <div id="@("btnDeleteFromGreenhouse"+ item.ID)" class="btn btn-default" data-id="@item.ID">
                        Obriši
                    </div>
                </div>
            </div>
        </div>
    }
```

Programski kod 5.14. Pogled *Greenhouse*

U pogledu *Vegetables* nalazi se tablica svih poljoprivrednih kultura koje su unesene u bazu podataka, te sučelje za unos nove kulture. U programskom kodu 5.15 prikazan je karakteristični dio koda koji za svaku kulturu u bazi podataka kreira novi red u tablici, te ga popunjava odgovarajućim vrijednostima.

```
@foreach (var item in Model)
{
    <tr>
        <td>@item.VegetableName</td>
        <td>@item.VegetableType</td>
        <td>@item.OptimalTemperatureMin</td>
        <td>@item.OptimalTemperatureMax</td>
        <td>@item.OptimalHumidityMin</td>
        <td>@item.OptimalHumidityMax</td>
        <td>@item.OptimalSoilMoistureMin</td>
        <td>@item.OptimalSoilMoistureMax</td>
        <td>@item.OptimalLuminosityMin</td>
        <td>@item.OptimalLuminosityMax</td>
        <td>
            <div id="@("btnDeleteVegetable"+ item.ID)" class="btn btn-default" data-id="@item.ID">
                Obriši
            </div>
        </td>
    </tr>
}
```

Programski kod 5.15. Pogled *Vegetables*

5.6. Implementacija podatkovne fuzije

Podatkovna fuzija je implementirana u metodi *GetDataFusionResults()* koja kao parametre prima DB Context i model *RecentValue* koji predstavlja trenutne očitane vrijednosti. Prvo se trenutna temperatura postavlja na vrijednost -100, da bi se dalje u kodu mogao razlikovati podatak koji ne sadrži trenutnu temperaturu od spremljene niske temperature, te se dohvaćaju optimalne vrijednosti i zadnje vrijednosti iz tablice *RecentValues* zbog podatka o trenutnim postavkama izvršnih uređaja u plasteniku (programski kod 5.16).

```
values.Temperature = -100;
var optimalValues = GetOptimalValuesForDisplay(db);
var recentValuesId = db.RecentValues.Select(x => x.ID).Max();
var recentValues = db.RecentValues.Find(recentValuesId);
```

Programski kod 5.16. Dohvaćanje podataka trenutnih i optimalnih vrijednosti

Nakon toga se postavlja status intenziteta svjetlosti (programski kod 5.17). Vrijednost -1 predstavlja nedovoljan intenzitet, 0 predstavlja dobar intenzitet i 1 predstavlja previsok intenzitet svjetlosti.

```
int luminosityStatus = 0;
if (values.Luminosity < optimalValues.OptimalLuminosityMin)
{
    luminosityStatus = -1;
}
else if (values.Luminosity >= optimalValues.OptimalLuminosityMin && values.Luminosity <= optimalValues.OptimalLuminosityMax)
{
    luminosityStatus = 0;
}
else if (values.Luminosity > optimalValues.OptimalLuminosityMax)
{
    luminosityStatus = 1;
}
```

Programski kod 5.17. Postavljanje statusa intenziteta svjetlosti

Varijable *temperatureLow*, *temperatureOk* i *temperatureHigh* predstavljaju broj zapisa u bazi podataka koji odgovaraju određenom uvjetu za pojedini slučaj. U programskom kodu 5.18 je prikazan postupak dohvaćanja broja zapisa kada je temperatura niska za očitane vrijednosti intenziteta svjetlosti.

```
var temperatureLow = db.RecentValues
    .Where(t => t
    .Temperature < optimalValues.OptimalTemperatureMin && t
    .Temperature > -100 && (luminosityStatus == -1 ? t
    .Luminosity < optimalValues.OptimalLuminosityMin : luminosityStatus == 0 ? t
    .Luminosity >= optimalValues.OptimalLuminosityMin && t
    .Luminosity <= optimalValues.OptimalLuminosityMax : t
    .Luminosity > optimalValues.OptimalLuminosityMax) && t
    .Light == recentValues.Light)
    .ToList().Count;
```

Programski kod 5.18. Dohvaćanje broja zapisa u slučaju da je temperatura niska

Za ostale slučajeve se na isti način dohvaća broj zapisa uz promjenu uvjeta za temperaturu. Nakon što se dobije broj zapisa za pojedine slučajeve, potrebno je dohvatiti ukupan broj slučaja za zadani intenzitet svjetlosti (programski kod 5.19).

```
var valuesTotalCount = db.RecentValues
    .Where(t => (luminosityStatus == -1 ? t
    .Luminosity < optimalValues.OptimalLuminosityMin : luminosityStatus == 0 ? t
    .Luminosity >= optimalValues.OptimalLuminosityMin && t
    .Luminosity <= optimalValues.OptimalLuminosityMax : t
    .Luminosity > optimalValues.OptimalLuminosityMax) && t
    .Temperature > -100 && t
    .Light == recentValues.Light)
    .ToList().Count;
```

Programski kod 5.19. Dohvaćanje ukupnog broja zapisa

Na kraju je potrebno izračunati vjerojatnosti za svaki slučaj (programski kod 5.20). Rezultat se zaokružuje na dva decimalna mjesta.

```
if (valuesTotalCount > 0)
{
    result.TemperatureLow = Math.Round(Convert.ToDouble(temperatureLow) / Convert.ToDouble(valuesTotalCount), 2);
    result.TemperatureOk = Math.Round(Convert.ToDouble(temperatureOk) / Convert.ToDouble(valuesTotalCount), 2);
    result.TemperatureHigh = Math.Round(Convert.ToDouble(temperatureHigh) / Convert.ToDouble(valuesTotalCount), 2);
}
```

Programski kod 5.20. Konačni izračun vjerojatnosti

5.7. Programsko rješenje klijenta na Arduinu

Prilikom programiranja klijenta na Arduinu korištene su biblioteke *SPI.h* i *Ethernet.h* za povezivanje mrežnog modula s Arduinom i mrežnu komunikaciju, *DHT.h* i *Wire.h* za senzor temperature i relativne vlažnosti zraka, te *BH1750.h* za senzor intenziteta svjetlosti. Prvo je bilo potrebno podesiti IP adrese poslužitelja i klijenta i MAC adresu klijenta (programski kod 5.21).

```
byte mac[] = { 0xDE, 0xAD, 0xBE, 0xEF, 0xFE, 0xED };
IPAddress server(192, 168, 1, 2); // numeric IP for server (no DNS)

// Set the static IP address to use if the DHCP fails to assign
IPAddress ip(192, 168, 1, 4);
```

Programski kod 5.21. Podešavanje IP adresa poslužitelja i klijenta i MAC adrese klijenta

U funkciji *setup()* definirani su izlazni priključci za releje grijanja, ventilacije, zalijevanja i svjetla (programski kod 5.22). S obzirom da se sklopka releja aktivira na logičku nulu, svi izlazi su početno postavljeni na logičku jedinicu.

```
void setup() {
  pinMode(5, OUTPUT); //pin za grijanje
  pinMode(6, OUTPUT); //pin za ventilaciju
  pinMode(2, OUTPUT); //pin za zalijevanje
  pinMode(3, OUTPUT); //pin za svjetlo
  digitalWrite(5, HIGH);
  digitalWrite(6, HIGH);
  digitalWrite(2, HIGH);
  digitalWrite(3, HIGH);
```

Programski kod 5.22. Postavljanje izlaznih priključaka

U funkciji *loop()* svakih 5 sekundi vrši se očitavanje vrijednosti sa senzora (programski kod 5.23) i spremaju se u varijable: *h* za relativnu vlažnost zraka, *t* za temeperaturu, *l* za intenzitet svjetlosti i *s* za vlažnost zemlje.

```
int h = dht.readHumidity();
int t = dht.readTemperature();
int l = luxometar.readLightLevel();
int s = analogRead(SMPIN);
```

Programski kod 5.23. Očitavanje vrijednosti sa senzora

Ovako dobivene vrijednosti su potrebne za formiranje HTTP GET upita prema poslužitelju. U programskom kodu 5.24 je prikazan postupak formiranja *URL-a* koji služi za slanje upita prema poslužitelju.

```
value += temperature;
value += "/";
value += humidity;
value += "/";
value += soilMoisture;
value += "/";
value += luminosity;
String url = "";
url += "GET /api/saveRecentValues/";
url += value;
url += " HTTP/1.1";
```

Programski kod 2.24. Postupak formiranja *URL-a*

S ovako dobijenim *URL*-om vrši se slanje upita prema poslužitelju (programski kod 2.25), te se čeka odgovor. Kada se dobije odgovor s poslužitelja, rezultat se raščlanjuje na dijelove i uzimaju se oni korisni iz kojih se iščitavaju vrijednosti prema kojima se upravlja izvršnim uređajima (programski kod 2.26).

```
client.println(url);
client.println("Host: 192.168.1.2");
```

Programski kod 2.25. Slanje upita prema poslužitelju

```
heating = getValue(resultString, ',', 0).toInt();
ventilation = getValue(resultString, ',', 1).toInt();
watering = getValue(resultString, ',', 2).toInt();
light = getValue(resultString, ',', 3).toInt();
```

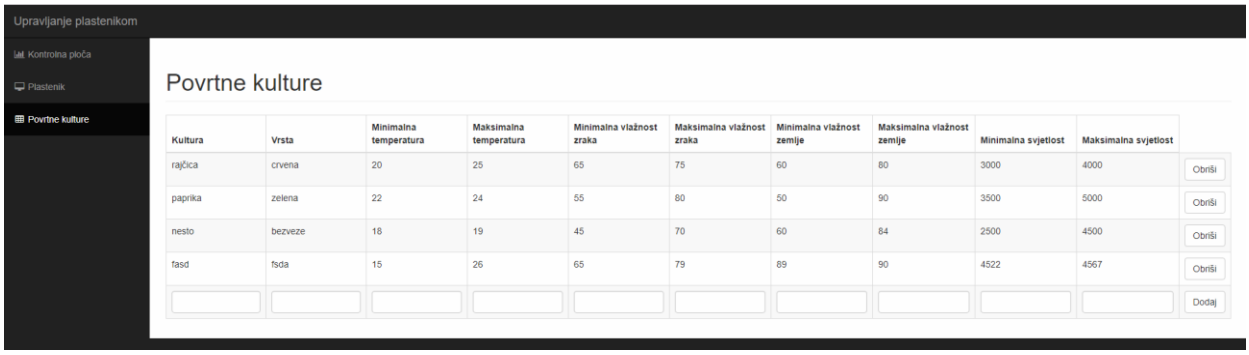
Programski kod 2.26. Raščlanjivanje dobivenog rezultata i uzimanje korisnih informacija

Pomoću ovako dobivenih vrijednosti koje su ili 1 ili 0 se pali ili gasi odgovarajući izvršni uređaj, odnosno šalje se logička nula ili logička jedinica na odgovarajući izlazni priključak.

6. PRIKAZ RADA SUSTAVA S TESTIRANJEM I ANALIZOM

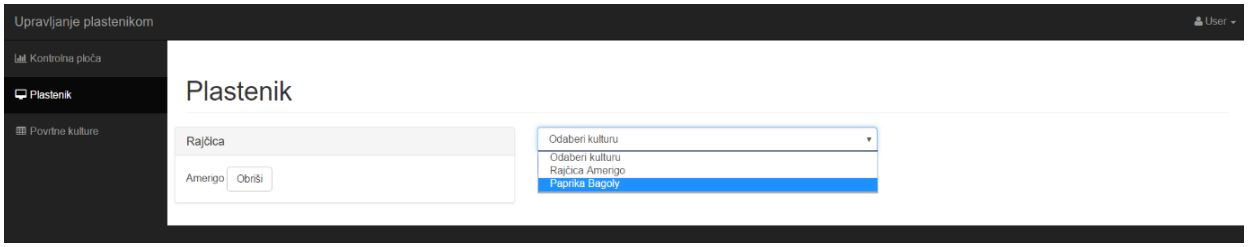
6.1. Unos podataka

Unos povrtnih kultura se vrši tako da se klikom na gumb „Dodaj“ otvara forma za unos naziva, vrste i optimalnih vrijednosti, te nakon unosa se klikne na gumb „Dodaj“, što je vidljivo na slici 6.1. Unesene povrtne kulture se prikazuju u tablici te je svaku moguće obrisati klikom na gumb „Obriši“.



Slika 6.1. Dodavanje povrtne kulture

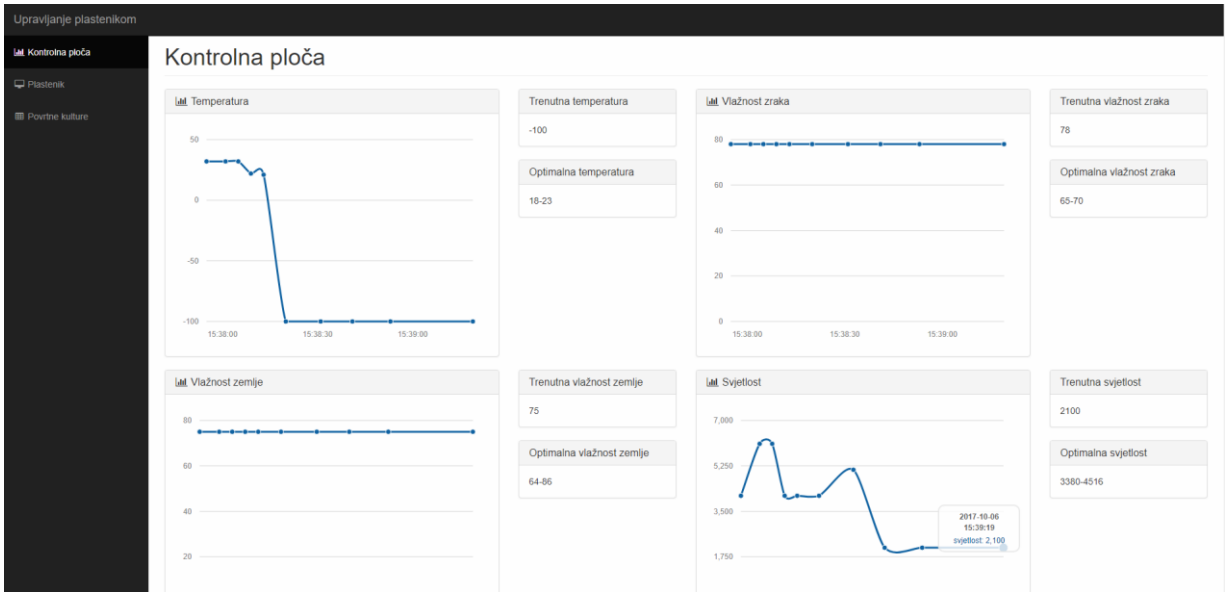
Dodavanje povrtne kulture u plastenik vrši se tako da se na stranici „Plastenik“ klikom na gumb „Dodaj“ otvori padajući izbornik u kojem se odabire željena kultura i nakon odabira potvrđuje se klikom na gumb „Dodaj“ (slika 6.2). Sve povrtne kulture koje se nalaze u plasteniku su prikazane kao kućica u kojoj se nalazi naziv kulture i vrsta. Klikom na gumb „Obriši“ kultura se može maknuti iz plastenika.



Slika 6.2. Dodavanje povrtne kulture u plastenik

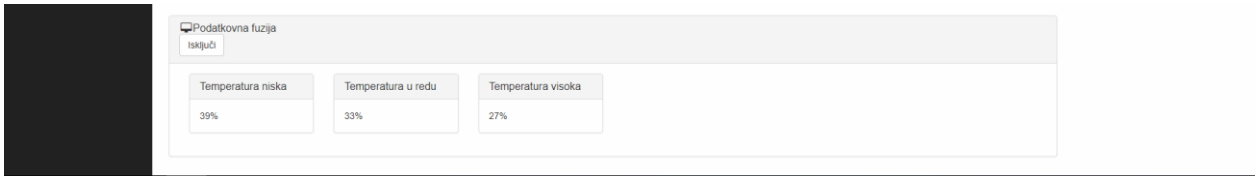
6.2. Prikaz trenutnih vrijednosti parametara sustava

Na početnoj stranici se nalaze četiri linijska grafa koji prikazuju zadnje izmjerene, te trenutnu vrijednost temperature, vlažnosti zraka, vlažnosti zemlje i svjetlosti. Kada se prijeđe strelicom miša preko grafa, ispisuju se vrijeme i vrijednost koja je bila u tom trenutku, što se vidi na slici 6.3.



Slika 6.3. Kontrolna ploča

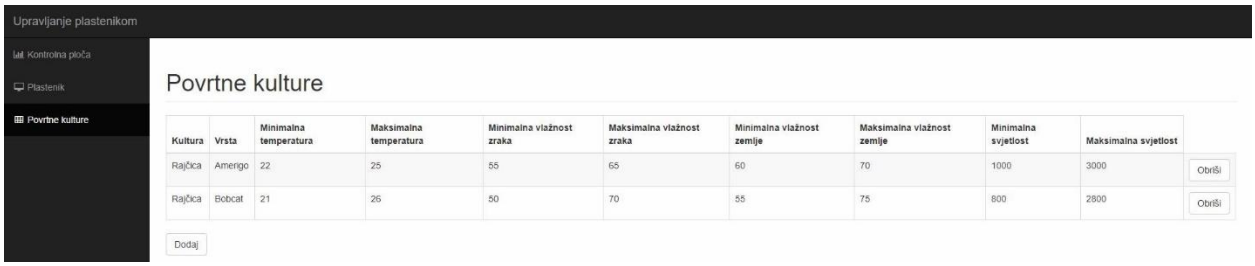
Pored grafova s desne strane, nalaze se po dvije kućice. U jednoj je podatak o trenutnoj vrijednosti, a u drugoj podatak o optimalnim vrijednostima. Na kontrolnoj ploči se nalazi i modul za podatkovnu fuziju koji se pritiskom na gumb može uključiti ili isključiti. U slučaju kada se podatkovna fuzija uključi, nakon izračuna, u odgovarajuće kućice se ispisuje podatak o vjerojatnosti da je temperatura niska, temperatura u redu ili temperatura visoka. Pritiskom na gumb za isključivanje podatkovne fuzije kućice u kojima se ispisuje rezultat nestaju i aplikacija se vraća u režim rada bez podatkovne fuzije. Na slici 6.4 je prikazan modul za podatkovnu fuziju s prikazom rezultata podatkovne fuzije kada je ona uključena.



Slika 6.4. Modul za podatkovnu fuziju

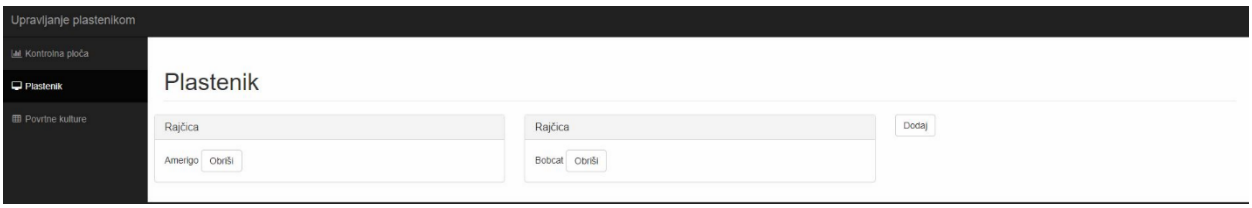
6.3. Postavke testiranja

Testiranje je vršeno u zatvorenom prostoru, pri sobnoj temperaturi od 22°C i prirodnom svjetlu. U prvom slučaju u web aplikaciji su dodane dvije kulture s optimalnim vrijednostima vidljivim na slici 6.5.



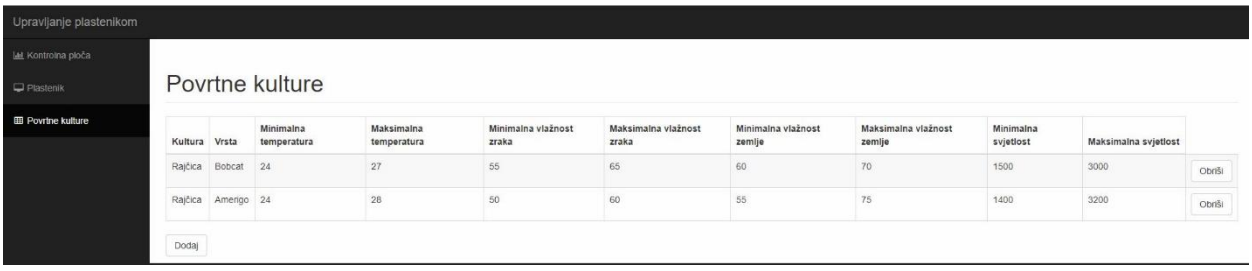
Slika 6.5. Postavke testiranja za povrtne kulture u prvom slučaju

U plastenik su dodane obje kulture (slika 6.6) te su izračunate optimalne vrijednosti bile: temperatura 22°C do 25°C, vlažnost zraka 55% do 65%, vlažnost zemlje 60% do 70% i svjetlost 1000 luxa do 2800 luxa.



Slika 6.6. Postavke testiranja za plastenik u prvom slučaju

U drugom slučaju su dodane dvije kulture s većim optimalnim vrijednostima temperature i intenziteta svjetlosti (slika 6.7). I u ovom slučaju obje kulture su dodane u plastenik te su izračunate optimalne vrijednosti iznosile: temperatura 24°C do 27°C, vlažnost zraka 55% do 60%, vlažnost zemlje 60% do 70% i svjetlost 1500 luxa do 3000 luxa.

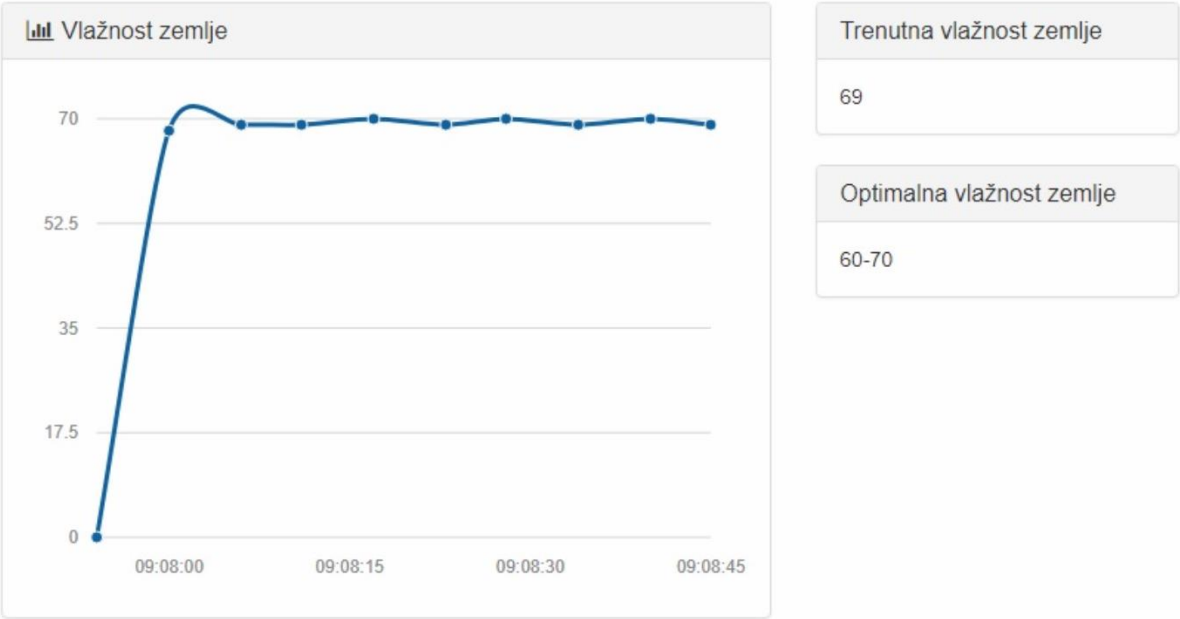


Slika 6.7. Postavke testiranja za povrtne kulture u drugom slučaju

U trećem slučaju se u plasteniku nalazila samo jedna kultura koja je imala optimalne vrijednosti: temperatura 21°C do 24°C, vlažnost zraka 50% do 60%, vlažnost zemlje 65% do 77% i svjetlost 1400 luxa do 3200 luxa.

6.4. Analiza rezultata

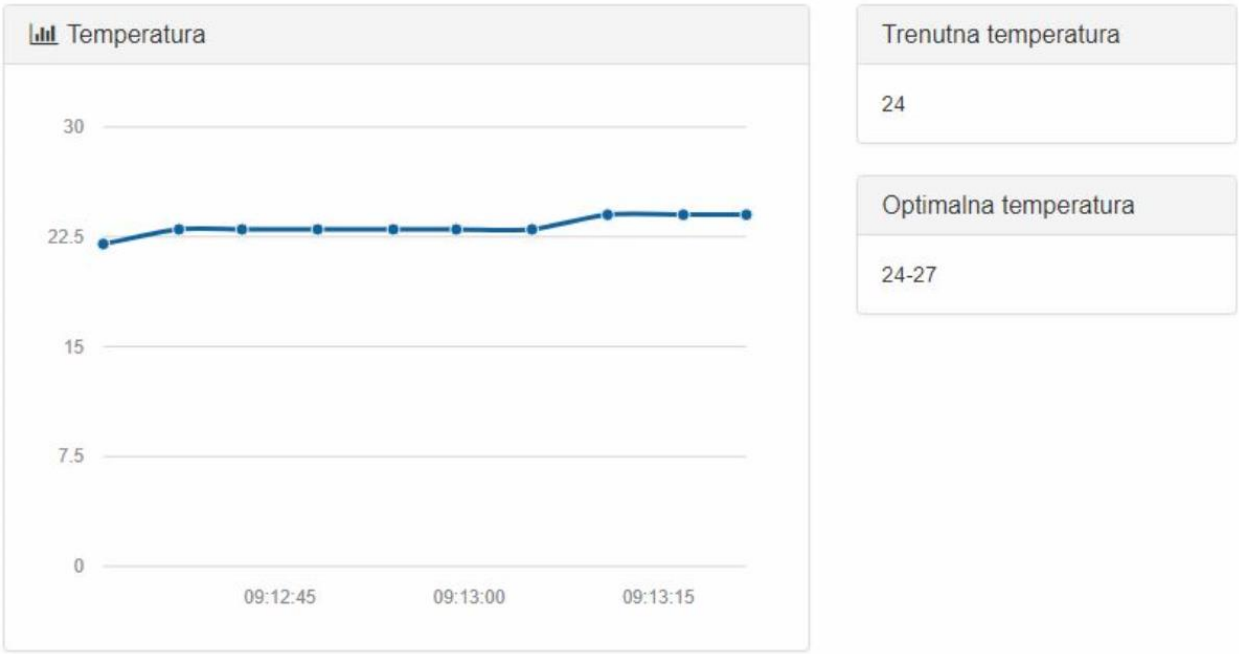
U prvom slučaju temperatura je bila jednaka sobnoj temperaturi koja je iznosila 22°C, te nije bilo promjena temperature i djelovanja na izvršne uređaje grijanja i ventilacije. Intenzitet svjetlosti je bio manji od optimalne vrijednosti čak i nakon što je dodatno svjetlo upaljeno, te je svjetlo ostalo upaljeno do kraja testiranja. S obzirom da je prvi put testirano, vlažnost zemlje prije testiranja je bila 0%. Odmah kod prvog mjerenja bilo je potrebno izvršiti regulaciju vlažnosti zemlje te je pumpa za navodnjavanje uključena. Promjena vlažnosti zemlje je vidljiva na slici 6.8.



Slika 6.8. Promjena vlažnosti zemlje prilikom regulacije

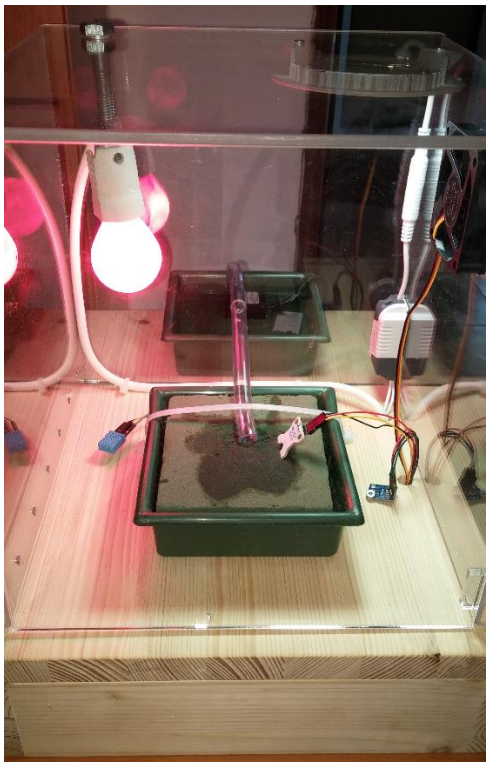
Drugi slučaj se razlikuje u optimalnim vrijednostima i trenutnoj vrijednosti vlažnosti zemlje koja je nakon prvog testiranja iznosila približno 70%, te se nije ni mijenjala prilikom drugog testiranja. Kao i u prvom slučaju, vrijednost intenziteta svjetlosti je bio ispod optimalnih vrijednosti, te je dodatno svjetlo bilo uključeno prilikom cijelog testiranja. Optimalna vrijednost temperature je viša od trenutne temperature u plasteniku te je bilo potrebno utjecati na uključivanje grijanja. Na slici

6.9 prikazan je graf temperature na kojemu se vidi porast trenutne vrijednosti temperature uslijed utjecaja grijača.



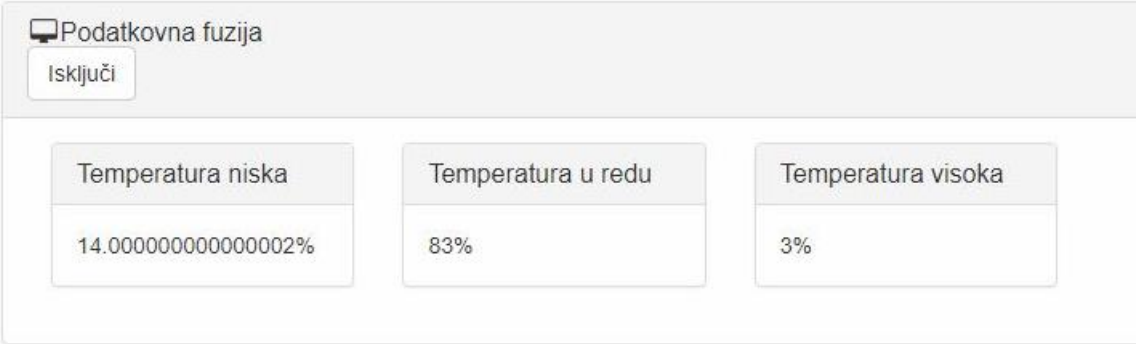
Slika 6.9. Promjena temperature uslijed djelovanja grijača

Iz grafa je vidljivo da je potrebno oko 30 sekundi da bi se temperatura podigla za 1 °C. Nakon što je temperatura dostigla optimalnu vrijednost, sustav je uspio održati temperaturu u optimalnim vrijednostima. Na slici 6.10 je prikazana maketa sustava u trenutku kada je uključen grijač i dodatno svjetlo.



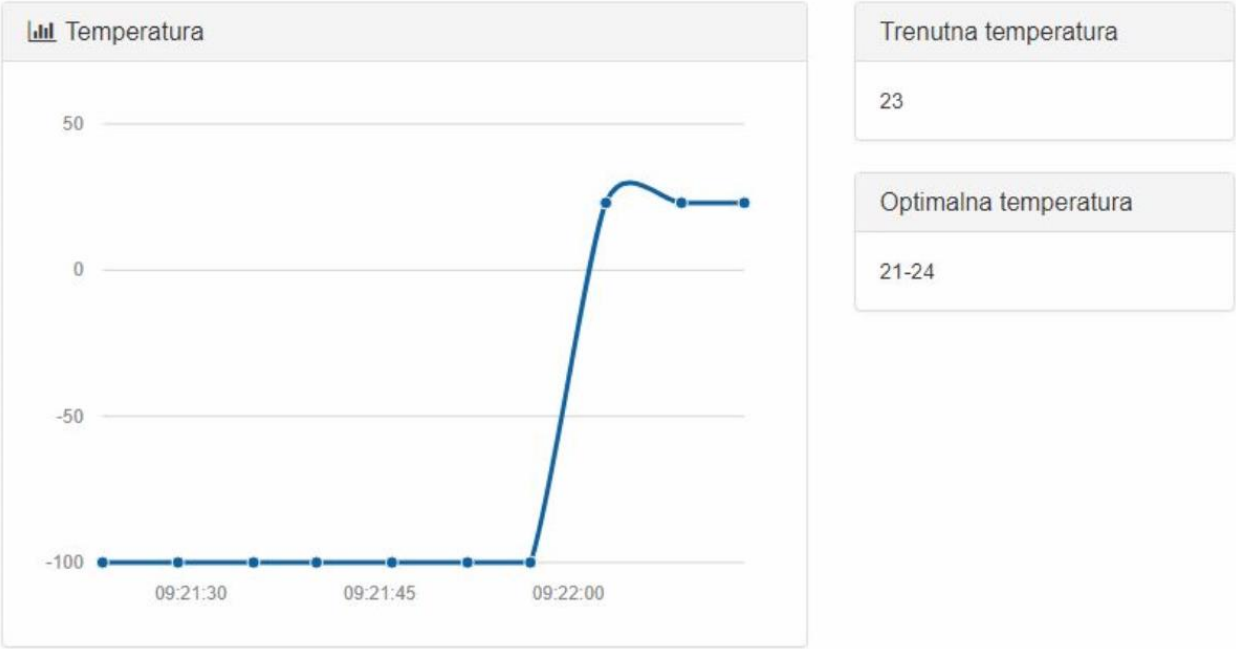
Slika 6.10. Prikaz makete sustava u trenutku kada je uključen grijač i dodatno svjetlo

U trećem slučaju sustav je testiran u režimu rada kada je uključena podatkovna fuzija. U trenutku kada je uključena podatkovna fuzija, prilikom prvog mjerenja prikazali su se rezultati podatkovne fuzije (slika 6.11).



Slika 6.11. Rezultat podatkovne fuzije

Iz rezultata podatkovne fuzije vidljivo je da je najveća vjerojatnost da je temperatura u redu, pa prema tome nije bilo utjecaja na izvršne uređaje. Nakon nekog vremena podatkovna fuzija je isključena i na grafu za temperaturu se vidi očitanje trenutne temperature sa senzora (slika 6.12), koja je u granicama optimalnih vrijednosti, što znači da je rezultat podatkovne fuzije u ovom slučaju bio točan i nije bilo potrebe utjecati na izvršne uređaje.



Slika 6.12. Vrijednost temperature nakon podatkovne fuzije

7. ZAKLJUČAK

Proizvodnja povrća je vrlo bitna grana poljoprivrede. Da bi proizvodnja bila što učinkovitija potrebno je poljoprivrednim kulturama osigurati najbolje moguće uvjete. Uzgojem povrća u zaštićenim prostorima drastično se povećava učinkovitost. Odabir pravog tipa zaštićenog prostora ovisi o potrebama proizvođača i vrsti kulture koja se uzgaja. Najpopularniji tip zaštićenog prostora je plastenik. Razlog je jednostavnost izrade, cijena i mogućnosti koje pruža. Plastenici omogućuju korištenje sustava za regulaciju mikroklime. Na primjeru postojećeg rješenja je vidljivo da se upravljanje mikroklimom izvršava pomoću ponuđenih programa ili unosom vlastitog programa, kojeg je potrebno, ovisno o fazi proizvodnje, mijenjati. Da bi se izbjeglo učestalo mijenjanje režima upravljanja potrebno je težiti pametnim sustavima, koji bi na osnovu unesenih i stečenih znanja sami odlučivali kada je potrebno utjecati na izvršne uređaje. Upravo to se postiže podatkovnom fuzijom, gdje se na osnovu podataka iz više različitih izvora dobiva korisnija i preciznija informacija nego što bi se dobilo samo s jednog izvora. Primjenom podatkovne fuzije u sustavu izrađenom u ovom diplomskom radu vidljivo je da je na osnovu podataka o intenzitetu svjetlosti dobivena točna informacija o trenutnoj temperaturi. Podatkovna fuzija daje preciznije rezultate što ima više ulaznih podataka. U ovom sustavu nije bilo mogućnosti direktnog utjecaja na relativnu vlažnost zraka, nego se ona posredno mijenjala utjecajem drugih veličina. Da bi se omogućio direktni utjecaj na relativnu vlažnost zraka potrebno je sustav opremiti odgovarajućim izvršnim uređajem. Rezultati testiranja pokazali su da se vrlo precizno može upravljati temperaturom i vlažnosti zemlje, dok je izvor svjetlosti bio preslab da bi se donio zaključak o uspješnosti upravljanja. Primjena podatkovne fuzije u sustavima za upravljanje mikroklime plastenika može čovjeku uvelike olakšati rad sa sustavima.

LITERATURA

[1] N. Parađiković, Ž. Kraljičak, Zaštićeni prostori – plastenici i staklenici, Sveučilište J.J. Strossmayera u Osijeku, Poljopriverdni fakultet u Osijeku, Osječko-baranjska županija, Osijek, 2008.

[2] D. Kantoci, Zaštićeni prostor (plastenici/staklenici), Glasnik zaštite bilja, No.5, Vol.37, str. 6 – 8, studeni 2014.

[3] Pro-Com Gutter Connect, https://www.atlasgreenhouse.com/product/pro-com-gutter-connect [2.10.2017.]

[4] J. Borošić, Plastenici: Kako uzgajati povrće u zaštićenim prostorima, veljača 2016., dostupno na: http://www.agrobiz.hr/agrosavjeti/plastenici-kako-uzgajati-povrce-u-zasticenim-prostorima-127 [2.10.2017.]

[5] Podno grijanje, http://www.gis-impro.hr/podno-grijanje [6.10.2017.]

[6] M. Popović, B. Lazić, Gajenje povrća u zaštićenom prostoru, Nolit, Beograd, 1989.

[7] Što je Arduino, a što Croduino, https://e-radionica.com/hr/blog/2015/10/08/sto-je-arduino-i-croduino [6.10.2017.]

[8] W5100 Datasheet, Version 1.1.6, WIZnet Co., 2008., dostupno na: https://www.sparkfun.com/datasheets/DevTools/Arduino/W5100_Datasheet_v1_1_6.pdf [6.10.2017.]

[9] J. Chadwick, T. Snyder, H. Panda, Programiranje ASP.NET MVC 4, Dobar Plan, Zagreb, 2013.

[10] Basic Web Application, studeni 2016., dostupno na: https://docs.microsoft.com/en-us/azure/architecture/reference-architectures/managed-web-app/basic-web-app [7.10.2017.]

[11] W. Elmenreich, Sensor Fusion in Time-Triggered Systems, Techische Universität Wien, Beč, listopad 2002., dostupno na: https://mobile.aau.at/~welmenre/papers/elmenreich_Dissertation_sensorFusionInTimeTriggeredSystems.pdf [6.10.2017.]

[12] R. R. Singh, An Introdution to Entity Framework for Absolute Beginners, travanj 2012., dostupno na: https://www.codeproject.com/Articles/363040/An-Introduction-to-Entity-Framework-for-Absolute-B [7.10.2017.]

[13] A. Stuart, K. Ord, Kendall's Advanced Theory of Statistics: Volume 1 – Distribution Theory, Edward Arnold, London, 1994.

[14] NMC-Junior Irrigation Control, Part 1, User Manual, Netafim, 2007.

[15] G. Martinović, J. Simon, Greenhouse microclimatic environment controlled by a mobile measuring station, NJAS Wageningen Journal of Life Sciences, Vol. 70, str. 61 – 70, 2014.

SAŽETAK

U ovom diplomskom radu obrađen je web sustav upravljanja mikroklime plastenika na temelju podatkovne fuzije. Opisani su sustavi za kontroliranu proizvodnju povrća, vrste zaštićenih prostora i osnovni mikroklimatski čimbenici, te je dan primjer postojećeg rješenja, kao i idejno rješenje vlastite okoline plastenika. Objašnjene su osnove podatkovne fuzije, te je na primjeru prikazana primjena algoritma podatkovne fuzije. Napravljena je potpuno funkcionalna web aplikacija s dva režima rada: bez podatkovne fuzije i s podatkovnom fuzijom. Opisano je korišteno sklopovlje i programske tehnologije. Osim aplikacije izrađena je i maketa stvarnog sustava, te je izvedeno testiranje i analiza dobivenih rezultata.

Ključne riječi: Arduino, plastenik, podatkovna fuzija, upravljanje, web aplikacija

ABSTRACT

Subject matter of this final thesis is web system managing greenhouse microclimate based on data fusion. Systems for vegetable production control, types of protected areas and basic microclimate factors are described. An example of existing solution and a conceptual solution are also given. Basics of data fusion are described and an example of usage of the algorithm is shown. A fully functional web application was made with two work regimes: without data fusion and with it. Used hardware and programming technologies are also described. Apart from the application, a model of a real system was made. Tests were conducted and an analysis of the results was made.

Keywords: Arduino, greenhouse, data fusion, control, web application

ŽIVOTOPIS

Perica Džijan rođen je 8.9.1991. god. u Vinkovcima. Osnovnu školu je završio u Vinkovcima 2006. godine. Iste godine upisuje Tehničku školu Ruđera Boškovića u Vinkovcima, smjer računalni tehničar u strojarstvu. Srednju školu završava 2010. godine, te iste te godine upisuje preddiplomski studij računarstva na Elektrotehničkom fakultetu u Osijeku. 2014. godine završava preddiplomski studij i upisuje diplomski studij računarstva, smjer procesno računarstvo.

PRILOZI (na CD-u)

Prilog 1. Docx i pdf rada

Prilog 2. Programski kod Arduino klijenta

Prilog 3. Programski kod web aplikacije u programskom okruženju Visual Studio