ВІДЗИВ НАУКОВОГО КЕРІВНИКА НА ВИПУСКНУ КВАЛІФІКАЦІЙНУ РОБОТУ

Студента (ки) **Хомицького Д.А.**
(прізвище та ініціали)
представлену на здобуття освітнього ступеня «бакалавр»
зі спеціальності 123 «Комп'ютерна інженерія»

1. Робота виконана на тему Проект IoT-системи захисту банківської комірки

2. Актуальність. За останні п'ять років, як пристрої зв'язку, лідирують смартфони, які допомагають вирішувати повсякденні завдання. Далі йдуть планшети, з мобільною операційною системою і сенсорним інтерфейсом. Як тільки Інтернет-речей стане повсякденним, збільшиться кількість споживчих додатків і бізнес-додатків, задіяних в цій новій системі, яку можна описати поняттям "підключене життя". Це включає будівельну і побутову автоматизацію, системи опалення та кондиціонування повітря, управління дорожнім рухом (наприклад, розумний світлофор), організацію турботи про літніх людей, системи безпеки, а також підключені до інтернету автомобілі і зовнішня реклама. В розвитку технологій Інтернету-речей задіяні не тільки виробники пристроїв, але і оператори стільникового зв'язку, які спрямовані в першу чергу на розробку і впровадження найбільш затребуваних послуг Інтернет-речей..

(зазначити, які питання теоретичного і практичного характеру визначають актуальність роботи)

3. Основний зміст роботи викладено на 78 сторінках тексту, містить 39 ілюстрацій. Список використаних літературних джерел налічує 15 найменувань.

4. Зміст розділів роботи *(зазначити за кожним розділом, які питання досліджені та з якою метою)*

Розділ 1. Аналітичний огляд літературних та інших джерел. У розділі обґрунтовано актуальність теми роботи по розробці моделі IoT-системи захисту банківської комірки, проведено аналіз архітектури IoT-систем, аналогів програмного продукту, можливостей Cisco Packet Tracker щодо моделювання IoT-систем, а також здійснено постановку завдання на розробку.

Розділ 2. Розробка моделі IoT-системи захисту банківської комірки. У розділі розроблено структурну схему IoT-системи захисту банківської комірки за допомогою витої пари технології Ethernet, алгоритми функціонування моделі IoT-системи захисту банківської комірки, здійснено вибір інструментів для реалізації моделі IoT-системи захисту банківської комірки.

Розділ 3. Оцінка працездатності та ефективності моделі IoT-системи захисту банківської комірки. Реалізовано модель IoT-системи захисту банківської комірки, досліджено працездатність та ефективність моделі IoT-системи захисту банківської комірки, здійснено аналіз отриманих результатів.

5. Висновки обґрунтовані, повністю відповідають поставленим завданням, мають теоретичне та практичне значення. Практичне значення отриманих результатів полягає у розробці моделі IoT-системи захисту банківської комірки.

(обґрунтованість, теоретичне і практичне значення)

6. Список використаних літературних джерел. Джерела опрацьовані у повному обсязі, відповідають змісту роботи.

(обсяг і повнота опрацювання за змістом роботи)

7. Додатки _____

(перелік додатків та повнота їх опрацювання за змістом роботи)

8. Таблиці та ілюстрації Ілюстрації носять інформаційний характер, є доповненням до змісту роботи. _____

(призначення таблиць та їх використання для підтвердження досліджень в роботі)

9. Якість оформлення роботи на достатньому рівні _____

10. Ставлення студента (ки) до виконання випускної кваліфікаційної роботи відповідально

11. Висновок

В цілому при написанні випускної кваліфікаційної роботи студент (ка) **Хомицький Д.А.**

(прізвище та ініціали)

вміння /невміння/ підібрати необхідну літературу, інформаційну базу, проаналізувати їх, зробити відповідні висновки і оформити свої дослідження у випускній кваліфікаційній роботі.

Випускна кваліфікаційна робота студента (ки) **Хомицького Д.А.**

(прізвище та ініціали)

написана на достатньо /недостатньо/ високому рівні та може /не може/ бути допущена до захисту в Екзаменаційній комісії.

Науковий керівник д. п.н., доцент, професор кафедри інженерії програмного забезпечення

(вчений ступінь, звання, посада)

(підпис)

Вакалюк Т.А.

(прізвище та ініціали)

«____»_____20____ р.

РЕЦЕНЗІЯ

на кваліфікаційну роботу
студента 4 курсу групи КІ-2 спеціальності 123 «Комп'ютерна інженерія»
факультету інформаційно-комп'ютерних технологій
Державного університету «Житомирська політехніка»
Хомицького Дмитра Анатолійовича
на тему: „Проект IoT-системи захисту банківської комірки"

Кваліфікаційна робота студента Хомицького Д.А. „Проект IoT-системи захисту банківської комірки" присвячена актуальній тематиці – розробці та впровадженню у практику діяльності банків IoT-систем.

Робота містить реалізований у середовищі моделювання Cisco Packet Tracer проект захисту комірки та пояснювальну записку наступного складу: вступ, три розділи основного тексту, висновки, список використаної літератури, два додатки.

У першому розділі пояснювальної записки проведено аналіз : літературних джерел, аналогів моделі, середовищ для моделювання , зокрема, у розділі розглянуто: архітектуру IoT-систем, можливості Packet Tracer для створення імітаційної моделі.

У другому розділі пояснювальної записки здійснено проектування моделі IoT-системи захисту , зокрема, у ньому розроблено структурну схему, алгоритм функціонування і підібрано інструменти Cisco Packet Tracer для створення даної моделі.

У третьому розділі пояснювальної записки здійснено реалізацію моделі та оцінку працездатності і ефективності IoT-системи. У розділі наведено опис процесу налагодження та тестування побудованої моделі. Також у цьому розділі здійснено перевірку функціоналу IoT-системи захисту банківської комірки.

Зміст роботи відповідає поставленому завданню. У цілому кваліфікаційна робота виконана на високому рівні. Тема роботи розкрита повністю. Студентом продемонстровані ґрунтовні теоретичні знання та практичні фахові навички, що отримані як під час навчання, так і у процесі виконання роботи. Матеріал пояснювальної записки викладено логічно, науково і технічно грамотно. Ілюстративний матеріал (рисунки), що наведений у роботі, оформлений якісно і наведений у відповідних місцях. Оформлення роботи у цілому відповідає вимогам стандартів. Проект мережі реалізовано у середовищі моделювання Cisco Packet Tracer. Його функціонал відповідає заявленому.

Позитивною стороною є те, що отримані результати мають практичну цінність – розроблений проект планується до впровадження в діяльності банківських систем.

У цілому кваліфікаційна робота студента Хомицького Д.А. „Проект IoT-системи захисту банківської комірки" відповідає вимогам рівня вищої освіти «бакалавр» спеціальності 123 «Комп'ютерна інженерія» і рекомендується до захисту в ЕК.

Пропонована оцінка роботи – «відмінно».

Рецензент,
декан факультету
інформаційно-комп'ютерних технологій,
к.т.н., доцент Н.М. Лобанчикова,
10 червня 2020 р.

Міністерство освіти і науки України
Державний університет «Житомирська політехніка»
Факультет інформаційно-комп'ютерних технологій
Кафедра комп’ютерної інженерії та кібербезпеки

ПОЯСНЮВАЛЬНА ЗАПИСКА

до випускної кваліфікаційної роботи бакалавра

на тему:

«Проект IoT-системи захисту банківської комірки»

Виконав студент 4-го курсу, групи КІ-2
спеціальності 123 «Комп’ютерна інженерія»
_____ Хомицький Д.А.
Керівник, д. пед. н., професор кафедри ІПЗ,
доцент _____ Вакалюк Т.А.
Консультант, к.т.н., завідувач кафедри ІПЗ,
доцент_____ Пулеко І.В.
Рецензент, декан ФІКТ, к. т. н., доцент
_____ Лобанчикова Н.М.

Житомир – 2020

ДЕРЖАВНИЙ УНІВЕРСИТЕТ «ЖИТОМИРСЬКА ПОЛІТЕХНІКА»

ФАКУЛЬТЕТ ІНФОРМАЦІЙНО-КОМП'ЮТЕРНИХ ТЕХНОЛОГІЙ

КАФЕДРА КОМП'ЮТЕРНОЇ ІНЖЕНЕРІЇ ТА КІБЕРБЕЗПЕКИ

СПЕЦІАЛЬНІСТЬ 123 «КОМП'ЮТЕРНА ІНЖЕНЕРІЯ»

ЗАТВЕРДЖУЮ
Завідувач кафедри
комп'ютерної інженерії
та кібербезпеки
Єфіменко А. А.
«06» квітня 2020 р.

ЗАВДАННЯ

на випускну кваліфікаційну роботу

Студента Хомицького Дмитра Анатолійовича

1. Тема роботи: Проект IoT-системи захисту банківської комірки: доктор педагогічних наук, доцент, професор кафедри інженерії програмного забезпечення Вакалюк Т.А.
2. Строк подання роботи студентом: "10" червня 2020р.
3. Зміст розрахунково-пояснювальної записки(перелік питань, які підлягають розробці)
   1. Обґрунтування актуальності теми досліджень.
   2. Аналіз особливостей архітектури IoT-систем.
   3. Розробка моделі IoT-системи захисту банківської комірки.
   4. Дослідження працездатності та ефективності розробленої моделі IoT-системи захисту банківської комірки.
   5. Розробка рекомендацій щодо реалізації IoT-системи захисту банківської комірки.
4. Вихідні дані до роботи: проект розробити на основі Cisco Packet Tracer.
5. Консультанти розділів проекту (роботи)

Розділ	Консультант	Підпис, дата	
		Завдання видав	Завдання прийняв
1,2,3	Пулеко І.В , кандидат технічних наук, доцент завідувач кафедри інженерії програмного забезпечення	06.04.2020 р.	11.05.2020 р.

Керівник _____ Вакалюк Т.А.

Календарний план

№ з/п	Назва етапів випускної кваліфікаційної роботи	Термін виконання етапів роботи	Примітка
1	Постановка задачі	6 квітня 2020 – 7 квітня 2020	Виконано
2	Пошук, огляд та аналіз аналогічних розробок	8 квітня 2020 – 11 квітня 2020	Виконано
3	Формулювання технічного завдання	12 квітня 2020 – 16 квітня 2020	Виконано
4	Опрацювання літературних джерел	17 квітня 2020 – 19 квітня 2020	Виконано
5	Проектування структури	20 квітня 2020 – 24 квітня 2020	Виконано
6	Написання програмного коду	25 квітня 2020 – 30 квітня 2020	Виконано
7	Налагодження	1 травня 2020 – 4 травня 2020	Виконано
8	Тестування роботи системи	5 травня 2020 – 9 травня 2020	Виконано
9	Аналіз результатів тестування	10 травня 2020 – 13травня 2020	Виконано
10	Написання пояснювальної записки	25 травня 2020 – 29 травня 2020	Виконано
11	Підготовка презентації та виступу	30 травня 2020 – 31 травня 2020	Виконано
12	Отримання відгуку керівника та рецензії на кваліфікаційну роботу	4 червня 2020 – 8 червня 2020	Виконано
13	Захист	13 червня 2020	Виконано

Студент _____Хомицький Д.А.

Керівник_____Вакалюк Т.А.

РЕФЕРАТ

Випускна кваліфікаційна робота бакалавра направлена на розробку IoT-системи захисту банківської комірки. Пояснювальна записка до випускної роботи містить 78 сторінок та 39 ілюстрацій.

Метою роботи є створення проекту IoT захисту банківської комірки за допомогою Cisco Packet Tracer .

В роботі визначено основні завдання на розробку системи, проаналізовано аналоги систем IoT, обґрунтовано вибір Packet Tracer, який містить все необхідне для створення IoT-системи. Наведено схему роботи, опис елементів і алгоритмів даного проекту.

КЛЮЧОВІ СЛОВА: IOT, СИСТЕМА, CISCO, PACKETTRACER, ЗАХИСТ, БАНКІВСЬКА КОМІРКА, РОЗУМНИЙ БУДИНОК.

Зм.	Арк.	№ докум.	Підпис	Дата	КІК.КР.Б-123-20-ПЗ		
Розроб.		Хомицький Д.А.		19.06.20	Проект IoT-системи захисту банківської комірки	Літ.	Арк. 4 / Аркушів 78
Керівник		Вакалюк Т.А.		19.06.20			
Консультан		Пулеко І.В.				ФІКТ, група КІ-2	
Рецензент		Лобанчикова Н.М.		19.06.20			
Зав. каф.		Ефіменко А.А.		19.06.20			

ABSTRACT

Bachelor's graduate work consists of an IoT bank cell protection system and documentation. Documentation for bachelor’s graduate work contains 80 pages and 39 illustrations.

The goal of the graduate work is to create an IoT project for bank cell protection using Cisco Packet Tracer.

The main tasks for system development are defined in the work, analogs of IoT systems are analyzed, the choice of Packet Tracer is substantiated, which contains everything necessary for creation of an IoT-system. The scheme of work, the description of elements and algorithms of the given project have resulted.

KEYWORDS: IOT, SYSTEM, CISCO, PACKET TRACER, PROTECTION, BANKING CELL, SMART HOUSE.

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

IoT - Internet of Things.

M2M - Machine-to-Machine.

ІТ - Інформаційні технології.

ІЧ - Інфрачервоний

RFID - Radio frequency identification.

DHCP - Dynamic Host Configuration Protocol.

GNS3 - Graphical Network Simulator-3.

MCU - Microcontroller Unit.

PT - Packet Tracer.

SBC - Session Border Controller.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ 4
ВСТУП 6
РОЗДІЛ 1. 8
АНАЛІТИЧНИЙ ОГЛЯД ЛІТЕРАТУРНИХ ТА ІНШИХ ДЖЕРЕЛ 8
1.1. Обґрунтування актуальності теми роботи по розробці моделі IoT-системи захисту банківської комірки 12
1.2. Аналіз архітектури IoT-систем 22
1.3. Аналоги програмного продукту 27
1.4. Аналіз можливостей Cisco Packet Tracer щодо моделювання IoT-систем 29
1.5 Постановка завдання на розробку 33
Висновок до розділу 1 34
РОЗДІЛ 2. 35
РОЗРОБКА МОДЕЛІ ІОТ-СИСТЕМИ ЗАХИСТУ БАНКІВСЬКОЇ КОМІРКИ 35
2.1. Розробка структурної схеми IoT-системи захисту банківської комірки допомогою витої пари технології Ethernet. 35
2.2. Розробка алгоритмів функціонування моделі IoT-системи захисту банківської комірки 40
2.3. Вибір інструментів Cisco Packet Tracer для реалізації моделі IoT-системи захисту банківської комірки 44
Висновок до розділу 2 49
РОЗДІЛ 3. 50
ОЦІНКА ПРАЦЕЗДАТНОСТІ ТА ЕФЕКТИВНОСТІ МОДЕЛІ ІОТ-СИСТЕМИ ЗАХИСТУ БАНКІВСЬКОЇ КОМІРКИ 50
3.1 Реалізація моделі IoT-системи захисту банківської комірки 50
3.2.Дослідження працездатності та ефективності моделі IoT системи захисту банківської комірки. Аналіз отриманих результатів. 54
Висновок до розділу 3 57
ВИСНОВКИ 58
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ 59
ДОДАТКИ 61

ВСТУП

Актуальність. В даний час інформаційні технології знаходяться на стадії бурхливого розвитку. Щодня у світі з’являється велика кількість нових систем, технологій, програмного забезпечення та пристроїв побудованих на основі технології інтернету речей. Кількість підключених пристроїв до глобальної мережі з кожним роком збільшується. За розрахунками консалтингового підрозділу Cisco IBSG в проміжку між 2008 і 2009 роками число підключених до інтернету предметів перевищило кількість людей, до 2015 року кількість підключених пристроїв досягло 25 мільярдів, а до 2020 - 50 мільярдів. Таким чином, відбувся перехід від «Інтернету людей »до« Інтернету речей », IoT (Internet of Things). У загальному випадку, під Інтернетом речей розуміється безліч різноманітних приладів, датчиків, пристроїв, які об'єднані в мережу за допомогою будь-яких доступних каналів зв'язку, що використовують різні протоколи взаємодії і єдиний протокол доступу до глобальної мережі. Спільним протоколом є IP. Інтернет речей проникає в різні пристрої і додатки, які можуть використовуватися у всіх галузях промисловості і таким чином з'єднує величезну кількість пристроїв.

За останні п'ять років, як пристрої зв'язку, лідирують смартфони, які допомагають вирішувати повсякденні завдання. Далі йдуть планшети, з мобільною операційною системою і сенсорним інтерфейсом. Як тільки Інтернет-речей стане повсякденним, збільшиться кількість споживчих додатків і бізнес-додатків, задіяних в цій новій системі, яку можна описати поняттям "підключене життя". Це включає будівельну і побутову автоматизацію, системи опалення та кондиціонування повітря, управління дорожнім рухом (наприклад, розумний світлофор), організацію турботи про літніх людей, системи безпеки, а також підключені до інтернету автомобілі і зовнішня реклама. В розвитку технологій Інтернету-речей задіяні не тільки виробники пристроїв, але і оператори стільникового зв'язку, які спрямовані в першу чергу на розробку і впровадження найбільш затребуваних послуг Інтернет-речей.

Метою випускної кваліфікаційної роботи бакалавра є створення моделі IoT-системи захисту банківської комірки.

Основні задачі дослідження:

1.Провести аналіз відомих ресурсів із даної предметної галузі.

2.Провести дослідження об'єкту.

3.Здійснити вибір методів та засобів для створення моделі.

*4.Провести аналіз архітектур*и *IoT.*

5.Реалізувати модель.

Об'єктом дослідження є – процес захисту банківської комірки створеної за технологіями IoT.

Предметом дослідження є використання Cisco Packet Tracer для реалізації моделі IoT-системи захисту банківської комірки.

Результатом випускної кваліфікаційної роботи бакалавра є завершена модель IoT-системи захисту банківської комірки.

РОЗДІЛ 1
АНАЛІТИЧНИЙ ОГЛЯД ЛІТЕРАТУРНИХ ТА ІНШИХ ДЖЕРЕЛ

Є мільярди пристроїв, які пов'язані один з одним чимось, що забезпечує інтелектуальну систему. Ці системи та інтелектуальні пристрої обмінюються даними через хмару, яка аналізується, щоб допомогти різним чином трансформувати бізнес, життя людей та світ. Банки можуть мати повне уявлення про фінанси клієнтів у режимі реального часу, оскільки клієнти використовуватимуть розумні пристрої для доступу до даних. Дані легко поділяються між мільйонами пристроїв від кінцевої сторони клієнта до кінцевої сторони банку. За допомогою даних, які банки збирають, вони можуть передбачати потреби клієнтів та надавати рішення, а також поради, які допомагають клієнтам приймати обґрунтовані та розумні рішення стосовно їхніх фінансів. Тому "Банк речей" стає потужним інструментом сприяння збільшенню лояльності клієнтів для банків, що, в свою чергу, приносить більше бізнесу для банків. Крім того, банки взаємодіють і спілкуються з клієнтом, часто надаючи поради через мобільні телефони. Ви можете побачити подібний підхід щодо звичок споживача у витратах. Банки можуть зв’язуватися зі своїм клієнтом різними способами - наприклад, пропонуючи поради, а також винагороджуючи клієнтів в інших сферах життя, не обов'язково у фінансах. Коли це станеться, банки можуть формувати лояльність клієнтів.

Що стосується банківської справи, IoT пропонує багато переваг[1]. Він надає корисні, легкі в доступі послуги не лише боржникам, але й клієнтам кредитних карток. Банки здатні проаналізувати, як клієнти використовують кіоски банкоматів у різних районах та зменшити або збільшити встановлення банкоматів у цих зонах залежно від обсягів використання. Крім того, банки також можуть використовувати IoT, щоб наблизити послуги на вимогу до клієнтів, встановивши кіоски та підвищивши доступність банківських послуг для клієнтів. IoT пропонує дані клієнтів, які допомагають банкам визначати потреби бізнесу своїх клієнтів, їх ланцюжок вартості, наприклад, роздрібні торговці,

постачальники та дистрибютори. Ці дані також допомагають банкам отримувати уявлення про клієнтів. Інформація про клієнтів, яка доступна через IoT, допомагає банкам надавати послуги з доданою вартістю, персоналізовані банківські послуги та продукти, а також фінансову допомогу для того, щоб забезпечити вигідну ситуацію для залучених сторін - клієнта та банку.

У сільськогосподарському секторі фермери, які є клієнтами банків, мають можливість отримати з Інтернету рішень. Наприклад, банки здатні проаналізувати виробництво сільського господарства на додаток до інших умов вирощування сільськогосподарських культур, що дозволяє банкам оцінювати вихідну продукцію сільськогосподарських культур. На основі врожаю, який оцінюється через IoT, банки здатні забезпечити гнучкість, коли мова йде про фінансові умови, що базуються на очікуваній врожайності, а також частоті та урожайності. Така інформація може допомогти вибудувати міцніші відносини між фермером та банкіром.

Крім того, банки також можуть передбачити шахрайство з переходом на кредитну картку та дебет, що дозволяє їм вжити відповідних заходів для запобігання проблеми. Коли клієнт розгортає свою картку, він забезпечує перевірку власника рахунку через місце розташування пристрою, і залежно від інформації, отриманої, банк може впевнено відхилити або затвердити транзакцію. Також сенсорні пристрої можуть бути встановлені на складах позичальників для відстеження сировини та інвентарю. Відстеженні дані можуть допомогти банку зменшити залишок на рахунку або забезпечити погашення позики, яка була отримана під час продажу товарів. Таким чином, банки можуть мінімізувати накладні витрати на відстеження та не дозволяти позичальникам поривatись у сумну практику.

Банки, яким потрібні додатки IoT, можуть найняти розробників IoT, які допоможуть не лише у створенні, але й у обслуговуванні цих додатків. Програми можна налаштувати для задоволення потреб банків та вирішення певної проблеми.

Інтернет-речей ґрунтується на трьох базових принципах. По-перше, повсюдно поширену комунікаційну інфраструктуру, по-друге, глобальну ідентифікацію кожного об'єкта і, по-третє, можливість кожного об'єкта відправляти і отримувати дані за допомогою персональної мережі або мережі Інтернет, до якої він підключений. В IoT кожна річ має свій ідентифікатор, які спільно утворюють континуум речей, здатних взаємодіяти один з одним, створюючи тимчасові або постійні мережі. Так речі можуть брати участь в процесі їх переміщення, ділячись інформацією про поточну геопозицію, що дозволяє повністю автоматизувати процес логістики, а маючи вбудований інтелект, речі можуть змінювати свої властивості і адаптуватися до навколишнього середовища. Інтернет-речі мають єдиний протокол взаємодії, згідно з яким будь-який вузол мережі рівноправний в наданні своїх сервісів. Кожен вузол мережі інтернет-речей надає свій сервіс, надаючи якусь послугу поставки даних. У той же час вузол такої мережі може приймати команди від будь-якого іншого вузла. Це означає, що всі інтернет-речі можуть взаємодіяти один з одним і вирішувати спільні обчислювальні завдання. Інтернет-речі можуть утворювати локальні мережі, об'єднані якоюсь однією зоною обслуговування.

Потужність пристроїв IoT безмежна . Однак також є і їх вразливість. Ця крихкість ускладнюється застарілою та складною системою, яка її формує. Ці системи містять високочутливу та цінну інформацію для споживачів, яка може отримати значну винагороду для хакерів. Все більше і більше цих даних буде надалі генеруватися та ставати все більш доступними та бажаними, оскільки кількість підключених пристроїв, користувачів та взаємодій зростає з експоненціальною швидкістю. Тому організації швидко повинні придумати спосіб зберігання інформації, її відстеження та захист[2].

Додатковим аспектом є те, що пристрої IoT часто «невидимі»; ми, як правило, про них забуваємо, і виробники пристроїв не є фахівцями з кібербезпеки, тому довіряти самій системі / пристрою безпеки - це проблема. Скажімо, існує вразливість і щось потрібно оновити; у багатьох випадках

організації пропускатимуть оновлення, оскільки це може поставити під загрозу операцію або може коштувати чималих грошей на організацію відключення.

Є багато організацій, які вкладають багато грошей у механізми захисту своїх мереж, периметрів і кінцевих точок, тому зловмисники будуть використовувати шлях найменшого опору щодо пристроїв, пов'язаних з поверхнею атаки, особливо в бездротовому середовищі. Однак організації не знають, що небезпека можлива не лише з корпоративної мережі, а також через її повітряний простір, який знаходиться під загрозою. Хакери можуть підключитися через P2P безпосередньо до цих активів і звідти потрапити у корпоративну мережу.

Отже потрібен захист, який швидко та легко ідентифікує, відстежує та захищає всі смарт-підключені пристрої, що працюють у бізнесі та навколо нього. Це спеціальне рішення щодо кібербезпеки повинне контролювати як пристрій IoT, так і його активність 24 х 7, та нейтралізувати загрозу. Роблячи це, організація зможе виявити, коли і які пристрої загрожують, а також зменшити загрозу в режимі реального часу, не шукаючи її фізично. Відповідь лежить не в самому пристрої, а в рішенні, яке надає команді операцій з безпеки видимість та контроль.

Безпека повинна здійснюватися поза операційною мережею чи інфраструктурою організації спеціалізованими співробітниками служби безпеки, щоб вони могли забезпечити правильне рішення, не довіряючи самій інфраструктурі чи операційній мережі.

Зібрані дані можуть бути дуже складними для прийняття правильних рішень банками або фінансовими установами. Банки повинні заглиблюватися в управління даними, безпеку і захищати інформацію про клієнтів, а також підтримувати стандарти конфіденційності.

В рамках IoT всі дані транзакцій, включаючи інформацію, що надсилається через інтелектуальні пристрої і розумні годинники, будуть доступні для банків і фінансових установ. Поряд з даними, банки також мають доступ до місця розташування клієнта, що може призвести до порушення конфіденційності.

Дотримання стандартів конфіденційності при ефективному використанні інформації є однією з таких проблемних областей.

Банки і фінансові установи збирають багато інформації від клієнтів через безліч різних каналів. Будь-яке порушення даних може призвести до серйозних наслідків для банків. Порушення даних і злом даних можуть завдати величезної шкоди клієнтам і зруйнувати їхні стосунки з банками.

Банки повинні використовувати найкращі і новітні технології захисту даних і повинні вживати превентивні і коригуючі заходи для забезпечення безпеки інформації даних.

1.1. Обґрунтування актуальності теми роботи по розробці моделі IoT-системи захисту банківської комірки

Сучасний світ, безумовно, не став би таким, яким ми звикли його бачити, якби так стрімко не розвивалися технології інформаційної сфери. З кінця 20 століття і до сьогоднішнього моменту здійснений справжній прорив в даному секторі, ми можемо переконатися в цьому глянувши навколо себе. Персональні комп'ютери практично в кожному будинку, бездротові мережі, можливості передачі великого обсягу даних в будь-який куточок планети за лічені секунди, все це підіткнув розвиток і колосальне зростання військової, економічної і багатьох інших сфер. В індустрії інформаційних технологій на даний момент задіяні в плідній роботі на благо майбутнього мільйони людей по всьому світі, податкові відрахування в казну держав обчислюються мільярдами доларів. Традиційні галузі економіки за темпами розвитку сильно відстають від індустрії інформаційних технологій. Результатом діяльності IT є інформаційні продукти у вигляді баз даних, програмного забезпечення і служб, технічних засобів і багато чого іншого. Прогрес інформаційних технологій розширює кругозір всіх, хто стикається з ним, дає можливість ефективно використовувати ресурси інформації і знань, накопичених суспільством за довгі роки існування. У тенденції розвитку інформаційних технологій, безсумнівно, входить мета

створення взаємозв'язку між фізичними, а також логічними складовими системи. Важливим фактором тут є той факт, що IT в основному створюють стандарти на програмні і апаратні засоби, багато в чому автоматизуючи виробництво на підприємствах. Також ними ж здійснюється інтеграція всіх систем. Часто розвиток навмисно стримується багатьма компаніями. Також тенденції розвитку інформаційних технологій включають в себе глобалізацію інформаційного бізнесу. Будь-яка людина споживає інформацію, тому можливості ринку безмежні, хоча доводиться здійснювати свою діяльність в умовах жорсткої конкуренції.

Інтернет речей відіграє ключову роль в наші дні, від управління пасажирськими потоками аеропортів до будівництва опалення та догляду за старшими.

Можливість мережевої електроніки стандартними засобами встановлена для модернізації інтелектуального управління пристроєм. Він представляє світ, описаний так званим IoT, де електронні машини передають дані в хмару через Інтернет за допомогою TCP / IP.

На сьогоднішній день Інтернет-домашня автоматизація можлива завдяки винаходу обчислювальних пристроїв з низькими витратами, таких як Raspberry Pi, радіочастотної мережі та інфрачервоного інтерфейсу для IP-інтерфейсів. Дотепер IoT мав великий вплив у нашій рутинній діяльності. Наприклад для тих хто зазвичай втрачає або забуває цінні речі, такі як ключі, пульт телевізора, смартфон або гаманець. Для таких людей з’явився IoT, який може допомогти. Маяки - це крихітний квадратний червоний пристрій, який підключається до вашого телефону на Android, щоб знайти будь-які речі, до яких ви приєднаєте його. Ви можете легко підключити TEKT Tracker до своїх предметів. Якщо ви щось втратили, можете відкрити додаток і отримати інформацію про розташування ваших цінностей.

IoT - це повстання, яке гарантує змінити життя людей. Це станеться через бум доступних обчислень. Насправді головний дизайнер процесора ARM Стів Фурбер каже, що IoT буде наступною великою площею зростання ARM.

Технологічна підтримка IoT може сприймати навколишнє середовище багатьма способами, такими як моніторинг тиску, температури, в'язкості та геопозиціонування. Навіть для приготування їжі використовуються ці системи. У минулому датчики підключались до локальної комп'ютерної системи та керувалися вбудованим модулем. Тепер IoT дозволяє нам використовувати доступну бездротову технологію та передавати дані в хмару на рівні компонентів. Тут також передбачено місце для збереження даних, а також управління та безпеку. IoT підтримує спосіб побудови систем гнучким та економічно ефективним шляхом за допомогою відкритого набору компонентів, які включаються та легко підключаються до вашого робочого місця. Останнім часом IoT ставати все більш популярною темою, особливо в прив'язці до 5G, яка дозволить розширити функціональні можливості і інфраструктуру «розумних» пристроїв. За розрахунками, на їх масове впровадження піде не менше п'яти років. Одна з головних перешкод є відсутність єдиних стандартів роботи для різноманітних пристроїв, що входять в IoT-системи. Однак, поки розробники шукають потрібне рішення, в певній мірі полегшити «спілкування» з «розумними» пристроями може звичайний смартфон.

Мабуть, на сьогодні це одне з оптимальних рішень для комунікації між різними IoT-пристроями і своєрідний пульт управління ними. Іншими словами, сьогодні смартфон не тільки полегшує спілкування з «розумними» речами, а й, будучи повноцінним шлюзом, виступає таким собі перехідником, або базовою платформою для IoT. Цьому сприяє ряд факторів, одним з яких є масовість його поширення. За даними британської компанії IHS Markit, до кінця 2016 року в світі було приблизно 4 млрд смартфонів, а до кінця 2020-го їх кількість може зрости до 6 млрд. Причому, сьогодні у світі проживає близько 7,4 млрд осіб і за прогнозами ООН до кінця 2020 року їх чисельність не перевищить 8 млрд. Тобто відсоток людей, що мають смартфони, буде рости дуже істотно.

Більшість сучасних моделей смартфонів є досконалими технічними пристроями. Їх програмне забезпечення і потужність процесора дають можливість підключати до них і керувати великою кількістю різноманітних IoT-

пристроїв. А наявний в смартфоні базовий набір датчиків і інших пристроїв дозволяє будувати мережу, розміри якої можуть набагато перевищувати «людський» Інтернет. Крім того, смартфони можна використовувати в якості хаба на шляху до хмари, де зберігатися вся необхідна інформація для повноцінної взаємодії IoT-пристроїв. Таким чином, не дивлячись на свою «навороченність», смартфон залишається простим у використанні, а головне, доступним масовому споживачеві девайсом, що виконує функції передачі / отримання даних, управління та конфігурації IoT-пристроїв. Але все ж, головною ланкою в комунікації залишається людина. Вона в більшості випадків є ініціатором взаємодії пристроїв в IoT-системі. А це говорить про те, що людина сама стає її частиною і може керувати потоками інформації, ініціювати або приймати команди управління. Тобто, з її участю утворюється нова система, яка отримала назву «Соціальний Інтернет речей» (SIoT).

Безумовно, з приходом зв'язку 5G і прогнозованому збільшенні в світі «розумних» пристроїв до 50-100 млрд буде необхідний і якісно новий рівень взаємодії IoT. А для цього знадобитися сумісність протоколів їх роботи, оскільки саме можливість безперебійної комунікації різних за своїм функціоналом «розумних» пристроїв буде головною причиною масового поширення IoT-систем. До речі, ще в 2015 році американська консалтингова компанія McKinsey & Company встановила, що близько 40% загальної економічної цінності різних IoT-систем становить здатність взаємодіяти один з одним. Обмежена сумісність різко зменшує цінність і корисність такої системи і призводить до її незатребуваності користувачами. Тому, для використання всіх переваг IoT-концепції і можливостей 5G, розрізнені сьогодні системи повинні навчитися взаємодіяти і обмінюватися інформацією. Вирішення цієї проблеми є складним і витратним за часом і фінансів завданням, але розробники додатків і виробники пристроїв спільно з компаніями, які впроваджують IT-системи, повинні будуть домовитися про необхідний мінімум загальних стандартів і протоколів. Так що в будь-якому випадку будуть знайдені рішення, які приведуть до їх уніфікації, приблизно так само, як це було з набором мережевих протоколів TCP / IP. Коли

це станеться, «розумні» пристрої в величезних кількостях зможуть організовуватися в глобальну мережу фізичних об'єктів, що підключаються до інтернету через шлюзи (хаби або спеціалізовані IoT-платформи). Останні виконують необхідну трансляцію між протоколами, які використовуються в мережах зв'язку і в самих пристроях. При цьому шлюзом зможе виступати і безліч апаратних платформ, починаючи від звичайних роутерів і закінчуючи спеціалізованими «коробковими» рішеннями. Безумовно, щоб забезпечити з'єднання і обслуговування такої кількості пристроїв, а також надати відповідні інструменти для обробки і зберігання отриманих від них даних, необхідна глобальна програмна платформа. Причому, в ідеалі, вона повинна бути на багато користувачів і мати можливість об'єднувати не тільки IoT-пристрої, але і людей. І найбільш підходящим варіантом тут бачиться SIoT-платформи. З їх допомогою користувачі зможуть не тільки керувати підключеним пристроями, а й створювати нові конфігурації пристроїв і всілякі сценарії їх використання. Дані будуть зберігатися в хмарі, а керовані «розумні» пристрої можуть перебувати на великих відстанях один від одного, навіть на різних континентах. На перший погляд, для звичайної людини все, що пов'язано з IoT, може здаватися чимось неймовірним. Але це вже буквально завтрашня реальність, наступ якої бажано не пропустити. Світ навколо змінюється поза нашою волею і в один прекрасний день ваш звичний телевізор раптом стає центром управління всього і вся, оскільки все необхідне для цього вже закладено виробником. У ньому є дисплей, мікроконтролер, мережева карта, операційна система і все що потрібно для комунікації з іншими мережевими пристроями. Прописавши за допомогою IoT Конструктора відповідні модулі і підключивши свій Smart TV до хмари, будь-який користувач може повноцінно і в потрібних параметрах взаємодіяти з новим, оточуючим його цифровим і бездротовим світом. І ця взаємодія може бути найнесподіванішою. Насправді IoT - це величезне поле для експериментів. І не дивлячись на те, що ера «розумних» пристроїв тільки починається, багато хто говорить про те, що майбутня IT революція буде пов'язана саме з Інтернетом речей.

Інтернет речей пропонує численні переваги для організацій. Більшість сучасних підприємств вже використовують ІОТ для автоматизації та спрощення багатьох своїх щоденних завдань. Підключені пристрої інтегруються в існуючі та розвиваються бізнес-процеси. Промисловий Інтернет речей орієнтований на автоматизацію - не тільки для зниження операційних витрат, а й для підвищення продуктивності, поліпшення бізнес-процесів, підвищення досвіду клієнтів, розширення на нові ринки та отримання додаткових потоків доходу. Промисловий IoT полягає в тому, щоб бути добре поінформованим для прийняття своєчасних і кращих рішень. Це може допомогти організаціям перейти від традиційного способу продажу апаратних чи програмних засобів до продажу всеосяжних рішень - продуктів, вбудованих датчиками та бездротовим зв'язком для отримання цінних даних. Платформи IoT призначені для збору та обробки великих обсягів даних, не завдаючи жодного впливу на продуктивність системи. Поряд зі збором даних, IoT має можливість виконувати аналітику даних та зробити її доступною для всіх потрібних команд.

Це виключає необхідність наймання дорогих бізнес-консультантів або фірм, що займаються дослідженнями ринку. Оскільки IoT генерує та аналізує величезну кількість даних, він є важливим рушієм великих проектів з аналізу даних. Зокрема, він може доставляти велику кількість даних у режимі реального часу. Це дає підприємствам ретельне уявлення про ефективність їхніх продуктів та послуг у реальному світі, що дозволяє швидко вносити необхідні вдосконалення. Цінна інформація, що надається платформами IoT, дозволяє організаціям легко оцінювати попит клієнтів та загальну поведінку при покупці. Незалежно від галузі, компанії можуть використовувати пристрої IoT для ефективного адаптації своїх пропозицій для задоволення інтересів клієнтів, надання розширеного обслуговування клієнтів та підвищення цінності бізнесу. Витрати на управління офісом - це основні витрати, які Інтернет речей може значно зменшити. Через IoT доступні різноманітні продукти, які автоматично контролюють споживання енергії та взаємодіють з іншими пристроями, щоб зменшити загальні витрати. Зустріньте підключений конференц-зал. За

допомогою IoT легко обмінюватися інформацією та ефективно співпрацювати з іншими членами команди, де б вони не були. Дані можна збирати, оновлювати та обмінюватися з усіма відповідними членами команди в режимі реального часу одним натисканням кнопки. За допомогою різних пристроїв IoT можна відстежувати результати роботи всіх працівників, а також вдосконалювати операції у всіх місцях. Аеропорт міста Лондон також використовує Інтернет речей, щоб покращити досвід подорожі для клієнтів. Для вимірювання швидкості подорожі вони використовують розпізнавання обличчя. В аеропорту технологія Hitachi використовується для вимірювання руху клієнтів.

Вони побудували систему для вимірювання проїзду через аеропорт та отримання даних у реальному часі, наприклад, у багажному залі, як працює пасажиропотік. Ця інформація зберігається в центральному контролі та розробці для покращення процедури. Це виявилося дуже корисним для них пріоритетом їх інвестицій.

Незалежно від майбутнього для IoT майстерність смарт-пристроїв стане звичним у нашому житті. Моральні наслідки полягали в тому, щоб такі пристрої вирішували щоденні завдання, які звичайно виконували б люди.

Колись консервативний і обережний до нововведень банківський сектор в останні роки став одним з найбільш автоматизованих, і багато в чому це відбулося завдяки рішенням на базі Інтернету речей (Internet Of Things, IoT). Банкіри бачать в цій технології безліч потенційних можливостей, які можуть допомогти банківському бізнесу перейти на новий, цифровий рівень монетизації інформації.

Поява покоління більш «просунутих» і вимогливих клієнтів змусило банки адаптуватися до швидких темпів розвитку IT і впроваджувати відповідні інноваційні рішення. Залишатися на колишньому рівні означало для них швидку втрату конкурентоспроможності перед стрімким натиском FinTech-компаній. Поява концепції Інтернету речей сприяла радикальній трансформації банківської галузі.

Сьогодні нові цифрові технології та інновації формують діючі операційні моделі банківського сектора і його екосистему. Найбільш популярними нововведеннями стали хмарні технології, Big Data, AI, ML, автоматизація процесів, блокчейн і IoT. Перехід на більш надійну цифрову базу стало пріоритетом для тих банківських установ, які прагнуть до розвитку і конкурентоспроможності на фінансовому ринку.

Як результат, зараз клієнти отримали можливість зв'язуватися з банком і управляти своїми рахунками через планшети, смартфони, ноутбуки і навіть «розумні» годинники.

Переваги, які дає бездротовий зв'язок, мотивують розробку нових банківських IoT-додатків, що дозволяють проводити збір інформації про переваги і потреби клієнтів банків.

У міру зростання клієнтського попиту на комфортний і мобільний банківський сервіс, банки намагаються якомога швидше його задовольнити. Застосування IoT-технологій забезпечує їм збір і аналіз великих масивів банківської інформації, що, в свою чергу, дозволяє надавати клієнтам індивідуальну лінійку необхідних послуг, або пропонувати знижки на вже існуючі. Ці ж дані допомагають банкірам розробляти індивідуальні програми заохочення клієнтів і таким чином підвищувати їхню лояльність.

Масштабний збір інформації, який забезпечує Інтернет речей, використовується сьогодні банками не тільки для кращого розуміння і відстеження поведінки клієнтів, але і як інструмент покращення внутрішніх процесів, особливо в частині прийняття рішень по продуктовій стратегії або сфері кредитування. Дані про купівельні переваги використовуються для прогнозування цільових пропозицій по банківським продуктам та визначення індивідуального кредитного ризику потенційних позичальників (особливо тих, які не мають власної кредитної історії) і прийняття зважених комерційних рішень.

Все це дозволяє банкам монетизувати отримані дані в прибуток, причому досить прогнозований і цілком керований. Таким чином, з появою інноваційних

рішень в області Інтернету речей банки змогли підняти персоніфіковане банківське обслуговування на якісно новий рівень.

Практична площина використання IoT в банківському середовищі досить різноманітна. Сьогодні встановлені у відділенні біометричні і позиційні датчики укупі з поліпшеними відеокамерами дозволяють розпізнати клієнта з моменту його входу в приміщення. Аналіз інформації датчиків, розміщених на банкоматах, дає можливість визначити оптимальні зони для установки апаратів. Оперуючи інформацією, заснованої на даних про клієнта і його місцезнаходження, банки тепер можуть визначати їхні потреби, пропонуючи свої продукти і допомагаючи їм приймати прийнятні для обох сторін фінансові рішення. Разом з тим банкіри вважають, що в найближчі два-три роки клієнти почнуть масово проводити транзакції з домашніх «розумних» пристроїв. Варіанти включають можливість здійснювати базові операції через спеціальні програми за допомогою переносних (фітнес-трекери, «розумні» годинники) і голосових (Amazon Echo) пристроїв, невидимих платежів за транспортні та ресторанні послуги (за прикладом Uber, Dine and Dash, Dash Replenishment Service і ін.) і автомобільних GPS-систем. Це досить прості приклади тих величезних можливостей, які дають традиційному банківському сервісу цифрові технології. Можна сказати, що цифрова банківська революція тільки починає набирати обертів. На сьогоднішньому першому її етапі більшість банківських установ вже пропонують обслуговування через веб-сайти і мобільні додатки. Далі слідує повний перехід на дистанційне обслуговування і догляд мобільних сервісів, що дозволяють здійснювати практично всі операції, починаючи від відкриття рахунку і закінчуючи платіжками будь-якої складності. Оскільки фізичної необхідності відвідувати банківські відділення не буде, відпаде потреба у великих мережах і сьогодні, у міру відходу банків в мережу, спостерігається стійкий тренд по їх закриттю. До речі, конкуренція на ринку фінансових послуг стає, мабуть, одним з головних двигунів модернізації банків. Увійшовши на ринок мобільних платежів, вони зіткнулися з активно працюючими там телеком операторами і такими IT-монстрами, як Facebook, Google і Apple. Тому в цій

жорсткій конкуренції за клієнта виживуть тільки ті банки, які зможуть правильно використовувати потоки даних, які генеруються IoT-пристроями, розробляючи на їх основі нові продукти і рішення. Безумовно, використання величезних масивів конфіденційної інформації принесло банкам і нові проблеми, пов'язані з безпекою даних. Наявність детальної персональної «картинки» клієнта і безліч різних типів IoT-пристроїв, означатиме для них необхідність в додаткових рівнях безпеки для усієї використовуваної екосистеми IoT. Тому, потреба в безпеці банківського обслуговування, зав'язаного на роботі з «розумними» пристроями, буде носити постійний характер. Як і інші сфери, вбудовані в IoT-систему, банківський сектор повинен гарантувати клієнтам збереження їх коштів і особистої інформації. Сьогодні у кіберзлочинців є можливість проникнення і злому мережі, використовуючи підключення IoT-пристроїв. Банківські дані вельми чутливі до стороннього втручання, будучи то кредитна історія клієнта або його майновий стан. Незалежно від того, де вони зберігаються і як переміщаються по мережі, ці дані повинні бути надійно захищені. Сьогодні банки застосовують для цього достатньо великий набір інструментів, серед яких шифрування і аутентифікація осіб, що мають доступ до даних. Також, багато хто з них почали впроваджувати біометричну складову авторизації клієнта, щоб переконатися що він той, за кого себе видає. Ну і, звичайно, використання IoT-пристроїв, що володіють високим ступенем вбудованої захисту від злому. На жаль, сьогодні поки що багато виробників не сильно переймаються над цією проблемою, вважаючи за краще перекладати її на користувачів або постачальників. Частково проблема може бути знівельована шляхом проведення роз'яснювальної роботи з клієнтами на предмет захисту «розумних» пристроїв і їх даних від зовнішніх зазіхань. Інтернет речей відкрив банківській сфері двері в цифрове майбутнє, де їх уже чекають найбільш просунуті покоління клієнтів. Але щоб реалізація можливостей IoT почала носити масштабний характер, багатьом банкам ще належить переглянути свою мережеву інфраструктуру, аналітичні системи і хмарну доступність. Орієнтованість на надання традиційних послуг не вимагало особливих змін у налаштуваннях, але для

переходу на новий інноваційний рівень їм потрібна глобально ремоделіровані процеси і якнайшвидше впровадження нових технологій. Безумовно, світ знаходиться на порозі нової технологічної ери, яка принесе величезні зміни в усі галузі, включаючи і банківську. Їх наслідки можна буде відчути вже в найближчі роки. З різних причин величезний потенціал Інтернету речей сьогодні поки ще прихований від користувача, але цифрова трансформація вже набирає швидкість і проглядається в стратегіях багатьох банківських установ і груп. Їх втілення залежатиме від інноваційної активності кожного конкретного банку і готовності його власників і менеджменту проводити зміни. Світова практика «цифрової перебудови» вже показала, що ті, хто в перших рядах почнуть перетворення, отримають серйозні конкурентні переваги, ну а ті, хто запізниться - залишаться на узбіччі фінансових потоків.

1.2. Аналіз архітектури IoT-систем

Коли мова йде про Інтернет речей, велика увага приділяється його потенціалу. Новини про те, що IoT зможе зробити і як це розширить можливості нашого життя, продовжують накопичуватися, але багатьом може здатися, що ці піднесені бачення не втілюються в реальність так швидко, як нам хотілося б. Проте, великі зміни дійсно відбуваються, повільним шляхом , а не гігантськими стрибками. Причина цього досить проста, але, як правило, вона залишається поза увагою громадськості: саме внутрішнє розмаїття систем IoT(рис.1.0) стримує прогрес і часто заважає з'єднанню всіх речей.

Будучи однією з двох, мабуть, найбільших проблем, що стоять перед IoT (інший є безпека), фрагментація лежить в основі Інтернету речей через різнорідної природи речей, до якої вона прагне підключитися. Для запуску будь-якої системи IoT потрібне використання всіх ресурсів, обладнання, програмного забезпечення і систем, якими б різними вони не були, в єдину структуру для формування інтегрованого, надійного і економічно ефективного вирішення[3]. Простіше кажучи, кожне розгортання IoT потребує надійної архітектури IoT,

щоб бути в змозі виконати своє призначення; результуюча ефективність і можливість застосування системи багато в чому залежить від якості розробленої інфраструктури.

Хоча кожна система IoT відрізняється, основа для кожної архітектури Інтернету речей, а також її загальний потік обробки даних приблизно однакові. Перш за все, він складається з речей, які є об'єктами, підключеними до Інтернету, які за допомогою вбудованих датчиків і виконавчих механізмів здатні сприймати навколишнє середовище і збирати інформацію, яка потім передається на шлюзи IoT. Наступний етап складається з систем збору даних IoT і шлюзів, які збирають величезну масу необроблених даних, перетворять їх в цифрові потоки, фільтрують і попередньо обробляють, щоб вони були готові до аналізу. Третій рівень представлений периферійними пристроями, такими, що відповідають за подальшу обробку і розширений аналіз даних. На цьому рівні також можуть з'явитися технології візуалізації і машинного навчання. Після цього дані передаються в центри обробки даних, які можуть бути або хмарними, або встановлені локально. Саме тут дані зберігаються, обробляються і аналізуються для більш глибокого розуміння.

Це чотири рівні архітектури IoT(рис.1.1), описані докладно:

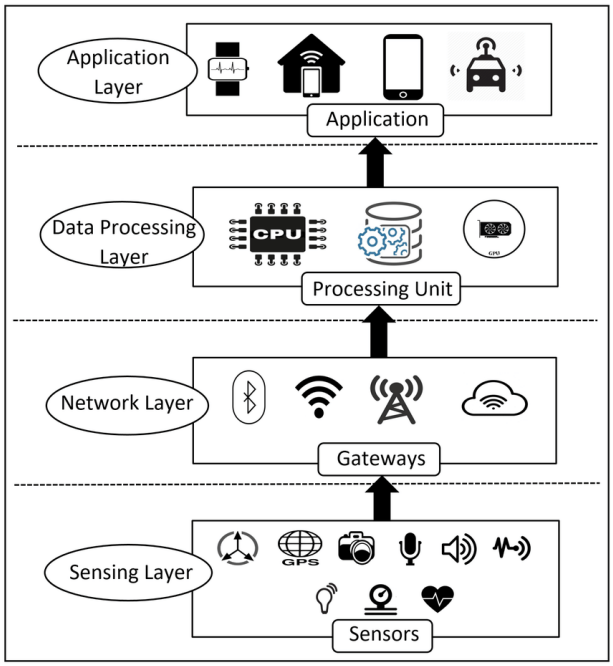


Рисунок 1.1 - Архітектура IoT

В якості основи для кожної системи IoT підключені пристрої відповідають за забезпечення сутності Інтернету речей, а саме даних. Щоб підібрати фізичні параметри в зовнішньому світі або всередині самого об'єкта, їм потрібні датчики. Вони можуть бути вбудовані в самі пристрої або реалізовані як окремі об'єкти для вимірювання та збору даних телеметрії[4]. Що також важливо, так це те, що підключені об'єкти повинні бути здатними не тільки підтримувати двосторонній зв'язок зі своїми відповідними шлюзами або системами збору даних, але також мати можливість розпізнавати і спілкуватися один з одним для збору та обміну інформацією та спільної роботи в режимі реального часу , щоб скористатися наявними можливостями розгортання. Зокрема, в разі пристроїв з обмеженими ресурсами і батареями, досягнення цього є непростим завданням, оскільки такий зв'язок вимагає великої обчислювальної потужності і споживає дорогоцінну енергію і смугу пропускання. Таким чином, надійна архітектура може забезпечити ефективне управління пристроями тільки в тому випадку, якщо вона використовує спеціалізовані, безпечні і легкі протоколи зв'язку, такі як Lightweight M2M, який став провідним стандартним протоколом для управління малопотужними легковагими пристроями, типовими для багатьох IoT варіантів використання(рис.1.2)



Рисунок 1.2 - Варіанти використання IoT

Хоча цей рівень як і раніше функціонує в тісній близькості з датчиками і виконавчими механізмами на певних пристроях, важливо описати його як

окремий етап архітектури IoT, оскільки він має вирішальне значення для процесів збору даних, фільтрації і передачі в прикордонну інфраструктуру і хмарні платформи.(рис.1.3) З огляду на величезні обсяги введення і виведення, які можуть бути створені при розгортанні мільйонів пристроїв, в центрі уваги повинні бути можливості для агрегації, вибору і передачі даних. У якості посередників між пов'язаними об'єктами і хмарою і аналітикою шлюзи і системи збору даних забезпечують необхідну точку з'єднання, яка пов'язує ці шари[5]. Перебуваючи на межі світів ВІД і ІТ, шлюзи полегшують зв'язок між датчиками і іншою частиною системи, перетворюючи дані датчиків в формати, які легко переносяться і можуть використовуватися іншими компонентами системи. Більш того, вони можуть контролювати, фільтрувати і вибирати дані, щоб мінімізувати обсяг інформації, яка повинна бути передана в хмару, що позитивно впливає на вартість передачі по мережі і час відгуку. Таким чином, шлюзи надають місце для локальної попередньої обробки даних датчика, які стискаються в корисні пакети, готові для подальшої обробки. Інший аспект, який підтримують шлюзи, - це безпека. Оскільки шлюзи відповідають за управління інформаційним потоком в обох напрямках, за допомогою належних інструментів шифрування і безпеки вони можуть запобігти витоку даних із хмари IoT, а також знизити ризик зловмисних зовнішніх атак на пристрої IoT.

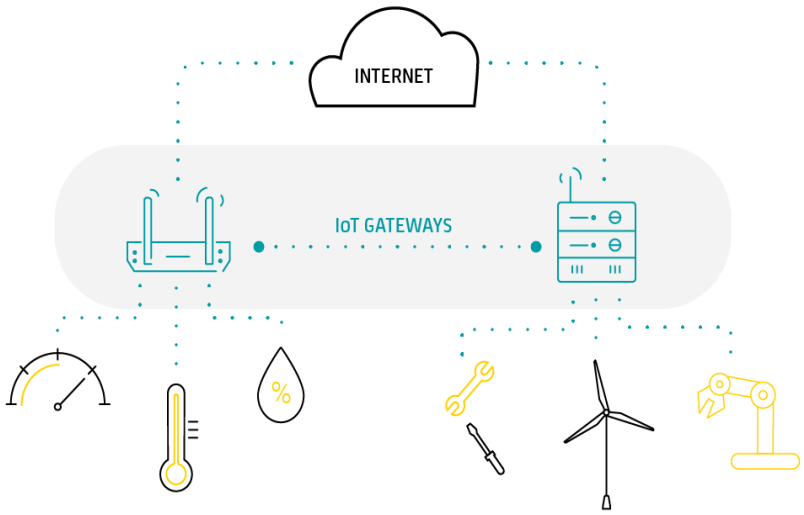


Рисунок 1.3 - Взаємодія елементів IoT

Незважаючи на те, що периферійні пристрої не є неминучим компонентом кожної архітектури IoT, вони можуть принести значні переваги, особливо в великомасштабних проектах IoT. В умовах обмеженої доступності та швидкості передачі даних хмарних платформ IoT прикордонні системи можуть забезпечити більш швидкий час відгуку і велику гнучкість при обробці і аналізі даних IoT. Оскільки швидкість аналізу даних є ключовою в деяких промислових додатках Інтернету речей, останнім часом різко зросла популярність сучасних обчислень в екосистемах промислового Інтернету речей. Оскільки прикордонна інфраструктура може бути розташована ближче до джерела даних у фізичному вираженні, їй простіше і швидше впливати на матеріал IoT в режимі реального часу і надавати вихідні дані у вигляді миттєвого дієвого інтелекту[6]. У цьому сценарії туди пересилаються тільки більші фрагменти даних, яким дійсно потрібна потужність хмари. Мінімізуючи вплив на мережу, можна значно підвищити безпеку, а зниження енергоспоживання і пропускної здатності сприяє більш ефективному використанню бізнес-ресурсів.

Якщо датчики - це нейрони, а шлюз - це основа IoT, то хмара - це мозок в тілі Інтернету речей. На відміну від крайніх рішень, центр обробки даних або хмарна система призначені для зберігання, обробки та аналізу величезних обсягів даних для більш глибокого розуміння з використанням потужних механізмів аналізу даних і механізмів машинного навчання, які прикордонні системи ніколи не зможуть підтримувати. Володіючи все більш широким впровадженням (особливо в архітектурі Industrial IoT) за останні кілька років, хмарні обчислення сприяють підвищенню продуктивності, скороченню незапланованих простоїв і енергоспоживанню, а також багатьом іншим переваг для бізнесу. Володіючи належними для користувача прикладними рішеннями, хмара може надавати можливості бізнес-аналітики та подання, які допомагають людям взаємодіяти з системою, контролювати і контролювати її, а також приймати обґрунтовані рішення на основі звітів, інформаційних панелей і даних, що переглядаються в режимі реального часу.

1.3. Аналоги програмного продукту

Перед тим, як розпочинати роботу над створенням проекту системи захисту, важливим етапом є аналіз та оцінка вже існуючих систем-аналогів. Так як на ринку існує велика кількість систем захисту побудованих на базі інтернету речей, доцільно буде врахувати переваги та недоліки таких систем та по результатам такого аналізу створити систему, яка відібрала найкращі якості.

Xiaomi

Система розумного будинку від одного з лідерів на ринку “Xiaomi” (рис.1.4) , відноситься до бюджетного класу обладнання, яке дозволяє зробити керування різноманітними пристроями, побутовою технікою та системою безпеки максимально просто та ефективно.

Перевагами даної системи є: можливість збільшувати кількість включених у систему датчиків, безпровідний протокол ZigBee, дуже зручне керування за допомогою смартфону, компактність та стильний дизайн.



Рисунок 1.4 - Система захисту “Xiaomi”

Не зважаючи на велику кількість переваг, дана система має ряд недоліків: дуже маленький радіус дії (всього 10 м). Для кожного датчика необхідно своє положення, відсутність резервного живлення.

Ajax

Ajax – це професійна безпровідна система безпеки. На сьогоднішній час, система складається з 27 пристроїв для захисту від пограбувань, пожеж, затоплень, а також для керування електроживленням. Датчики працюють до семи років від комплектної батареї. Для зв'язку пристроїв у системі, використовується розроблений компанією пропрієтарний двохсторонній радіопротокол Jeweller. Радіус дії системи до 2000 метрів, час передачі сигналу тривоги складає 0,15 секунд. Дані про безпеку користувач отримує у вигляді сповіщення, смс повідомлення та звінком від головного хаба (центральний контроллер).(рис.1.5)



Рисунок 1.5 – Система захисту Ajax

Незважаючи на переваги даної системи, існують недоліки такі як: початковий набір не задовільняє всі вимоги повноцінної охорони, вартість пристрою є дещо більшою а ніж інші системи охорони.

SmartOn

SmartOn – це не лише система розумного будинку та охоронна система але й компанія, яка може перетворити будь яке приміщення чи будинок у повноцінну розумну систему. Завдяки великому досвіду компанія налаштовує будь яке обладнання а також створює власне використовуючи мікрокомп'ютери. Управління освітленням, аудіо системами, системою охороною, постачанням води, газу та електроенергії і все це лише маленький список можливостей даної системи.

Недоліками системи є складність використання, та звісно ж ціна. Кожна система є унікальною і розробляється для кожного будинку чи організації індивідуально. Дана система має складний інтерфейс та є ще недостатньо протестована, для того щоб обирати її для захисту та управлінням банківської комірки.

1.4. Аналіз можливостей Cisco Packet Tracer щодо моделювання IoT-систем

У побудові розумної мережі може бути реалізована концепція імітації, розроблена на Cisco Packet Tracer. Деякі інструменти, необхідні для побудови інтелектуальної домашньої мережі, - це домашні шлюзи, які функціонують як шляхи передачі медіа та забезпечують автоматичну адресацію до декількох пристроїв, підключених через бездротові мережі, смартфони, які служать інтерфейсами для керування та контролю електронних пристроїв, а також деякі пристрої, якими слід керувати на основі умов, встановлених на смартфоні .Cisco - світовий лідер у галузі мережевих технологій, що змінюють способи людського спілкування, зв'язку і спільної роботи. Діяльність компанії зосереджена на п'яти

основних технологічних напрямках: магістральна маршрутизація, комутація та послуги; рішення для спільної роботи; віртуалізація центрів обробки даних і хмарні обчислення; відеотехнології; архітектури для трансформації бізнесу. Packet Tracer - це потужний мережевий симулятор, який можна використовувати для підготовки до сертифікації по мережі, дозволяючи студентам створювати мережі з практично необмеженою кількістю пристроїв і усувати неполадки, не купуючи справжні маршрутизатори або комутатори Cisco. Інструмент створений зусиллям Cisco Systems[7]. Мета Packet Tracer - запропонувати учням інструмент для вивчення принципів роботи в мережі, а також для розвитку навичок, специфічних для технології Cisco. Проте, він не повинен використовуватися в якості заміни для маршрутизаторів або комутаторів. Його можна завантажити безкоштовно, якщо у вас є обліковий запис Netacad. Як говорить Cisco, кращий спосіб дізнатися про мережу - це зробити її. Викладачі використовують Packet Tracer для демонстрації складних технічних концепцій і мережевих систем. Студенти використовують Packet Tracer для виконання завдань, працюючи самостійно або в команді. Cisco Packet Tracer має в своєму розпорядженні відмінні інструменти для моделювання та візуалізації мереж, полегшених як для навчання студентів, так і для представників, які працюють під керівництвом фірм Cisco. Програмний симулятор дозволяє налаштовувати (віртуально) різне телекомунікаційне обладнання фірми Cisco (комутатори, маршрутизатори, ip-телефони, розблокування, сервер, міжмережевий екран Cisco ASA і багато іншого) [8].

Компоненти та можливості які надає Cisco Packet Tracer для проектування розумних будинків, розумних міст (рис.1.6)

1 - мережеві пристрої

2 - кінцеві пристрої

3 - компоненти

4 - з'єднання

5 - інше

6 - з'єднання мультикористувачів

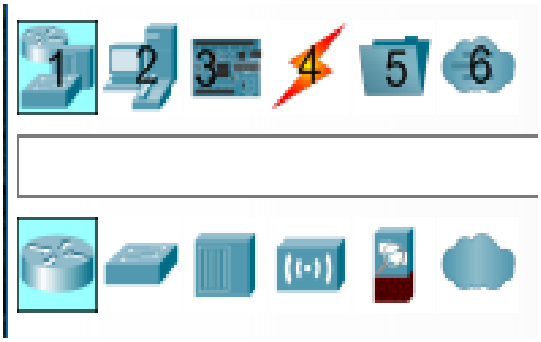


Рисунок 1.6 - Панель елементів

Розглянемо кінцеві пристрої і компоненти.

На вкладці кінцевих пристроїв, що зображена на (рис.1.7), ми маємо кілька категорій:

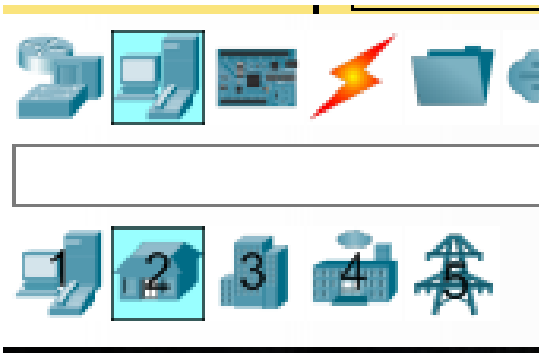


Рисунок 1.7 - Кінцеві пристрої

1 - Кінцеві пристрої (ПК, смартфон, телефон, сервер, телевізор)(рис.1.8):



Рисунок 1.8 - Елементи розумного будинку

3 - розумне місто, який включає в себе такі елементи (рис. 1.9):



Рисунок 1.9 - Елементи розумного будинку

4 - вузли та компоненти електронні (рис. 1.10):



Рисунок 1.10 - Промислові компоненти

5 - Енергосистема (рис.1.11):акумулятор;



Рисунок 1.11 - Елементи енергосистеми

Компоненти (рис. 1.12):

1. Плати MCU і SBC, розумний пристрій

За допомогою плат можна запрограмувати розумний пристрій і створити свій компонент з новими функціями

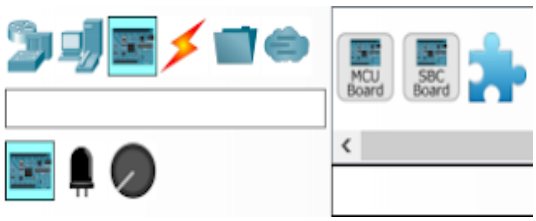


Рисунок 1.12 - Компоненти

2. Виконавчі механізми (рис.1.13)



Рисунок 1.13 - Виконавчі механізми

3. Сенсори (рис. 1.14)



Рисунок 1.14 - Сенсори

Для проектування розумних будинків або розумних міст, сенсори та інші компоненти можна налаштовувати.

1.5 Постановка завдання на розробку

Об'єкт – система керування обладнанням у кімнаті банку.

Умови роботи системи керування обладнанням у кімнаті банку.

Банківські комірки знаходяться в великій кімнаті. У кімнаті дві лампи, $Л_1$ – на початку й $Л_2$ – в кінці, над столом для роботи з банківськими комірками.

На початку кімнати знаходяться двері з RFID Reader, за ними встановлені дискретні датчики $Д_1$ і $Д_2$ для контролю проходження людини. Відстань між

датчиками така, що людина не може помістится між зонами їхньої роботи. Двері блокуються для запобігання проходу іншої людини. Перед столом є площа з дискретним датчиком ваги $Д_3$.

Як спрацює датчик $Д_3$,необхідно включити лампу $Л_1$, а після проходження дверей (тобто сходу людини з датчика $Д_2$) необхідно їх заблокувати. Коли людина відкриє банківську комірку, візьме її вміст і підійде до столу (при цьому датчик ваги $Д_3$ подасть сигнал), потрібно включити лампу $Л_2$. Вимикання лампи $Л_2$ над столом виконується по сигналу датчика $Д_3$, а розблокування дверей по сигналу датчика $Д_2$.

Висновок до розділу 1

У першому розділі був здійснений огляд літературних та інших джерел. У результаті огляду джерел було виявлено актуальність даної роботи. Також була обгрунтована актуальність теми роботи по розробці проекту IoT-системи захисту банківської комірки. У результаті аналізу архітектури IoT-систем, наведено аналоги програмного продукту і показано які переваги і недоліки вони мають .У результаті розгляду можливостей Cisco Packet Tracer було наведено інструменти саме для проектування IoT-систем.

РОЗДІЛ 2

РОЗРОБКА МОДЕЛІ ІОТ-СИСТЕМИ ЗАХИСТУ БАНКІВСЬКОЇ КОМІРКИ

2.1. Розробка структурної схеми IoT-системи захисту банківської комірки допомогою витої пари технології Ethernet.

Інтернет речей (англ. Internet of Things, IoT) є концепцією мережі, яка є системою взаємопов'язаних обчислювальних пристроїв, механічних і цифрових машин, об'єктів, тварин або людей, які мають програмне забезпечення, що дозволяє здійснювати передачу та обмін даними між фізичним світом та комп'ютерними системами за допомогою стандартних протоколів зв'язку. Така мережа може мати датчики та виконавчі пристрої, вбудовані у фізичні об'єкти і пов'язані між собою через дротові чи бездротові мережі. Ці взаємопов'язані пристрої можуть зчитувати та приводити в дію, програмувати та ідентифікувати пристрої без участі людини за рахунок використання інтелектуальних інтерфейсів.

Основною концепцією протоколу IP є можливість підключення різних об'єктів (речей), які людина може використовувати у повсякденному житті. Це можуть бути такі пристрої, як кавоварка, холодильник, кондиціонер, автомобіль. Ці об'єкти (речі) повинні мати вбудовані сенсори для обробки інформації, що 6 надходить з навколишнього середовища, обмінюватися нею і виконувати різні дії в залежності від отриманої інформації. Прикладом є система «розумний будинок» або «розумна ферма». Ця система аналізує дані навколишнього середовища і, наприклад, в залежності від показників, регулює температуру в приміщенні. У зимовий період регулюються інтенсивність опалення, у спеку будинок має механізми відкривання і закривання вікон, завдяки чому будинок провітрюється, при чому все це відбувається без втручання людини.

Для об'єднання побутових речей у мережу необхідно використати декілька технологій. Система унікальної ідентифікації дозволяє збирати та накопичувати інформацію про предмети. Це можна забезпечити за допомогою мікросхем RFID

(Radio-Frequency IDentification). Вони здатні без власного джерела струму передавати інформацію приладам зчитування. Кожна мікросхема має індивідуальний номер. Альтернативою даної технології для ідентифікації об'єктів може бути використання QRC (Quick Response Code). Для визначення точного місцеположення пристрою використовується технологія GPS (Global Positioning System). Для відслідковування змін у стані елементу або оточуючого середовища об'єкти оснащуються сенсорами. Для обробки та накопичення даних з сенсорів використовується вбудований комп'ютер (наприклад, Raspberry Pi, Intel Edison тощо). Для обміну інформацією між пристроями можуть бути використані технології бездротових мереж (Wi-Fi, Bluetooth, ZigBee, 6LoWPAN).

Інтеграція з мережею Інтернет передбачає використання IP-адреси як унікального ідентифікатора. Через обмеження адресного простору в IPv4 пристроям потрібно використовувати IPv6, який забезпечує унікальними адресами мережевого рівня не менше 300 млн. пристроїв на одного жителя Землі. Об'єктами в IP будуть не тільки пристрої із сенсорними можливостями, але й пристрої, які виконують дії (наприклад, лампочки або замки, якими можна керувати через мережу Інтернет). Важливими характеристиками бездротової передачі даних для побудови IoT є ефективність, відмовостійкість, адаптивність та можливість самоорганізації. Стандарт IEEE 802.15.4 управляє доступом для організації енергоефективних 7 персональних мереж. Наприклад, сонячне випромінювання заряджає сонячний акумулятор, що надсилає електроенергію на акумулятор для зберігання та розподілу енергії. Вимірювач потужності, з'єднаний між ними, зчитує та відображає кількість енергії, яку захоплює сонячна панель.

Оскільки усі пристрої підключені, вони реєструються на сервері реєстрації, дозволяючи користувачеві контролювати всю систему з веб-браузера. Коли пристрої належним чином підключені, вони повинні бути налаштовані. Оскільки система покладається на IP мережу, пристрої повинні бути налаштовані з правильною інформацією про IP адреси. Оскільки сервер налаштований також на роль сервера DHCP, пристрої IoT повинні бути налаштовані як клієнти DHCP,

щоб автоматично вивчати інформацію IP. Користувач може контролювати енергоспоживання системи не тільки локально, але і дистанційно. Ще один крок – підключити мікроконтролер та написати код для вимкнення одного або декількох світлодіодів, коли рівень заряду акумулятора стане нижчим заздалегідь визначеного порогу. Цей режим енергозбереження дозволить батареї деякий час перезаряджатись.

Середовище моделювання Cisco Packet Tracer (CPT) має наступні можливості для створення та налаштування IoT рішень:

• підтримка SPAN / RSPAN;

• підтримка L2NAT;

• підтримка PTP;

• підтримка REP;

• підтримка LLDP;

• підтримка розподілу адрес на основі портів DHCP-сервера;

• PoEsupport;

• підтримка протоколу IoTSwitch.

На рис. 2.1 зображені пристрої, які можна використовувати як рішення IoT для моделювання та тестування їх роботи в середовищі Cisco Packet Tracer.

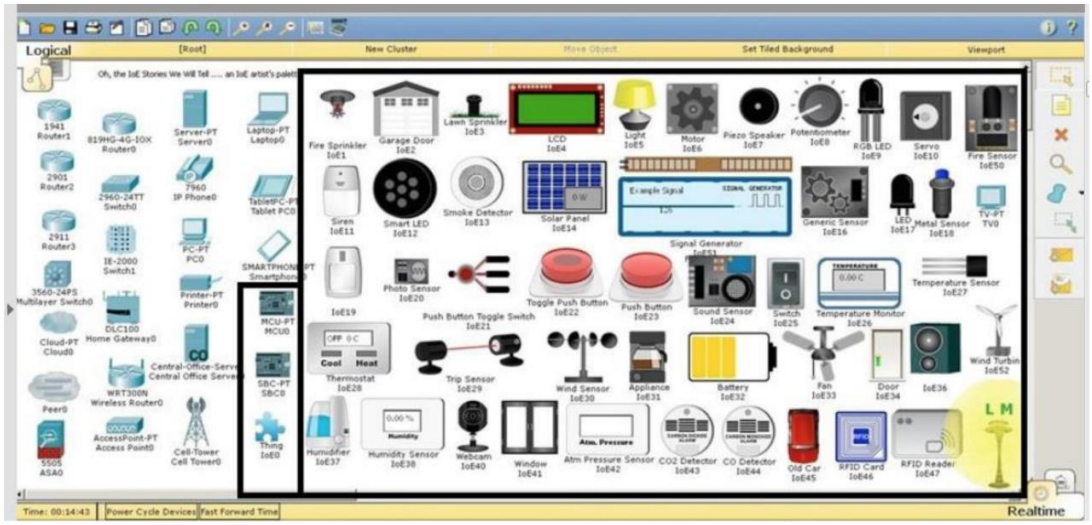


Рисунок 2.1 - Пристрої IoT у середовищі моделювання Cisco Packet Tracer

Для керування розумним будинком або системами, які базуються на основі Інтернету речей використовують системи з об'єднанням великої кількості

сенсорів. Для створення прототипу найчастіше використовують мікрокомп’ютери такі як Raspberry Pi та Arduino Uno, які є основними на ринку.

Для керування пристроями та системою в цілому найчастіше використовують модулі Wi-Fi та Bluetooth, що допомагають здійснювати контроль та нагляд з мобільних пристроїв.

Raspberry Pi - одноплатний мікрокомп’ютер розмірами як банківська карта (рис. 2.2). Спочатку цей мікрокомп’ютер було створено як бюджетна система і пристрій для вивчення інформатики але пізніше набув своєї популярності та почав широко використовуватись у великих проектах.



Рисунок 2.2 - Мікрокомп’ютер Raspberry Pi

Arduino Uno - апаратна обчислювальна платформа для аматорського конструювання, основними компонентами якої є плата мікроконтролера з елементами вводу/виводу та середовище розробки Processing/Wiring на мові програмування, що є спрощеною підмножиною C/C++ (Рис. 2.3). Arduino дозволяє комп'ютеру вийти за рамки віртуального світу в фізичний і взаємодіяти з ним. Пристрої на базі Arduino можуть отримувати інформацію про навколишнє середовище за допомогою різних датчиків, а також можуть управляти різними виконавчими пристроями. Мікроконтролер на платі програмується за допомогою мови Arduino (заснований на мові Wiring) і середовища розробки Arduino (заснована на середовищі Processing). Проекти пристроїв, засновані на

Arduino, можуть працювати самостійно, або ж взаємодіяти з програмним забезпеченням на комп'ютері (Flash, Processing, MaxMSP). Плати можуть бути зібрані користувачем самостійно або куплені в зборі. Програмне забезпечення 19 доступне для безкоштовного скачування. Вихідні креслення схем (файли CAD) є загальнодоступними, користувачі можуть застосовувати їх на свій розсуд. У 2006 Arduino отримала визнання в категорії Digital Communities на фестивалі Ars Electronica Prix.



Рисунок 2.3 - Мікрокомп’ютер Arduino Uno

Плати розширення, що встановлюються на платформи, це плати які розширюють функціональність Arduino для управління різними пристроями, отримання даних і т.д.

• Плата розширення WiFi використовується для з'єднання з бездротовими мережами стандарту 802.11 b / g.

• Плата розширення Xbee Shield забезпечує за допомогою модуля Maxstream Xbee Zigbee бездротовий зв'язок декільком пристроям Arduino в радіусі до 35 метрів (в приміщенні) і до 90 метрів (поза приміщенням).

• Плата розширення Motor Shield забезпечує управління двигунами постійного струму і читання датчиків положення.

• Плата розширення Ethernet Shield забезпечує підключення до інтернету.

За допомогою об'єднання програмного та апаратного забезпечення можна створити систему захисту комірки в банку, що і є основною метою випускної кваліфікаційної роботи бакалавра.



Рисунок 2.4 - Схема керування системою

2.2. Розробка алгоритмів функціонування моделі IoT-системи захисту банківської комірки

Для реалізації та правильної роботи системи було створено алгоритм функціонування. Для того, щоб детально дослідити послідовність дій, алгоритм було розділено на стани (кроки), для чіткого розуміння як працює система.

Отож початковим станом є закриті двері при вході до банківської комірки.(рис.2.5)

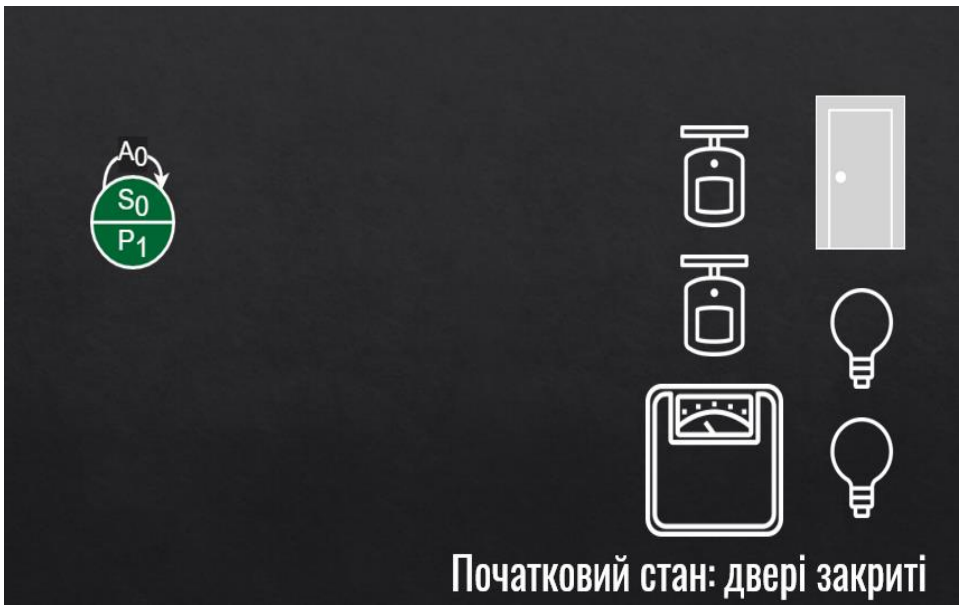


Рисунок 2.5 – Початковий стан

Система оснащена різними видами датчиків зокрема датчиками Інфрачервоного випромінювання. Коли клієнт прикладає RFID чіп до зчитувача, двері відкриваються і він заходить до кімнати де відразу спрацьовує перший інфрачервоний датчик, який помічає перешкоду (клієнта)і вмикає світло. (рис.2.6)



Рисунок 2.6 – Перший датчик помічає клієнта

Після чого спрацьовує другий датчик, який також реагує на рух.(Рис.2.7)



Рисунок 2.7 – Другий датчик помічає клієнта

Після того як клієнт пройде два інфрачервоних датчика він потрапляє до банківської комірки де виконує банківські операції. (рис.2.8)



Рисунок 2.8 – Перший ІЧ датчик перестає помічати перешкоду

При проходженні другого ІЧ датчику блокуються двері від стороннього втручання(тобто інший клієнт не зможе відкрити дані двері за допомогою свого RFID чіпу коли всередині знаходиться інша людина).(рис.2.9)



Рисунок 2.9 – Другий датчик іч перестає помічати перешкоду

Поруч з коміркою знаходиться датчик ваги який вмикає та вимикає світло у кімнаті. (2.10)

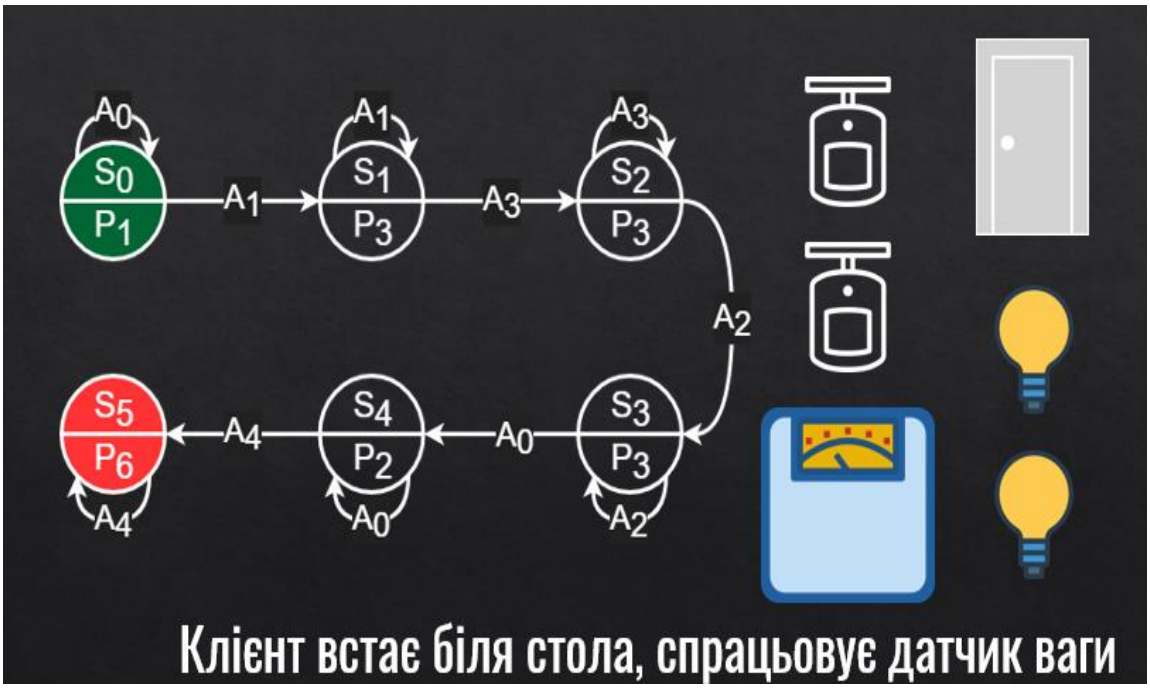


Рисунок 2.10 - Спрацьовування датчику ваги

На зворотному шляху клієнт проходить ті ж самі датчики та виходить з банку.

Завдяки таким етапам рівень безпеки є найвищим адже під час виконання кожного кроку відстежуються та контролюються всі дії клієнта, що дозволяє безпечно виконати ту чи іншу операцію.

Отже для максимального рівня захисту необхідно облаштувати середовище (банківську комірку) датчиками, які будуть контролювати всі процеси.

2.3. Вибір інструментів Cisco Packet Tracer для реалізації моделі IoT-системи захисту банківської комірки

Ідея розумної системи дуже приваблива, тому було зроблено огляд подібних досліджень які проводилися з метою розуміння понять та архітектури IoT. Для впровадження розумної системи використовувався Cisco Packet Tracer. В нещодавно випущеній версії Cisco з'явилася підтримка функції домашньої безпеки, а також забезпечення програмування оточення, крім функцій мереж. В PT підтримуються такі мови програмування як visual basic, java scripts, and python[9]. Смартфон та home gateway використовуються для керування пристроями, такими як смарт-вікно, розумний вентилятор, розумне сміття та датчики. Розумні пристрої підключені до порту home gateway IoT і смартфону який може спілкуватися зі смарт-пристроями.(рис.2.11)

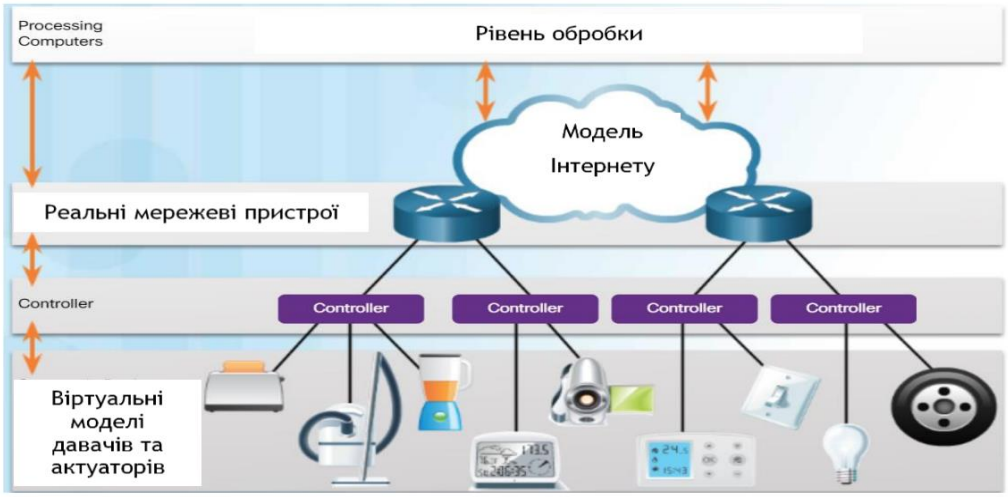


Рисунок 2.11 - Комбінована модель інтернету речей

Cisco Packet Tracer дозволяє використовувати різні фізичні середовища, «розумні пристрої», датчики і активатори, Smart Home, Smart City, Industrial, Power Grid, редагувати існуючі або програмувати власні пристрої, використовуючи технології Python, Javascript, Blockly, SBC та MCU, Home Gateway та RegServer, налаштовувати правила роботи пристроїв[10]. Мікроконтролери змішаного типу також є інтегруючими аналоговими компонентами, вони необхідні для управління нецифровими електронними системами. Cisco Packet Tracer забезпечує підтримку емулятора MCU. Користувач може запрограмувати PT MCU для виконання завдань, подібних до MCU у реальному світі. Щоб спростити процес, PT MCU може бути запрограмований за допомогою Java і Python[11]. PT MCU – це плата, що має порт USB, шість цифрових портів вводу-виводу та чотири аналогові порти вводу-виводу. Цифрові входи / виходи на PT MCU дозволяють користувачеві підключати цифрові датчики та виконавчі пристрої. Аналогові порти вводу-виводу дозволяють користувачеві підключати аналогові датчики та виконавчі пристрої. Python працює на різних комп'ютерних платформах. Cisco Packet Tracer використовує цю перевагу та реалізує пристрої IoT, що підтримують Python. Радіочастотна система ідентифікації (RFID) - це автоматична технологія що допомагає машинам або комп’ютерам ідентифікувати об'єкти, записувати метадані або керувати окремою ціллю за допомогою радіохвиль. Підключаючи RFID-зчитувач до терміналу Інтернету, зчитувачі можуть ідентифікувати, відстежувати та відслідковувати об'єкти, приєднані з тегами глобально, автоматично та в режимі реального часу, якщо це потрібно. Це так званий Інтернет речей (IOT). RFID часто розглядають як необхідну умову IOT. Передбачуване мережеве суспільство - це таке, в якому повсякденні об'єкти доповнюються за допомогою функцій зондування, взаємодії в мережі та обробки даних, взаємодіючих в Інтернеті речей (IoT). Радіочастотна ідентифікація (RFID) по-різному описується як ключовий фактор, що сприяє IoT. Зважаючи на їх традиційне застосування у відстеженні активів та управлінні запасами, теги RFID, вбудовані в розумні об'єкти, забезпечують спосіб ідентифікації та

відстеження цих об'єктів у режимі реального часу. Все більш актуальною програмою є те, в якій теги RFID використовуються в якості датчиків для збору відповідних даних про оточення, в яке розміщуються розумні об'єкти. Промислові прогнози передбачають, що невеликі датчики низької потужності в майбутньому будуть рахувати більшість бездротових з'єднань, при цьому численні застосування в транспорті, охороні здоров'я, виробництві, серед інших. Однією з найважливіших ідей, що сприяють застосуванню технології RFID в бізнес-додатках, є концепція так званого "Інтернету речей". Оскільки RFID дозволяє ідентифікувати речі за допомогою комп'ютерних систем, він дозволяє додаткам ставати «відомими».Як результат, RFID - одна з ключових технологій, від якої залежить Інтернет речей. RFID-теги можна використовувати для того, щоб розповісти програмам, що сталось, де вони, якщо вони перемістилися, хто перемістив їх чи використав. Для виконання даного проекту нам необхідно використовувати Cisco Packet Tracer. Cisco Packet Tracer є дуже зручним засобом моделювання мереж передачі даних. Основні інструменти його інтерфейсу розташовані і організовані цілком логічно, що спрощує процес знайомства з програмою, а також значно скорочує час, необхідний для попередньої підготовки. Є деякі особливості використання специфічних для Cisco Packet Tracer інструментів, але ці інструмент використовуються не часто. Для створення проекту IoT-системи захисту банківської комірки було викростано:



Рисунок 2.12 - Розумні двері

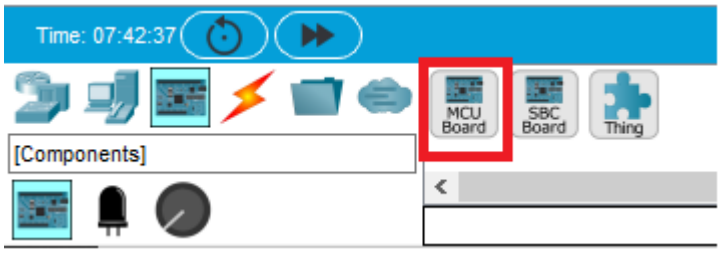


Рисунок 2.13 - Плата для запрограмовування

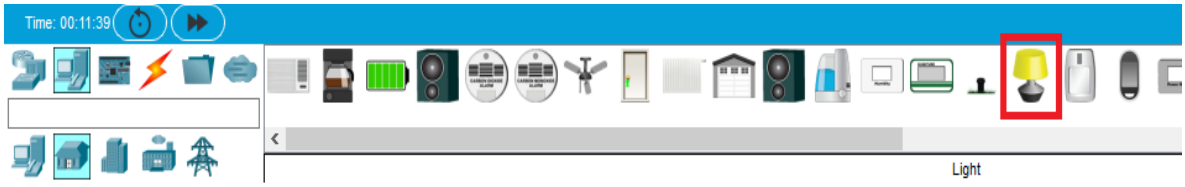


Рисунок 2.14 - Джерело світла

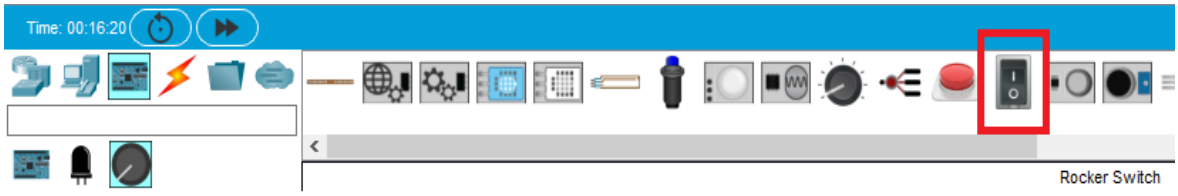


Рисунок 2.15 - Електричний вимикач

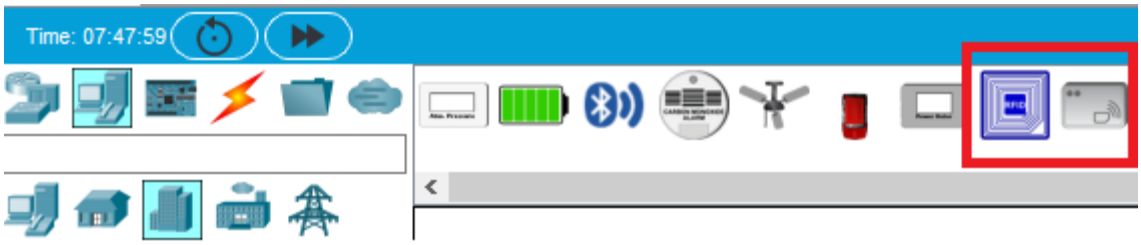


Рисунок 2.16 - RFID зчитувач та чіп

Комп'ютерна мережа, представлена в цій роботі, була розроблена в середовищі моделювання Cisco Packet Tracer. Даний програмний продукт розроблений компанією Cisco і рекомендований для використання при вивченні телекомунікаційних мереж і мережевого устаткування. За допомогою даного програмного продукту можна створювати і конфігурувати мережі і проводити в них пошук несправностей. Packet Tracer дає можливість більш детально представляти новітні технології, тим самим роблячи навчальний процес надзвичайно корисний з точки зору засвоєння отриманого матеріалу.

Cisco Packet Tracer спеціально розроблений для вивчення сучасних телекомунікаційних систем, і більше а ніж інші симулятори відповідає цьому завданню. Даний симулятор дозволяє проектувати свої власні мережі, створюючи і відправляючи різні пакети даних, зберігати і коментувати свою роботу. Відмінною особливістю даного симулятора є наявність в ньому "Режиму моделювання". В даному режимі всі пакети, що пересилаються всередині мережі, відображаються графічно.

При розробці комп'ютерної мережі використовувалися такі структурні елементи як: робочі станції PC-PT, сервери, маршрутизатори 1841, комутатори та набір елементів пакету IoT (лампи, сигналізація, пожежна сигналізація, розумні вікна та двері, кондиціонери).

Робоча станція як місце роботи фахівця є повноцінним комп'ютером або комп'ютерний термінал (пристрої введення-виведення, відокремлені і часто віддалені від керуючого комп'ютера), набір необхідного ПО, в разі потреби доповнюються допоміжним обладнанням: принтер, зовнішній пристрій зберігання даних на магнітних і / або оптичних носіях, сканер штрих-коду тощо.

Сервер - це модуль, який постійно очікує приходу з мережі запитів від клієнтів, і прийнявши запит, намагається його обслужити, як правило, за участю локальної операційної системи; один сервер може обслуговувати запити відразу декількох клієнтів (по черзі або одночасно).

Маршрутизатор представляє собою програмований обчислювальний пристрій, який працює під управлінням спеціалізованої операційної системи, оптимізованої для виконання операцій побудови таблиць маршрутизації і просування пакетів на їх основі.

Висновок до розділу 2

У другому розділі було створено структурну схему моделі IoT-системи захисту банківської комірки.У результаті було розроблено алгоритм роботи даної системи. Переглянуто та вибрано інструменти Cisco Packet Tracer для створеня даної моделі. У результаті коротко розказано про них і обгрунтовано їхнє використання. Було розглянуто технологію RFID, яка створена для ідентифікації об’єктів та буде використана під час моделювання системи захисту банківської комірки.

РОЗДІЛ 3

ОЦІНКА ПРАЦЕЗДАТНОСТІ ТА ЕФЕКТИВНОСТІ МОДЕЛІ ІОТ-СИСТЕМИ ЗАХИСТУ БАНКІВСЬКОЇ КОМІРКИ

3.1 Реалізація моделі IoT-системи захисту банківської комірки

Для створення моделі IoT-системи захисту банківської комірки було вирішено обрати середовище розробки Cisco Packet Tracer.

Cisco Packet Tracer – це мультифункціональна програма моделювання мереж, яка дозволяє експериментувати з різною поведінкою мереж і оцінювати можливості різних сценаріїв. Цей інструмент доповнює також фізичне обладнання, яке дозволяє створювати мережі з майже необмеженою кількістю пристроїв та допомагає отримувати практичні навички зі створення та налагодження мереж[12].

Нещодавно, Cisco випустили нову версію програми, що включає в себе пакет IoT мережевих пристроїв. До цієї версії також було додано мікроконтролери (MCU-PT) та одноплатні комп'ютери (SBC-PT), які забезпечують програмну розробку та керування підключених пристроїв.

Перевагами нової версії Cisco Packet Tracer є:

- пропонує реалістичну візуалізацію та моделювання IoT пристроїв;
- дозволяє користувачам проектувати, створювати та налаштовувати "розумне місто", тим самим надаючи різні розумні об'єкти для його створення;
- забезпечує платами для керування розумними об'єктами;
- дозволяє студентам досліджувати детальніше галузь "Інтернет речей";

З метою впровадження розумної системи у проект, було вирішено використати нову версію Cisco Packet Tracer в яку включено такі розумні пристрої: розумне вікно, двері, освітлення, датчики вогню та диму, датчики руху та інші[13].

Для створення системи захисту банку яка також включає банківську комірку було використано Cisco Packet Tracer 7.3.0. Під час створення структурної схеми було використано різні типи розумних пристроїв з вбудованого пакету IoT.(рис.3.1)



Рисунок 3.1 - Архітектура моделі банку яка включає банківську комірку

На Рис.3.1 зображено спроектовану модель з використанням розумних пристроїв та датчиків IoT. Дана система включає в себе два мікроконтролера MCU-PT, які використовуються для контролю над пожежною сигналізацією та системою захисту від несанкціонованого доступу.

Для того щоб керувати розумними об’єктами та датчиками використовується мікроконтролер (MCU-PT) та Home Gateway, оскільки вони забезпечують середовище програмування для управління підключеним до нього розумним об'єктом та забезпечують механізми управління, реєструючи розумний пристрій відповідно до Home Gateway.

Home Gateway має 4 порти Ethernet в додаток до Wireless порту. Для безпечного бездротового зв'язку WEP / WPA-PSK / WPA2 можна налаштувати Home Gateway. Пристроєм IoE можна дистанційно керувати через розміщений веб-інтерфейс. Внутрішня (локальна) IP-адреса домашнього шлюзу - 192.168.25.1, але до неї також можна отримати доступ через Інтернет-адресу, що звертається до Інтернету[14].

Мікроконтролерна плата використовується для з'єднання різних розумних об'єктів та забезпечення середовища програмування різною мовою - JavaScript, Python та Visual Basic, для управління підключеним розумним об'єктом.

Мікроконтроллер який використовується для захисту банківської комірки, додатково оснащений резервною батареєю , яка живиться від сонячної енергії. Завдяки такому рішенню, система захисту банківської комірки буде працювати не залежно від електропостачання в офіс банку, та зможе бути захищеною навіть в екстрених ситуаціях (рис. 3.3).

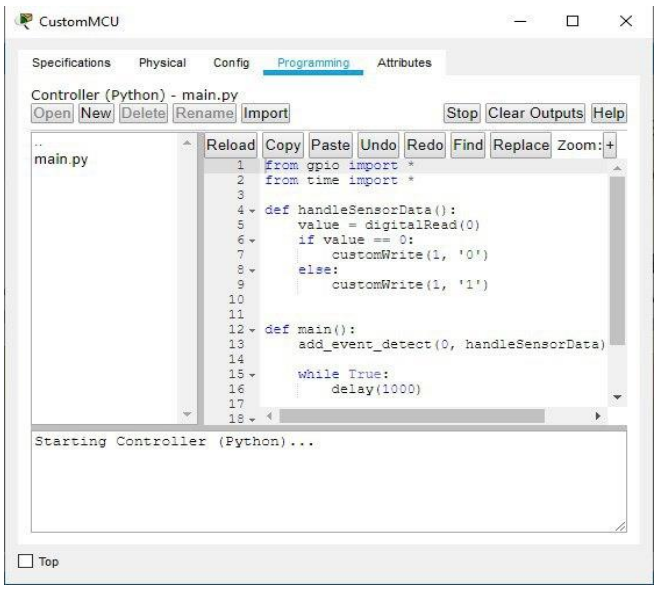


Рисунок 3.2 - Мікроконтролер MCU-PT

На рис. 3.3 зображено систему резервного живлення, яка складається з маршрутизатора, ноутбука, сонячної панелі, батареї та LED-індикаторів.

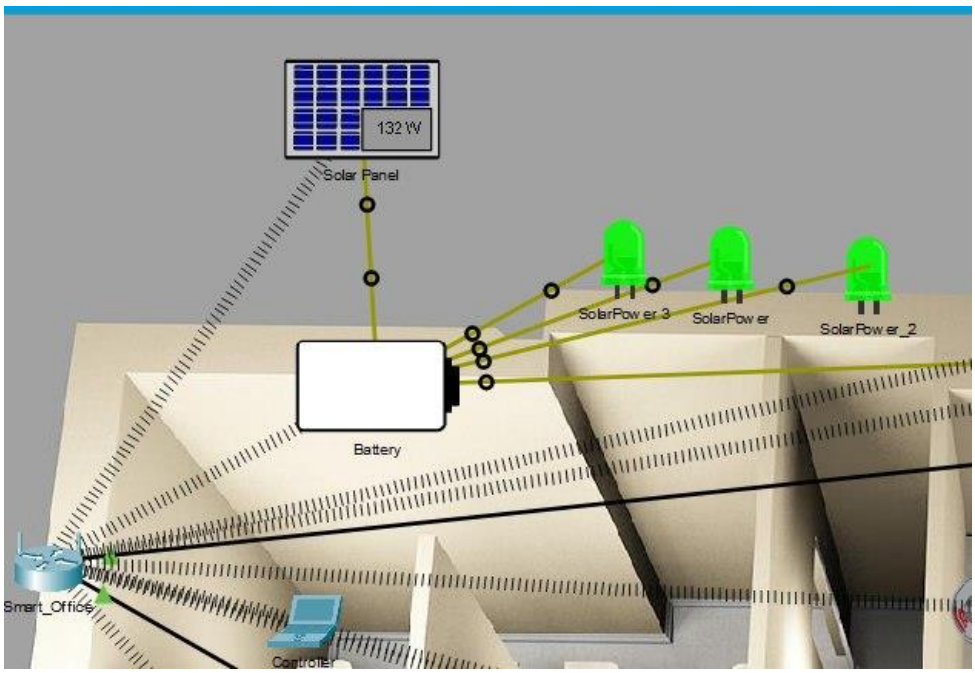


Рисунок 3.3 - Система резервного живлення

Сонячна панель працює та накопичує енергію у батарею, яка в свою чергу живить плату MCU-PT. Моніторинг роботи сонячної панелі, додаткову інформацію, та кількість накопиченої енергії можна здійснювати за допомогою підключеного у мережу ноутбука. В даному випадку панель працює з 8 ранку до 3 години дня. Завдяки LED – індикаторам можна побачити стан роботи сонячної батареї. Якщо індикатори світяться зеленим, то це означає що сонячна батарея акумулює енергію у даний момент часу. Коли сонячної енергії недостатньо, то відповідно робота сонячної панелі припиняються та індикатори перестають світитись.

Банківська комірка є одним з головних об’єктів у системі банку. Саме тому, необхідно забезпечити захист максимального рівня, за для безпеки клієнтів а також надійності зберігання. Для того, щоб захистити банківську комірку від несанкціонованого доступу, було використано ряд розумних пристроїв та датчиків.(рис.3.4)

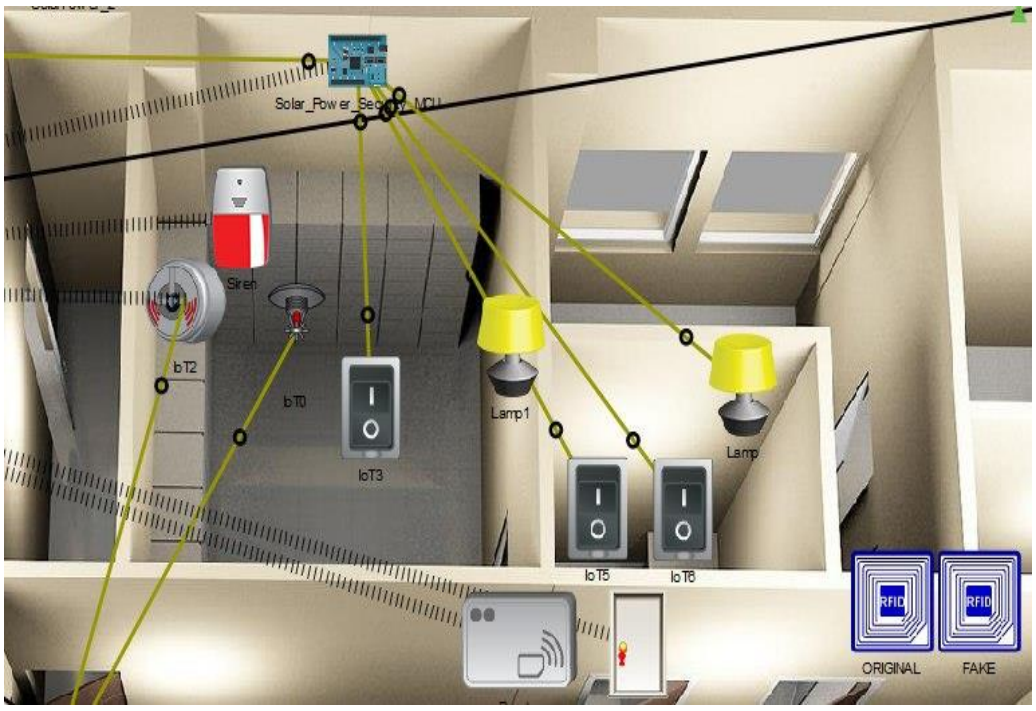


Рисунок 3.4 - Система захисту банківської комірки

.

3.2.Дослідження працездатності та ефективності моделі IoT системи захисту банківської комірки. Аналіз отриманих результатів.

Для того щоб дана система була придатна до використання у банківських системах , її роботу попередньо потрібно протестувати[15]. В даному проекті IoT було використано технологію RFID. На рис.3.5 і 3.6 показано стабільну роботу даної системи.

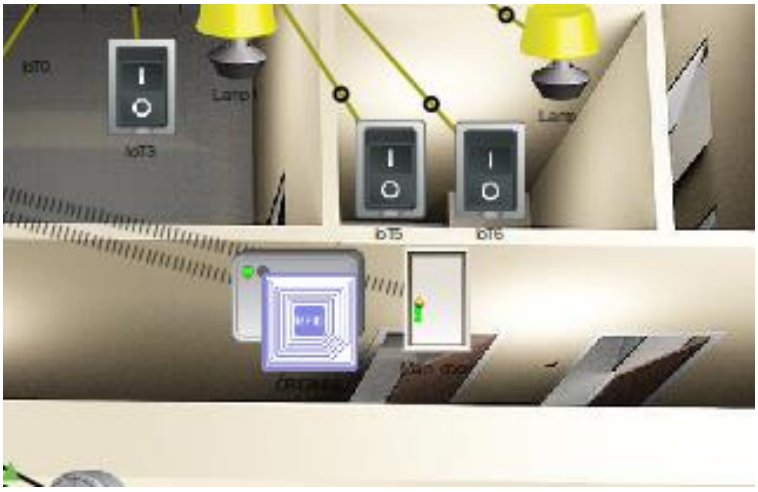


Рисунок 3.5 – Використання оригінального RFID чіпу(двері відкриті)

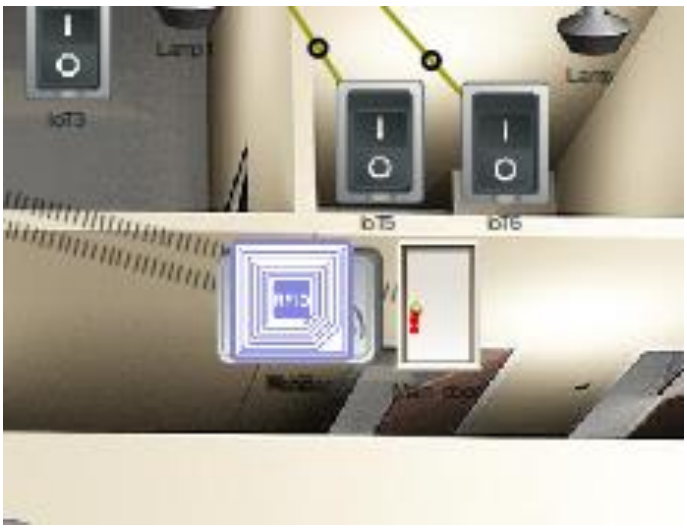


Рисунок 3.6 – Використання неоригінального RFID чіпу(двері закриті)

Далі клієнт заходить в сховище і потрапляє на ІЧ датчик який вмикає світло в даній кімнаті.(рис 3.7)

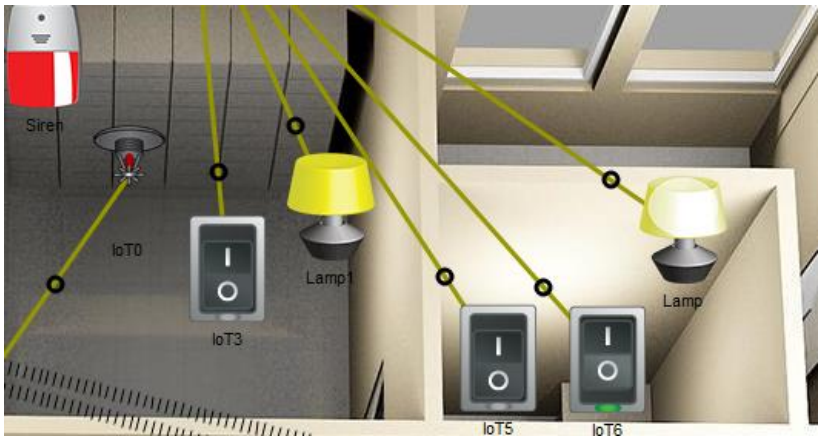


Рисунок 3.7 – Робота ІЧ датчика

Далі клієнт підходить до комірки де наступає на площу з датчиком ваги пілся чого також вмикається світло але вже біля комірки (рис.3.8).



Рисунок 3.8 – Робота датчика ваги

Окрім системи захисту банківської комірки було додатково встановлено систему пожежогасіння з датчиком диму(рис.3.9)

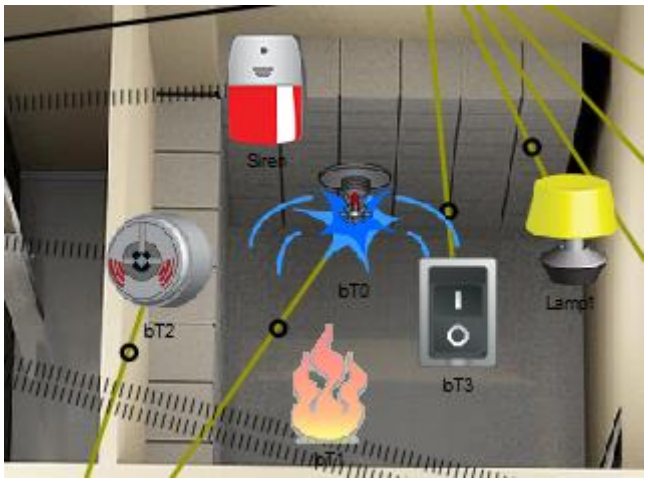


Рисунок 3.9 - Робота системи пожежогасіння

Отже, на даних рисунках було відображено справність всіх елементів захисту банківської комірки і також їхня ефективність і готовність до використання в реальних банківських системах.

Висновок до розділу 3

Реалізована схема банку системи захисту банківської комірки, використовуючи новий випущений пакет Cisco, оскільки ця версія включала різні пристрої IoT, які використовуються для автоматизації. Було використано Home Gateway для реєстрації розумних пристроїв на ньому. У створеній системі використовується мікроконтролер (MCU-PT) для взаємозв'язку різних датчиків та пристроїв IoT. Також, MCU-PT надає середовище програмування для управління різними пристроями. Було використано мови програмування JavaScript та Python для керування пристроями але також є можливість використовувати й інші мови. Результатом третього розділу є готова до використання ситема захисту банківської комірки, здійсненно тестування її працездатності та ефективності.

ВИСНОВКИ

В даній випускній кваліфікаційній роботі бакалавра було розроблено проект IoT-системи захисту банківської комірки.

У першому розділі було досліджено актуальність роботи та проаналізовано літературні джерела. Було проведено аналіз існуючих рішень. Щоб показати всі переваги і недоліки розроблюваної системи, проведено аналіз та порівняння систем з іншими існуючими аналогами. Також було проведено аналіз архітектури IoT-систем. Результатом аналізу є підбір необхідних елементів для створення даного проекту.

Дана модель може використовуватись в банках, тому що це вигідне і надійне рішення. Щоб отримати даний результат, було переглянуто багато джерел і підібрано оптимальне рішення для створення моделі.

Для реалізації проекту було використано Cisco Packet Tracer і навчальні матеріали Cisco Academy.

Результатом випускної кваліфікаційної роботи бакалавра є готова модель, яку можна застосовувати у банках, а саме для захисту банківських комірок.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. MaciejKranz,BuildingtheInternetofThings— 2016. — № 1. — с.59–67.

2. BruceSinclair, IoT Inc: How Your Company Can Use The Internet Of Things — 2017. — № 17. — с. 14–16.

3. Cisco Systems: документация [Электронный ресурс]. Режим доступа:

http://www.cisco.com/en/US/docs .

4. Cisco Systems: конфигурации [Электронный ресурс]. Режим доступа:

http://www.cisco-conf.ru/ .

5. Jamil Y.Khan, and Mehmet R. Yuce, "Internet Of Things : Systems and Applications", USC/Information Sciences Institute, 2019.

6. QusayF. Hassan,Internet Of Things A to Z / 2018. — 104 с.

7. MarcoGarbati, EtiennePerret,Chipless RFID Reader Design for Ultra Wideband Technology —2018. — 50 с.

8. Mulayam Singh,.Network with Practical: ALL PACKET TRACER LABS // — 2020. — № 12. — с. 21–26.

9. Документация компании Cisco [Электронный ресурс]. – Режим доступа к ресурсу: http://www.cisco.com/ en/US/docs/ios/12_2/ip/configuration/guide/1cfipadr.html#wp 1001061.

10. Савельев А. Современные протоколы маршрутизации. [Электронный ресурс]. – Режим доступа к ресурсу: http://www.citforum.ru/nets/articles/ sovr_marsh.shtml.

11.Margaret Rouse. Internet Of Things. [Электронныйресурс].- Режим доступа к ресурсу:https://internetofthingsagenda.techtarget.com/definition/Internet-of-Things-IoT.

12.Concept of IoT (Internet of thing) on Cisco Packet Tracer. [Електронний ресурс]. – Режим доступу: http://telecomacadmey.com/concept-iot.

13.Andrea Finardi IoT Simulations with Cisco Packet Tracer. [Електронний ресурс]. – Режим доступу: https://www.theseus.fi/bitstream/handle/10024/150158/Andrea%20Finardi%20%20Master%20of%20Engine ering%20%20Information%20technology.pdf?sequence=1&isAllowed=y.

14.Приложения GNS3 и Cisco Packet Tracer Student. [Електронний ресурс]. – Режим доступу: https://upweek.ru/prilozheniya-gns3-i-cisco-packet-tracer-student-chto-nam-stoit-set-postroit-narisuembudem-zhit.

15.А. Соммер, Интернет вещей: Arduino в связке с облаком. [Електронний ресурс]. – Режим доступу: https://habr.com/ru/company/microsoft/blog/343450

ДОДАТКИ

```
from options import Options
from time import *
from physical import *
from gpio import *
from ioeclient import IoEClient


DELAY_TIME = 1000       # var DELAY_TIME
current_time = 0        # var current_time

X_READ_DISTANCE = 50        # var X_READ_DISTANCE
Y_READ_DISTANCE = 50        # var Y_READ_DISTANCE
cardID = 0        # var cardID
lastCardID = 0        # var lastCardID
state = 2 # waiting        # var state


def setup ():

    # Registration Server Setup

    IoEClient.setup({
        "type": "RFID Reader",
        "states": [{
                "name": "Card ID",
                "type": "number",
                "unit": '',
                "controllable": False
            },
            {
                "name": "Status",
                "type": "options",
                "options": {
                    "0": "Valid",
                    "1": "Invalid",
                    "2": "Waiting"
                },
                "controllable": True
        }]
    })

    IoEClient.onInputReceive( lambda rinput: processData(rinput, True) )



def loop ():
    global cardID, lastCardID, state
    devices = devicesAt(getCenterX(), getCenterY(), X_READ_DISTANCE, Y_READ_DISTANCE)        # var devices
```

```
    found = False       # var found
    for i in xrange(0, len(devices)) :
        if devices[i] is getName():
            continue

        cardID = getDeviceProperty(devices[i], 'CardID')
        found = True
        break


    if not found:
        cardID = lastCardID = 0
        setState(2)
    else:
        if lastCardID != cardID:
            lastCardID = cardID
            sendReport()


    delay(DELAY_TIME)



def setState (newState):
    global  state
    if state != newState:
        state = newState
        analogWrite(A1, state)
        sendReport()




def sendReport ():
    report  = str( int(cardID)) + "," + str(state)       # var report
    IoEClient.reportStates(report)



def processData (data, bIsRemote):
    if len(data) <= 0:
        return
    data = data.split(",")
    setState(int(data[1]))



if __name__ == "__main__":
    setup()
    while True:
        loop()
        idle()
```

Rfid cards

```
from time import *
from physical import *


CARD_ID = 1001        # var CARD_ID


def setup ():
            global CARD_ID
            CARD_ID = restoreProperty('CardID', CARD_ID)
            setDeviceProperty(getName(), 'CardID', CARD_ID)

def restoreProperty(propertyName, defaultValue):
    value = getDeviceProperty(getName(), propertyName)
    if  not (value is "" or value is None):
        if  type(defaultValue) is int :
            value = int(value)

        setDeviceProperty(getName(), propertyName, value)
        return value
    return defaultValue

if __name__ == "__main__":
    setup()
    while True:
        sleep(3600)
```


```
from time import *
from physical import *


CARD_ID = 1001        # var CARD_ID


def setup ():
            global CARD_ID
            CARD_ID = restoreProperty('CardID', CARD_ID)
            setDeviceProperty(getName(), 'CardID', CARD_ID)

def restoreProperty(propertyName, defaultValue):
    value = getDeviceProperty(getName(), propertyName)
    if  not (value is "" or value is None):
        if  type(defaultValue) is int :
            value = int(value)

        setDeviceProperty(getName(), propertyName, value)
        return value
    return defaultValue

if __name__ == "__main__":
    setup()
    while True:
        sleep(3600)
```

```
from time import *
from physical import *
from gpio import *
from environment import Environment
from ioeclient import IoEClient

ENVIRONMENTS = ["Argon", "CO", "CO2", "Hydrogen", "Helium", "Methane", "Nitrogen", "O2", "Ozone",
"Propane", "Smoke"]
ENVIRONMENT_MAX_IMPACT = -0.02 # 2% max when door opens
TEMPERATURE_TRANSFERENCE_MULTIPLIER = 1.25 # increase speed 25% when door open
HUMIDITY_TRANSFERENCE_MULTIPLIER = 1.25
GASES_TRANSFERENCE_MULTIPLIER = 2

doorState = 0 # 0 is closed, 1 is opened
lockState = 0  # 0 is unlocked, 1 is locked

def on_event_detect_0 () :
    processData(customRead(0), False)

def on_input_receive(input) :
    processData(input, True)

def setup ():
    IoEClient.setup({
        "type": "Door",
        "states": [{
            "name": "Open",
            "type": "bool"
        },
        {
            "name": "Lock",
            "type": "options",
            "options": {
                0: "Unlock",
                1: "Lock"
            },
            "controllable": True
        }]
    })

    IoEClient.onInputReceive(on_input_receive)

    add_event_detect(0, on_event_detect_0)

    setDoorState(doorState)
    setLockState(lockState)


def mouseEvent (pressed, x, y, firstPress):
    if firstPress:
        if  isPointInRectangle(x, y, 10,40,5,10) :
            if  lockState == 0 :
                setLockState(1)
            else:
```

```
            setLockState(0)
        else:
            if  doorState == 0 :
                openDoor()
            else:
                closeDoor()

def processData (data, bIsRemote):
    if  len(data) < 1 :
        return

    print data

    data = data.split(",")
    doorStateData = int(data[0])
    lockStateData = int(data[1])
    if  lockStateData > -1 :
        setLockState(lockStateData)

    if  doorStateData > -1 and not bIsRemote :
        if  doorStateData == 0 :
            closeDoor()
        else:
            openDoor()


def sendReport ():
    report = str(doorState)+","+str(lockState)    # comma seperated states
    customWrite(0, report)

    IoEClient.reportStates(report)
    setDeviceProperty(getName(), "door state", doorState)
    setDeviceProperty(getName(), "lock state", lockState)


def closeDoor ():
    setDoorState(0)
    updateEnvironment()


def openDoor ():
    if  lockState == 0 :
        setDoorState(1)
        updateEnvironment()
    else:
        print "can't open locked door"


def setDoorState (state):
    global doorState
    if  state == 0:
        digitalWrite(1, LOW)
        setComponentOpacity("led", 1)    # show the led
    else:
```

```
        digitalWrite(1, HIGH)
        setComponentOpacity("led", 0)    # hide the led

    doorState = state
    sendReport()


def setLockState (state):
    global lockState
    if  state == 0 :
        digitalWrite(2, LOW)
    else:
        digitalWrite(2, HIGH)

    lockState = state
    sendReport()



def updateEnvironment ():
    rate, emax = 0, 0
    if  doorState == 1:
        for e in ENVIRONMENTS:
            emax = Environment.get(e) * ENVIRONMENT_MAX_IMPACT
            # the emax is reached in an hour, so we divide by 3600 to get seconds
            # then this rate is also based on 100,000 cubic meters (approx. coporate office size)
            rate = emax / 3600 * 100000 / Environment.getVolume()
            Environment.setContribution(e, rate, emax, True)
            Environment.setTransferenceMultiplier(e, GASES_TRANSFERENCE_MULTIPLIER)


        Environment.setTransferenceMultiplier("Ambient Temperature",
TEMPERATURE_TRANSFERENCE_MULTIPLIER)
        Environment.setTransferenceMultiplier("Humidity", HUMIDITY_TRANSFERENCE_MULTIPLIER)
    else:
        for e in ENVIRONMENTS:
            Environment.setContribution(e, 0, 0, True)
            Environment.removeCumulativeContribution(e)
            Environment.setTransferenceMultiplier(e, 1)

        Environment.setTransferenceMultiplier("Ambient Temperature", 1)
        Environment.setTransferenceMultiplier("Humidity", 1)


def isPointInRectangle (x,y, rx, ry, width, height):
    if width <= 0 or height <= 0:
        return False
    return (x >= rx and x <= rx + width and y >= ry and y <= ry + height)


if __name__ == "__main__":
    setup()
    while True:
        sleep(0)
```

```
from gpio import *
from time import *
from physical import *
from ioeclient import *
from environment import *


ENVIRONMENT_IMPACT_DIM = 10
VOLUME_AT_RATE = 100000

state = 0;  # 0 off, 1 low, 2 high
lastTimeInSeconds = 0

def main():
    setup()
    while True:
        loop()


def setup():

    IoEClient.setup({
        "type": "Light",
        "states": [
        {
            "name": "Status",
            "type": "options",
            "options": {
                "0": "Off",
                "1": "Dim",
                "2": "On"
            },
            "controllable": True
        }
        ]
    })

    IoEClient.onInputReceive(onInputReceiveDone)
    global state
    add_event_detect(0, detect)
    state = restoreProperty("state", 0)
    setState(state)

def detect():
    processData(customRead(0), False)

def onInputReceiveDone(analogInput):
    processData(analogInput, True)

def restoreProperty(propertyName, defaultValue):
    value = getDeviceProperty(getName(), propertyName)
    if  not (value is "" or value is None):
        if  type(defaultValue) is int :
            value = int(value)
```

```
        setDeviceProperty(getName(), propertyName, value)
        return value
    return defaultValue


def mouseEvent(pressed, x, y, firstPress):
    global state
    if firstPress:
        setState(state+1)

def loop():
    updateEnvironment()
    sleep(1)


def processData(data, bIsRemote):
    if  len(data) <= 0 :
        return
    setState(int(data))


def setState(newState):
    global state
    if newState >= 3 :
        newState = 0
    state = newState

    analogWrite(A1, state)
    customWrite(0, state)
    IoEClient.reportStates(state)
    setDeviceProperty(getName(), "state", state)

def updateEnvironment():
    global VOLUME_AT_RATE
    global ENVIRONMENT_IMPACT_DIM
    volumeRatio = float(VOLUME_AT_RATE) / Environment.getVolume()
    if  state is 0 :
        Environment.setContribution("Visible Light", 0,0, True)
    elif  state is 1:
        Environment.setContribution("Visible Light", ENVIRONMENT_IMPACT_DIM*volumeRatio,
ENVIRONMENT_IMPACT_DIM*volumeRatio, False)
    elif  state is 2 :
        Environment.setContribution("Visible Light", ENVIRONMENT_IMPACT_DIM*2*volumeRatio,
ENVIRONMENT_IMPACT_DIM*2*volumeRatio, False)

if __name__ == "__main__":
    main()

var ENVIRONMENT_IMPACT_DIM = 10;
var VOLUME_AT_RATE = 100000;

var state = 0;       // 0 off, 1 low, 2 high
var lastTimeInSeconds = 0;
```

```
function setup() {

	IoEClient.setup({
		type: "Light",
		states: [
		{
			name: "Status",
			type: "options",
			options: {
				"0": "Off",
				"1": "Dim",
				"2": "On"
			},
			controllable: true
		}
		]
	});

	IoEClient.onInputReceive = function(input) {
		processData(input, true);
	};

	attachInterrupt(0, function() {
		processData(customRead(0), false);
	});

	state = restoreProperty("state", 0);
	setState(state);
}

function restoreProperty(propertyName, defaultValue)
{
	var value = getDeviceProperty(getName(), propertyName);
	if ( !(value === "" || value == "undefined") ){
		if ( typeof(defaultValue) == "number" )
			value = Number(value);

		setDeviceProperty(getName(), propertyName, value);
		return value;
	}

	return defaultValue;
}

function mouseEvent(pressed, x, y, firstPress) {
	if (firstPress)
		setState(state+1);
}

function loop() {
	var val=customRead(0);
	if(val>0){
		setState(2);
```

```
        }
        else{
                setState(0);
        }
        updateEnvironment();

        delay(1000);
}

function processData(data, bIsRemote) {
        if ( data.length <= 0  )
                return;
        setState(parseInt(data));
}

function setState(newState) {
        if (newState >= 3)
                newState = 0;
        state = newState;

        analogWrite(A1, state);
        customWrite(0, state);
        IoEClient.reportStates(state);
        setDeviceProperty(getName(), "state", state);
}

function updateEnvironment()
{
        var volumeRatio = VOLUME_AT_RATE / Environment.getVolume();
        if ( state === 0 )
                Environment.setContribution("Visible Light", 0,0);
        else if ( state === 1)
                Environment.setContribution("Visible Light", ENVIRONMENT_IMPACT_DIM*volumeRatio,
ENVIRONMENT_IMPACT_DIM*volumeRatio, false);
        else if ( state === 2 )
                Environment.setContribution("Visible Light", ENVIRONMENT_IMPACT_DIM*2*volumeRatio,
ENVIRONMENT_IMPACT_DIM*2*volumeRatio, false);
}

from gpio import *
from time import *
from physical import *

is_open = False
P1 = 0
PImage = 1

def main():
   global is_open
   update(is_open)
   while True:
        delay(1000)
```

```
def mouseEvent(pressed, x, y, firstPress):
    global is_open
    # if mouse is first pressed, then flip it
    if firstPress == True:
        update(not is_open)

def update(open):
    global is_open
    global PImage
    global P1
    is_open = open
    if is_open == True:
        digitalWrite(PImage, HIGH)
    else:
        digitalWrite(PImage, LOW)

    if is_open == True:
        digitalWrite(P1, HIGH)
    else:
        digitalWrite(P1, LOW)

    setDeviceProperty(getName(), "state", is_open)

if __name__ == "__main__":
            main()


from time import *
from physical import *
from environment import Environment
from ioeclient import IoEClient
from gpio import *
from pyjs import JsObject

g_sensorData = None

g_processValue = lambda  unprocessedValue, deviceIDS: None

g_processMouse = lambda  pressed, x, y, firstPress: None

g_wasMouseClicked = False

g_delayTime = -1


def environmentType():
    return 'ENVIRONMENT'

def propertyType():
    return 'PROPERTY'


def analogType():
    return 'ANALOG'
```

```
def customType():
    return 'CUSTOM'

def digitalType():
    return 'DIGITAL'

def setWaitTimeMSec(timeMSec):
    global g_delayTime
    g_delayTime = timeMSec

def setCallbackOnProcessMouse(callback):
    global g_processMouse
    g_processMouse = callback


def setCallbackOnProcessValue(callback):
    global g_processValue
    g_processValue = callback

def setCallbackOnProcessValue(callback):
    global g_processValue
    g_processValue = callback

def updateSensor():
    if None is not g_sensorData:
        unprocessedValue = None
        value = None
        deviceIDS = []

            if environmentType() is g_sensorData.type:
            unprocessedValue = Environment.get(g_sensorData.propertyName)
        elif propertyType() == g_sensorData.type:
            if None is not g_sensorData.searchData:
                if getattr(g_sensorData.searchData, "w", None) and getattr(g_sensorData.searchData, "h", None):
                    # Search an area.
                    x = getX()
                    y = getY()

                    if 'CENTER' is g_sensorData.searchData.offset:
                        x = getCenterX()
                        y = getCenterY()

                        x -= (g_sensorData.searchData.w/2)
                        y -= (g_sensorData.searchData.h/2)
                    elif 'WORKSPACE' is g_sensorData.searchData.offset:
                        x = 0
                        y = 0


                    devices = devicesAt(x, y, g_sensorData.searchData.w,  g_sensorData.searchData.h)

                    deviceProperty = None
```

```
                unprocessedValue = []
                for dev in devices:
                    if g_sensorData.searchData.deviceIDs:
                        for dev_id in g_sensorData.searchData.deviceIDs:
                            deviceProperty = getDeviceProperty(dev, g_sensorData.propertyName)
                            if (dev is dev_id) and deviceProperty:
                                unprocessedValue.append(deviceProperty)

                                deviceIDS.append(dev)
                                break
                    else:
                        deviceProperty = getDeviceProperty(dev, g_sensorData.propertyName)
                        if deviceProperty:
                            unprocessedValue.append(deviceProperty)
                            deviceIDS.append(dev)

        if None is not unprocessedValue:
            value = g_processValue(unprocessedValue, deviceIDS)

            if g_sensorData.signalData:
                if analogType() is g_sensorData.signalData.type:
analogWrite(g_sensorData.signalData.slot, value)
                elif customType() is g_sensorData.signalData.type:
customWrite(g_sensorData.signalData.slot, value)
                elif digitalType() is g_sensorData.signalData.type:
digitalWrite(g_sensorData.signalData.slot, HIGH if value > 0 else LOW)

        g_sensorData.lastKnownValue = unprocessedValue
        g_sensorData.lastProcessedValue = value

    if g_sensorData.updateRegistrationServer:
        IoEClient.reportStates(g_sensorData.lastProcessedValue)


        if g_sensorData.updateRegistrationServer:
        IoEClient.reportStates(g_sensorData.lastProcessedValue)

updateMouseClick()

    setDeviceProperty(getName(), "state", g_sensorData.lastProcessedValue);

    if g_delayTime > 0:
        delay(g_delayTime)
    elif g_delayTime == 0:
sleep(0)

def mouseEvent(pressed, x, y, firstPress):
    global g_wasMouseClicked
    if firstPress: # we don't need the False branch for this if
        g_wasMouseClicked = firstPress
    if g_processMouse:
        g_processMouse(pressed, x, y, firstPress)
```

```
def updateMouseClick():
    global g_wasMouseClicked
    g_wasMouseClicked = False


def getMouseClicked():
    return g_wasMouseClicked


def setupDetectEnvironment(propertyName, signalData):
    global g_sensorData
    g_sensorData = JsObject({
        "type": environmentType(),
        "propertyName": propertyName,
        "signalData": JsObject(signalData),
        "lastProcessedValue": None,    # Store the last processed sensor value.
        "lastKnownValue": None,        # Stores the last unprocessed sensor value.
        "updateRegistrationServer": False # True to auto handle registration server updating.
    })


def setupDetectProperty(propertyName, searchData, signalData):
    global g_sensorData
    g_sensorData = JsObject({
        "type": propertyType(),
        "propertyName": propertyName,
        "searchData": JsObject(searchData),
        "signalData": JsObject(signalData),
        "lastProcessedValue": None,    # Store the last processed sensor value.
        "lastKnownValue": None,        # Stores the last unprocessed sensor value.
        "updateRegistrationServer": False # True to auto handle registration server updating.
    })


def createPropertySearch (deviceID, w, h, offset):
    if (None is w or None is h):
        w = 4000
        h = 4000
        offset = 'WORKSPACE'

    search = JsObject({"deviceIDs": [], "w": w, "h": h, "offset": offset})
    if (None is not deviceID):
search.deviceIDs[0] = deviceID

    return search


def createPropertySearchDevices(deviceIDs, w, h, offset):
    if (w is None or h is None):
        w = 4000
        h = 4000
        offset= 'WORKSPACE'

    search = JsObject({"deviceIDs": [], "w": w, "h": h, "offset": offset})
```

```
    if None is not deviceIDs:
search.deviceIDs = deviceIDs

    return search


def createAnalogWrite (slot):
    return {"type": analogType(), "slot": slot}


def createCustomWrite(slot):
    return {"type": customType(), "slot": slot}


def createDigitalWrite(slot):
    return {"type": digitalType(), "slot": slot}


def getLastKnownValue():
    return g_sensorData.lastKnownValue


    return g_sensorData.lastProcessedValue


def setupRegistrationServer(sensorName, propertyName):
    if g_sensorData:
        g_sensorData.updateRegistrationServer = True

        sensorType = None
        if analogType() is g_sensorData.signalData.type:
            sensorType = 'number'
        elif digitalType() is g_sensorData.signalData.type:
            sensorType = 'bool'
        else:
            g_sensorData.updateRegistrationServer = False
            print 'Slot type not applicable for registration'

        if g_sensorData.updateRegistrationServer:
            IoEClient.setup({
            'type': sensorName,
            'states': [{
                'name': propertyName,
                'type': sensorType,
                'controllable': False
                }]
```