УНИВЕРЗИТЕТ „ГОЦЕ ДЕЛЧЕВ“ - ШТИП

ФАКУЛТЕТ ЗА ИНФОРМАТИКА

Вештачка интелигенција и роботика

Штип

ВЛАТКО ИВАНОВ

ИЗРАБОТКА НА МОДЕЛ НА ПАМЕТЕН И БЕЗБЕДЕН ДОМ СО ТЕХНИКИ НА ВЕШТАЧКА ИНТЕЛИГЕНЦИЈА

-МАГИСТЕРСКИ ТРУД-

Штип, февруари, 2021

Комисија за оценка и одбрана

Ментор: проф. д-р Цвета Мартиновска-Банде

Факултет за информатика при Универзитет „Гоце Делчев“ - Штип

Член/претседател: доц. д-р Доне Стојанов

Факултет за информатика при Универзитет „Гоце Делчев“ - Штип

Член: доц. д-р Наташа Стојковиќ

Факултет за информатика при Универзитет „Гоце Делчев“ - Штип

Февруари, 2021 год.

Благодарност

По повод изработката на овој магистерски труд сакам да изразам голема благодарност до сите оние кои на некој начин придонесоа овој труд да ја добие потребната содржина.

Пред сè, упатувам огромна благодарност до мојот ментор проф. д-р Цвета Мартиновска-Банде, за перманентната стручна помош и сугестии што ми ги даваше од почетокот до финалната изработка на трудот.

Исто така, им се заблагодарувам на членовите на комисијата за одбрана доц. д-р Доне Стојанов и доц. д-р Наташа Стојковиќ кои со своите стручни сугестии придонесоа за изработката на овој труд.

Искрено им се заблагодарувам на членовите на моето семејство и моите најблиски за несебичната помош и поддршка во текот на изработката.

Влатко Иванов

Листа на објавени трудови

1. Ivanov Vlatko, Ivanov Ratko, Martinovska Bande Cveta (2020). Real time alert system in Smart Home. *International Journal of Scientific and Research Publications, Volume 10, Issue 9, September 2020 ISSN 2250-3153 pp (676-682)*
2. Ivanov Ratko, Ivanov Vlatko, Martinovska Bande Cveta (2020). Energy efficiency in Smart Home system. *International Journal of Scientific and Research Publications, Volume 10, Issue 9, September 2020 ISSN 2250-3153 pp (683-688)*

ИЗРАБОТКА НА МОДЕЛ НА ПАМЕТЕН И БЕЗБЕДЕН ДОМ СО ТЕХНИКИ НА ВЕШТАЧКА ИНТЕЛИГЕНЦИЈА

Краток извадок

Аспирација на човекот отсекогаш била да се олеснат секојдневните обврски. Со напредокот на технологијата е овозможено нивно олеснување, односно овозможена е автоматизација на нив. Во денешно време стануваат популарни паметните домови со кои скоро може и да се заборават некогашните секојдневни обврски во домот. *Smart home*, односно „Паметен дом“ претставува систем кој вклучува повеќе аспекти од домот и овозможува далечинска контрола преку мобилни или веб-апликации. Овде спаѓаат осветлувањето, греењето, ладењето, домаќинските апарати, вентилацијата, системите за обезбедување и сл. Овој систем донесува сигурност, комфор, но воедно и заштеда на енергија, време и пари. Преку мобилни апликации или софтвери корисникот има достапност во својот дом во секое време и од секое место.

Популарноста на автоматизирањето на домовите е во постојан пораст во последниве години поради зголемената достапност, едноставност, зголемувањето на квалитетот и комфорот на живот. Идејата за контролирање на аспекти од нашите домови и можностите одредени елементи да реагираат автоматски на настани, станува сè попопуларна и неопходна за безбедносни и финансиски цели.

Затоа, апликациите за паметни домови играат сè поголема улога во денешницата. Со брзиот развој на интернетот на нештата (IOT), паметниот дом сè повеќе го привлекува вниманието на луѓето поради подобрувањето на квалитетот на животот.

Со зголемениот економски развој се зголемува и животниот стандард. Модерното општество бара безбедност, економичност и удобен живот кој е идеален за секое семејство. Автоматизацијата на домовите претставува идеја која ветува. Главна предност, освен зголемениот комфор и безбедност, е порационалната употреба на енергијата и другите ресурси, што обезбедува значителна заштеда.

Изработката и автоматизацијата на овие паметни домови претставува вистински предизвик за секој инженер.

Клучни зборови

Паметен дом, IoT, NodeMcu.

MAKING AND CONTROL OF A MODEL OF SECURE SMART HOME WITH ARTIFICIAL INTELLIGENCE TECHNIQUES

ABSTRACT

Human evolution is in constant development to ease daily tasks in each aspect of life. The technological advances of the modern world are vast and ever-growing. Combining the two we now have new solutions to old problems. The possibility to provide an automated answer to a given issue is examined in this thesis with the ability to remotely control many appliances via a single system, directly in our homes.

The popularity of home automation has been growing steadily in recent years due to increased availability, simplicity, increased quality and comfort of life. The idea of controlling aspects of our homes and allowing certain elements to react automatically to events is becoming increasingly popular and necessary for security and financial purposes. Smart lighting, heating, cooling, home appliance, ventilation, security systems and many more are part of the smart homes. This system brings security, comfort, but also saves energy, time and money.

That's why smart home applications are playing an increasingly important role today. With the rapid development of the Internet of Things (IoT), the smart home is increasingly attracting people's attention for improving the quality of life. With the increased economic development, the standard of living increases as well.

Modern society requires security, economy and a comfortable life that is ideal for every family. Home automation is a promising idea. The main advantage, apart from increased comfort and safety, is the rational use of energy and other resources, providing significant savings.

The construction and automation of these smart homes is real challenge for every engineer.

KEY WORDS

Smart Home, IoT, NodeMcu

СОДРЖИНА

1. Вовед

Со постојаниот пораст и прогрес на телекомуникациските технологии, ерата на „Интернет на компјутерите“ (*Internet of Computer*) премина во „Интернет на нештата“ (*Internet of Things*). Интернет на нештата претставува мрежна инфраструктура на уреди кои меѓусебно споделуваат податоци преку интернет. IoT е главниот дел од идејата за поврзан свет. Во денешно време, нашите домови, клиники, фабрики се подобруваат со уреди кои имаат компјутерски и мрежни способности. Интернетот-на-нештата вклучува значителни развојни активности како паметна мрежа, паметна логистика, тестирање на животната средина и безбедност, интелигентен транспорт, индустриска контрола и автоматизација, финансии и услуги, воена одбрана, здравствена заштита, добро земјоделство и паметни домови.

Оваа парадигма овозможува да се чуваат, обработуваат и да се даваат изобилни количини на податоци во стручно-толкувачка форма без човечки ум. Појавата на IoT фрла и ново светло на концептот на „паметен дом“. IoT-куќната опрема овозможува паметниот дом да биде поинтелигентен, контролиран од далечина и меѓусебно поврзан.

Глобалниот пазар за паметни домови се очекува да достигне околу 246.42 милијарди долари до 2025 г. [2]. Компании како „Гугл“, „Амазон“ и „Самсунг Електроникс“ (*Google*, *Amazon*, *Samsung Electronics)* полека се појавуваат на големата сцена и големиот пазар, а со себе носат и нови сервиси и продукти со цел да добијат предност на растечкиот пазар. Многу „start-up“ компании исто така прават напори да се приклучат на овој растечки пазар. Паметниот дом привлекува внимание поради IoT, но тоа не е нов концепт. Всушност, концептот за паметен дом се дискутира од 1980 година и тој еволуираше од традиционалната автоматизација на домот (т.н. мрежни домови, сеприсутни домови и интелигентни и интерактивни домови).

И покрај долгата историја и зголемениот интерес, услугите кои ги нудат паметните домови не се широко распространети. Постојат многу причини (на пример, високи цени на уредите, ограничена побарувачка на потрошувачите и долги циклуси на замена на уредите) што спречуваат ширење на идејата за паметен дом.

Целта на ова истражување е изработка на модел на паметен дом и систем за безбедност и алармирање во паметните домови. Со ова може да се долови што сè може да се имплементира во нашите домови на паметен и ефективен начин и да се покаже какви предности придонесуваат употребите на модерните алатки и како сето тоа ни го олеснува начинот на живеење и овозможува посмирен и удобен начин на живеење. Исто така, целта е да се прикажат различни безбедносни алатки кои можат да се имплементираат, со што ќе се добие подобра слика за безбедноста во паметниот дом. Целта е со користењето на техники од вештачката интелигенција да се прикаже квалитетот на работа на еден сензор и контролирање на уред преку гласовни команди на македонски јазик.

2. Преглед на литература

Според извештајот на Светската банка за глобалното здравје и старост во 2019 година, приближно 700 милиони луѓе, што претставува 9 % од светската популација, се на возраст од 65 години или постари. Се проценува дека оваа бројка до 2050 година ќе достигне 1,5 милијарди (околу 18 % од светската популација) [1]. Покрај тоа, СЗО проценува дека 650 милиони луѓе во светот живеат со попреченост [2,3]. Затоа, неопходно е истражувачите да развиваат пријателски настроени паметни околини за промовирање на подобар и поквалитетен живот за постари лица и лица со попреченост.

Од распространетата литература и насоки во паметните домови, ги издвоивме следниве: Џефри Сор, Цујоши Хамано и Јосхиказу Фуџисава предлагаат решение за приватноста и безбедноста на постарите лица во паметен дом во Австралија. Тие опишуваат како *RFID* (*Radio Frequency Identification*) технологијата може да се користи за следење на локација на нешта, луѓе, лекови итн. [3].

Христос Стергуа [4] ги споил пресметувањето во облак (*Cloud Computing*) и Интернет на нештата, сè со цел да покаже како технологијата на *Cloud computing* ја подобрува функционалноста на ИнН. Мажид Ал-Кувари [5] се фокусирал на вграден IoT за користење на анализирани податоци за далечинско извршување на команди на домашни апарати во паметен дом. Триша Дата [6] предлага библиотека за зачувување на приватноста, сè со цел да се визуализира сообраќајот во домашните апарати. Жиан Мао [7] ги искористил алгоритмите на

машинското учење во безбедноста на екосистемот на паметен дом. Фајсал Саид [8] предлага користење на сензори кои ќе насетат и откријат пожар во реално време со висока точност.

3. Интернет на нештата

Во принцип, не постои прифатена дефиниција за Интернет на нештата. Всушност, има многу различни групи на луѓе кои го дефинирале поимот, иако почетната употреба му се препишува на експертот за дигитална иновација Кевин Ештон [9]. Од сите дефиниции за IoT може да се заклучи дека првата верзија на IoT е за податоци создадени од луѓето, додека следната верзија е за податоци добиени од нешта, па затоа е наречено Интернет на нештата.

Подолу се наведени некои од дефинициите:

IoT генерално е дефиниран како „динамичка глобална мрежна инфраструктура со самоконфигурирање и способности засновани на стандарди и интероперативни протоколи; физичките и виртуелните „нешта“ во IoT имаат идентитети и атрибути и се способни да користат интелигентни „интерфејси“ и да бидат интегрирани како информациска мрежа“ [10].

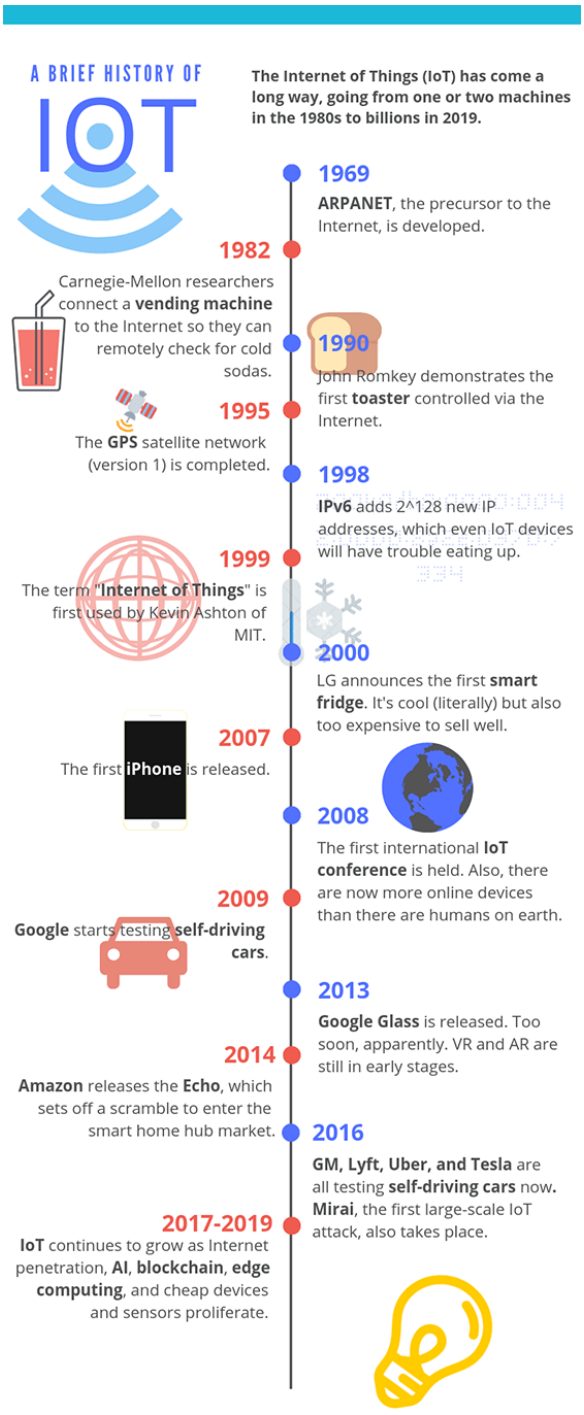
Целта на IoT е да ги зголеми функциите на првата верзија на интернет и да го направи покорисен. Со IoT, корисниците можат да ги споделат и двете информации: обезбедени од луѓето кои се вклучени во базите на податоци и исто така информации обезбедени од „нештата“ на физичкиот свет [11]. Можеме да го опишеме IoT како врска на физички „нешта“ на интернет и едни со други за разни корисни цели преку различни интелигентни технологии, создавајќи „паметен“ екосистем од сеприсутни компјутери (*Pervasive computing*). Исто така, може да се опише како вклучена вградена интелигенција во одделни објекти што може да забележи промена во нивната физичка состојба.

Заедничката дефиниција за IoT е дека компјутерите, сензорите и објектите комуницираат едни со други и процесираат податоци, затоа можеме да наведеме дека IoT е нов систем комбиниран од голем број на информациски технологии. Интернетот на нештата комбинира различни технологии во полуавтономна мрежа. Тоа поврзува индивидуални уреди со мрежата и ги поврзува едни со други. Постојат и контролорни системи во мрежата (софтвери и сервиси) кои дејствуваат како мозоци на системот за обработка на податоци

од анализирањето и користење на податоците собрани од поврзаните уреди за донесување одлуки и иницирање акции од исти или други уреди [12].

Централната цел на IoT е да ни овозможи уникатно идентификување, означување пристап и контрола на нештата во секое време и од каде било, користејќи интернет [13]. Уредите поврзани на интернет може да резултираат во носење на голем број интелигентни и автономни апликации и услуги носејќи значителни лични, професионални и економски придобивки.

3.1. Времеплов на избрани големи настани во IoT



Слика 1: Големи настани во IoT [14]

Figure 1 Big events in IoT [14]

1969: APRANET, претходник на современиот интернет е развиен и ставен во употреба од страна на *DARPA*, агенција за напредни истражувачки проекти за одбрана на САД. Ова е основно за „интернет“ делот од Интернет на нештата.

1980-ти: *ARPANET* е отворен за јавноста од страна на трговски провајдери, овозможувајќи им на луѓето да ги поврзуваат работите доколку сакаат.

1982: Програмерите на Универзитетот „Карнеги Мелон“ поврзуваат автомат за кока-кола на интернет, овозможувајќи им да проверат дали машината има ладни газирани пијалаци пред да набават. Ова често се наведува како еден од првите IoT уреди.

1990: Џон Ромки, како одговор на предизвикот, поврзува тостер на интернет и успева да го вклучи и исклучи, приближувајќи нѐ уште повеќе на она што го сметаме за модерни IoT уреди.

1993: Инженерите на Универзитетот во Кембриџ, поддржувајќи ја веќе воспоставената традиција за комбинирање на интернет со апарати и храна, развиваат систем кој фотографира машина за кафе трипати во минута, овозможувајќи им на работниците да го следат статусот. Ова е првата светска веб-камера.

1995: Конечно е завршена првата верзија на долгогодишната *GPS* сателитска програма водена од американската влада, голем чекор кон докажување на една од највлијателните компоненти за многу IoT уреди: локација.

1998: IPv6 станува нацрт-стандард, што овозможува повеќе уреди да се поврзат на интернет отколку што претходно дозволувал IPv4. Додека 32 битниот IPv4 обезбедувал доволно уникатни идентификатори за околу 4.3 милијарди уреди, 128 битниот IPv6 има доволно уникатни идентификатори до 2^128.

1999: Ова е голема година за IoT, бидејќи фразата најверојатно за првпат е употребена. Кевин Ештон, раководител на лабораториите за автоидентификација на МИТ, го вклучува во презентација пред директорите на „Проктор и Гембел“ (*Proctor & Gamble*) како начин да се илустрира потенцијалот на технологијата за следење на RFID.

2000: *LG* најавува еден од квинтесенцијалните IoT уреди: интернет фрижидер. Интересна идеја, комплетно со екрани и следење, сè со цел да ви помогнат да имате евиденција за тоа што сте имале во вашиот фрижидер, но неговата цена од повеќе од 20 000 долари била пречка за корисниците.

2004: Фразата „Интернет на нештата“ започнува да се појавува во наслови на книги и почнува да се појавува во медиумите.

2007: Првиот *iPhone* се појавува на сцената, нудејќи сосема нов начин за интеракција на поширокатa јавност со светот и уредите поврзани на интернет.

2008: Првата меѓународна IoT конференција се одржува во Цирих, Швајцарија. Годината е пригодна бидејќи е исто така прва година кога бројот на уреди поврзани на интернет го надминува бројот на луѓето на земјата.

2009: *Google* започнува со тестирање на (*self-driving*) автономни возила, а Медицинскиот центар *St. Jude`s* објавува „*pacemakers*“ поврзани на интернет. Уредот *St. Jude`s* ќе продолжи да испишува историја со тоа што е првиот IoT медицински уред што претрпува сериозно нарушување на безбедноста во 2016 година (без жртви, за среќа). Исто така, Биткоин започнува со работа, претходник на блокчејн-технологиите кои веројатно ќе бидат голем дел од IoT.

2010: Кинеската влада го именува IoT како клучна технологија и најавува дека тоа е дел од нивниот долгорочен план за развој. Во истата година *Nest* објавува паметен термостат што ги учи вашите навики и автоматски ја прилагодува температурата во вашиот дом, ставајќи го во центарот на вниманието на концептот „Паметен дом“.

2011: Фирмата за истражување на пазарот *Gartner* го додава IoT во нивниот „*hype cicle*“ што е график кој се користи за мерење на популарноста на технологијата наспроти нејзината вистинска корисност со текот на времето. Во 2018 година, IoT излезе од врвот на „надуените“ очекувања и може да се насочи кон проверка на реалноста во „коритото“ на разочарување пред да биде погоден од платото на продуктивноста.

2013: *Google Glass* е објавено како револуционен чекор во IoT и технологија што може да се носи, но најверојатно е пред своето време. Па така користењето на овој продукт опаѓа нагло.

2014: *Amazon* го објавува *Echo*, отворајќи го „брзиот пат“ кон пазарот на паметни домови. Од друга страна, на конзорциумот за индустриски IoT стандарди се демонстрира потенцијалот за IoT да го промени начинот на работа на кој било процес и на производствениот синџир.

2016: *General Motors, Lyft, Tesla* и *Uber* тестираат автомобили кои се управуваат самостојно. За жал, исто така, потврден е и првиот масовен напад со малициозен софтвер, со тоа што ботнет „*Mirai*“ ги напаѓа уредите на IoT со произведувачките стандардни најавувачки информации, преземајќи ги и користејќи ги за *DDoS* напад на популарните веб-страници.

2017-2019: Развојот на IoT станува поевтин, полесен и пошироко прифатен што доведува до мали бранови на иновации низ целата индустрија. Автомобилите што се управуваат самостојно продолжуваат да се подобруваат, блокчејнот и вештачката интелигенција започнуваат да се интегрираат во IoT платформите, а зголемената „пенетрација“ на паметни телефони продолжува да го прави IoT атрактивен предлог за во иднина.

Ако може да се верува на *Gartner Hype Cicle*, може да се очекува да ги прилагодиме нашите очекувања во следните неколку години. Сепак, долгорочно IoT веројатно ќе биде „нормално“ во секојдневниот живот. Контролирањето на домовите преку паметните телефони е прилично уредено. Да се добие слика во реално време за секоја ставка во снабдувањето со продукти исто така би било корисно. Вештачката интелигенција и блокчејн сè повеќе се користат за да ги направат нашите уреди сè понезависни и подобро поврзани. Но, за масовно усвојување на хардвер е потребно време, па затоа целосното применување на Интернетот на нештата ќе биде постепено.

4. Паметен дом

Терминот „Паметен дом“ се користи за да се опише куќа која содржи комуникациска мрежа што поврзува различни апарати и им овозможува да се контролираат од далечина, да се следат и да се пристапува, според Одделот за трговија и индустрија [63].

Паметните уреди се поврзуваат на интернет, а многу од нив имаат апликации за паметни телефони што ви овозможуваат да пристапувате и да ги контролирате од далечина преку *Wi-Fi*.

Широкопојасниот интернет е побрз, посигурен и подостапен од кога било досега. Подобрениот опсег на сигнали на *Wi-Fi*-рутери значи дека еден рутер може да понуди безжично покритие низ повеќе простории во нашите домови, овозможувајќи да се поврзат повеќе уреди и да се поврзе целиот дом.

Користејќи ја технологијата во куќата, може да се контролира и да се види што се случува во домот, дури и кога нема никој присутен таму, користејќи паметен телефон, таблет или компјутер. На пример, може да се инсталира камера во домот што овозможува да се проверуваат децата, домашните миленици и сè останато.

Исто така, постои потенцијал да се направи голема разлика во цената на сметките за комунални услуги, бидејќи со паметниот дом нема потреба да се троши енергија доколку се заборави греењето. Со следење на греењето, вода и енергетската ефикасност, има големи шанси да се намалат сметките за комунални услуги.

Паметните домови исто така можат да ги заштитат најранливите во општеството. Некои уреди можат да предизвикаат тревога ако постара личност не ги пие своите таблети, падне или се однесува надвор од нивната вообичаена рутина.

4.1. Историја на автоматизација на домовите

Домашната автоматизација му била достапна на просечниот потрошувач скоро 35 години. Концептот на автоматизација на домовите бил обемно истражен во 20 и 21 век. Табелата 1.0 дава преглед на развојот на автоматизацијата на домовите. Во 1975 година, компанијата „Пико електроникс“ го развива и патентира X10, технологија која работела на веќе постоечките електрични жици. Компанијата пробала девет различни пристапи без успех за да успеат во десеттиот обид, па затоа и продуктот е наречен X10. Идејата зад X10 била да се пренесе сигнал од 120 kHz на електрична линија. Секој сигнал бил специфично кодиран со код на куќа и *Unit* код. Во 1978 е издаден X10 протоколот. Поради фактот што преносот на податоците бил направен со повторна употреба на веќе постоечките далноводи, потрошувачката на енергија и користењето на дополнителни компоненти се намалил. Осумдесеттите години од 20-от век се отскочна штица за автоматизацијата на домот. Во 1983 година, Мурата, Намекава и Хамабе предложуваат план за стандардизација на системите за автоматизација на домот затоа што дотогаш немало компатибилност помеѓу различните производители кои биле поборници за системи на автоматизација на домови во Јапонија. Истражувачката група подоцна била наречена ХБС студиска група. Во 1984 година, по две години работа, седум производители постигнале

договор за стандардизација. Тие предложиле систем за автоматизација на дом (HBS) *Home bus system* кој бил составен од три опсези: основно ниво за контролирање на сигналот, подниво за голема брзина на преносот на податоци и FM / TV ниво за визуелизација на информациите. Во споредба со x10 протоколот, HBS користел коаксијален кабел кој според студиската група имал помала цена за инсталација од другите протоколи. Во таков случај, доколку еден дом веќе има инсталација со коаксијален кабел, било возможно да го вклучите HBS на веќе постоечкиот систем со мали измени.

Табела 1: Времеплов на автоматизација на домовите

Table 1 Timeline of home automation

Година / Year	**Автор / Author**	**Придонес / Contribution**
1975	Пико Електроникс [15]	Пронаоѓање и патентирање на X10 технологијата
1978	Пико Електроникс [15]	Првите x10 продукти претставени на јавноста
1983	Масаши Мурата [16]	Јапонски предлог за стандардизиран протокол наречен HBS
1983	Христос Далгерис [17]	Американски предлог за стандардизиран протокол наречен CEBUS
1985	Масахиро Инуе [18]	Првиот развиен систем базиран на HBS
1986	Руџи Канабе [19]	Обновување на стандардизираниот протокол HBS во Јапонија
1988	Христос Далгерис [17]	Објава на CEBus стандардот
1989	IEC и ISO [20]	Организација на заеднички комитет наречен HAS
1992	Христос Далгерис [17]	Завршување и пуштање во употреба на CEBus
1993	Smart Home Inc. [21]	Маркетинг на паметен дом
1996	Питер Коркоран [22]	Демонстрација за пристап на CEBus преку WWW
2002	Н. Сришкантан [23]	Автоматизиран систем за контрола на паметен дом преку блутут

2002	Europeans [24]	Одлука за правење еден стандардизиран протокол
2004	А. Алхераиш [25]	Предлог за систем базиран на M2M преку GSM
2005	А. Ал-Али [26]	Предлог за Јава-базиран систем за автоматизација
2005	SmartLabs [27]	Insteon
2005	ZigBee Alliance [28]	ZigBee
2005	ZenSys [29]	Z-Wave
2006	Arduino [30]	Развој на Ардуино
2011	Google [31]	Вовед во Android@home
2014	Apple [32]	Apple HomeKit
2014	Amazon [33]	Alexa Echo Dot
2016	Google [34]	GoogleHome
2017	Google [34]	Google Home Mini
2017	Google [34]	Google Home Max
2017	Amazon [33]	Amazon Tap
2017	Amazon [33]	Amazon Echo Look
2018	Google [34]	Google Home Hub / Nest Hub
2018	Apple [32]	HomePod
2018	Amazon [33]	Amazon Echo Show
2018	Samsung [35]	SmartThings Hub
2019	Google [34]	Google Nest Hub Max
2019	Google [34]	Google Nest Mini (2nd generation)

Во 1985 година, водејќи се по истражувањата на Мурата од 1984 година, Инуе, Уемура, Минагава, Есаки и Хонда изработуваат систем за автоматизација на дом базиран на *HBS* стандардот. Системот бил составен од четири потсистеми:

1. Потсистем за мониторирање и контролирање на соба
    a. Систем за контролирање на задачи во домот и безбедност.
2. Телефонски потсистем
    a. Телефонски и безбедносен аларм.
3. Потсистем за телефонско контролирање
    a. Контролирање на уреди и безбедносни сензори преку телефони кои не се наоѓаат во домот.
4. Потсистем за внатрешна видео контрола
    a. Систем за добивање видео сигнали од поставени телефони. [18]

По три години успешно експериментирање и два направени модели за автоматизиран дом, Мурата го ревидира и обновува барањето направено во 1983 година. Новото барање за *HBS* содржи еден опсег помалку од претходниот (*band*).

Паралелно на Јапонците, производителите во САД исто така ги увиделе проблемите на стандардизацијата. Многу нови електрични уреди им биле понудени на потрошувачите, но ниту еден од нив немал можност да се интегрира секој уред во една централна мрежа. Единствено решение било потрошувачите да ги набавуваат уредите само од еден произведувач. Во 1983 година ЕИС (Електронска индустриска асоцијација) организира комитет за развивање на стандард кој ќе ги надмине проблемите на стандардизацијата [17].

Дејвид мек Фојден бил пронаоѓачот и главен извршен директор на *Upper Marlboro, Md*., конзорциум. Во 1984 тој го претставува концептот на паметен дом. Додека другите компании на пазарот нудат систем за автоматизиран дом кој луѓето можат да го инсталираат во својот дом, Дејвид предлага нов пристап. Конзорциумот започнува со развој на веќе преинсталирани системи за автоматизација на домови и до 1987 година планирале да имаат 5000 домови подготвени за продажба. Домовите би биле програмирани и секој уред би можел да комуницира со друг преку компјутерски мрежи [36].

Во меѓувреме, исто така во раните 80-ти, Електронската индустриска асоцијација (ЕИА) [37] работи на *Consumer Electronic Bus (CEBus)*. По пет години постојано истражување и развивање тие го објавуваат работниот нацрт-стандард. Според ЕИА, *CEBus* е дизајниран да им овозможи на производителите да произведуваат електронски уреди кои можат да комуницираат со производи од други производители преку различни медиумски канали. Карактеристика на *CEBus* била можноста за именување на уредите. На пример, корисникот може да го именува својот телевизор како „*Televizor*“. Оваа карактеристика ја отстранува потребата за познавањето на нумеричкиот код за секој уред [37].

CEBus имал модел со четири мрежни нивоа

1. Апликациско ниво
    a. Уредот испраќа барање за *APDU (Application Layer Protocol Data Unit)* и ова се проследува до мрежното ниво.

2. Мрежно ниво
    a. Го прима *APDU* и го претвора во *NPDU (Network Layer Protocol Data Unit)*. Потоа го испраќа до *LLC (Logical Link Control)* потсистемот кој е дел од податочното ниво.
3. Податочно ниво
    a. *Logical Link Control* потсистем додава заглавје со дополнителен информациски сервис до *NPDU* за да го направи *LPDU* и да го повика сервисот на *Medium Access Control (MAC).*
    b. *Medium Access Control*, овде *LPDU* се претвора во *MPDU* и се испраќа на физичкото ниво.
4. Физичко ниво
    a. За примање на видео сигнали од телефони. [17]

По воведувањето на *CEBus* стандардот, истражувањата во ЕИА продолжиле. Иако *Smart House* планирале 5000 домови да бидат спремни до 1987 година, нивните планови пропаднале поради низа од пречки. На пример, една од пречките е кога *Smart House* компанијата пробала да направи еден голем кабел што содржи телефонски жици, ТВ жици, напојувачки жици и жици за комуникација. На крај, жицата била премногу густа што не било изводливо да се протне меѓу ѕидовите. На крајот од 1987, пронаоѓачот и извршниот директор Дејвид, ја напушта компанијата поради постојани повторувачки проблеми со жиците. Подоцна, производните инженери се фокусирале на додавање нови карактеристики на *Smart House* наместо да ги пуштат своите први производи на пазарот [36]. Кон крајот на 80-те имало четири клучни играчи во стандардот за автоматизација на дом: *HBS, Smart House, Esprit Home System* и *CEBus*. Јапонските и европските групи сакале да го направат својот стандард како светски стандард [20]. Па така, во 1988 година, бил создаден комитет со име *HES (Home Electronic Systems*). Овој комитет бил составен од Меѓународната електротехничка комисија *IEC (International Electrotechnical Commision)* и Меѓународната организација за стандардизација *ISO (International Organization of Standardization).* Како цел имале да наведат стандарди за автоматизиран дом. До крајот на 1989 година, тимот на САД бара од *HES* да го разгледа нивниот систем за автоматизација на дом. Подоцна, во 1992 година веќе постоеле седум контролни системи за домови вклучени во *HES*.

1. *BatiBUS* (Франција)
2. *CEBus* (САД)
3. *D2B* (Холандија)
4. *EIB* (Германија)
5. *ESPRIT Home Systems* (Глобален пазар)
6. *HBS* (Јапонија).

Исто така, во 1992 година е завршувањето и објавувањето на *CEBus* стандардот IS-60 (18). 1993 била критична година за автоматизација на *SMART HOUSE*. Тие го објавиле првиот продукт на пазарот. Понатаму, првиот модел на *SMART HOUSE* се отворил во Напервил, САД [21]. Во меѓувреме, *World-Wide-Web (WWW)* била развиена од научници во *CERN* лабораторијата 1990. Оваа технологија станала избор за интернет апликации и во 1996 година Питер М. Коркоран, Џо Дезбонет и Карл Лустед предложиле начин за далечински пристап до *CEBus* мрежите преку *HTTP* протокол. Тие вовеле систем за дијагностицирање на грешки. Хардверот кој бил дел од имплементацијата се состоел од *CEBus* мрежа, активен *CEBus* јазол и компјутерски базиран сервер. Софтверот вклучувал модифициран *CEBus* протокол, модификација на *WWW* серверот, *serverend CGI* и интерфејси на крајот на клиентот. Според Коркоран, системот може да се опише како *„'активен' CEBus-јазол што содржи посебен микроконтролер со модифициран протоколен софтвер за да се олесни тестот и следењето на активности. Поднивото Medium Access Control (MAC) на податочното ниво бил репрограмиран за да му овозможи на системот да ги прима и снима сите мрежни пакети. Овие зафатени пакети се пренесуваат на Апликациското ниво. HTTP серверот работел на стандарден компјутер, кој исто така бил поврзан на мрежата CEBus со помош на „активен" CEBus јазол. „Демон" програм ја следи активноста на CEBus и нуди низа услуги. Основни услуги вклучувајќи го и најавувањето на мрежен сообраќај до крајна серверска база на податоци и пренесување на избран мрежен сообраќај до оддалечен клиент на WWW. Софтверот исто така бил способен да прифати низи од мрежни пакети од далечен WWW клиент за пренос на CEBus мрежата преку „активниот" јазол (22)".* Затоа ова се смета за пионер на онлајн далечинското управување, не само на протоколот *CEBus*, туку и на сите системи за автоматизација на домот.

До 1997 година имало осум национални стандарди за домашна автоматизација и еден меѓународен систем за автоматизација на домови.

1. *X-10* (САД)
2. *CEBus* (САД)
3. *LonTalk* (САД)
4. *BatiBUS* (Европа)
5. *EIB* (Европа)
6. *European Home Systems* (Европа)
7. *HBS* (Јапонија)
8. *HES* (Меѓународен).

Производителите имале огромен број различни стандарди за избор, па поради тоа, развивањето на паметен дом и автоматизацијата на продуктите бил одложен.

До 2002 година, САД имале осум стандарди, 12 отворени протоколи и 10 комерцијални протоколи. Оваа разновидност била комплицирана за производителите да развијат комерцијален производ. За среќа, поради напорите, инженерите од Европа сфатиле дека големиот број на протоколи го забавува развојот на технологијата за автоматизација на домот и одлучиле да ги комбинираат трите ривалски протоколи во еден наречен Конекс (*Konnex*). Спротивно на тоа, пристапот на САД е спротивен, односно направиле притисок врз развојот на повеќе протоколи. Тоа било негативен знак за многу технологии. Проблемот го воочил Кенет Векс кој вели: *„Пазарот за домашни системи не е доволно голем за да поддржи толку многу протоколи. Затоа многумина нема да траат [24]“*.

Иако луѓето знаеле дека на пазарот веќе постојат премногу протоколи и технологии за автоматизација на домовите, не престанале да измислуваат и предлагаат нови системи во 2000-тите. Едно од таквите решенија е предложено од Срискантан, Тан и Каранде. Ти предложиле систем за автоматизација на дом кој се заснова на блутут (*Bluetooth*). Тие го избрале блутутот бидејќи според нив ги покривал сите основни потреби за автоматизација на дом. На пример, работел преку нелиценцирана и достапна фреквенција од 2,4 GHz и можел да ги поврзе уредите во опсег од 10 метри. Развиениот систем се состоел од контролер на

домаќин, односно *Host Controller* кој бил имплементиран на персонален компјутер и микроконтролер што можел да комуницира со „домаќинот“ преку блутут. Тие го именувале како *The Home Automation Protocol (HAP)* и предвиделе голема иднина за предложеното решение. Срискантан вели: „Со нашиот домашен систем, кој се состои од „*HC*“, кој обично е во форма на компјутер, лесно може да се воспостави поврзување на интернет со што би се овозможила контролата. Напорите во таква насока ќе помогнат да се реализира целосно безжичен и целосно автоматизиран систем за автоматизација на дом [23]“.

Во периодот што следувал биле презентирани различни решенија за автоматизација на дом. На пример:

1. Во 2004 година А. Алхераш предложува *M2M (machine-to-machine, man-to-machine or mobile-to-machine)*, односно машина до машина, човек до машина и мобилен телефон до машина решение базирано на *GSM* мобилна комуникациска мрежа [25],
2. Во 2005 година А. Р. Ал-Али и М. Ал-Русан презентираат систем за домашна автоматизација базиран на *Java* програмскиот јазик [26].

Во 2005 година имало голема експанзија во системите за автоматизација на дом. Повеќето системи кои денес се многу популарни, за првпат се претставени во истата година. Прво, системот за автоматизација на дом, наречен Инстеон (*Insteon*), бил развиен од компанијата наречена *SmartLabs.* Главната карактеристика на Инстеон била тоа што имал мрежеста топологија која се состоела од *RF* и *PLC*. Накратко, можело да се користи само *RF*, само *PLC* или и двете истовремено.

Второ, технологијата за безжично вмрежување наречена ЗигБи (*ZigBee*) била претставена од *ZigBee Alliance*. Таа била развиена за мал број податоци и апликации со краток опсег. Една од главните карактеристики е тоа што мрежа заснована на *ZigBee* лесно можело да се скалира без потреба од користење на моќни предаватели. И, на крај е претставена *Z-Wave*. Тоа е безжичен протокол произведен од *ZenSys*. Производителот наведува дека: „Протоколот *Z-Wave* е протокол со половина дуплекс со мал опсег дизајниран за сигурна и стабилна безжична мрежа на евтина мрежа за контрола. Главната цел на протоколите е

да комуницираат со кратки контролни пораки испратени на сигурен начин од контролната единица до еден или повеќе јазли во мрежата [29].

Различни технологии почнале да се појавуваат со помала цена и станале достапни за сите. Во 2006 година, бил претставен микроконтролер со единична плоча (*open-source-single-board*) од група италијански студенти кои го именувале Ардуино. Мотивацијата за создавање на Ардуино била да се развие уред кој ќе биде поевтин од останатите прототипни системи и лесно вградлив. Ардуино не бил наменет исклучиво за автоматизација на дом, но многу луѓе пронашле начин како да го искористат во своите системи за домашна автоматизација. Систем за автоматизација на дом кој е базиран на Ардуино плоча е *DomoticHome* кој е претставен во 2009. Авторот Матија Липрери, сакал да развие едноставен и ефикасен начин за автоматизација на светла и гаражна врата во својот дом. Слично на тоа, во 2011 година *Google* објавува дека развива свој систем за автоматизација на дом кој им овозможува на *Android* апликациите да „откриваат“, да се поврзат и да комуницираат со електрични апарати и уреди во домот. *Android@Home* користи протокол што функционира на 900 MHz фреквенција исто како *Z-Wave* [31]. Безжичниот протокол што се користи во презентацијата на *Android@home* во „*Google I/O Developers Conference*“ е базиран на *SNAP* од *Synapse Wireless* [34].

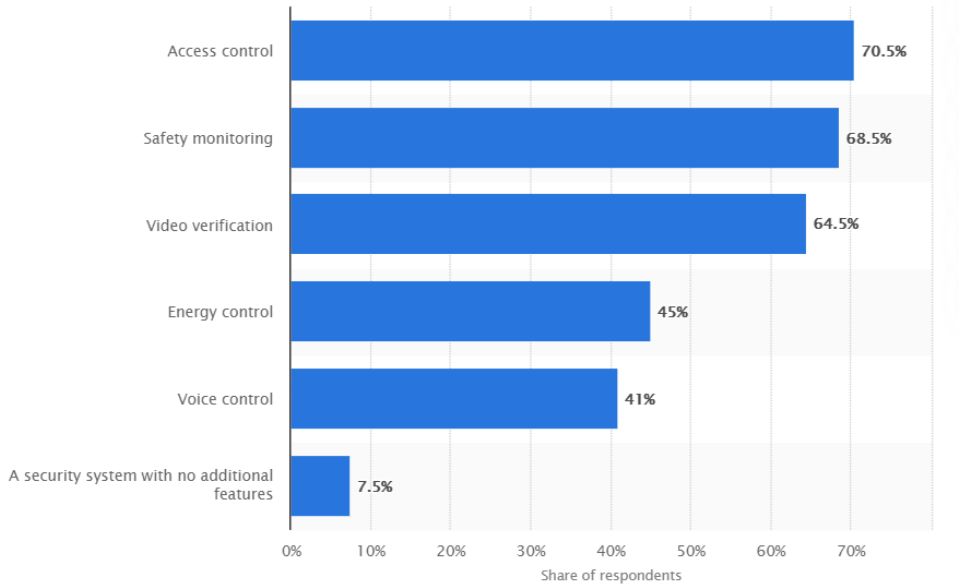
4.2. Безбедност во паметниот дом

Една од главните гранки на паметниот дом е безбедноста. Паметниот дом не може целосно да биде комплетиран доколку не се вклучи оваа гранка. Без безбедност не може да се постигне удобност и сигурност во домот кон кои се стреми секој човек. Конвенционалните системи за безбедност ги чуваат сопствениците, нивниот дом и нивниот имот безбедни од натрапници. [38]. Паметните системи за безбедност во домот нудат многу придобивки. Откривање на пожар, истекување на гас, истекување на вода, следење на камера, паметни светла, аларми и предупредување на сопственикот преку е-пошта, СМС и сл. Во случај на пожар, системот за паметен дом може да го извести сопственикот и да ги извести службите за итни случаи.

Програмите од вештачката интелигенција можат да ја пронајдат локацијата на пожарот и да ги достават информациите до противпожарните служби. Контрола

за пристап, видео верификација, проверка на отпечаток од прст се посакувани од луѓето денес.

Повеќе од 70 % од корисниците на домаќинства во САД се изјасниле дека сакаат контрола за пристап како одлика на нивната безбедност во домот. Посакуваните контролни способности за нов систем за безбедност на паметни домови според домаќинствата во САД од 2019 година се прикажани на следнава слика.



Слика 2: Анкета за безбедност во домот [64]

Figure 2 Poll for security in home [64]

Поставена камера на влезот, која со користење на техниките на вештачката интелигенција врши препознавање на лик, сензор за препознавање на отпечаток од прст и систем за препознавање на глас може да се искористат за контрола за пристап во домот. Камера поставена во дворот и сензор за детектирање на движење можат да придонесат во безбедноста. Кога сензорот за детектирање на движење ќе биде активиран, односно ќе забележи одредено движење, камерата може да започне со снимање и испраќање фотографии до сопственикот. Откако ќе биде известен сопственикот, тој преку команда може да го вклучи алармот.

Покрај безбедноста од обивање, потребна е безбедност од пожар, поплавување и штетен гас. Со поставување на сензори за пожар, за гас, за ниво на вода може

да се спречат големи оштетувања и да се спасат човечки животи. Откако некој од сензорите ќе се активира, системот го известува корисникот преку СМС, е-пошта или гласовна порака и ги известува соодветните служби. Со поставување на термални камери исто може да се следи пожарот, односно овие информации да се достават до службите за полесно справување со пожарот.

6. Вештачка интелигенција

Во компјутерската наука, терминот вештачка интелигенција (ВИ) се однесува на секоја интелигенција слична на човекот, изложена од компјутер, робот или друга машина. Во популарна употреба, вештачката интелигенција се однесува на можноста на компјутер или машина да ги имитира можностите на човечкиот ум - учење од примери и искуство, препознавање предмети, разбирање и реагирање на јазик, донесување одлуки, решавање проблеми - и комбинирање на овие и други можности за извршување функции што може да ги извршува човекот. Од развојот на дигиталниот компјутер во 40-тите години од минатиот век, докажано е дека компјутерите можат да бидат програмирани да извршуваат многу сложени задачи, како на пример, откривање докази за математички теореми или играње шах со голема вештина.

По неколку децении во кои вештачката интелигенција била сметана за „научна фантастика“, денес таа е дел од нашето секојдневие. Бранот на развој на ВИ е овозможен со ненадејната достапност на големи количини на податоци и соодветниот развој и широката достапност на компјутерските системи кои можат да ги процесираат сите податоци побргу и попрецизно отколку луѓето. ВИ ги комплетира нашите зборови додека ги пишуваме, обезбедува насоки за возење, чисти подови, препорачува што следно да купиме или да гледаме итн.

Сепак, и покрај постојаниот напредок во брзината на компјутерската обработка и капацитетот на меморија, сè уште нема програми што може да парираат на човечката флексибилност во однос на поширок домени или во задачи кои бараат секојдневно учење. Од друга страна, некои програми достигнуваат нивоа на перформанси на човечки експерти и професионалци при извршувањето на одредени специфични задачи, така што вештачката интелигенција може да се најде во разновидни апликации како медицинска дијагноза, компјутерски пребарувачи, препознавање глас или ракопис, предвидување базирано на претходни податоци и сл.

6.1. Апликации со вештачка интелигенција

Како што е споменато претходно, денес вештачката интелигенција е насекаде. Во прилог се прикажани неколку примери:

- Препознавање на говор: често се нарекува и говор во текст (*STT, speech-to-text)* - препознавањето на говор е технологија од вештачката интелигенција што ги препознава изговорените зборови и ги претвора во дигитализиран текст. Препознавање на говор се користи за диктати, гласовно управување на уреди, пораки со гласовен текст итн.
- Процесирање на природен јазик (*NLP): NLP* овозможува софтверска апликација, компјутер или машина да разбере, толкува и генерира човечки текст. *NLP* е вештачка интелигенција што стои зад дигиталните асистенти како *Siri* и *Alexa*, чет-ботови и друга виртуелна помош заснована на текст. Некои *NLP* користат анализа на чувства за откривање на расположението, ставот или други субјективни квалитети на јазикот.
- Препознавање на слика: технологија на вештачката интелигенција што може да идентификува и класифицира предмети, луѓе, пишување па дури и дејства во рамките на неподвижни и подвижни слики. Во принцип, управувано од длабоките невронски мрежи, препознавањето на слики се користи за системи за идентификација на отпечатоци од прст, апликации за проверка на мобилни телефони, анализа на видео и медицински слики, автомобили што управуваат самостојно и многу повеќе.
- Препораки во реално време: Некои веб-страници користат невронски мрежи за да препорачаат дополнителни набавки или реклами што веројатно ќе им се допаднат на клиентите врз основа на претходната активност на клиентот, претходни активности на други клиенти и огромен број други фактори, вклучувајќи и временски период од денот и временски услови. Препораките преку интернет можат да ја зголемат продажбата од 5 % до 30 %.
- Откривање аномалии во работата на сензори и други производи.

6.2. Вештачка интелигенција во паметен дом

Автоматизацијата во домот не е новитет и повеќе не е луксуз бидејќи во денешно време е неизбежно да се има дом во кој ќе има „паметни“ уреди кои позитивно го менуваат начинот на живеење. Како и да е, технологијата игра исклучително

голема улога во нашите животи. Со усвојувањето на поновите технологии како Интернет на нештата (IoT) и вештачката интелигенција (ВИ), автоматизацијата во домот станува уште поголем дел од нашето секојдневие. Навистина, паметните домови стануваат моќно средство за решавање на секојдневните рутински задачи.

6.2.1. Вештачка интелигенција и Интернет на нештата во паметните домови

Во денешно време, големите индустриски компании како што се *Apple, Google* и *Amazon* сè повеќе се фокусираат на своите паметни асистенти или вештачка интелигенција за домашна автоматизација.

Вештачката интелигенција, покрај другите работи, подразбира и капацитет што може автоматски да извршува обврски и задачи за корисникот. Способноста за извршување задачи генерално се потпира на информациите што ги собрала рамката и ги „добила од“ или биле „подготвени за“ искористување на асортиман на алгоритми за машинско учење или длабоко учење. IoT или Интернет на нештата се однесува на иновацијата што ги овластува уредите и апаратите да пренесуваат информации преку интернет. Значи, тие можат да се контролираат од далечина, со говорни команди на корисникот или со испраќање податоци за управување на нивниот статус преку команди.

Уредите и апаратите се генерално електрични уреди како основно осветлување, вентилатори, машини за перење, фрижидери итн. Во случај со паметните домови, рамката на IoT ги дава информациите и вештачката интелигенција ги користи тие информации да изврши специфични активности кои ќе ги олеснат човечките оптоварувања. Откако се интегрираат ВИ и IoT, паметните уреди почнуваат далечински да одговараат на гласовните команди на корисникот или преку однапред програмирана команда. Еден исклучително забележителен случај за ова е термостатот на *Google – Nest*, кој постојано добива информации од однесувањето на жителите за тоа како се работи, а потоа ги користи тие податоци за автоматско поставување на температурата за да се гарантира удобност додека жителите се дома и обезбедува енергетска ефикасност доколку никој не е присутен во домот.

6.2.2. Како интеграцијата на ВИ и IoT влијае врз паметните домови

ВИ во паметните домови може да ги претвори податоците на сензорите од поврзаниот паметен уред во дизајн на однесување што е важен за нашето секојдневие. Уредите интегрирани со ВИ ја учат рутината на жителите и почнуваат да предвидуваат од наученото искуството што соодветствува. На пример, ако нема никој присутен во домот, уредот нема да го вклучи греењето, вентилаторот или светлата и автоматски ќе ги заклучи вратите.

Потенцијалот што го нудат IoT и ВИ не е само ограничен за нови домови, бидејќи има голем избор што дозволува постојните уреди да се прилагодат, за да им се даде далечински пристап преку паметни апликации или ВИ зависна од облак сервери и сл. ВИ & IoT во *Smart Home* се дефинитивно добитна комбинација за паметни домови.

Со интеграција на ВИ и IoT, паметните домови многу значајно помагаат за заштеда на енергијата и обезбедување поголема безбедност. Може да се заклучи дека паметните домови даваат чувство на задоволство и сигурност за следното ниво на високотехнолошко живеење. Во иднина се претпоставува дека IoT и ВИ ќе направат чуда за автоматизација на паметните домови и дека ќе бидат истрајни во тоа.

6.3. Детекција на аномалии

Како луѓе, нашиот мозок е секогаш подготвен да забележи нешто надвор од „нормалното“ или од „вообичаеното“. Накратко, некоја аномалија што не одговара на вообичаеното. Со обилниот раст на податоците, алатките за изучување на податоци исто така бараат аномалии кои не се совпаѓаат со нормалниот проток на податоци. На пример, „невообичаено висок“ број на обиди за најавување на некоја веб-страница или систем може да укаже на потенцијален напад во мрежата или големо покачување на трансакциите со кредитни картички за краток период, потенцијално може да биде измама со кредитни картички.

Во исто време, откривањето на аномалии на постојан прилив на неструктурирани податоци од различни извори има свои предизвици. Пример за предизвик е да се претпостави дека мнозинството трансакции со кредитни картички се легитимни и соодветни, додека се бараат големи отстапувања во неколку трансакции што не се во рамките на „нормалната“ граница.

Благодарение на растот на различни технологии за длабоко учење, денес откривањето на аномалии користејќи машинско учење (*ML*) е практично решение. Алгоритмите за машинско учење можат да се употребат за да се дефинираат нормални податочни шеми кои користат модели од машинското учење за да пронајдат отстапувања и аномалии.

6.3.1. Што е откривање на аномалии?

Откривањето на аномалии, често е поврзувано со откривање на нетипични/гранични податоци (*outlier detection*). Тоа е едноставен начин на откривање и идентификување на аномални податоци во кој било настан или набљудување на податоци кои се разликуваат од останатите податоци. Аномалните податоци можат да бидат клучни за откривање на ретка податочна шема (*data pattern*) или потенцијален проблем во финансиски измами, медицински состојби или опрема која работи неправилно.

Слика 3: Пример за детектирање на аномалии

Figure 3 Example of anomaly detection

6.3.1.1. Вообичаен пристап за откривање аномалии

Постоечките модели се тренираат да видат што претставува „нормално“, а потоа, сè што не одговара на оваа дефиниција го етикетираат како аномалија. Скоро секој алгоритам има свој начин на дефинирање на нормална точка/примерок. Некои го прават тоа преку статистички методи, други користат класификација или кластерирање, но на крајот, процесот останува ист: дефинирај „нормално“ и филтрирај сè останато.

Проблем кај вообичаените методи е што тие не се оптимизирани за откривање аномалии. Тие се оптимизирани за да најдат нормални случаи, поради што резултатот на откривање на аномалија или содржи премногу лажни позитивни или може да детектира премалку аномалии. Многу од овие методи се компјутерски сложени и оттаму одговараат на ниско-димензионални и/или мали податоци.

6.3.2. *Isolation Forest*

Алгоритмот „*Isolation Forest*“ ги опфаќа и горенаведените проблеми и обезбедува ефикасен и точен начин за откривање на аномалии.

Јадрото на овој алгоритам е да се „изолираат“ аномалиите со создавање на дрва на одлука над случајни атрибути. Случајната поделба произведува забележливи пократки патеки за аномалии бидејќи:

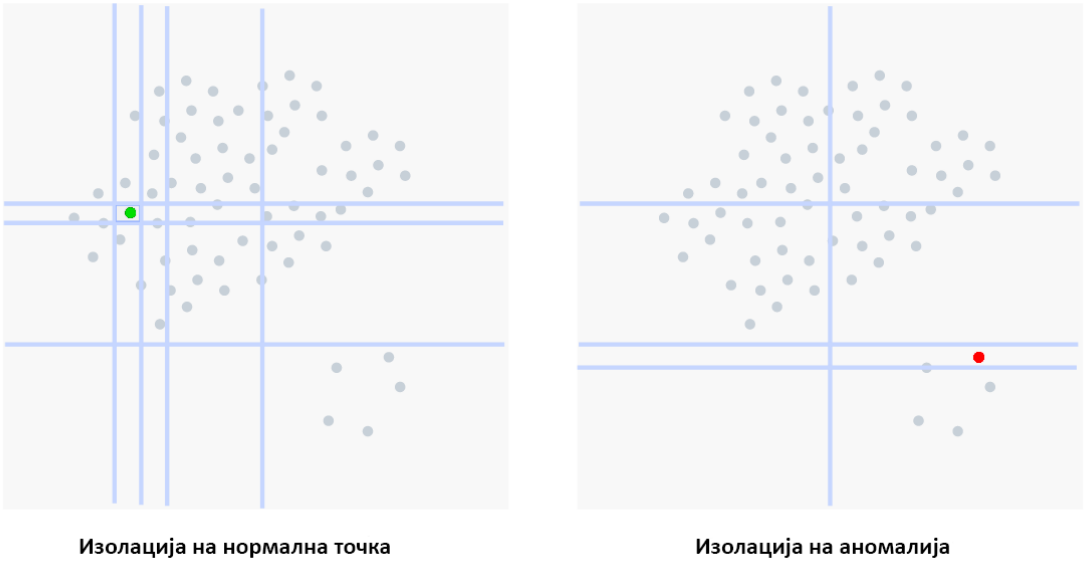
1. помалку инстанци (на аномалии) резултираат во помали партиции
2. различните атрибутни вредности се со поголема веројатност да бидат одделени при рано партиционирање.

Наједноставно објаснување за овој алгоритам е:

1. се зема произволна карактеристика
2. се партиционира произволно во границите

постпката се повторува се додека секоја точка не се изолира.

Оттука, кога „шума“ од случајни дрва колективно произведува пократки должини за некои посебни точки, тогаш многу веројатно се работи за аномалии.



Слика 4: Приказ на изолација на нормална точка и аномалија

Figure 4 Display of isolation of normal point and anomaly

Дијаграмот прикажан на сликата погоре го покажува бројот на поделби потребни за изолирање на нормална точка и аномалија. За изолација на нормална точка (точка во граници) вообичаено се потребни повеќе партиционирања односно поделби што е приметливо и на слика 4, а наспрати тоа за изолирање на аномалија (вредност која отстапува од границите) се потребни помалку. Поделбите претставени преку сини линии, се случуваат по случаен избор врз случаен атрибут и во процесот на градење на дрво на одлука. Бројот на поделби го одредува нивото на кое се случува изолацијата и ќе се искористи за да се генерира резултат на аномалија.

Процесот се повторува повеќепати и се бележи нивото на изолација за секоја точка/примерок. Откако ќе завршат повторувањата се генерира резултат на аномалија за секоја точка/примерок, што укажува на неговата веројатност да биде аномалија. Резултатот е функција на просечното ниво на кое точката била изолирана. Најголемите (*The top*) „*m*“ точки собрани врз основа на резултатот, се означени како аномалии.

6.3.2.1. Конструкција на дрво на одлука

Дрвото на одлука се конструира со поделба на точките/примероците на подпримерокот врз поделена вредност на случајно избраниот атрибут, така што примероците чија соодветна вредност на атрибутот е помала од вредноста на поделбата, одат лево, а другите одат десно. Процесот продолжува рекурзивно сè додека дрвото не се изгради целосно. Вредноста на поделбата се избира случајно помеѓу минималните вредности на избраниот атрибут.

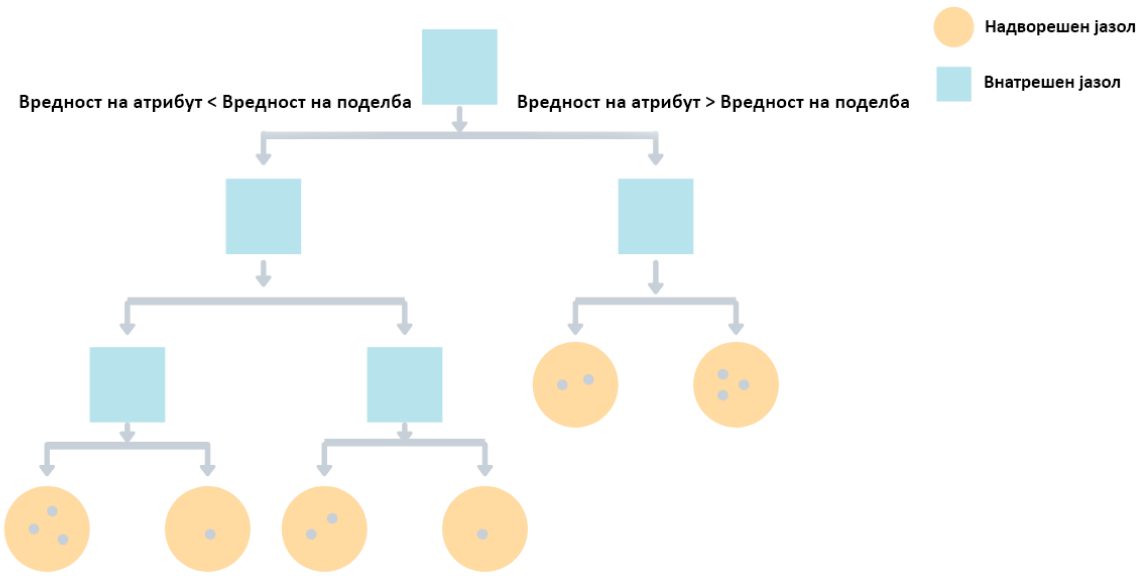
Постојат два вида на јазли во дрвата на одлука:

Внатрешен јазол

Внатрешните јазли се без лисја и содржат поделена вредност, атрибут на поделба и покажувачи на две под-дрвја на деца. Внатрешен јазол е секогаш родител на две под-дрва на деца што го прави целото дрво на одлука правилно бинарно дрво.

Надворешен јазол

Надворешните јазли се јазли на лисјата кои не можат да се поделат понатаму и се наоѓаат на дното од дрвото. Секој надворешен јазол ја задржува големината на неизграденото под-дрво што се користи за пресметување на резултатот на аномалијата.

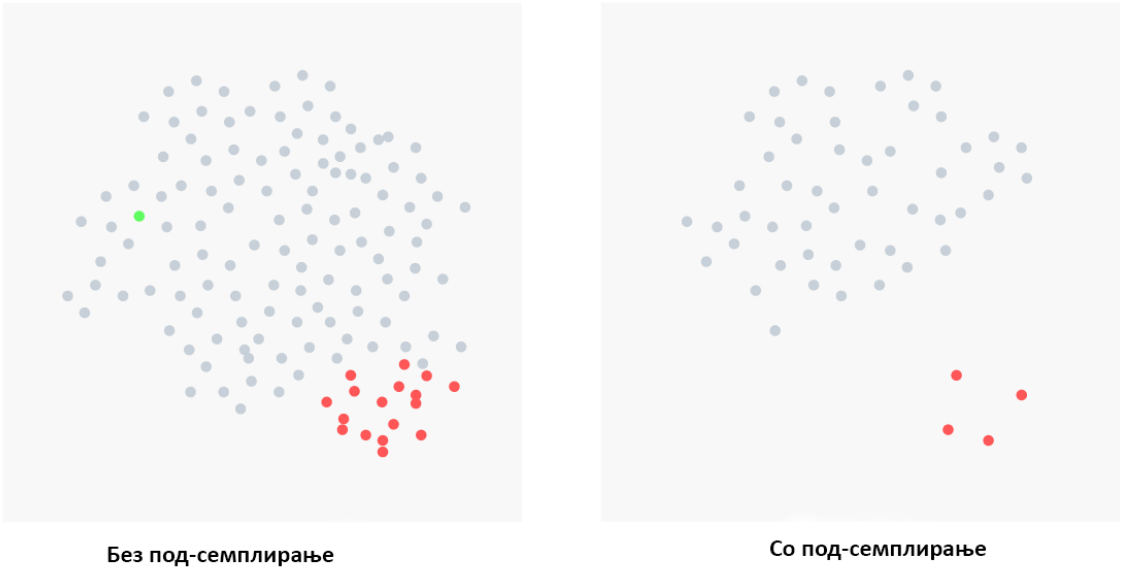


Слика 5: Конструкција на дрво на одлука

Figure 5 Construction of decision tree

6.3.2.2. Зошто помага под-сземплирањето на податоци?

Алгоритмот „*Isolation Forest*“ работи добро кога дрвата се креирани, не од целиот датасет, туку од збир на податоци од под-семплираниот (*sub-sampling*) датасет. Семплирање е селектирање на мал број на податоци од некој голем сет од податоци. Под-семплирањето се користи бидејќи „*Isolation Forest*“ нема потреба да ги изолира сите нормални точки, односно голем дел од тренинг примероците ги игнорира. Како последица на тоа, работи многу добро кога големината на земани примероци е мала, својство што е многу различно од скоро сите други техники кои напредуваат врз основа на податоците и бараат повеќе податоци за поголема точност. Под-семплирањето на податоци во овој алгоритам се користи бидејќи нормалните примероци може да се мешаат во процесот на изолирање со тоа што ќе бидат поблиску до аномалиите.



Слика 6: Приказ како под-семплирањето прави поделба помеѓу нормалните точки и аномалиите

Figure 6 Overview on how sub-sampling make difference between normal points and anomalies

На сликата погоре е прикажано како под-семплирањето на примероци всушност прави јасна поделба помеѓу нормалните точки и аномалиите. Во оригиналниот сет од податоци, може да се види дека нормалните точки и точки блиску до аномалиите го прават откривањето потешко и неточно (со многу лажни негативни примероци). Поради под-семплирањето, може да се види јасна

поделба на аномалиите и нормалните случаи. Ова го прави целиот процес на откривање аномалија ефикасен и точен.

6.3.2.3. Оптимизирање на конструкцијата на дрвата за одлука

Бидејќи аномалиите се подложни на изолација и имаат тенденција да се наоѓаат поблиску до коренот на дрвото на одлука, истото се конструира сè додека не достигне одредена висина *max_height* и не се разгранува понатаму. Оваа висина е висина на столбот за кој сме (скоро) сигурни дека не може да има никакви аномалии.

6.3.2.4. Конструирање на шумата

Процесот на изградба на дрво се повторува повеќепати и секој пат кога ќе се избере случаен под-примерок се конструира дрво. Не постојат строги правила за одредување на бројот на повторувања, но генерално, може да се каже колку повеќе, толку подобро. Бројот на под-земање е исто така параметар и може да се промени во зависност од сетот од податоци.

Додека се конструира дрвото, се надминува *max_height* висината како *log2* (број на јазли) така што тоа е просечната висина на соодветното бинарно дрво што може да се конструира од бројот на јазли. Бидејќи аномалиите се наоѓаат поблиску до коренот на јазолот, многу малку е веројатно дека некоја аномалија ќе се изолира откако дрвото ќе ја достигне максималната висина. Ова помага да се заштедат многу пресметки и изградби на дрвја што го прави овој алгоритам ефикасен и компјутерски и мемориски.

6.3.2.5. Резултат на аномалии

Секој алгоритам за откривање аномалија мора да ги оцени своите податочни точки/примероци и да ја измери довербата што алгоритмот ја има на потенцијалните аномалии. Генерираната оценка за аномалијата треба да биде ограничена и споредлива. Во *Isolation Forest*, фактот дека аномалиите секогаш остануваат поблиску до коренот, станува упатство и дефинирање што помага да се изгради функцијата за пресметка на резултатот. Резултатот на аномалии е функција од должината на патеката и е дефинирана како:

*Должина на патот **h (x)** на точка **x** е бројот на рабови **x** што поминуваат од коренскиот јазол.*

Бидејќи максималната можна висина на дрвото расте по редослед од **n**, просечната висина расте по **log (n)** – ова го прави нормализирањето на функцијата за резултат малку незгодно. За да се поправи ова, се користи увид во структурата на дрвото на одлука. Дрвото на одлука има два вида јазли, односно внатрешни и надворешни, така што надворешните немаат деца, додека внатрешното е родител на точно два јазли – што значи дека дрвото на одлука е правилно бинарно дрво и затоа може да се заклучи:

*Просечната должина на пат **h (x)** за завршување на надворешниот јазол е иста со просечната должина на патеката за неуспешно пребарување во BST (бинарно структурирано дрво).*

Во *BST*, неуспешното пребарување секогаш завршува на покажувачот *NULL* и ако го третираме надворешниот јазол на дрвото на одлука како *NULL* на *BST*, тогаш може да се каже дека просечната должина на пат на извршување на надворешен јазол е иста со просечната должина на пат на неуспешно пребарување во *BST* (конструиран само од внатрешни јазли на дрвото на одлука), дадено од:

$$c(n)=2H_{n-1} - \frac{2(n-1)}{n}$$

каде што **H (i)** е „хармоничен број“ и може да се процени со **ln (i) + 0.5772156649** (константа по Ојлер-Машерони). Просекот на должината на патот **h (x)** на даден **n** е наведена како **c (n)** и се користи за нормализирање на **h (x).**

Оценката на аномалиите на примерокот **x** се дефинира како:

$$S(x,n) = 2^{-\frac{E(h(x))}{c(n)}}$$

каде x е нова точка (data point), n е големина на примерок (sample size), S е вредност за оценката на аномалија, E е очекувана вредност, h(x) е просечна висина на пребарување за x од разни изолациони дрва кои се конструирани, c(n) е просечна вредност за h(x) што значи должината на потребниот пат за да се најде било кој јазол низ сите дрва (ова кажува колкава е длабочината на

пронаоѓање на x низ сите изолациони дрва кои се изградени, споредена со длабочината на пронаоѓање на просечна точка).

Од функцијата за оценка дефинирана погоре, може да се заклучи дека:

$$E(h(x) \ll c(n) \Rightarrow S(x,n) \cong 1$$

Доколку резултатот е блиску до 1, тогаш тие дефинитивно се аномалии.

$$E(h(x) \cong c(n) \Rightarrow S(x,n) \cong 0.5$$

Доколку резултатот е многу помал од 0.5, тогаш тие се сосема безбедни за да се сметаат за нормални примероци.

Доколку сите примероци се околу 0.5, тогаш целиот сет од податоци навистина нема посебна аномалија.

6.3.2.6. Оценување на аномалиите

Во фазата на проценка, резултатот од аномалиите се изведува од очекуваната должина на патеката **E (h (x))** за секој примерок на тестот. Со користење на функцијата *get_path_length*, должината на единствената патека **h (x)** се пресметува со поминување низ дрвото на одлука.

Доколку повторувањето завршува на надворешен јазол, каде што големината е поголема од 1, тогаш повратната вредност е **e** (број на рабови пресечени од тековниот јазол плус прилагодување **c (големина),** проценето од формулата прикажана погоре. Ова прилагодување е за неизградено дрво на одлука (за ефикасност) над максималната висина. Кога се добива **h (x)** за секој јазол од дрвото, резултатот на аномалија се добива со пресметување на **s (x, големина на примерок).** Со подредувањето на примероците според резултатите **s** по опаѓачки редослед и добивањето на врвот **m** се добиваат аномалиите.

Алгоритмот *Isolation Forest* се заснова на под-семплирањето на податоци и нема потреба да го гради дрвото од целиот сет од податоци, односно работи добро со податоци од под-семплирањето. Додека се конструира дрвото, не треба да се гради дрво повисоко од максималната висина. Бидејќи алгоритмот не зависи од операции кои се „скапи“ за пресметување како растојание или густина на пресметка, тој се извршува навистина брзо. Фазата на обука има линеарна

сложеност во времето со мала константа и затоа може да се користи и во онлајн систем во реално време (*real-time*).

7. Користени технологии

Со цел да се постигне лесен за користење, убаво дизајниран, брз и ефикасен систем се искористени најнови технологии. Детален опис за користените технологии е прикажан во точките подолу.

7.1. Софтвер

7.1.1. Spring Boot

Spring Boot обезбедува добра платформа за развивачите кои го користат *Java* програмскиот јазик да развијат самостојна и продукциска апликација. Може да се започне со минимални конфигурации без потреба за целосно поставување на *Spring*.

Spring Boot, на своите развивачи им ги нуди следниве предности [39]:

1. лесно разбирање и развивање *Spring* апликации
2. ја зголемува продуктивноста
3. го намалува времето за развој.

Оваа платформа е дизајнирана со следниве цели:

1. да се избегне комплексната конфигурација на *XML* во *Spring*
2. на полесен начин да се развијат апликации спремни за продукција
3. да се намали времето за развој и апликацијата да се изврши независно.

Spring Boot може да се избере поради своите карактеристики и поволности што ги нуди. Некои од нив се [40]:

1. Овозможува флексибилен начин на конфигурирање на *Java Beans, XML* конфигурации и трансакции со бази на податоци.
2. Директно вграден сервер *Tomcat, Jetty* или *Undertow* (нема потреба да се додаваат *WAR* датотеки).
3. Овозможува моќно обработувачко процесирање и управува со *REST* крајните точки (*endpoints*).
4. Во *Spring Boot*, сè е автоматски конфигурирано, односно не се потребни рачни конфигурации
5. Нуди *Spring* апликација базирана на прибелешки (*Annotation-based*).
6. Го олеснува управувањето со зависноста (*dependency management*).

7. Вклучува контејнер за вграден *Servlet (Embedded Servlet Container*).
8. Обезбедува карактеристики спремни за продукција како метрики, „здравствени проверки“ и надворешна конфигурација.

7.1.2. *Angular*

Ангулар е *JavaScript* рамка за градење веб апликации и апликации во *JavaScript*, *HTML* и *TypeScript*, со што претставува одличен сет на *JavaScript*. Ангулар обезбедува вградени карактеристики за анимација, *HTTP* услуга и материјали кои за возврат имаат особини како што се автоматско комплетирање, навигација, лента со алатки, менија итн. [41]. Кодот е напишан во *TypeScript* кој се базира на *JavaScript* и го прикажува истото на пребарувачот.

Главни карактеристики на Ангулар се:

1. Детална документација – Ангулар содржи детална документација каде што развивачите може да ги пронајдат сите потребни информации. Како резултат, развивачите можат брзо да дојдат до своите технички решенија и да ги решат итните проблеми.
2. Подржан е од *Google* – многу програмери сметаат дека поддршката на *Google* е друга придобивка на Ангулар, што ја прави платформата доверлива. На ng-conf2017, развивачите на Ангулар потврдија дека *Google* ќе го подржува Ангулар на долгорочна основа.
3. Одличен екосистем на 3d-party компоненти [42] – популарноста на Ангулар резултираше со појава на илјадници дополнителни алатки и компоненти што може да се користат во Ангулар апликациите. Како резултат може да се добијат дополнителни функционалности и подобрувања во продуктивноста.
4. Архитектура заснована на компоненти – во втората верзија, Ангулар се префрли од *MVC* (*Model-View-Controller*) во архитектура базирана на компоненти. Според оваа архитектура, една апликација е поделена на независни логички и функционални компоненти. Овие компоненти лесно можат да се заменат и раздвојат, како и повторно да се користат во други делови од апликацијата. Покрај тоа, независноста на компонентите го олеснува тестирањето на веб-апликација и гарантира дека секоја компонента работи непречено.

5. Предвремен компајлер (*Ahead-of-time compiler*) – компајлерот на Ангулар АОТ ги претвора *TypeScript и HTML* во *JavaSript* за време на процесот на градење. Ова значи дека кодот е составен пред прелистувачот да ја вчита веб-апликацијата, така што таа ќе се прикажува многу побрзо. Компајлерот АОТ е исто така многу побезбеден од навремениот (*just-in-time*) *JIT* компајлер.
6. *Command Line Interface* (*CLI*) [43] – ова е веројатно најпосакуваната одлика на мнозинството Ангулар програмери. Го автоматизира целиот процес на развој, правејќи ги иницијализацијата, конфигурацијата и развојот на апликациите што е можно полесно. *CLI* овозможува да се креира нов Ангулар проект, да се додадат карактеристики (*features*), извршна единица и *крај-до-крај* (*end-to-end*) тестови со неколку едноставни команди. Не само што го зголемува квалитетот на кодот, туку и во голема мера го олеснува развојот.
7. Ангулар може да се користи за изработка на апликации за различни платформи како веб, мобилен веб, мобилен телефон и компјутер.

7.1.3. *MQTT* против *HTTP*: кој е најдобар за IoT?

HTTP е најпопуларниот и најчесто користениот протокол. Но, во последниве години *MQTT* зема залет. Програмерите често имаат дилема околу изборот помеѓу овие два протокола кога станува збор за развој на IoT [44].

7.1.3.1. Дизајн и пораки

MQTT е податочно ориентиран, додека *HTTP* е документски ориентиран. *HTTP* е барање - одговор (*request-response*) протокол за компјутерски клиент - сервер (*client-server*) и не секогаш е оптимизиран за мобилни уреди. Главните придобивки на *MQTT* во овие услови се леснотиите (*MQTT* пренесува податоци како низа од бајти) и содржи „објави/претплати“ (*publish/subscribe*) модел, што го прави совршен за уредите со ограничени ресурси и помага во заштедата на енергија.

И покрај тоа, моделот на „објави/претплати“ (*publish/subscribe*) им овозможува на клиентите да бидат засебни и да не зависат едни од други. Кога еден уред ќе се исклучи, целиот систем може да продолжи да работи правилно.

7.1.3.2. Брзина и испорака

Според мерењата во 3G мрежите, пропусната моќ на *MQTT* е 93 пати побрза од *HTTP*. Покрај тоа, во споредба со *HTTP*, *MQTT* протоколот обезбедува високи гаранции за испорака.

Постојат 3 нивоа на квалитет на услугите:

1. најмногу еднаш: гарантира дека пораката ќе биде доставена најмногу еднаш
2. барем еднаш: гарантира дека пораката ќе биде доставена барем еднаш
3. точно еднаш: гарантира дека пораката ќе биде примена само еднаш.

MQTT исто така на корисниците им ги овозможува опциите за „последен завет и тестамент“ (*Last will & Testament*) и задржани пораки (*Retained messages*). Првото значи дека во случај на неочекувано исклучување на клиент, сите претплатени (*subscribed*) клиенти ќе добијат порака од брокерот. Задржани пораки (*Retained messages*) значи дека новопретплатените клиенти ќе добијат инстантно ажурирање на статусот.

HTTP протоколот не содржи ниту една од овие способности.

7.1.3.3. Комплексност и големина на пораките

MQTT има прилично кратка спецификација. Постојат само *CONNECT, PUBLISH, SUBSCRIBE, UNSUBSCRIBE* и *DISCONNECT* типови кои се значајни за програмерите, додека пак спецификациите на *HTTP* се поголеми.

MQTT има кратко заглавие на порака и најмала големина на пакет од само 2 бајта. Користењето на формат на текстуална порака му овозможува на *HTTP* протоколот да составува долги заглавија и пораки. Тоа помага да се отстранат проблемите (*troubles*) бидејќи луѓето можат да ги читаат, но во исто време не е потребно за уредите кои се ограничени со ресурси.

Табела 2: Споредба на MQTT и HTTP

Table 2 Comparison between MQTT and HTTP

Карактеристики / Feautures	MQTT	HTTP
Целосна форма	Message Queue Telemetry Transport	Hyper Text Transfer Protocol
Архитектура	Работи на објави/претплати (publish/subscribe) модел	Работи на барање/одговор (request/response) модел
Комплексност	Едноставен	Покомплексен
Протокол на горниот слој (работи на)	Работи на TCP	Работи на UDP
Дизајн на протокол	Податочно ориентиран	Документски ориентиран
Големина на порака	Мала	Голема
Формат на порака	Бинарна порака со заглавје од 2 бајти	Во ASCII формат
Безбедност на податоци	Овозможува безбедност на податоците со SSL/TLS	Нема, но затоа е направен HTTPS
Нивоа на услуги	3	1
Дистрибуција на податоци	Еден на повеќе	Еден на еден
Број на порта	Работи на 1883 порта	Работи на 80 или 8080 порта

MQTT протоколот е лесен за употреба. Тоа е од суштинско значење кога времето на одговор, пропусната моќност, помалата употреба на батеријата и ширината на опсег се на прво место за идните решенија. Исто така е совршено во случај на наизменична поврзаност.

HTTP е достоен и може да се надградува. Но, *MQTT* е посоодветен кога станува збор за развој на Интернет на нештата. Затоа нашиот избор за протокол се сведува на *MQTT*.

7.1.4. *MQTT*

MQTT или *Message Queuing telemetry Transport* е лесен, отворен и едноставен комуникациски протокол за испраќање пораки за употреба во случаи кога на клиентите им е потребен мал „отпечаток“ од кодови и се поврзани на несигурни мрежи или мрежи со ограничени ресурси. Првично е користен за комуникација на машина со машина (M2M) или за комуникација во IoT [45].

MQTT првично е создаден од д-р Енди Стенфорд-Кларк и Арлен Нипер во 1999 година. Првичната цел на овој комуникациски метод била да им овозможи на уредите за следење што се користат во нафтената и гасната индустрија да ги испраќаат своите податоци до оддалечени сервери. Во многу случаи, таквите уреди за следење се користеле на оддалечени локации каде што било тешко да се воспостави каква било фиксна, жична врска или радио-преносна врска. Во тоа време, единствена опција за таквите случаи биле сателитските комуникации кои биле доста скапи и се наплаќале врз основа на тоа колку податоци се користени. Со илјадници сензори на терен, индустријата имала потреба од облик на комуникација што може да обезбеди пренос на податоците користејќи минимален опсег на пораките. Основната цел им била да развијат протокол кој ќе има минимален пропусен облик и минимална потрошувачка на енергија. Покрај тоа имале за цел и да го направат лесен за имплементација, да може да пренесува различни типови на податоци, да има способност да одржува сесии и да обезбеди квалитет на сервисот при доставата.

Во 2013, според (*OASIS*), *MQTT* е стандардизиран како *open source*. *OASIS* сè уште управува со *MQTT* стандардот [46].

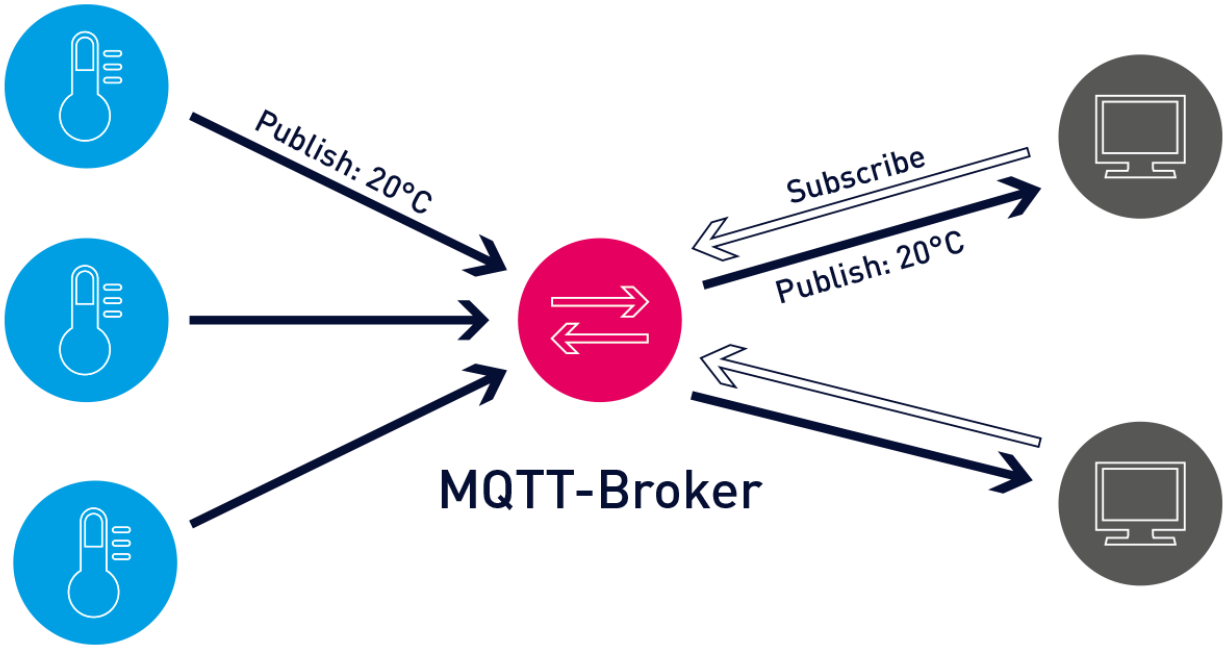
MQTT работи на *TCP/IP* користејќи објави/претплати (*PUSH/SUBSCRIBE*) топологија. Во архитектурата на *MQTT*, постојат два вида на системи:

1. клиенти
2. брокери.

Брокер е серверот со кој комуницираат клиентите. Брокерот прима комуникации од клиентите и ги испраќа тие комуникации до другите клиенти. Клиентите не

комуницираат директно едни со други, туку се поврзуваат на брокерот. Секој клиент може да биде објавувач (*Publisher*), претплатувач (*Subscriber*) или обете.

MQTT е протокол управуван од настани (*event-driven protocol*). Нема периодично или тековно пренесување на податоци. Ова го сведува пренесувањето на минимум. Клиент објавува само кога има информации, а брокер испраќа само кога пристигнуваат нови податоци.



Слика 7: Пример на MQTT-Broker

Figure 7 Example of MQTT-Broker

7.1.4.1. Архитектура на пораки.

Друг начин за *MQTT* да ги минимизира своите преноси е со мала и цврсто дефинирана конструкција на пораки. Секоја порака има фиксно заглавје од само 2 бајти. Може да се користи изборно заглавје, но се зголемува големината на пораката. Товарот на пораката (*message payload*) е ограничен на само 256 MB (мега бајти). Трите различни нивоа на квалитет на услуга (*QoS*) им овозможуваат на мрежните дизајнери да изберат помеѓу минимазирање на преносот на податоци и максимизирање на сигурноста.

***QoS* 0** – нуди минимален износ на пренос на податоци. Со ова ниво, секоја порака се доставува до претплатникот еднаш без потврда. Нема начин да се знае дали претплатниците ја примиле пораката. Овој метод понекогаш се нарекува „запали и заборави“ (*fire and forget*) или „*at most once delivery*“, односно

„најмногу едно доставување“. Бидејќи ова ниво претпоставува дека испораката е завршена, пораките не се зачувуваат за испорака до исклучените клиенти кои подоцна повторно се поврзуваат.

QoS 1 - брокерот се обидува да ја достави пораката и потоа чека одговор за потврда од претплатникот. Ако потврдата не е примена во одредена временска рамка, пораката се испраќа повторно. Користејќи го овој метод, претплатникот може да ја прими пораката повеќе од еднаш ако брокерот не добие навремена потврда од претплатникот. Ова понекогаш се нарекува „барем едно доставување“ (*at least once delivery*).

QoS 2 – клиентот и брокерот користат „ракување“ со четири чекори за да се осигурат дека пораката е примена и дека е примена само еднаш. Ова понекогаш се нарекува „точно едно доставување“ (*exactly once delivery*).

За ситуации кога комуникациите се сигурни, но ограничени, QoS 0 може да биде најдобра опција. За ситуации кога комуникациите не се веродостојни, но кога врските не се толку ограничени со ресурси, тогаш QoS 2, ќе биде најдобра опција. QoS 1 обезбедува еден вид најдобро решение и за двете страни, но бара апликацијата што ги прима пораките да знае како да се справи со дупликати.

И за QoS 1 и за QoS 2, пораките се зачувуваат или чекаат во редици за клиенти кои не се активни (*offline*) и имаат воспоставена постојана сесија. Овие пораки се испраќаат (според соодветното ниво на QoS) откако клиентот ќе стане активен [48].

7.1.4.2. Топици (теми)

Пораките во рамките на *MQTT* се објавуваат како топици. Топиците се структури во хиерархија користејќи црта (/) како карактер за разграничување. Оваа структура личи на дрвото на компјутерскиот датотечен систем. Структура како што се сензори */lights/livingRoom* му овозможува на претплатникот да одреди каде треба да се испраќаат податоците од клиенти што објавуваат на топикот *livingRoom* или за поширок поглед, можеби сите податоци од клиентите што ги објавуваат на која било тема на сензори/lights. Топиците не се експлицитно креирани во *MQTT*. Ако брокерот прима податоци објавени на топик кој не постои во моментов, топикот едноставно се креира и клиентите може да се претплатат на него.

7.1.4.3. Задржани пораки

За да се задржи мал отпечаток, приманите пораки не се чуваат во брокерот сè додека не се означени со знаменцето за задржување (*retained flag*). Ова се нарекува задржана порака. Корисниците кои сакаат да ги зачувуваат примените пораки ќе треба да ги зачувуваат некаде надвор од *MQTT* протоколот. Постои еден исклучок.

Како протокол управуван од настани, можно е, дури и веројатно, претплатникот да добие многу малку пораки за даден топик, дури и за подолг временски период. Во претходно споменатата структура на топици, можеби пораките до топикот */livingRoom* се испраќаат само кога корисникот ќе запали светло. Под претпоставка дека тоа е светло кое не се пали често, може да поминат и неколку денови пред клиентот да објави порака на тој топик.

За да биде сигурен дека новиот претплатник ја добил пораката од топикот, брокерот може да ја задржи последната порака испратена до секој топик. Ова се нарекува задржана порака. Секогаш кога нов клиент ќе се претплати на топик или кога веќе постоечки клиент ќе стане активен, задржаната порака се испраќа до претплатниците, со што се осигурува дека претплатата е активна и ги има најновите информации.

7.1.4.4. „Последен завет и тестамент“

Таму каде комуникациите не се сигурни, можно е објавувачот да се исклучи од мрежата без предупредување. Објавувачот може да регистрира порака што треба да им биде испратена на претплатниците во случај тој неочекувано да се исклучи, ова е т.н. последен завет и тестамент. Оваа порака е зачувана на брокерот и е испратена до претплатниците доколку објавувачот неправилно се исклучи. Типично, таквата порака вклучува информации што овозможуваат идентификување на исклучениот објавувач за да може да се преземат соодветни активности.

7.1.4.5. *MQTT* пораки

За да се задржи минималната големина на протоколот, има само четири можни активности со каква било комуникација: објави, претплати, отпиши се или пинг.

Publish **(објави)** – испраќа блок од податоци што ја содржат пораката што треба да се испрати. Овие податоци се специфични за секоја имплементација, но може

да бидат едноставни како индикација за вклучување/исклучување (*on/off*) или вредност на одреден сензор. Во случај да не постои топикот на кој се објавува, топикот се креира од брокерот.

Subscribe **(претплати)** – го претвора клиентот во претплатник на одреден топик. Топиците може да бидат претплатени на специфичен топик или преку валјд-карти кои овозможуваат претплата до целата гранка од која било гранка на топикот. За да се претплати, клиентот испраќа *SUBSCRIBE* (претплати) пакет и за возврат добива *SUBACK* пакет. Доколку пакетот содржи задржана порака за топикот, новиот претплатник ја прима и таа порака.

PING **(пинг)** – клиентот може да го пингува брокерот. Претплатникот испраќа пакет *PINGREQ* и за возврат добива *PINGRESP* пакет. Пинг опцијата може да се користи за да се осигури дали врската сè уште е активна и дали сесијата *TCP* не е неочекувано затворена од друга мрежна опрема како рутер или *gateway*.

DISCONNECT **(дисконект)** – претплатникот или објавувачот може да испратат *DISCONNECT* порака до брокерот. Оваа порака го информира брокерот дека повеќе нема да има потреба да испраќа или чека пораки за тој претплатник и дека повеќе нема да добива податоци од издавачот. Овој вид на исклучување му овозможува на клиентот повторно да се поврзе користејќи го истиот идентитет како порано. Кога клиентот ќе се исклучи без да се испрати порака за исклучување, „последниот завет и тестамент“ им се испраќа на претплатниците.

Бидејќи MQTT е бинарно базиран протокол и има формат за потврда со команди. Секогаш кога клиентот испраќа команда до брокерот, тој враќа потврда. Овој комуникациски протокол всушност е базиран на TCP/IP протокол. Затоа, прво ќе има остварување на TCP конекција, па MQTT конекција, па потоа трансферот на податоци ќе започне, по што се гаси TCP конекцијата.

За секоја функција, клиентот мора да испрати команда до брокерот. Податоците се испраќаат како пакети. Кога клиентот треба да објави порака, се извршуваат следниве чекори:

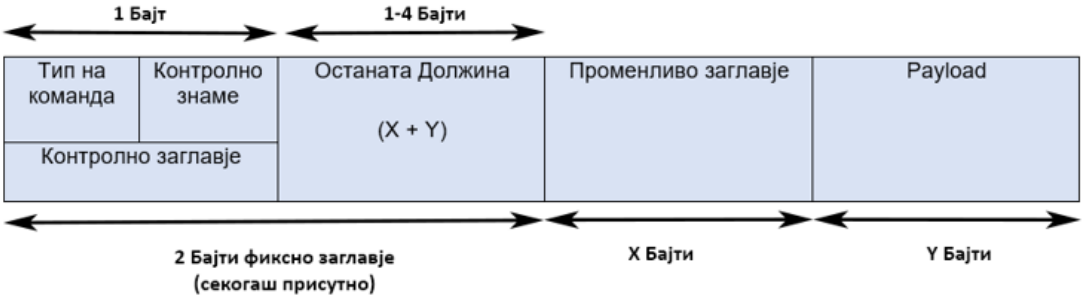
1. Клиентот треба да воспостави врска со брокерот преку испраќање на пакет за поврзување (со корисничко име и лозинка, доколку е потребно).

2. Се чека потврда за да се провери дали врската е овозможена или има некоја грешка.
3. Клиентот испраќа пакет за објавување што содржи име на тема (topic) и пораката што треба да се објави.
4. Се чека пакетот за одговор во зависност од нивото на квалитет (QoS).
    a. QoS 0: Нема да има никаков одговор
    b. QoS 1: PUBACK – Одговор за потврда на објавата
    c. QoS 2:
        i. се чека PUBREC – Објавата е примена
        ii. се испраќа назад PUBREL – Објавување на пораката
        iii. се чека PUBCOMP – Објавувањето е завршено
5. Доколку комуникацијата е завршена, клиентот може да се исклучи од брокерот со испраќање на пакет за исклучување.

Слична е постапката и со претплатувањето на тема (topic):

1. Се испраќа Connect пакет до брокерот за поврзување (Со корисничко име и лозинка, доколку е потребно).
2. Се чека за потврден Connect пакет од брокерот за да се провери дали врската е овозможена или има некоја грешка.
3. Кога ќе се прими потврдата, се испраќа пакет за претплата со соодветно име на тема (topic).
4. Се чека потврден пакет за претплатникот.

MQTT пакетот се состои од фиксно заглавје од 2 бајти, променливо заглавје и payload. Фиксното заглавје е секогаш присутно во сите пакети, а останатите две не.



Слика 8: Структура на заглавје

Figure 8 Header structure

7.1.4.6. Безбедност

Првичната цел на протоколот *MQTT* е да овозможи најмал и најефикасен пренос на податоци преку скапи и несигурни комуникациски линии. Како таков, безбедноста не била примарна грижа за време на дизајнот и имплементацијата на *MQTT*.

Сепак, постојат некои опции за безбедност на располагање по цена на поголем пренос на податоци и поголем „отпечаток“.

- **Безбедност на мрежата** – Ако самата мрежа може да биде безбедна, тогаш преносот на небезбедни податоци од *MQTT* е ирелевантен. Во овој случај, безбедносните проблеми треба да се појават од самата мрежа, можеби преку „лош актер“ (*bad actor*) односно натрапници или друга форма на мрежна пенетрација односно мрежно пробивање.
- **Корисничко име и лозинка** – *MQTT* дозволува кориснички имиња и лозинки за клиентот да овозможи врска со брокерот. За жал, тие се испраќаат како јасен текст (*clear text*). Сепак, денес пресретнувањето на многу видови безжични мрежни комуникации е тривијално, поради што таквата автентикација е бескорисна. Многу случаи на употреба бараат корисничко име и лозинка не како заштита од лоши намери, туку како начин да се избегнат несакани врски.
- ***SSL/TLS*** – Работи на врвот на *TCP/IP* и е очигледно решение за обезбедување преноси помеѓу клиенти и брокери. За жал, ова додава значителни трошоци за инаку лесните комуникации.

7.1.4.7. Брокер и клиент

Како брокер за *MQTT* во овој труд се користи *ActiveMQ* [49]. *Apache ActiveMQ* е брокер за пораки со отворен код, напишан во *Java*, заедно со целосен клиент за *Java Message Service (JMS)*. Овозможува „опции за претпријатија“ што во овој случај значи повикувања на комуникациите од повеќе клиенти или сервери. [3] Поддржува различни типови на клиенти. Комуникацијата се управува со карактеристики како компјутерско групирање (*computer clustering*) и можност да се користи која било база на податоци како *JMS persistence provides* покрај виртуелната меморија и кешот (*cache*). [4]

Користењето на пораките како стил на интеграција или комуникација доведува до многу придобивки како што се:

1. Дозволувањето на апликациите развиени во различни јазици и на различни оперативни системи да комуницираат едни со други.
2. Транспарентност на локацијата – апликациите на клиентите не мора да знаат каде се наоѓаат услужните апликации.
3. Сигурна комуникација – производителите/потрошувачите на пораки не мора да бидат достапни во исто време, или одредени сегменти од патот на пораката може да се намалат и да се појават без да влијаат на преносот на пораката.
4. Скалирање – може да скалира хоризонтално додавајќи повеќе сервиси (услуги) кои можат да се справат со пораки доколку пристигнуваат премногу пораки.
5. Асинхрона комуникација – клиентот може да испрати порака и да продолжи во друго процесирање наместо да блокира додека сервисот не врати повратен одговор. Може да се справи со повратната порака само кога пораката ќе биде спремна.

ActiveMQ има поддршка за *MQTT* протоколот и автоматски прави мапирање помеѓу *JMS/NMS* и *MQTT* клиентите.

Како клиент во овој труд е искористен *Eclipse Paho* клиентот [47]. *Eclipse Paho project* е библиотека која обезбедува имплементација на отворен код, главно од страна на клиентот, на *MQTT* и *MQTT-SN* на различни програмски јазици [5]. Во нашиов случај, преку програмскиот јазик *Java.*

7.1.5. *Python*

Python е интерпретиран, интерактивен, објектно ориентиран програмски јазик на високо ниво со динамична семантика. Создаден е од Гвидо ван Росум (*Guido van Rossum*) во 1991 година, а името го добил по култната британска комедија „*Monthy Python*“. Овој програмски јазик се карактеризира со висококвалитетна структура на податоци, што во комбинација со динамично пишување и поврзување, го прави многу привлечен за брз развој на разни апликации [52].

Неговата флексибилност овозможува да се постигнат многу резултати. Може да се користи за пишување едноставни програми, но исто така ја поседува потребната моќ за создавање сложени операции кои се користат во глобални компании како *Google, Facebook, Reddit, Dropbox, YouTube, Instagram, Pinterest* и ред други.

7.2 Хардвер

7.2.1. NodeMCU

Развојните плочи како Ардуино и Распбери Пи, најчесто се избираат за работа со нови IoT уреди. Овие развивачки плочи во суштина се мини компјутери кои можат да се поврзат и програмираат од стандарден компјутер. Откако ќе се програмираат, развојните плочи може да се поврзат и да ги контролираат поставените сензори [50].

Поради „И“ во IoT што претставува интернет, на овие развојни плочи им треба начин за да се поврзат на интернет. Најдобар начин да се конектираат е со користење на безжични мрежи. Сепак, Ардуино и Распбери Пи немаат вградена поддршка за безжични мрежи. Програмерите ќе треба да додадат *wifi* или мобилен модул на плочата и да напишат код за пристап до безжичниот модул.

Поради тоа, најдобар избор за изработка на IoT системи, комуникација со сензори и управување на уреди е *NodeMCU*. *NodeMCU* е развојна плоча IoT развивачка плоча со отворен код. Една од најуникатните карактеристики на *NodeMCU* е тоа што има вградена поддршка за поврзување со wifi, а со тоа и го олеснува развојот на апликациите за IoT. Останати предности на *NodeMCU* спрема Ардуино се, помала цена, помала големина на плочата и помала потрошувачка на енергија.

NodeMCU (Node MicroController Unit) е отворена развивачка околина за хардвер и софтвер што е изградена околу евтиниот систем на чип *SoC (System-on-a-Chip)* наречен *ESP8266*. Овој чип е дизајниран и произведен од *Espressif Systems* во 2013 година и ги содржи сите клучни елементи на современиот компјутер: процесор, RAM меморија, вмрежување (*wifi*), па дури и модерен оперативен систем и *SDK*. Ниската цена го прави уште подостапен и еден од најдобрите избори за IoT проекти од сите видови.

Сепак, како чип, *ESP8266* е исто така тежок за пристап и употреба. Жиците мора да се залемат со соодветен аналоген напон, до неговите пинови за наједноставни задачи како напојување или испраќање на „команда“ до „компјутерот“ на чипот. Исто така, мора да се програмира на ниско ниво со машински упатства што може да се толкуваат од хардверот на чипот. Иако ова ниво на интеграција не е проблем доколку *ESP8266* се користи како вграден чип контролер во масовно произведена електроника, сепак тоа претставува огромен товар за развивачите. За да се спречат овие недостатоци се користат „*jumper-wires*“ скокачки жици кои лесно се вклопуваат во пиновите или се користи основна плоча за *NodeMCU* каде што може да се користат и обични жици.

LoLin NodeMCU v3 е IoT платформа која го користи скриптирачкиот јазик *Lua*. Проектот *eLua* е основа на плочата и е изграден на *ESP8266 SDK 1.4. NodeMCU* користи многу проекти со отворен код, како што се *lua-cjson* и *spiffs* [51].

Овие плочи дополнително имаат вградено и *CH340G USB/UART* конвертор, како и раздвоено напојување со *LDO*. Исто така на плочата има две минијатурни копчиња за притискање.

Важни карактеристики на *NodeMCU:*

1. хардвер сличен на Ардуино
2. може да се програмира исто како Ардуино, но интерактивно во Луа скрипта
3. АПИ за интернет апликации управувано од настани што го олеснува пишувањето на код
4. интегрирани *GPIO, PWM, IIC, 1-Wire* и *ADC* на самата плоча
5. 10 *GPIO*, каде секој *GPIO* може да биде *PWM*, I2C, 1-*wire*
6. 4M *Flash* меморија
7. вградена *WiFi* антена.

Табела 3: Спецификации на *NodeMCU*

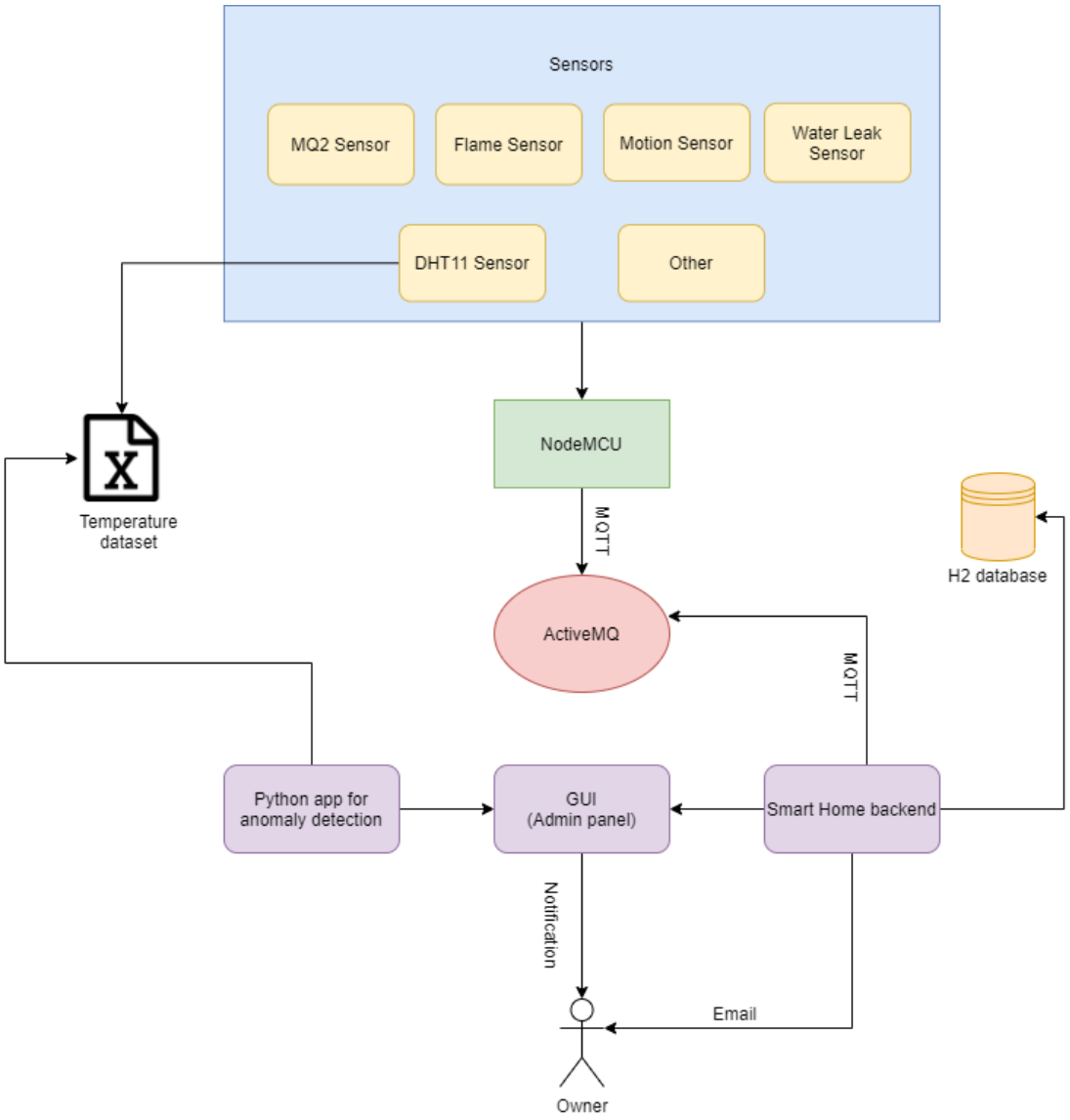
Table 3 NodeMCU specifications

Микроконтролер	ESP8266
Работна волтажа	3.3V
Напојување	7V – 12V
Моментална потрошувачка	15 µA – 400 mA
Потрошувачка во „длабок сон“	0.5 µA
Дигитални пинови	16
Дигитални пинови со PWM	16
Аналогни пинови	1
SPI / I2C / I2S / UART	2 /1 /2 /2
Еднонасочна потрошувачка по влезни/излезни пинови	12 mA
POM меморија (Флеш)	4 MB
SRAM	64 KB
EEPROM	512 бајти
Брзина на процесирање (Clock Speed)	80 MHz
Должина	48 mm
Ширина	31 mm
WiFi	Да
Micro USB	Да

8. Опис на проектот

За да се доловат дел од работите кои може да се имплементираат во нашите домови на паметен и ефективен начин и да се покажат придобивките од употребата на модерни алатки е изработен систем. Целта на системот е алармирањето и известувањето при активирање на некој од сензорите. Исто така цел е и со користењето на техники од вештачката интелигенција да се прикаже квалитетот на работа на еден сензор и контролирање на уред преку гласовни команди на македонски јазик. За изработката на овој систем се направени и искористени неколку компоненти кои заедно прават една целина која има улога на систем за безбедност на паметен дом.

За подобро да се долови овој систем е изработена архитектурна слика која ги прикажува главните компоненти и нивната поврзаност и комуникација.



Слика 9: Архитектура на системот

Figure 9 Architecture of the system

Како што може да се види на архитектурната слика, овој проект се состои од неколку компоненти:

1. *Smart Home backend* апликација
2. *Angular* апликација
3. *Python* апликација за детектирање на аномалии
4. *NodeMCU* програма.

8.1. Софтвер

8.1.1. *Smart Home backend* апликација

Во овој дел ќе биде претставен развојот на *Smart Home backend* апликацијата. Како што е наведено претходно, оваа апликација е јадрото на целиот систем. Овде се чуваат сите аларми, се препраќаат команди од графичкиот интерфејс до брокерот и се препраќаат мерењата од сензорите од *NodeMCU* до графичкиот интерфејс. Оваа апликација ги собира пораките испратени на брокерот од страна на *NodeMCU*. Па така доколку се активира некој од сензорите, *NodeMCU* ќе испрати порака на брокерот за аларм, која *Smart Home* апликацијата ќе ја преземе и ќе креира аларм, ќе го испрати алармот преку *WebSocket* до Ангулар апликацијата. Алармот се зачувува во базата на податоци и за истиот се испраќа е-пошта до сопственикот преку *Smart Home* апликацијата.

За развојот на *Smart Home backend* апликацијата е искористено *STS* студио за креирање на *Java* и *Spring Boot* апликации. Верзијата на *Java* користена во овој проект е:

```
<properties>
	<java.version>1.8</java.version>
</properties>
```

Верзијата на *Spring Boot* која е искористена во проектот *Smart Home* е:

```
<parent>
	<groupId>org.springframework.boot</groupId>
	<artifactId>spring-boot-starter-parent</artifactId>
	<version>2.1.0.RELEASE</version>
	<relativePath />
</parent>
```

Како што може да се види на архитектурната слика, оваа апликација користи H2 (X2) датабаза.

H2 е лесна база на податоци за *Java* со отворен код [53]. Може да биде вградена во Јава апликации или да работи во режим клиент-сервер. Оваа база на податоци може да работи како база на податоци во меморија, што значи дека податоците нема да опстојуваат на дискот.

Главни карактеристики на H2 базата на податоци:

1. екстремно брза база на податоци
2. има отворен код кој е напишан во *Java*

3. поддржува стандардни *SQL* и *JDBC API*. Може да користи и *PostgreSQL ODBC* драјвер
4. има вграден режим, и режим на клиент / сервер
5. поддржува групирање (кластери)
6. има силни безбедносни карактеристики.

За поврзување со H2 базата претходно треба во документот со библиотеки pom.xml да се додаде библиотеката за оваа база. Библиотеката се додава како *maven dependency* (зависност) на следниов начин:

```
<dependency>
	<groupId>com.h2database</groupId>
	<artifactId>h2</artifactId>
	<scope>compile</scope>
</dependency>
```

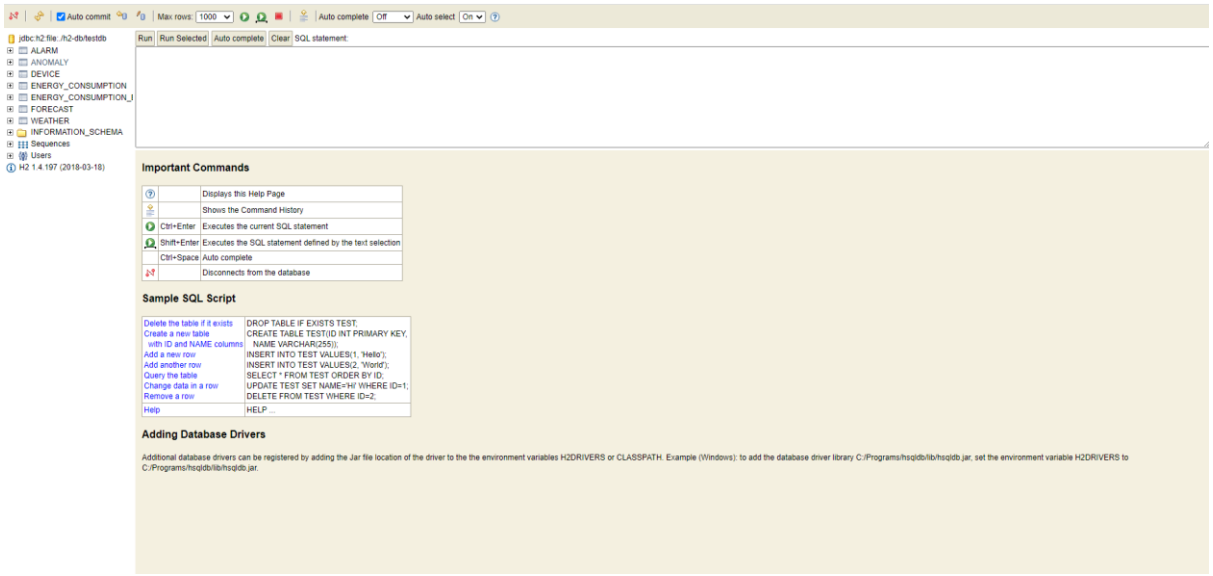
Параметрите кои треба да се додадат во конфигурацискиот документ, односно *application.properties* со цел да се овозможи конекција до самата база на податоци се:

```
# H2
spring.h2.console.enabled=true
spring.h2.console.path=/h2

spring.datasource.url=jdbc:h2:file:./h2-
db/smarthome;DB_CLOSE_ON_EXIT=FALSE;AUTO_RECONNECT=TRUE
spring.datasource.driverClassName=org.h2.Driver
spring.datasource.username=smarthome
spring.datasource.password=password

spring.jpa.database-platform=org.hibernate.dialect.H2Dialect
spring.jpa.hibernate.ddl-auto=update
```

Параметрите кои во себе содржат *console* како:

- `spring.h2.console.enabled=true`
- `spring.h2.console.path=/h2`

служат за овозможување на графичката конзола на базата на податоци, односно првиот од нив е за овозможување на самата конзола, а вториот за патеката на која може да пристапи истата.



Слика 10: Веб-конзола на h2 датабаза

Figure 10 Web console of h2 database

Останатите параметри се за самата база на податоци, за дијалектот и за *hibernate*.

Во параметарот:

spring.datasource.url=jdbc:h2:file:./h2-db/smarthome;DB_CLOSE_ON_EXIT=FALSE;AUTO_RECONNECT=TRUE се подесува дали базата да биде во меморија или како надворешен документ. Во нашиов случај е документ file.

Параметарот:./h2-db/smarthome; е подесен за име на базата на податоци. Параметарот DB_CLOSE_ON_EXIT=FALSE е подесен за самата база на податоци да не се исклучи доколку се исклучи апликацијата.

Параметарот AUTO_RECONNECT=TRUE е подесен доколку настане прекин во конекцијата со базата на податоци, апликацијата повторно да се поврзе со базата на податоци автоматски.

Параметарот spring.datasource.driverClassName=org.h2.Driver е за драјверот за поврзување со самата база на податоци.

Параметарот spring.datasource.username=smarthome е за подесување на корисничко име за самата база на податоци со кое се пристапува на истата.

Параметарот spring.datasource.password=password е за подесување на корисничка лозинка за самата база на податоци со која се пристапува на истата.

Параметарот spring.jpa.database-platform=org.hibernate.dialect.H2Dialect е за подесување на дијалектот на базата на податоци.

Параметарот spring.jpa.hibernate.ddl-auto=update доколку овој параметар е ставен на *create*, со секое рестартирање на апликацијата, датабазата се креира од почеток и сите табели се повторно креираат. Ова се користи во случај на промени во базата на податоци или во некоја табела. Доколку се подесува *update* како во нашиот случај, тогаш на рестартирање на апликацијата, табелите го зачувуваат својот статус и сите податоци се присутни.

Табелата за аларми содржи:

1. Id (идентификатор кој се генерира автоматски)
2. Date (време на зачувување, односно креирање аларм)
3. Info (информации за алармот)
4. Severity (сериозност)
5. Status (состојба).

ID	DATE	INFO	SEVERITY	STATUS
1	2020/10/07 22:48:00	Fire alarm	critical	Active
2	2020/10/07 22:50:03	Water leak detected	critical	Active

Слика 11: Запис во табелата за аларми

Figure 11 Data in alarm table

Испраќањето на електронска пошта до сопственикот е дел од оваа апликација. Електронската пошта се формира со користење на *spring-boot-starter-mail* библиотеката. За да се искористат функционалностите од библиотеката, истата треба да биде поставена во документот за библиотеки. Па затоа *maven dependency* (мавенска зависност) од библиотеката е поставена во *pom.xml* документот.

```
<dependency>
	<groupId>org.springframework.boot</groupId>
	<artifactId>spring-boot-starter-mail</artifactId>
</dependency>
```

Оваа библиотека во себе покрај други библиотеки ја содржи и библиотеката javax.mail од com.sun.mail.

```
<dependency>
    <groupId>com.sun.mail</groupId>
```

```
    <artifactId>javax.mail</artifactId>
    <version>1.6.2</version>
    <scope>compile</scope>
</dependency>
```

Од оваа библиотека се искористени следниве класи:

```
import javax.mail.Message;
import javax.mail.MessagingException;
import javax.mail.PasswordAuthentication;
import javax.mail.Session;
import javax.mail.Transport;
import javax.mail.internet.AddressException;
import javax.mail.internet.InternetAddress;
import javax.mail.internet.MimeMessage;
```

Со цел да се креира аларм, креиран е методот, односно функцијата createAlarm(Alarm alarm);

```
public String createAlarm(Alarm alarm) throws AddressException,
MessagingException, IOException {
	Date date = new Date();
	alarm.setDate(sdf.format(date));
	alarmRepo.save(alarm);
	sendMail();
	return "Successfully created";
}
```

Овде се подесуваат податоците за алармот, се зачувува алармот во базата на податоци преку alarmRepo.save(alarm). И како последно се извршува функцијата sendMail().

```
private void sendMail() throws AddressException, MessagingException, IOException {
		Properties props = new Properties();
		props.put("mail.smtp.auth", "true");
		props.put("mail.smtp.starttls.enable", "true");
		props.put("mail.smtp.host", "smtp.gmail.com");
		props.put("mail.smtp.port", "587");

		Session session = Session.getInstance(props, new
javax.mail.Authenticator() {
		   protected PasswordAuthentication getPasswordAuthentication() {
		      return new PasswordAuthentication(EMAIL, PASSWORD);
		   }
		});
	    Message msg = new MimeMessage(session);
		msg.setFrom(new InternetAddress("email_od_isprakjac@gmail.com",
false));
		msg.setRecipients(Message.RecipientType.TO,
InternetAddress.parse("email_od_primac@gmail.com"));
		msg.setSubject("Alarm");
		msg.setContent(alarm.getInfo(), "text/html");
		msg.setSentDate(new Date());
		Transport.send(msg);
}
```

Во оваа функција се подесуваат потребните информации за електронскиот сервис, информации за електронската пошта од која ќе се испраќа пораката и информации за електронската пошта на сопственикот, каде што ќе пристигнуваат пораките. Како предмет (*Subject*) на пораката е Аларм, а како содржина е информацијата која ќе ја добиеме од *NodeMCU* каде што се креира пораката за аларм.

Функцијата createAlarm(Alarm alarm) се повикува откако ќе стигне порака на брокерот и *MQTT* претплатувачот ќе ја преземе. Целата оваа функционалност се извршува во функцијата mqttSubscriber().

```
public void mqttSubscriber(IMqttClient client, String topic)
                    throws MqttSecurityException, MqttException {
              client.subscribe(topic, new IMqttMessageListener() {

                    public void messageArrived(String topic, MqttMessage message)
throws Exception {
                              String msg = new String(message.getPayload());
                              Alarm alarm = new Alarm();
                              alarm.setInfo(msg);
                              alarm.setSeverity("Critical");
                              alarm.setStatus("Active");
                              alarmService.createAlarm(alarm);
                              template.convertAndSend("/topic/alarm", msg);
                              }
              });
      }
```

messageArrived() е функција која се повикува во самиот mqttSubscriber, а таа за параметри ги прима: темата и пораката. Овде се креира аларм, се повикува createAlarm(Alarm alarm) функцијата на која е предаден истиот аларм и каде што се испраќа електронската пошта. По извршувањето на функцијата пораката се испраќа преку *WebSocket* на Ангулар апликацијата, за креирање на нотификација за аларм.

Слика 12: Приказ на е-пошта од системот за алармирање

Figure 12 Overview of e-mail from the alarming system

Вака изгледа електронската пошта која се испраќа од *Spring Boot* апликацијата. Може да се забележи дека времето на зачувување на аларм во базата на

податоци е скоро еднакво со времето на пристигнување на електронската пошта, што укажува на ефикасноста и брзината на дејствување на системот.

8.1.2. Angular App

Оваа апликација претставува графички кориснички интерфејс на целиот систем. За изработката на оваа апликација како главна технологија е искористена Angular развивачката рамка со верзија 9.0.0-rc.0. За изгледот на самата апликација е искористена *Bootstrap* библиотека. Останатите библиотеки кои се користат ќе бидат објаснети дополнително во секој од деловите каде се користат. *Html5, CSS3* со *JavaScript* се основните блокови на апликацијата. Апликацијата е изработена во *Visual Studio Code*.

На графичкиот интерфејс ќе биде прикажана табела од аномалиите кои се добиени од *Python App* апликацијата за пронаоѓање на аномалии. Истите ќе бидат преземени преку *Rest API* од *Smart Home backend* апликацијата.

Известувањата за аларми се добиваат од *Smart Home backend* апликацијата преку *WebSocket*. За истото се искористени следниве библиотеки:

1. angular-websocket 2.0.1
2. sockjs-client 1.4.0
3. stompjs 2.3.3

Функцијата за поставување на *WebSocket* е:

```
connect() {
  const socket = new SockJS('ws');
  this.stompClient = Stomp.over(socket);

  const _this = this;
  this.stompClient.connect({}, function (frame) {
    _this.setConnected(true);
    console.log('Connected: ' + frame);

    _this.stompClient.subscribe('/topic/reports', function (message) {
      _this.onError(JSON.parse(message.body).alarm);
    });
  });
}
```

Слика 13: Функција за поставување на WebSocket

Figure 13 Function for establishing WebSocket

Доколку стигне порака за аларм, истата се прикажува во горниот десен агол на корисничкиот интерфејс.

8.1.3. Python App

Оваа апликација е направена со цел да се провери функционалноста на сензорот за температура и да се прочисти сетот од податоци за подобро предвидување на температурата. Како алгоритам од вештачката интелигенција за пронаоѓање на аномалии е избран *Isolation Forest*. Како функционира алгоритмот е опишано во делот за користени технологии.

Податоците во кои се пронаоѓаат аномалиите се земени од сензорот за температура и зачувани во .csv формат.

Библиотеки кои се искористени за изработка на проектот:

```
In [1]: import numpy as np
        import pandas as pd
        import seaborn as sns
        import matplotlib.pyplot as plt
        import matplotlib.gridspec as gridspec
        from sklearn.ensemble import IsolationForest
        import datetime as datetime
```

Слика 14: Користени библиотеки

Figure 14 Used libraries

Првин се вчитува самиот сет од податоци со користење на библиотеката *pandas*.

```
In [2]: df_sen = pd.read_csv('weatherHistory.csv', index_col='Date')
        df_sen.head()
```

Out[2]:

Date	Temperature
4/1/2009 0:00	9.472222
4/1/2009 1:00	9.355556
4/1/2009 2:00	9.377778
4/1/2009 3:00	8.288889
4/1/2009 4:00	8.755556

Слика 15: Читање на сетот од податоци

Figure 15 Reading of the dataset

Како индексирачка колона е *Date* колоната. Потоа се прави конвертирање на форматот на индексите во *DateTimeIndex* формат.

```
In [3]: df_sen.index = pd.to_datetime(df_sen.index)
        type(df_sen.index)

Out[3]: pandas.core.indexes.datetimes.DatetimeIndex
```

Слика 16: Конверзија во DateTimeIndex

Figure 16 DateTimeIndex conversion

Потоа се прави конвертирање на *dataFrame* вредностите во *numpy* низа. Потоа се дефинира *IsolationForest* како класификатор. Откако ќе го дефинираме *IsolationForest* можеме да ја искористиме функцијата fit_predict на numpy низата со податоци. Откако ќе се изврши оваа функција се повикува функцијата decision_function која не прикажува само вредности од -1 и 1, туку ни прикажува и вредности кои може да се сметаат како *health*, односно мерка колку некоја опсервација е аномалија. Колку вредноста е помала, толку се поголеми шансите истата да биде аномалија. Колку бројката е поголема, толку се поголеми шансите истата да биде нормална.

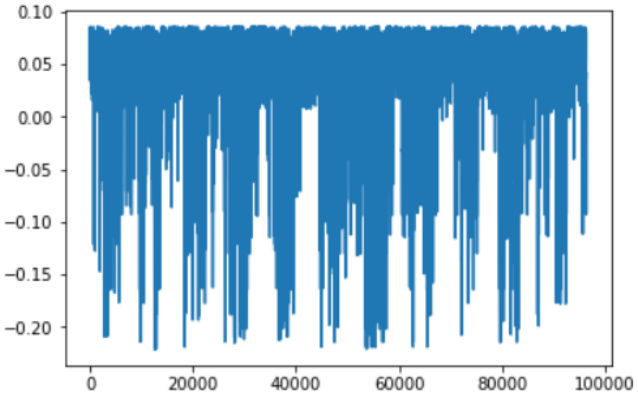
За најдобро да се прикаже истото, резултатот се става во вектор.

```
In [4]: x = df_sen.values

        clf = IsolationForest(contamination=.1)
        predictions = clf.fit_predict(x)
        dec_fun = clf.decision_function(x)

        plt.plot(dec_fun)
```

Слика 17: Креирање на вектор

Figure 17 Creating of the vector

Функцијата plt.plot е за графички приказ.



Слика 18: Графички приказ на пресметките

Figure 18 Result chart

На графиконот може да се забележат позитивни и негативни броеви. X-оската е опсервацискиот број кој повеќе не е во *time* формат.

Вредностите кои ги добиваме од функцијата за одлука (*decision_function*) ги додаваме во *dataFrame* како *Health* колона.

По извршување на df.head() функцијата може да се види новиот dataFrame со новата колона.

```
In [5]: df_sen['health'] = dec_fun
        df_sen.dropna()
        df_sen.head()
```

Out[5]:

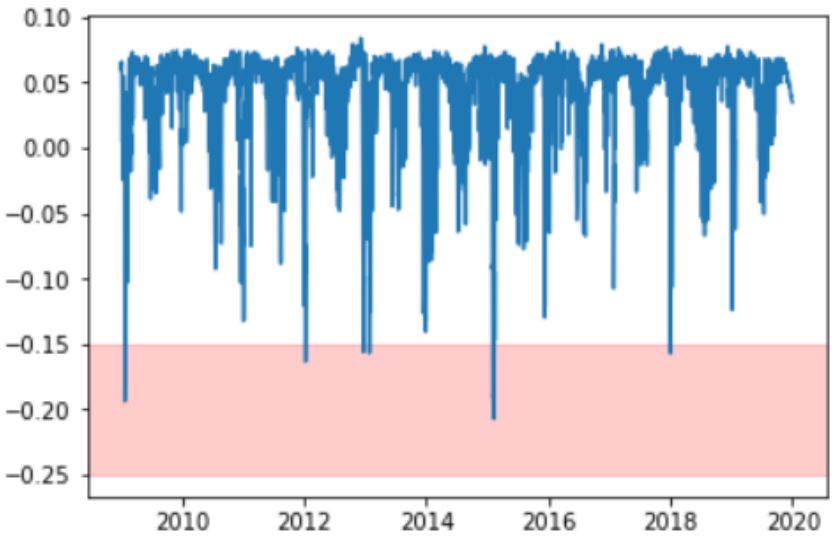
Date	Temperature	health
2009-04-01 00:00:00	9.472222	0.035612
2009-04-01 01:00:00	9.355556	0.035530
2009-04-01 02:00:00	9.377778	0.034360
2009-04-01 03:00:00	8.288889	0.053645
2009-04-01 04:00:00	8.755556	0.058385

Слика 19: Додавање на health колона

Figure 19 Adding health column

Исцртувањето на *health* колоната каде што се прави *resample*, односно се прелимениираат податоците на секои 24 h. Функцијата axhspan е искористена за приказ на вредностите (*thresholds*) во кои рамки би биле аномалиите.

```
In [13]: plt.plot(df_sen['health'].resample('1D').mean())
         plt.axhspan(-0.25, -0.15, alpha=0.2, color='red')

Out[13]: <matplotlib.patches.Polygon at 0x14e89dab488>
```



Слика 20: Графички приказ на податоците и аномалиите

Figure 20 Anomalies and data chart

На графиконот јасно може да се види кои од податоците спаѓаат во рамката на аномалии.

8.1.4. Програмирање на *NodeMCU*

За контролирање на уредите, земање на податоци од мерењата, испраќање аларми и поставување граници за истите, плочата *NodeMCU* треба да биде програмирана да ги извршува истите. За програмирање на *NodeMCU* е искористено *Arduino IDE* и *Arduino* јазик. Ардуино јазикот е базиран на јазиците C и C++ користејќи посебни правила за структурирање на кодот. За да можат да се контролираат уредите и да се препраќаат алармите и мерењата до брокерот, *NodeMCU* мора да е конектирано на *WiFi* мрежа. За поврзување на *WiFi* е искористена *EspMQTTClient* библиотека која воедно е и за поврзување со брокерот.

8.2. Хардвер

8.2.1. Сензори и уреди

За изработката на системот за безбедност и алармирање се искористени следниве сензори и уреди:

8.2.1.1. Сензор за детектирање на гас

Сензорскиот модул за детектирање гас MQ-2 е корисен за откривање на истекување на гас во домот [54]. Погоден е за откривање на метан, бутан, ТНГ и чад. Има висока чувствителност и брзо време на одговор. Чувствителноста на сензорот може да се прилагоди со потенциометар. Аналогниот излезен напон од сензорот се менува пропорционално на концентрацијата на гас. Колку е поголема концентрацијата на гас, толку е поголем излезниот напон.

Главни карактеристики на овој сензор се [55]:

1. работен напон +5V;
2. може да се користи за мерење или откривање на метан, бутан, ТНГ и чад;
3. аналоген излезен напон: 0V до 5V;
4. дигитален излезен напон: 0V до 5V;
5. времетраење на загревање: 20 секунди;
6. може да се користи како дигитален или аналоген сензор;
7. чувствителноста може да се менува со употреба на потенциометар.

Табела 4: Спецификации на сензорот за гас

Table 4 Gas sensor specifications

Дел / Unit	Параметар / Parameter	Мин. / Min	Вообичаено / Usually	Макс. / Max	Мерна единица / Measurement unit
VCC	Работен напон	4.9	5	5.1	V
PH	Потрошувачка при загревање	0.5	-	800	mW
RL	Отпорност на оптоварување		Прилагодливо		
RH	Отпорност на грејач	-	33	-	Ω
Rs	Отпорност на чувствителност	3	-	30	kΩ

Слика 21: Сензор за гас (MQ-2)

Figure 21 Gas sensor (MQ-2)

Кога сензорот ќе детектира истекување на гас, *NodeMCU* испраќа порака за нотификација до брокерот преку *MQTT*, која се презема од Ангулар апликацијата, односно Админ панелот каде што истата се прикажува. Потоа Ангулар апликацијата испраќа аларм до *Spring Boot* апликацијата за креирање на аларм. *Spring Boot* апликацијата го зачувува алармот и испраќа електронска пошта до сопственикот.

8.2.1.2. Сензор за детектирање на пожар

Сензор кој е најчувствителен на нормална светлина е познат како сензор за пламен. Затоа овој сензорски модул се користи во алармирањето за пожар. Овој сензор открива пламен во бранова должина во опсег од 760 nm - 1100 nm од изворот на светлината. Бидејќи лесно може да се оштети на висока температура пожелно е истиот да биде поставен на одредено растојание од пламенот. Откривањето на пламенот може да се изврши од 1 метар растојание. Излезот на овој сензор може да биде аналоген сигнал или дигитален сигнал [56]. Главни карактеристики на сензорот за детектирање на пожар се:

1. висока фотосензитивност
2. брзо време на одговор
3. едноставен за употреба
4. чувствителноста е прилагодлива
5. аголот на откривање е 600
6. одговара на пламенскиот опсег
7. точноста може да се прилагоди
8. работниот напон е 3,3V до 5V
9. компактен.

Слика 22: Сензор за оган
Figure 22 Flame sensor

8.2.1.3. Сензор за детектирање истекување на вода

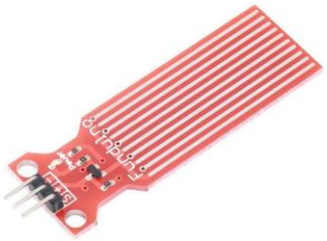
Работата на сензорот за нивото на водата е прилично јасна. Серијата изложени паралелни спроводници, заедно делуваат како променлив отпорник (исто како потенциометар) чиј отпор варира во зависност од нивото на водата. Промената на отпорноста одговара на растојанието од горниот дел на сензорот до површината на водата. Отпорот е обратнопропорционален на висината на водата [57]:

1. Во колку повеќе вода е потопен сензорот, резултира со подобра спроводливост и со помал отпор.
2. Во колку помалку вода е потопен сензорот, резултира со слаба спроводливост и поголема отпорност.

Сензорот произведува излезен напон според отпорот, со чие мерење можеме да го одредиме нивото на водата.

Спецификации на сензорот:

1. работен напон: DC3-5V
2. работна струја: помалку од 20mA
3. тип на сензор: Аналоген
4. област за откривање: 40mm x 16mm
5. работна температура: 10℃-30℃
6. влажност: 10% до 90% некондензирано.



Слика 23: Сензор за ниво на вода
Figure 23 Water level sensor

8.2.1.4. Сензор за детектирање на движење

ПИР сензорите или сензорите за детектирање на движење се користат за да се детектира нечие движење. Тие се мали, евтини, со мала моќност, лесни за употреба и истрајни. Поради тоа, обично се наоѓаат во апарати и гаџет и што се користат во домови или деловни објекти. Често се нарекуваат и ПИР „пасивни инфрацрвени“, „пироелектрични“ или „ИР за движење“ сензори. ПИР сензорите ја откриваат телесната топлина (инфрацрвена енергија) со тоа што бараат промени во температурата. Овој тип на сензор е најкористен во системите за безбедност. Откако сензорот ќе се загрее, односно ќе работи одреден период, тој може да открие движење и топлина во опсегот кој го покрива создавајќи заштитна „мрежа“. Ако предмет кој се движи блокира премногу мрежни зони и нивото на инфрацрвена енергија брзо се менува, тогаш сензорот активира аларм.

Спецификации на сензорот [58]:

1. широк опсег на влезен напон кој варира од 4V до 12V (се препорачува 5V)
2. излезен напон висок/низок (3.3V TTL)
3. може да направи разлика помеѓу движење на предмет и човек
4. има режим на работа – повторлив (H) и неповторлив (H)
5. растојание кое го покрива: од околу 120° и 7 метри
6. ниска потрошувачка на енергија: 65 mA
7. работна температура од -20 °C do +80 °C.

Слика 24: Модул за детекција на движење (PIR)
Figure 24 PIR module

8.2.1.5. Базер

Базерот е мала ефикасна компонента за додавање звучни карактеристики во системот. Тоа е многу мала и компактна 2-пинска структура, па затоа може лесно да се поврзе и користи [59]. Кога некој од сензорите прикажани погоре ќе се активира, односно кога ќе се открие пожар, истекување на гас, истекување на вода или несакан влез во домот, системот го активира базерот за да направи бучава како аларм.

Карактеристики и спецификации на базерот:

1. номинален напон: 6V DC
2. работен напон: 4-8V DC
3. номинална струја: <30mA
4. тип на звук: континуиран сигнал
5. резонантна фреквенција: 2300 ~ Hz
6. мал и уреден
7. лесен за поврзување
8. компактен.

Слика 25: Базер
Figure 25 Buzzer

8.2.1.6. Лед диода

Диода што емитира светлина (ЛЕД) е полупроводнички извор на светлина што емитира светлина кога низ неа протекува електрична енергија [60]. Кога ќе се активира звучната сигнализација од базерот, се активира и лед диодата и започнува да трепка.

Слика 26: Лед диода
Figure 26 Led diode

8.2.1.7. Сензор за температура и влажност

DHT11 - дигитален сензор за температура и влажност е калибриран излез од дигитален сигнал од комбиниран сензор за температура и влажност. Користи посебна технологија за снимање дигитални модули и технологија која овозможува голема сигурност и долгорочна стабилност. Сензорот вклучува отпорен елемент и чувство за влажна (*NTC*) температура со 8-битен микроконтролер со високи перформанси.

Температурата се мери со употреба на термистор со негативен температурен коефициент (*NTC*), а релативната влажност се мери со капацитивен сензор. Тие сензорни елементи се претходно калибрирани и излезот се обезбедува како дигитален сигнал [65].

Сензорот DHT11 е компатибилен со повеќето популарни плочки за развој на микроконтролери, како *Arduino*.

Табела 5: Идентификација на пинови и конфигурација

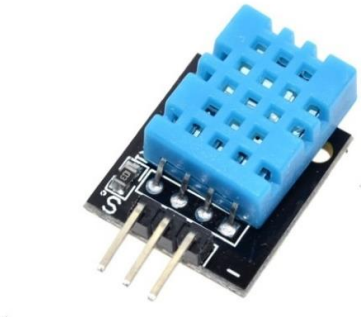
Table 5 Pin identification and configuration

Број / Number	Име на пин / Pin Name	Опис / Description
1	Vcc	Напојување од 3.3V до 5V
2	Data	Ги дава мерењата и за температурата и за влажноста
3	Ground	Се поврзува на заземјувањето во електричното коло

Спецификации на DHT11 [66]:

1. опсег на осетливост на влажност: 20 – 90% RH
2. точност на мерењето на влажност: ± 5 % RH
3. опсег на мерењето на температура: 0 - 60 °C
4. точност на мерењето на температура: ± 2 °C
5. напојување: 3.3 V – 5 V.

Сензорот за мерење на температура и влажност е искористен за детектирањето на аномалии. Мерењата од овој сензор се зачувуваат во .csv формат и подоцна се обработуваат со цел да се пронајдат аномалии и да се осигури работата на сензорот.



Слика 27: Сензор за температура и влажност

Figure 27 Temperature and humidity sensor

8.2.1.8. Elechouse V3 Voice Recognition Module

Овој модул за препознавање на глас е еден од најкомпактните на пазарот. Постојат два начина на користење на овој модул.

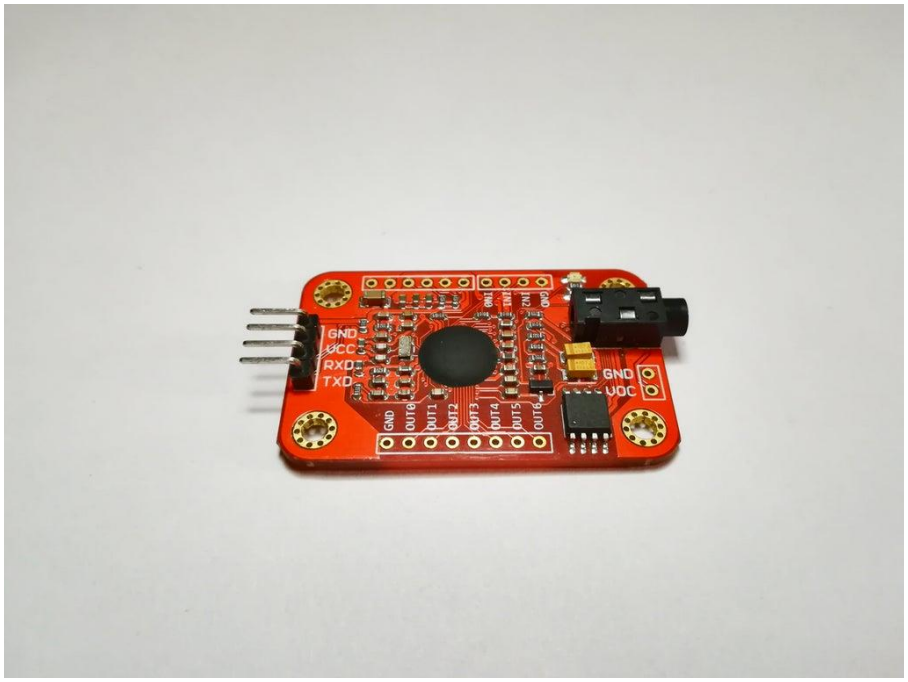
1. користење на сериската порта
2. користење преку вградените пинови.

Оваа плоча има капацитет да складира 80 гласовни команди и времетраење од 1500 милисекунди за секоја од нив. Ова не ги претвора командите во текст, но ќе ги спореди со веќе снимен збир од гласови. Ова значи дека нема јазични пречки за да се користи производот. Може да се снимаат команди на кој било јазик или може да се сними каков било звук за да се користи како команда. За да распознава гласовни команди, прво треба да се обучи [61].

Ако се користи начинот со користење на вградените *GPIO* пинови, тогаш модулот ќе испорачува излези за само 7 команди од 80. За овој метод треба да се изберат и вчитаат 7 команди во препознавачот и препознавачот ќе испрати излези до соодветните *GPIO* пинови доколку некоја од овие гласовни команди биде препознаена.

Уредот работи во опсег на влезен напон од 4,5 V – 5 V и има потрошувачка на енергија од 40 mA. Може да работи со прецизност на препознавање до 99 % доколку се користи во идеални услови. Изборот на микрофон и бучавата во околината имаат витална улога во влијанието врз перформансите на модулот. Со цел да се добијат максимални перформанси од модулот, подобро е да се избере микрофон со добра чувствителност и да се намали бучавата доколку е возможно за време на процесот на давање команди.

Модулот за препознавање на глас ќе биде искористен заедно со Ардуино и со лед диода, со чија помош преку гласовни команди диодата ќе биде активирана и деактивирана, односно ќе биде прикажан случај на вклучување и исклучување светилка во домот преку гласовни команди на македонски јазик.



Слика 28: Модул за препознавање говор

Figure 28 Voice recognition module

8.2.1.9. Arduino Uno R3

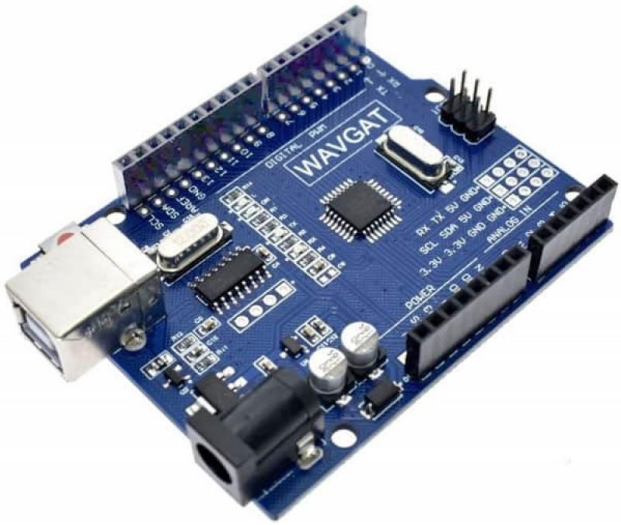
Arduino Uno R3 е микроконтролерска плоча базирана на *ATmega328P*. Се одликува со 14 дигитални пинови за влез/излез (од кои 6 може да се користат како *PWM* излези), 6 аналогни влезови, кварцен кристал од 16 MHz, *USB* приклучок, приклучок за напојување, *ISCP* заглавие и копче за ресетирање. Содржи сè што е потребно за поддршка на микроконтролерот [63].

Спецификации:

1. работен напон: 5V
2. влезен напон (препорачано): 7-12V
3. моќност 5V: преку приклучок
4. дигитални влезни/излезни пинови: 14 (од кои 6 може да бидат *PWM* излези)
5. дигитални влезни/излезни пинови: 3.3V логички нивоа
6. *PWM* дигитални влезни излезни пинови: 6
7. аналогни влезни пинови: 6 (3.3V логички нивоа)
8. *ISP* програмирање: достапно преку *TX/RX* портите
9. *DC* струја на пиновите при излез: 20 mA
10. *DC* струја за пиновите од 3.3 V: 50 mA

11. флеш меморија: 32 KB
12. *SRAM*: 2 KB
13. *EEPROM*: 1 KB
14. брзина на процесирање: 16 Mhz осцилатор.

Ардуиното ќе биде искористено како микроконтролер на кој ќе биде поврзан модулот за препознавање на говор и лед диодата која ќе биде управувана. Самото Ардуино ќе биде програмирано да ги извршува операциите на модулот за препознавање на говор.



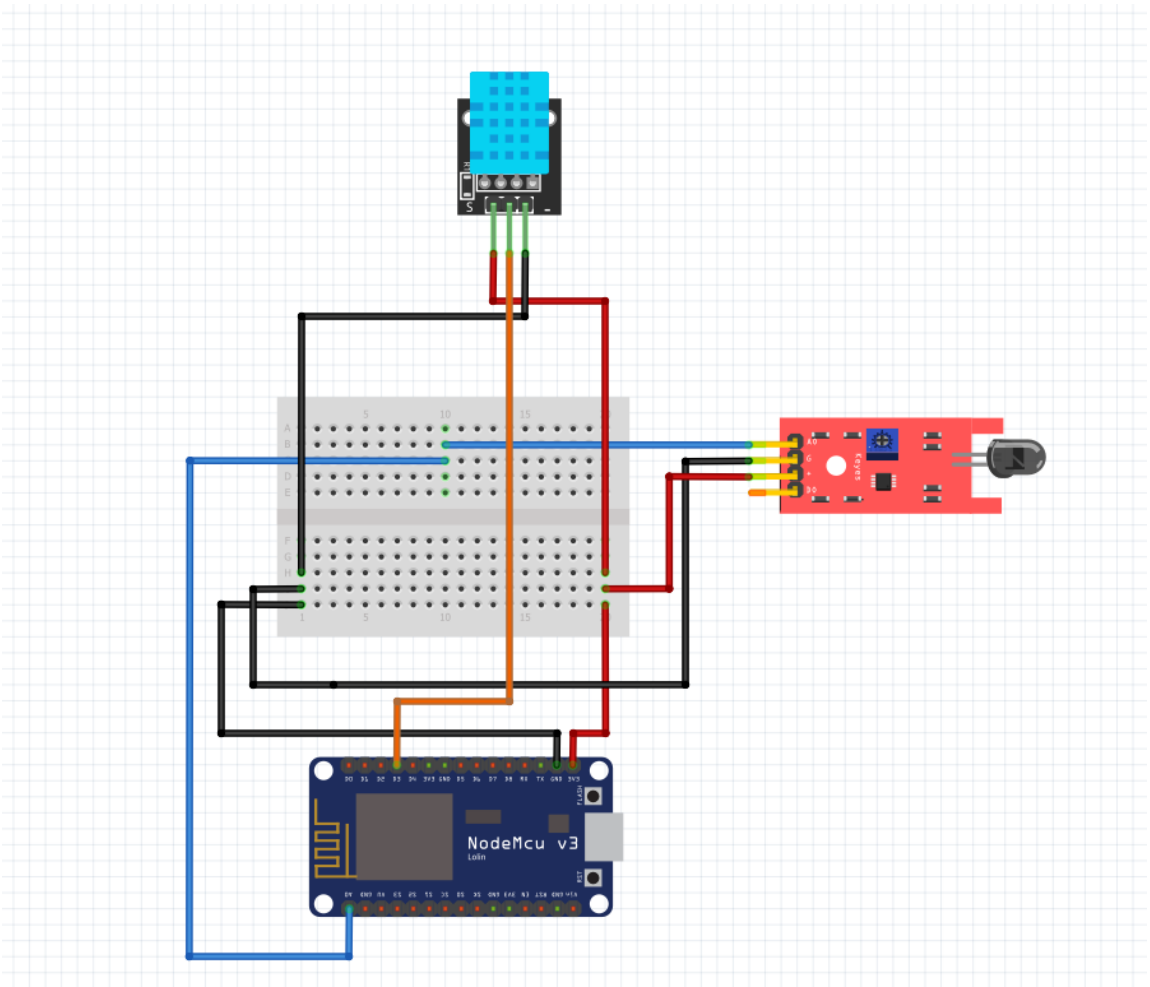
Слика 29: Ардуино УНО

Figure 29 Arduino UNO

8.3 Шематски прикази од опфатените сценарија

Во прилог се прикажани шематски прикази од системот за безбедност во паметен дом:

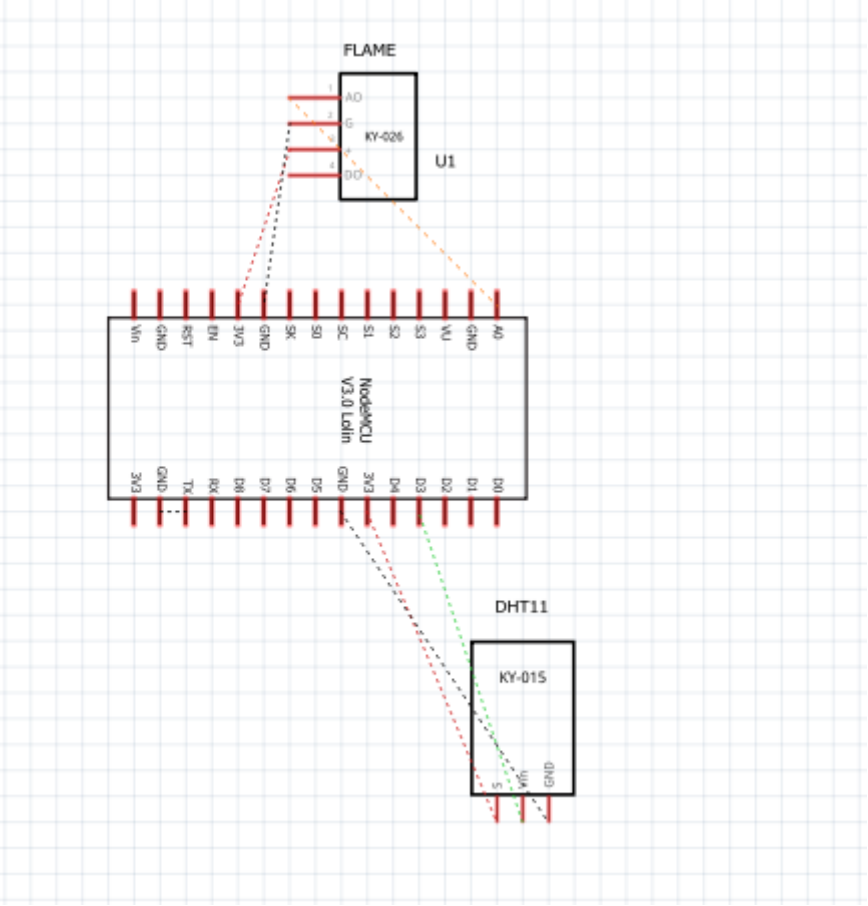
На следнава шема е прикажано поврзувањето на сензорот за температура и влажност и сензорот за оган. Како што може да се види од шемата, за сензорот за температура и влажност е искористен дигиталниот пин D3, заземјување и напојување со вредност од 3.3V. За сензорот за оган е искористен аналогниот пин A0, заземјување и 3.3V напојување. Од NodeMCU се извлечени заземјувањето и напојувањето и се поврзани со жици на протобордот (развивачката плоча). Црните жици од прикажаната слика се за заземјувањето, а црвените за напојувањето. За поврзување на дигиталниот пин од сензорот за температура и влажност и D3 пинот од NodeMCU е искористена портокалова жица. За поврзување на аналогниот пин од сензорот за оган и NodeMCU е искористена сина жица.



Слика 30: Поврзување на сензорот за оган и сензорот за температура и

влажност со *NodeMCU*

Figure 30 Connecting of flame sensor and temperature and humidity sensor with NodeMCU

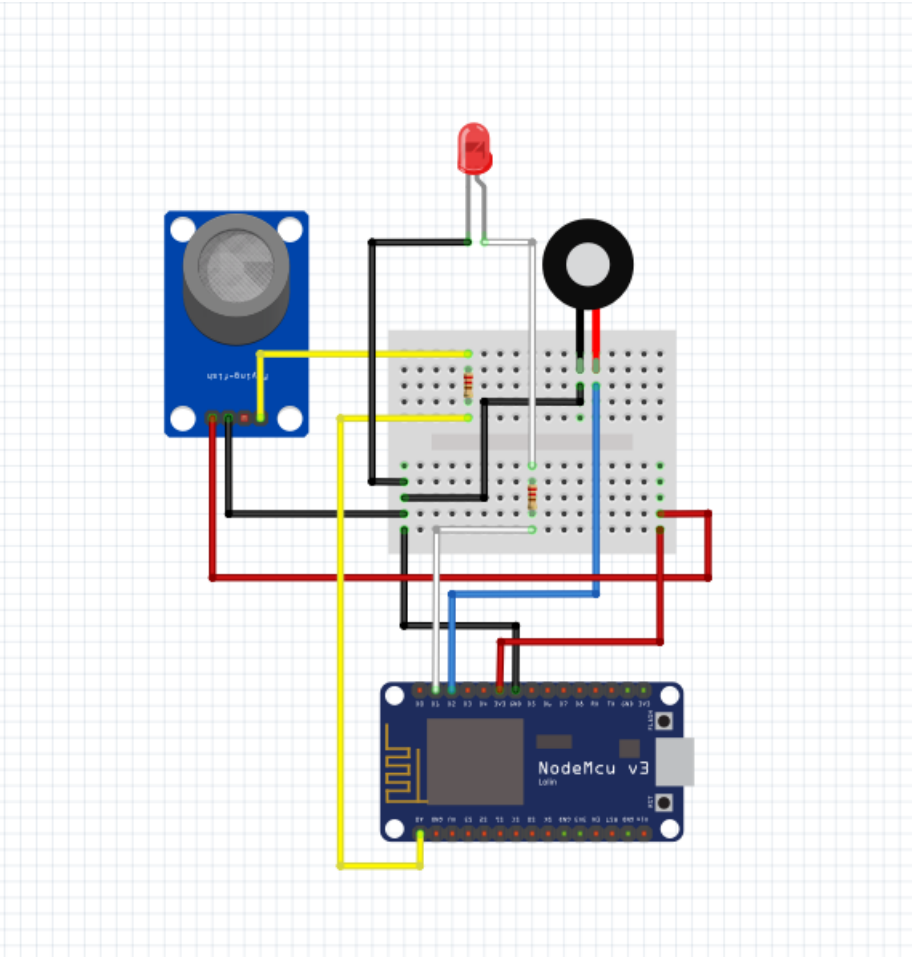


Слика 31: Шематски приказ на поврзувањето на сензорот за оган и сензорот за температура и влажност со *NodeMCU*

Figure 31 Schematic representation of the connection of the flame sensor and the temperature and humidity sensor to the NodeMCU

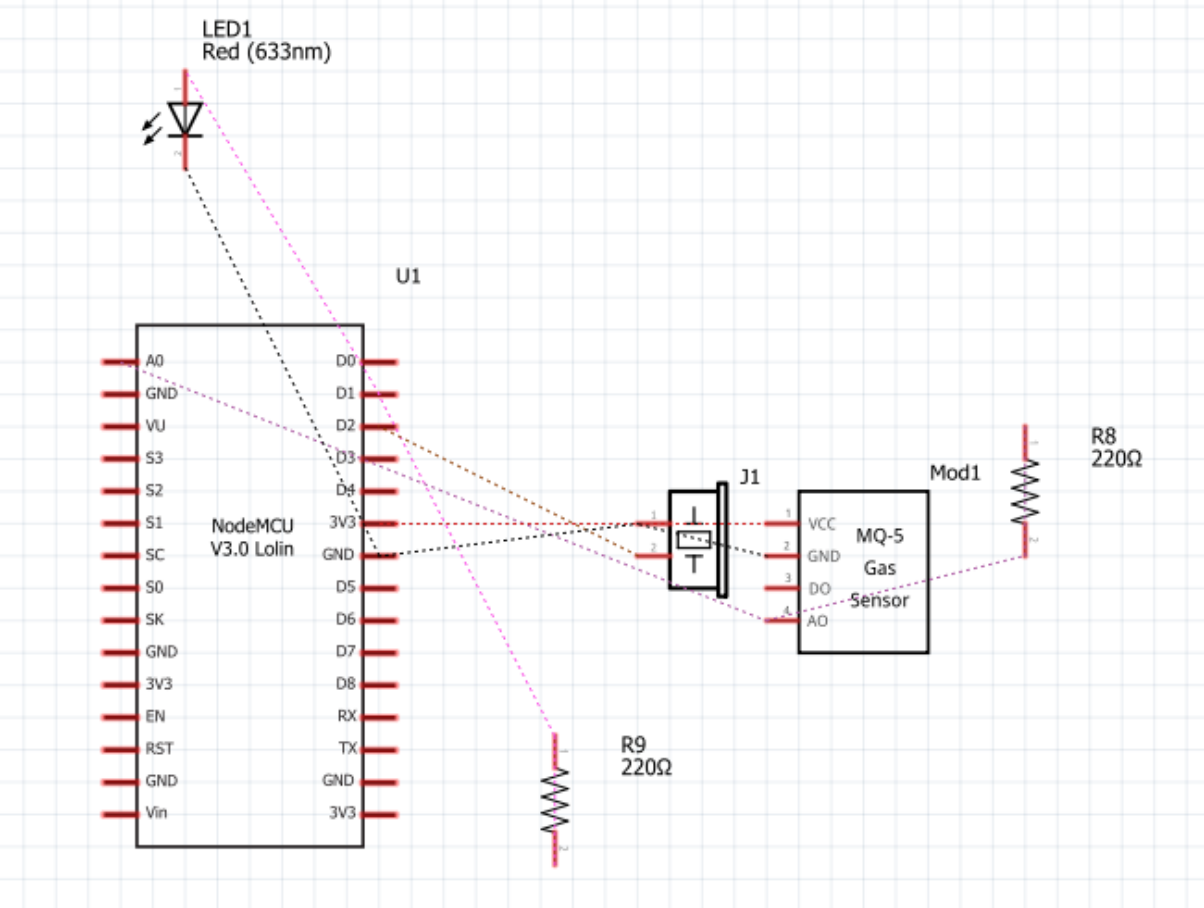
На следнава шема е прикажано поврзувањето на сензорот за откривање на истекување на гас, лед сијалица и базер. Како што може да се види од шемата, за сензорот за октривање на истекување на гас е искористен аналогниот пин AO, заземјување и напојување со вредност од 3.3V. За лед сијалицата е искористен дигиталниот пин D1 и заземјување. За базерот е искористен дигиталниот пин D2

и заземјување. Од NodeMCU се извлечени заземјувањето и напојувањето и се поврзани со жици на протобордот (развивачката плоча). Црните жици од прикажаната слика се за заземјувањето, а црвените за напојувањето. За поврзување на аналогниот пин од сензорот за откривање на истекување на гас A0 пинот од NodeMCU е искористена жолта жица и отпорник со вредност од 500 оми. За поврзување на лед сијалицата и дигиталниот D1 пин на NodeMCU е искористена бела жица и отпорник со вредност од 330 оми. За поврзување на базерот и дигиталниот пин D2 од NodeMCU е искористена сина жица.



Слика 32: Поврзување на сензорот за гас, базерот и лед диода со *NodeMCU*

Figure 32 Connecting of the gas sensor, buzzer and led diode to the NodeMCU

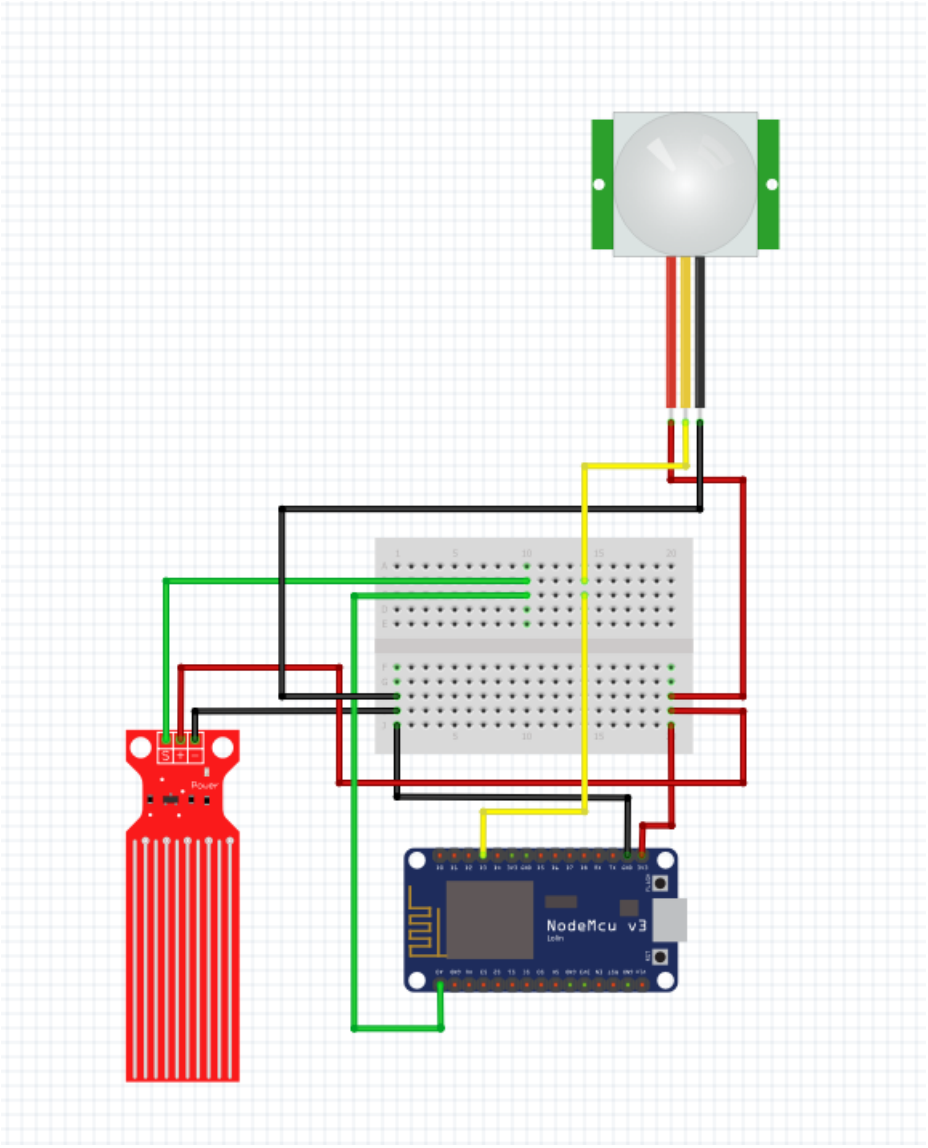


Слика 33: Шематски приказ на поврзувањето на сензорот за гас, базерот и лед диода со *NodeMCU*

Figure 33 Schematic representation of the connection of the gas sensor, buzzer and led diode to the NodeMCU

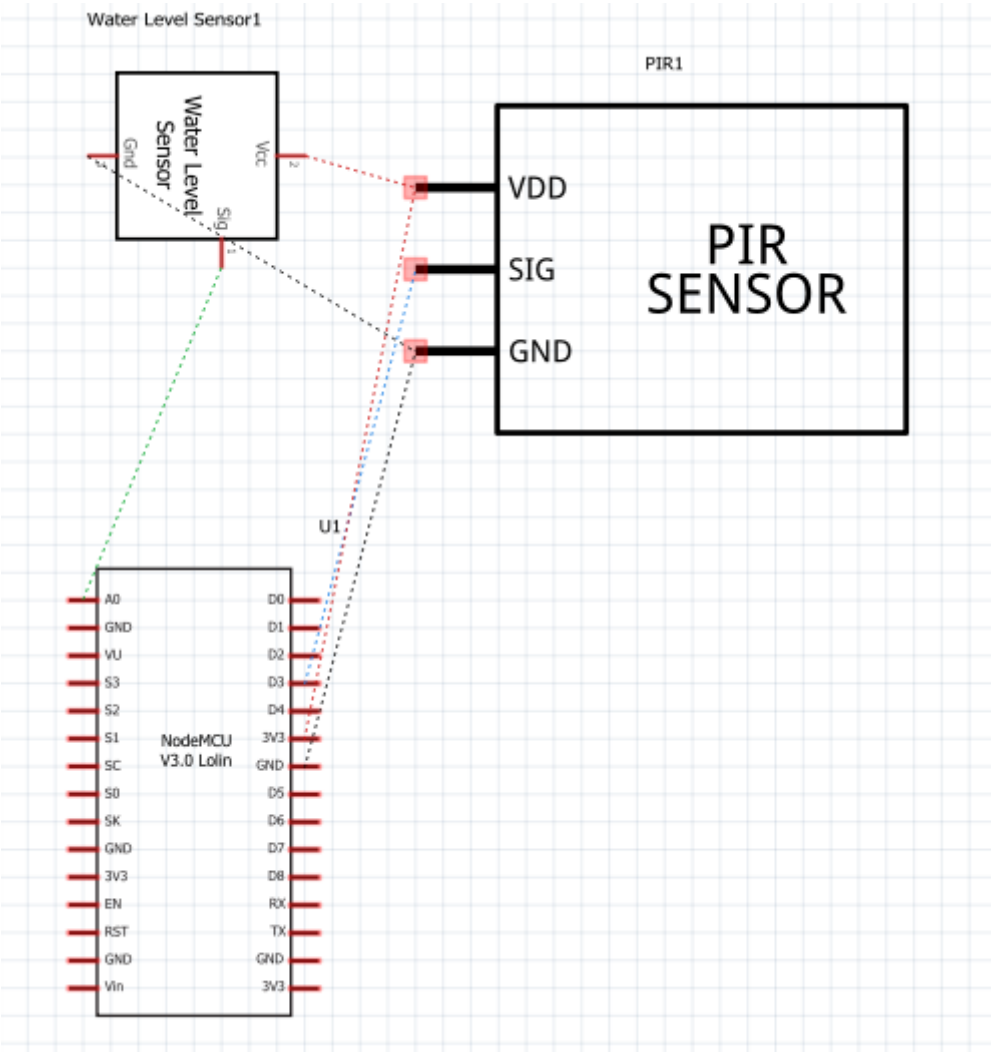
На следнава шема е прикажано поврзувањето на сензорот за детектирање на движење и сензорот за детектирање на истекување на вода. Како што може да се види од шемата, за сензорот за детектирање на движење е искористен дигиталниот пин D3, заземјување и напојување со вредност од 3.3V. За сензорот за детектирање на истекување на вода е искористен аналогниот пин A0, заземјување и 3.3V напојување. Од NodeMCU се извлечени заземјувањето и напојувањето и се поврзани со жици на протобордот (развивачката плоча). Црните жици од прикажаната слика се за заземјувањето, а црвените за напојувањето. За поврзување на дигиталниот пин од сензорот за детектирање на движење и D3 пинот од NodeMCU е искористена жолта жица. За поврзување

на аналогниот пин од сензорот за детектирање на истекување на вода и NodeMCU е искористена зелена жица.



Слика 34: Поврзување на сензорот за ниво на вода и сензорот за детектирање на движење со *NodeMCU*

Figure 34 Connecting of the water level sensor and the motion sensor to the NodeMCU

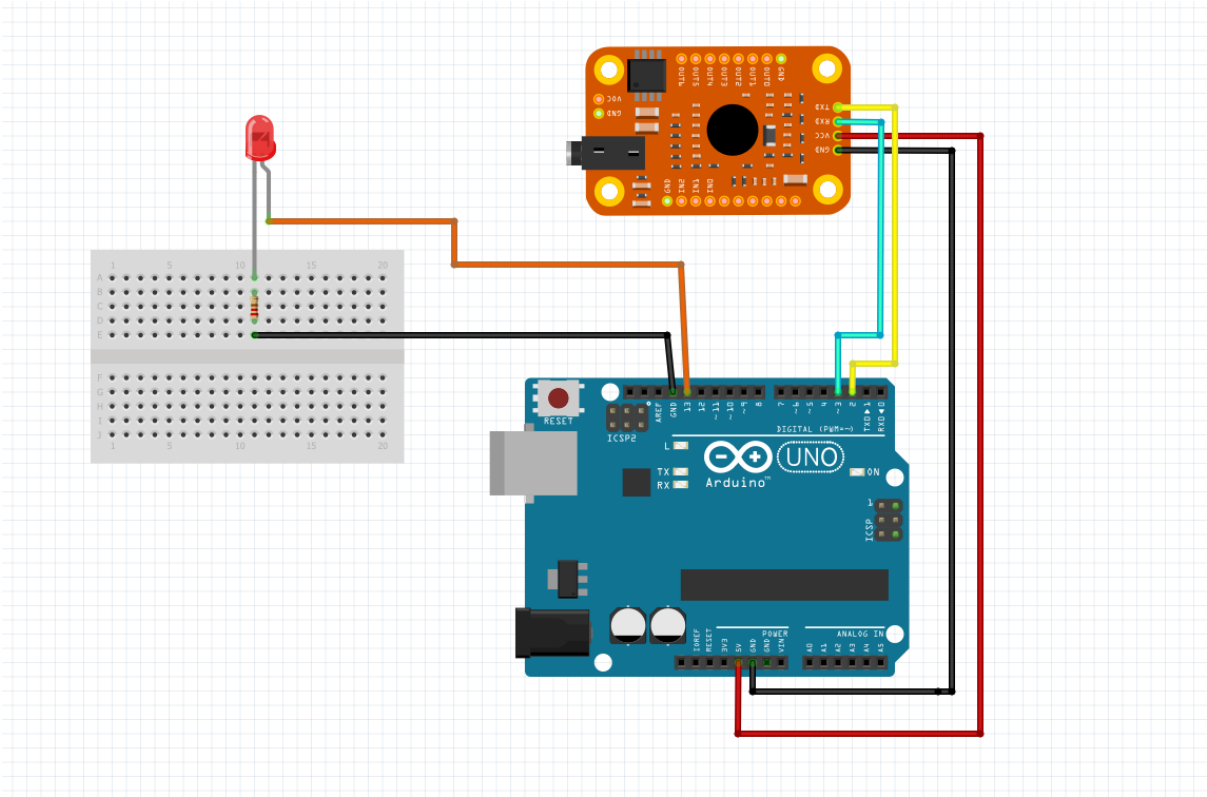


Слика 35: Шематски приказ на поврзување на сензорот за ниво на вода и сензорот за детектирање на движење со *NodeMCU*

Figure 35 Schematic representation of the connection of the water level sensor and the motion sensor to the NodeMCU

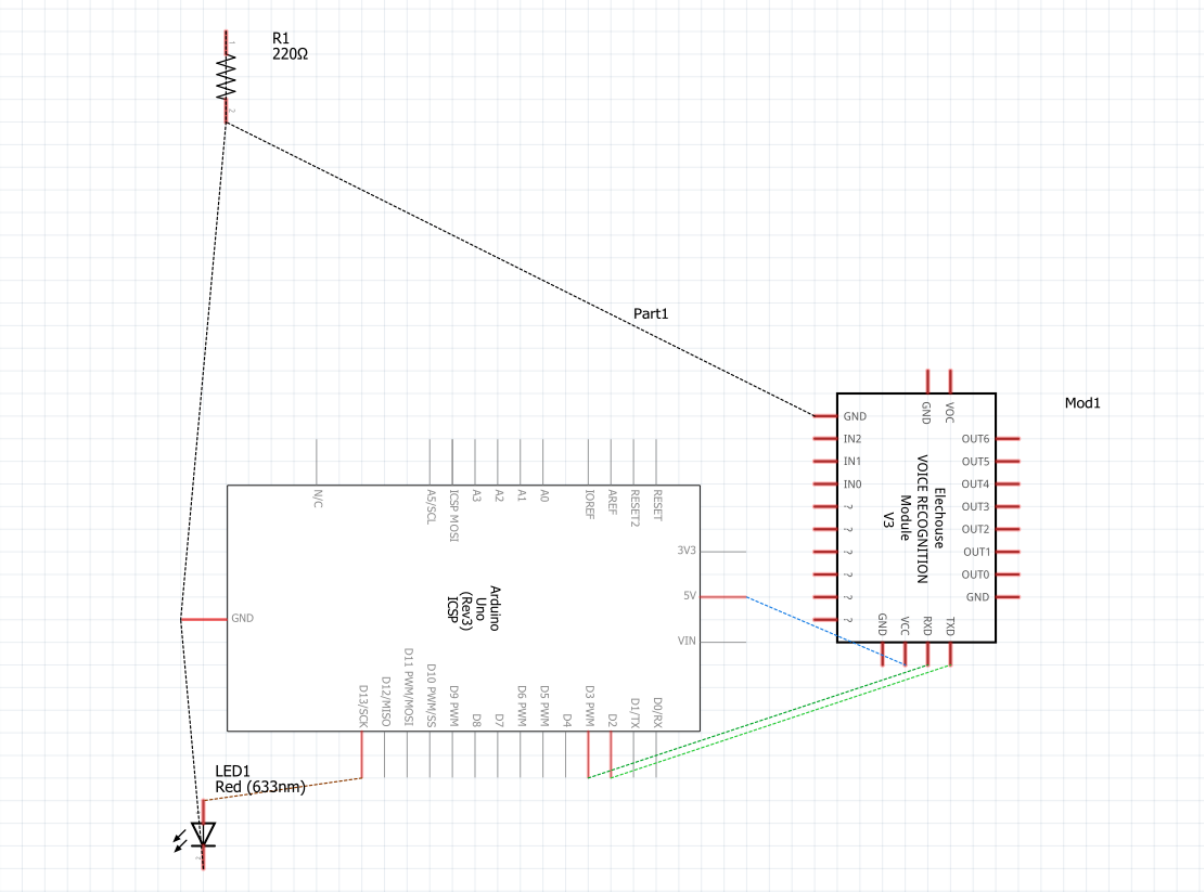
На следнава шема е прикажано поврзувањето на модулот за препознавање на глас и лед сијалица. Како што може да се види од шемата, за модулот за препознавање на глас се искористени дигиталните пинови D2 и D3, заземјување и напојување со вредност од 5V. За лед сијалицата е искористен дигиталниот пин D3, заземјување и 5V напојување. Модулот за препознавање на глас има 4

излези. Со црна жица е поврзано заземјувањето кое се зема од Arduino. За поврзување на напојувањето е искористена црвена жица. TX пинот од модулот е поврзан со дигиталниот пин D2 на Arduino Црните, за што е искористена жолта жица, а RX пинот од модулот е поврзан со дигиталниот пин D3 од Arduino за што е искористена сина жица. За поврзување на лед сијалицата е искористена портокалова жица од дигиталниот пин D13 на Arduino до лед сијалицата. Заземјувањето е поврзано со црна жица од Arduino до самата лед сијалица.



Слика 36: Поврзување на сензорот за препознавање на глас и лед диода со *Arduino UNO*

Figure 36 Connecting of the voice recognition module and led diode with Arduino UNO



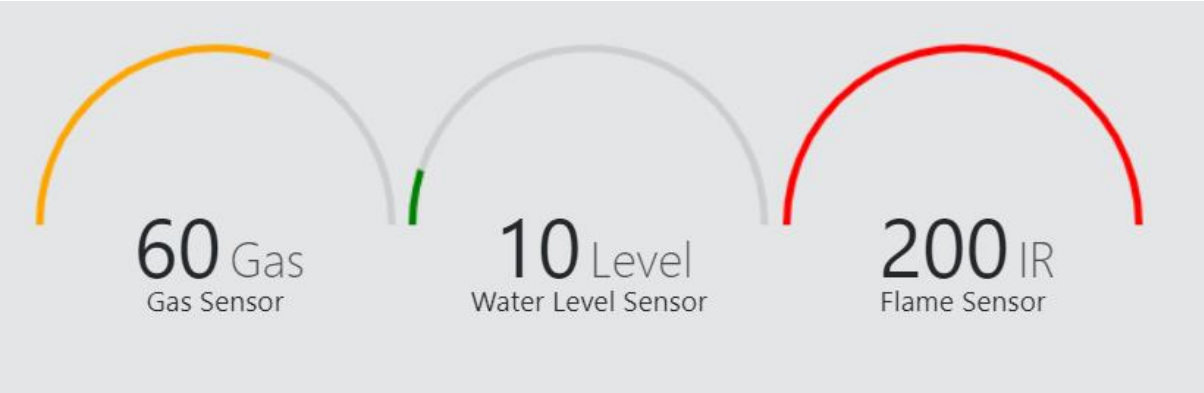
Слика 37: Шематски приказ на поврзување на сензорот за препознавање на глас и лед диода со *Arduino UNO*

Figure 37 Schematic representation of the connection of the voice recognition module and led diode with Arduino UNO

9. Резултати и дискусија

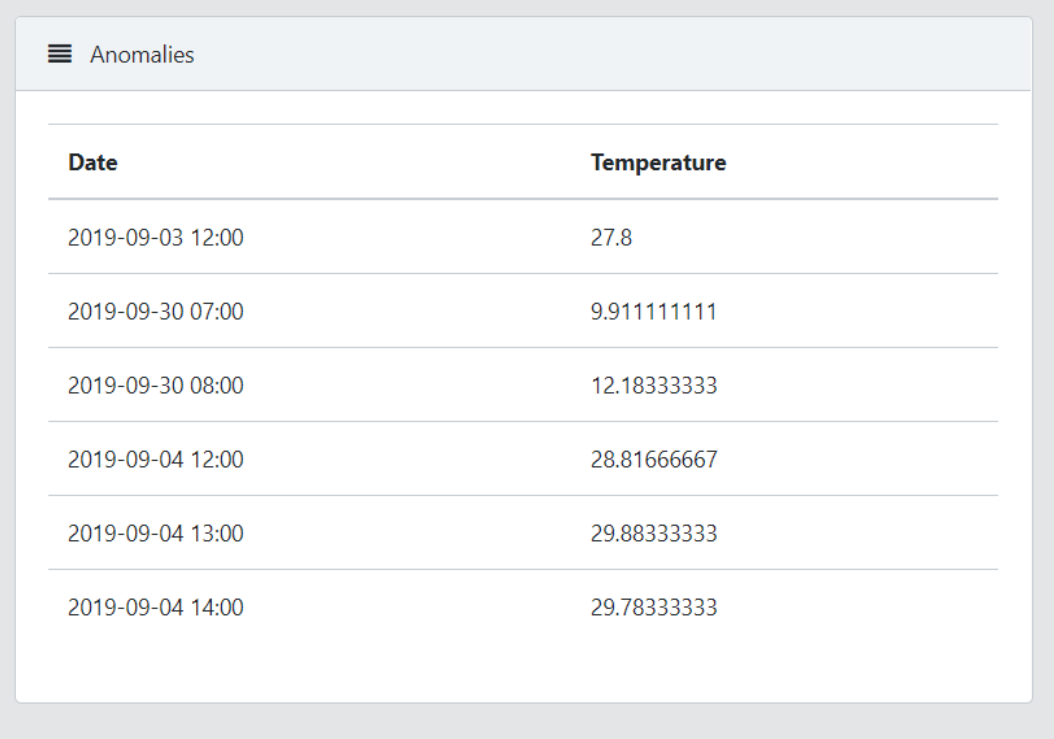
Од горенаведениот систем се постигнати следниве резултати:

1. Добиен е графички приказ за следење на мерењата на сензорите.



Слика 38: Графички приказ од мерењата на сензорите
Figure 38 Representation of the current values from the sensors

2. За приказ на работата на системот за откривање на аномалии, искористен е сензор за температура и влажност. Искористени се мерењата од сензорот од еден месец во нормални собни услови. За подобро функционирање на системот за откривање аномалии потребен е поголем сет од податоци. Затоа останатите податоци се симулирани. Од добиениот сет од податоци се пронајдени аномалии во работата на сензорот за температура и влажност. Дел од пронајдените аномалии се намерно предизвикани со поставување топол предмет до сензорот, а дел потекнуваат од работните услови. Добиените оценки за аномалиите од системот се прикажани на слика 20, а нивните вредности се прикажани на графичкиот интерфејс.



Слика 39: Приказ на аномалиите

Figure 39 Overview of the anomalies

3. Овозможено е контролирање на лед диода преку гласовна команда на македонски јазик.
4. По детектирање на истекување на гас, системот активира аларм и на графичкиот интерфејс се добива известување за истиот и се добива електронска пошта со известување за алармот.
5. По детектирање на оган, системот активира аларм и на графичкиот интерфејс се добива известување за истиот и се добива електронска пошта со известување за алармот.
6. По детектирање на истекување на вода, системот активира аларм и на графичкиот интерфејс се добива известување за истиот и се добива електронска пошта со известување за алармот.
7. По детектирање на движење, системот активира аларм и на графичкиот интерфејс се добива известување за истиот и се добива електронска пошта со известување за алармот.
8. За сите од наведените аларми е добиен резултат на пристигнување на пораката преку електронска пошта во времетраење помало од 30 секунди

и прикажување на известување на системот во времетраење помало од 5 секунди.

ID	DATE	INFO	SEVERITY	STATUS
1	2020/10/07 22:48:00	Fire alarm	critical	Active
2	2020/10/07 22:50:03	Water leak detected	critical	Active

Слика 40: Запис во табелата за аларми

Figure 40 Data in alarm table

Слика 41: Приказ на е-пошта од системот за алармирање

Figure 41 Overview of e-mail from the alarming system

10. Заклучок

Удобноста и сигурноста во домот за кои се стреми секој, не можат да се постигнат без безбедност. Системите за безбедност во паметен дом нудат многу придобивки како откривање пожар, истекување на гас, истекување на вода, следење преку камера, паметни светла, аларми во домот, алармирање на сопственикот преку е-пошта, СМС-пораки и сл.

За овој магистерски труд беше изработен модел на паметен и безбеден дом со користење на повеќе апликации. Во него се претставени сензори и уреди за детектирање на гас, пожар, истекување на вода, движење, базер, лед диода и сензор за температура и влажност. Додатно е искористен и *Elechouse V3 Voice Recognition Module* преку кој можат да се задаваат команди на македонски јазик.

Од изработениот систем може да заклучиме дека брзото и ефикасно алармирање од безбедносните системи е од големо значење за сигурноста и удобноста во секојдневниот живот.

Ниту еден паметен дом не е навистина „паметен“ без добро имплементирано безбедносно решение.

11. Користена литература

1. https://data.worldbank.org/indicator/SP.POP.65UP.TO.ZS?end=2019&start=1960&view=chart (Access Date: 07.02.2020)
2. https://www.who.int/ageing/publications/global_health/en/ (Access Date: 08.02.2020)
3. https://eprints.usq.edu.au/3795/ (Access Date: 11.02.2020)
4. Stergioua C, Psannis KE, Kimb B-G, Gupta B. Secure Integration of IoT and Cloud Computing. Elsevier, Future Generation Computer Systems, Vol. 78. Part 3. January 2018. pp. 964-975
5. Al-Kuwari M, Ramadan A, Ismael Y, Al-Sughair L, Gastli A, Benammar M. Smart-Home Automation Using IoT-Based Sensing and Monitoring Platform, IEEE. 2018. Available from: ieeexplore.ieee.org
6. Datta T, Apthorpe N, Feamster N. Developer-friendly library for smart home IoT privacy-preserving traffic obfuscation, IoT S&P 18. In: Proceedings of the 2018 Workshop on IoT Security and Privacy. ACM; 2018. pp. 43-48
7. Mao J, Lin Q , Bian J. Application of Learning Algorithms in Smart Home IoT System Security. American Institute of Mathematical Sciences; 2018. DOI: 10.3934/mfc.2018004
8. Saeed F, Paul A, Rehman A, Hong WH, Seo H. IoT-based intelligent modeling of smart home environment for fire prevention and safety. Journal of Sensor and Actuator Networks. 2018; (1):11. DOI: 10.3390/jsan7010011
9. Ashton, K. (2009). That ‘internet of things’ thing. RFiD Journal, 22(7), 97-114.
10. Li, S., Da Xu, L., & Zhao, S. (2015). The internet of things: a survey. Information Systems Frontiers, 17(2), 243-259.
11. Zhong, Y. (2015). I2oT: Advanced Direction of the Internet of Things. ZTECOMMUNICATIONS, 3.
12. Miller, M. (2015). The internet of things: How smart TVs, smart cars, smart homes, and smart cities are changing the world. Pearson Education.
13. N. K. Suryadevara and S. C. Mukhopadhyay, Smart Homes: Design, Implementation and Issues, vol. 14. Springer, 2015
14. https://www.iottechtrends.com/history-of-iot/?fbclid=IwAR02ivZAcxn8Gvr0hRgs7dOXnuew8LGFALr8w-JYvXQjLllpkCBAQah8Obg (Access Date: 09.03.2020)

15. Edward B.Driscoll, Jr., The History of X10. URL http://home.planet.nl/~lhendrix/x10_history.htm 9, 10, 2
16. M. Murata, T. Namekawa, R. Hamabe, A proposal for standardization of home bus system for home automation, Consumer Electronics, IEEE Transactions on CE29 (4) (1983) 524 –530. doi:10.1109/TCE.1983.356359. 9, 10
17. C. Douligeris, J. Khawand, C. Khawand, Communications and control for a home automation system, in: Southeastcon '91., IEEE Proceedings of, 1991, pp. 171 –175 vol.1. doi:10.1109/SECON.1991.147729. 10, 11, 12
18. Popular science (1993). 10, 12, 13 [16] M. Inoue, K. Uemura, Y. Minagawa, M. Esaki, Y. Honda, A home automation system, Consumer Electronics, IEEE Transactions on CE-31 (3) (1985) 516 – 527. doi:10.1109/TCE.1985.289966. 10, 11
19. R. Hamabe, M. Murata, T. Namekawa, A revised new proposal for standardization of home bus system for home automation, Consumer Electronics, IEEE Transactions on CE-32 (1) (1986) xi –8. doi:10.1109/TCE.1986. 290110. 10, 11
20. K. Wacks, The impact of home automation on power electronics, in: Applied Power Electronics Conference and Exposition, 1993. APEC '93. Conference Proceedings 1993., Eighth Annual, 1993, pp. 3 –9. doi:10.1109/ APEC.1993.290658. 10, 13, 14
21. C. Tribune, Smart house model opens in naperville. URL https://www.chicagotribune.com/news/ct-xpm-1993-11-20-9311200042-story.html 10, 14
22. P. Corcoran, J. Desbonnet, K. Lusted, Cebus network access via the world-wide-web, in: Consumer Electronics, 1996. Digest of Technical Papers., International Conference on, 1996, p. 236. doi:10.1109/ICCE.1996. 517285. 10, 14
23. N. Sriskanthan, F. Tan, A. Karande, Bluetooth based home automation system, Microprocessors and Microsystems 26 (6) (2002) 281 – 289. doi:10.1016/ S0141-9331(02)00039-X. URL http://www.sciencedirect.com/science/article/pii/ S014193310200039X 10, 16
24. K. Wacks, Home systems standards: achievements and challenges, Communications Magazine, IEEE 40 (4) (2002) 152 –159. doi:10.1109/35.995865. 10, 15

25. A. Alheraish, Design and implementation of home automation system, Consumer Electronics, IEEE Transactions on 50 (4) (2004) 1087 – 1092. doi:10.1109/TCE.2004. 1362503. 10, 16
26. A. Al-Ali, M. Al-Rousan, Java-based home automation system, Consumer Electronics, IEEE Transactions on 50 (2) (2004) 498 – 504. doi:10.1109/TCE.2004.1309414. 10, 16
27. P. Darbee, Insteon (8 2005). URL https://en.wikipedia.org/wiki/Insteon 10, 19, 20
28. P. Kinney, Zigbee technology: Wireless control that simply works (10 2003). URL https://www.mouser.com/pdfdocs/ZigBeeTechnology.pdf pdf 10, 26
29. JFR, Z-wave protocol overview. https://docs.google.com/viewer?a=v&pid=sites&srcid=ZGVmYXVsdGRvbWFpbnxvbmxpbmVrdWhhcmljYXxneDpkMzc4NzZlNzA0NjA5MGY 10, 16, 23
30. https://www.oreilly.com/library/view/arduino-a-technical/9781491934319/ch01.html (Access Date: 15.03.2020)
31. T. Ricker, Editorial: Android@home is the best worst thing that could happen to home automation (5 2011). URL https://www.engadget.com/2011-05-11-editorial-android-home-is-the-best-worst-thing-that-could-happe.html 10, 17
32. https://en.wikipedia.org/wiki/HomeKit (Access Date: 15.03.2020)
33. https://en.wikipedia.org/wiki/Amazon_Echo (Access Date: 16.03.2020)
34. https://en.wikipedia.org/wiki/Google_Nest_(smart_speakers) (Access Date: 16.03.2020)
35. https://www.samsung.com/us/smart-home/smartthings/hubs/ (Access Date: 16.03.2020)
36. Popular science (1993). 10, 12, 13
37. B. Markwalter, C. Russell, Consumer electronics bus, a robust communications system, in: Consumer Electronics, 1988. Digest of Technical Papers. ICCE., IEEE 1988 International Conference
38. Rosslin John Robles1 and Tai-hoon Kim, “A Review on Security in Smart Home Development,“ in International Journal of Advanced Science and Technology, vol. 15, February, 2010
39. https://spring.io/projects/spring-boot (Access Date: 17.04.2020)
40. https://www.tutorialspoint.com/spring_boot/spring_boot_introduction.html (Access Date: 17.04.2020)

41. https://www.tutorialspoint.com/angular6/index.htm (Access Date: 19.05.2020)
42. https://yalantis.com/blog/when-to-use-angular/ (Access Date: 19.05.2020)
43. https://angular.io/ (Access Date: 19.05.2020)
44. https://medium.com/mqtt-buddy/mqtt-vs-http-which-one-is-the-best-for-iot-c868169b3105 (Access Date: 26.05.2020)
45. https://mqtt.org/ (Access Date: 26.05.2020)
46. https://en.wikipedia.org/wiki/MQTT (Access Date: 26.05.2020)
47. https://www.eclipse.org/paho/ (Access Date: 26.05.2020)
48. https://internetofthingsagenda.techtarget.com/definition/MQTT-MQ-Telemetry-Transport (Access Date: 29.05.2020)
49. http://activemq.apache.org/ (Access Date: 29.05.2020)
50. https://developer.ibm.com/technologies/iot/tutorials/iot-nodemcu-open-why-use/ (Access Date: 06.06.2020)
51. https://www.tdgulf.com/product/nodemcu-v3-lolin/ (Access Date: 06.06.2020)
52. https://www.python.org/doc/essays/blurb/ (Access Date: 03.07.2020)
53. https://www.h2database.com/html/main.html (Access Date: 06.07.2020)
54. https://components101.com/mq2-gas-sensor (Access Date: 26.07.2020)
55. https://lastminuteengineers.com/mq2-gas-senser-arduino-tutorial/ (Access Date: 26.07.2020)
56. http://rogerbit.com/wprb/wp-content/uploads/2018/01/Flame-sensor-arduino.pdf (Access Date: 03.08.2020)
57. https://lastminuteengineers.com/water-level-sensor-arduino-tutorial/ (Access Date: 04.08.2020)
58. https://cdn-learn.adafruit.com/downloads/pdf/pir-passive-infrared-proximity-motion-sensor.pdf (Access Date: 05.08.2020)
59. https://components101.com/buzzer-pinout-working-datasheet (Access Date: 06.08.2020)
60. https://en.wikipedia.org/wiki/Light-emitting_diode (Access Date: 06.08.2020)
61. http://www.elechouse.com/elechouse/index.php?main_page=product_info&cPath&products_id=2254 (Access Date: 12.08.2020)
62. https://store.arduino.cc/arduino-uno-rev3 (Access Date: 14.08.2020)
63. https://www.housinglin.org.uk/_assets/Resources/Housing/Housing_advice/Smart_Home_-_A_definition_September_2003.pdf (Access Date: 20.08.2020)

64. https://www.statista.com/statistics/1014296/us-consumers-desire-smart-home-security-control-capability/ (Access Date: 22.08.2020)
65. https://components101.com/dht11-temperature-sensor (Access Date: 22.08.2020)
66. https://www.openimpulse.com/blog/products-page/product-category/dht11-temperature-and-humidity-sensor-module/ (Access Date: 26.08.2020)